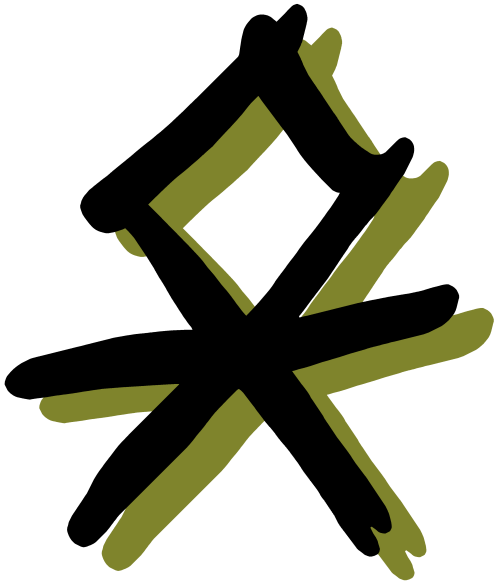


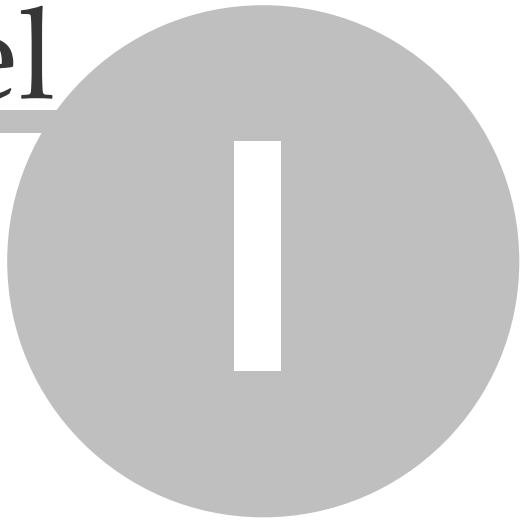
TrySim-Hilfe

Testen Sie Ihr SPS-Programm, bevor sich das erste Rädchen dreht!

Inhaltsverzeichnis

Einführung

Kapitel



1 Einführung

1.1 Einführung

TrySim ist ein Entwicklungswerkzeug für SPS-Programme. Im Gegensatz zu vielen anderen Simulationsprogrammen wird nicht nur die SPS nachgebildet, sondern auch die zu steuernde Maschine. So kann das mit TrySim erstellte Programm unter realitätsnahen Bedingungen getestet und optimiert werden, ohne dass irgendetwas anderes notwendig wäre, als ein PC mit Win98/NT/2000/XP/VISTA.

Optional können auch externe SPS-Simulationen oder reale Steuerungen angeschlossen werden. In TrySim wird dann nur die Maschine simuliert.

Für professionelle Programmierer bedeutet dies eine erhebliche Verkürzung der Inbetriebnahm Programme für die verschiedensten Maschinen zu schreiben und zu testen, ohne auf den Arbeitsplatz in seiner Ausbildungsstätte angewiesen zu sein.

Für die Simulation der Maschine steht eine Vielzahl von Elementen wie Motoren, Zylinder, Endschalter, Lichtschranken usw. zur Verfügung. Die Elemente können dreidimensional angeordnet werden, die graphische Darstellung entspricht einer technischen Zeichnung. Großen Wert haben wir auf Übersichtlichkeit und Geschwindigkeit gelegt, verzichtet haben wir auf optische Effekte, die den Nutzen nicht steigern. Zur besseren Veranschaulichung der simulierten Anlage kann diese echt dreidimensional dargestellt werden.

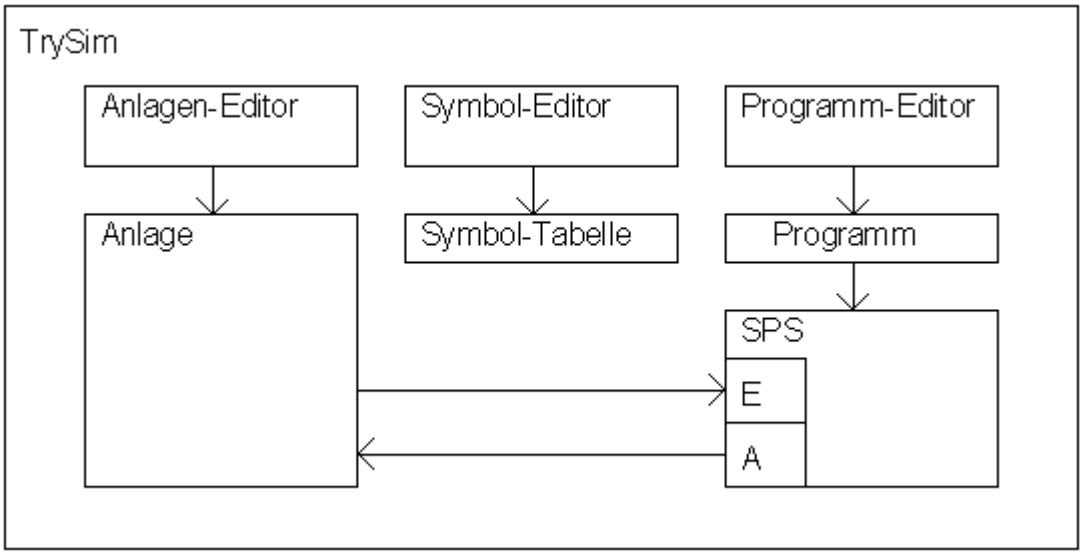
Die simulierte SPS verfügt über den Befehlssatz der Siemens S7-400 - Baureihe. Der Programmeditor ist nur leicht an das STEP®7-Entwicklungssystem angelehnt, es ist nicht unsere Absicht, die Ausbildung an einem Original Siemens PG zu ersetzen. Falls das fertige Programm auf einer wirklichen SPS laufen soll, kann es leicht auf das STEP®7-System übertragen werden. Auch der umgekehrte Weg, ein Programm auf dem Siemens STEP®7-System zu schreiben und dann zum Testen nach TrySim zu übertragen, ist ohne Weiteres möglich.

STEP®7 und S7-400 sind eingetragene Warenzeichen der Siemens AG.

1.2 Aufbau des Systems

TrySim besteht aus 2 großen Funktionseinheiten:
1. der Anlage
2. der SPS

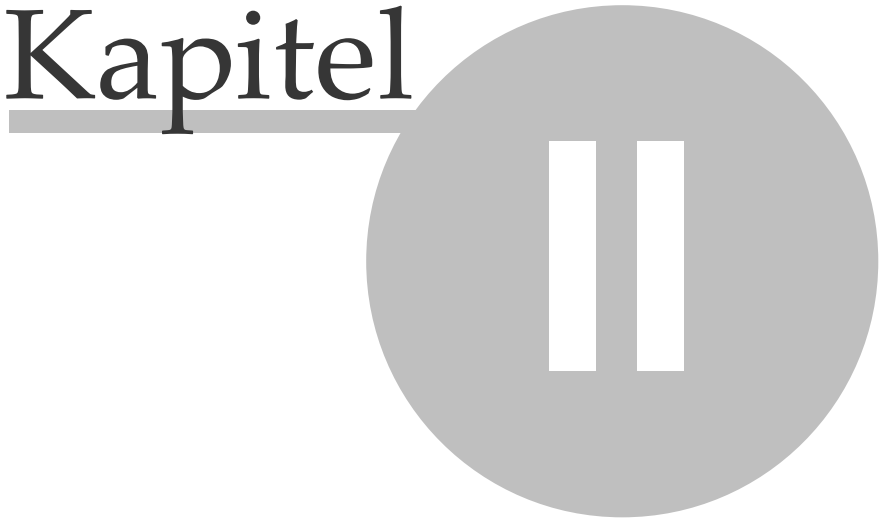
Mit dem Anlageeditor bauen Sie sich Ihre Maschine, mit dem Programmeditor schreiben Sie das dazugehörige SPS-Programm. Die Verbindung zwischen Maschine und SPS wird durch die Ein- und Ausgänge der SPS geschaffen, die der leichteren Lesbarkeit halber über die Symboltabelle mit Namen versehen werden können.



Nach dem Starten der Simulation werden die Anlage und das Programm zyklisch bearbeitet. Ein Zyklus besteht aus folgenden Schritten:

1. Alle Aktoren der Anlage lesen den Zustand der ihnen zugeordneten Ausgänge der SPS und verändern daraufhin ihre Position.
2. Alle Sensoren der Anlage überprüfen ihre Betätigung und setzen die entsprechenden Eingänge der SPS.
3. Das Programm wird mit den neuen Eingangsdaten einmal bearbeitet und die Ausgänge werden, wie von Ihnen programmiert, gesetzt.

Anlage



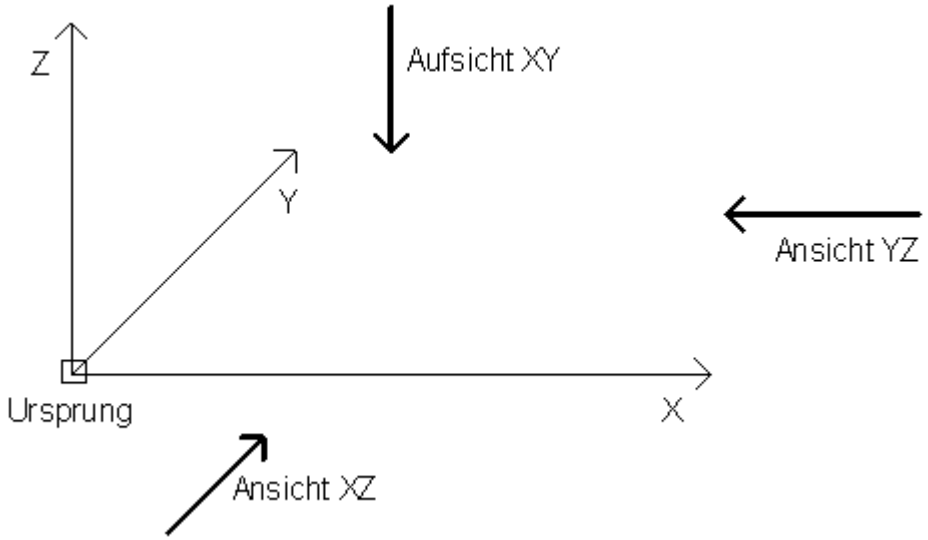
2 Anlage

2.1 Koordinatensystem

Die drei Raumkoordinaten werden mit x, y und z bezeichnet. Die x- und y-Richtungen liegen auf dem Boden, die z-Richtung zeigt nach oben. Sie können sich die Anlage aus allen drei Raumrichtungen anzeigen lassen, auch gleichzeitig aus mehreren Richtungen. Dabei liegt der Nullpunkt immer unten links. Das ist für erfahrene Computerbenutzer etwas gewöhnungsbedürftig, denn normalerweise liegt der Nullpunkt des Bildschirms oben links. Eine Beibehaltung dieser Konvention hätte aber dazu geführt, dass die z-Werte von oben nach unten zunehmen, was wiederum sehr befremdlich wäre.

Die Maßeinheit, in der die Positionen der Elemente angegeben werden, können Sie zwischen 0.1 mm, cm und m wählen. Standardmäßig sind mm vorgegeben.

Die Positionen der Elemente werden immer bezüglich des Nullpunktes angegeben.



2.1.1 Ursprung

Dies ist ein Element, das Sie nie zu sehen bekommen. Es ist in jeder Anlage automatisch vorhanden und dient als Befestigungspunkt für alle Elemente, die Sie der Anlage hinzufügen. Der Ursprung hat die Koordinaten (0,0,0). In den Grafik-Fenstern ist er immer links unten.

2.2 Arbeiten mit dem Anlageneditor

2.2.1 Kurzanleitung

Beispiele anschauen: Projekt | Öffnen

Simulation starten / stoppen mit den Symbolen  und 

Elemente editieren durch Rechts-Klick (bei stehender Simulation)

Status Beobachten: Ein SPS-Fenster aktivieren und  (Auge) klicken

Beispielsitzung durchführen! Zu finden in Kapitel 8 im Handbuch.

Anlage in der 3D-Ansicht mit der Maus drehen und schieben, in der online-Hilfe sind alle Navigationsmöglichkeiten aufgeführt.

Programmstellen finden: Auf das Element mit rechts klicken, **QL** (**Q**uerverweis**l**iste) anklicken, gewünschte Programmstelle durch Doppelklick aktivieren.

Weitere Verwendungen eines Operanden im Programm finden: Operand im SPS-Programm markieren, auf die Statusleiste doppelklicken -> Querverweisliste erscheint, gewünschte Programmstelle durch Doppelklick aktivieren.

Nutzen Sie den Element-Baum: Anlage|Elementbaum

Bitte verwenden Sie auch die kontextsensitive Hilfe mit der Taste 

Bei Problemen können Sie unsere Hotline +49 (0)4961 / 91 63 93 anrufen

oder uns eine e-mail an info@cephalos.de senden.

2.2.2 Neue Elemente erzeugen

Wählen Sie **Anlage|Neues Element** (Abkürzung:  bei aktivem Grafik-Fenster). Es erscheint ein Auswahlfenster, in dem die Elemente in Registerkarten geordnet sind. Nach der Wahl des entsprechenden Registers ziehen Sie das gewünschte Element auf ein Grafikfenster. Wenn Sie das Element an einem anderen befestigen wollen, lassen Sie die Maustaste los, wenn der Mauszeiger dem zukünftigen Vater so nahe ist, dass dieser markiert wird. Danach können Sie das Element an die endgültige Position schieben.

Dann klicken Sie mit der rechten Maustaste auf das neue Element und editieren seine Eigenschaften.

Im Auswahlfenster steht auch ein Kontextmenü zur Verfügung, mit dem Sie entweder

die Symbole oder die Namen der Elemente abwählen können. Wenn Sie mit TrySim etwas vertraut sind, lässt sich das Auswahlfenster dann kleiner machen und steht nicht so sehr im Weg herum.

Nützlich bei der Erzeugung neuer Elemente ist das Kreuz, mit dem Sie den Wert der unsichtbaren Koordinate bestimmen können.

Sie können die Liste der angebotenen Elemente nicht selbst erweitern. Wenn Sie ein Element benötigen, das nicht vorhanden ist und auch nicht durch eine Kombination von anderen Elementen simuliert werden kann, dann sprechen Sie bitte mit uns. Sie erreichen uns unter oben genannter Nummer oder via e-mail an info@cephalos.de.

Dynamiks (das Material, welches die Anlage bearbeitet) werden während der Laufzeit durch einen Generator erzeugt.

2.2.3 Auswählen von Elementen

Um ein Element zu markieren, klicken Sie bei stehender Simulation mit der linken Maustaste darauf. Wenn Sie mehrere Elemente markieren wollen, halten Sie die Shift-Taste oder die Strg-Taste gedrückt. Sie können auch mehrere Elemente markieren, indem Sie mit der Maus einen Kasten um sie ziehen.

Sie können mehrere markierte Elemente mit der Maus verschieben.

Linienförmige Elemente lassen sich schlecht markieren, wenn sie exakt entlang der Blickrichtung ausgerichtet sind. Hier hilft nur einkreisen mit der Maus oder wählen Sie im Menüpunkt **Ansicht|Dimension** die gewünschte Blickrichtung.

Wenn Sie Schwierigkeiten haben, bestimmte Elemente in einem großen Haufen anderer Elemente zu markieren, wechseln Sie in eine andere Ansicht, dort ist es häufig einfacher. Hilfreich ist auch die Verwendung der Editierhilfe Streifen, um störende Elemente auszublenden. Falls auch dies nicht zum Erfolg führt, können Sie die Editiermaske des Elementes immer noch über Anlage|Elementbaum oder über die Adressentabelle erreichen.

Versuchen Sie auch **Ansicht|Stark vergrößern,** der danach angeklickte Bereich wird automatisch in die Mitte des Fensters gerückt.

Wenn Sie in einer großen Anlage häufig die gleichen Elemente markieren wollen, sollten Sie dazu Gruppen anlegen.

Während des Baus einer Anlage ist es in vielen Fällen leichter, Elemente zu markieren, wenn Sie vor dem Einfügen neuer Elemente das Kreuz setzen, damit die Elemente in allen Raumrichtungen bereits in der Nähe ihrer endgültigen Position erzeugt werden.

In der 3D-Ansicht können Sie keine Elemente markieren.

Im Fenster "QuickSelected" werden alle zur Zeit markierten Elemente aufgelistet.

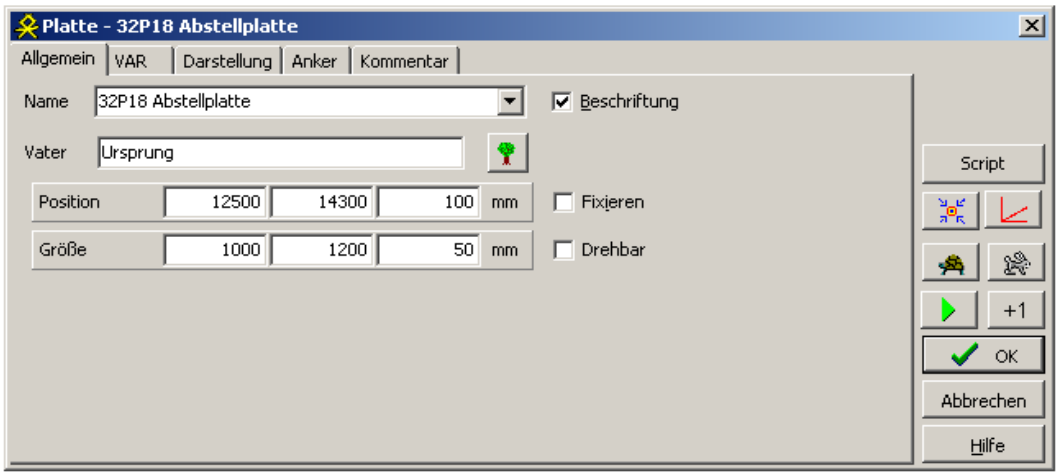
2.2.4 Editieren der Eigenschaften

Jedes Element verfügt über bestimmte Eigenschaften, die im Editiermodus geändert werden können. Dafür müssen Sie:

1.) Das zu editierende Element mit rechter Maustaste anklicken oder
2.) Im Menü Anlage|Elementbaum die gesamte Liste aller Elemente aufrufen, ein Element auswählen und auf den Knopf **Edit** drücken (Doppelklick tut es auch) oder
3.) Stellen Sie sicher, dass das richtige Element im Grafikeditor ausgewählt ist und drücken Sie die **Eingabetaste.**

Sie können auch die Eigenschaften mehrerer markierter Elemente gleichzeitig editieren. Dann werden aber nur die Editierfelder zur Verfügung gestellt, die bei allen markierten Elementen vorhanden sind.

Beispiel für eine Editiermaske:



OK-Button

Beim Drücken dieses Knopfs werden die Änderungen akzeptiert und das Eigenschaftenfenster geschlossen.

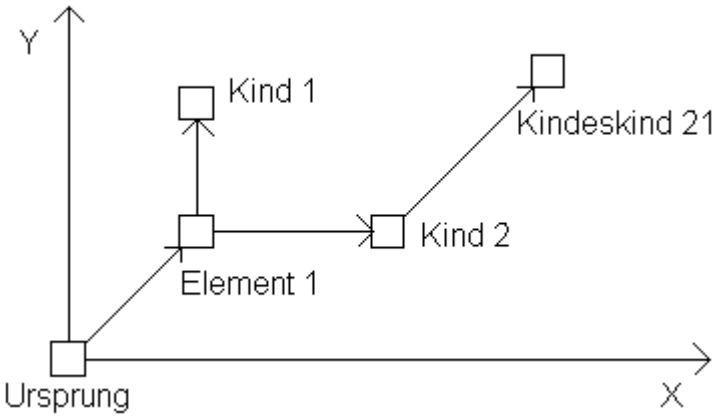
Abbrechen-Button

Beim Drücken dieses Knopfes werden die Änderungen verworfen und das Eigenschaftenfenster geschlossen.

2.2.5 Befestigen von Elementen

Wenn Sie eine wirkliche Anlage bauen, müssen Sie jedes Teil irgendwo befestigen. Genauso muss in TrySim jedes Teil befestigt werden, indem sein Vater festgelegt wird. Standardmäßig wird jedes Element am Ursprung befestigt, es bewegt sich dann nicht. Wenn Sie ein Element beweglich machen wollen, müssen Sie es an

einem bewegenden Element befestigen (z.B. Linearbeweger). Durch dieses Verfahren wird eine Baumstruktur aufgebaut, die im Elementbaum dargestellt werden kann. Die Festlegung eines Vaters ist auch zur **Strukturierung** großer Anlagen unerläßlich.



Im Bild ist der *Ursprung* der Vater von *Element 1*, das wiederum der Vater von *Kind 1* und *Kind 2* ist. *Kind 2* seinerseits ist der Vater von *Kindeskind 21*. Nehmen wir z.B. an, dass *Kind 2* ein Linearbeweger sei, dann kann sich *Kindeskind 21* relativ zum Ursprung bewegen.

Sie können für jedes Element festlegen, wo es am Vater befestigt werden soll (Anker). Dies ist nur relevant, wenn eine neue Anlage aufbauend auf eine bereits vorhandene oder mithilfe der Bibliothek aufgebaut werden soll. Wenn dabei die Größe der Elemente aus der Vorlage geändert wird, lässt sich durch die Festlegung des Ankers viel Arbeit sparen. Details hierzu unter Anker.

Mithilfe des Stickers können Sie auch während der Laufzeit den Befestigungspunkt von Elementen ändern oder Elemente zeitweilig an dem Material befestigen, das die Anlage bearbeitet.

2.2.6 Editierfenster "Statisches Element"

Dieses Editierfenster erhalten Sie, wenn Sie mehrere unterschiedliche Elemente markiert haben und dann Editieren gewählt haben. Wenn mehr als ein Element markiert ist, können Sie nur die Eigenschaften editieren, die bei allen markierten Elementen vorhanden sind und die gleiche Funktion haben. Z.B. können drehbare und nicht-drehbare Elemente nicht gemeinsam editiert werden, da bei den Drehbaren die Position die Mitte bezeichnet, während sie bei den Normalen die Ecke links unten bezeichnet.

Wenn Sie z.B. die Geschwindigkeit von mehreren Förderbändern gleichzeitig ändern wollen, dürfen in Ihrer Auswahl nur Förderbänder vorhanden sein. Um diese Auswahl zu erleichtern gibt es die Gruppen.

Auch wenn Sie mehrere Segmente einer Kette markiert haben, erscheint diese

Editiermaske. Markieren Sie nur ein Segment und versuchen Sie es erneut.

2.2.7 Löschen eines Elements

Es gibt drei Wege ein, Element zu löschen:

1. Sie wählen es in dem Grafikeditor mit der Maus aus und drücken **Entf.**
2. Sie klicken mit der rechten Maustaste so lange auf das Element, bis das Kontextmenü erscheint und wählen den Punkt “Löschen”.
3. Sie gehen zum Menüpunkt Anlage|Elementbaum, wählen das gewünschte Element aus der Liste aus und drücken auf die Schaltfläche **Löschen.**

Dynamiks können Sie auch während der Laufzeit mit einem Vernichter löschen.

2.2.8 Andere Ansicht wählen

Klicken Sie auf das Symbol  oder wählen Sie im Menüpunkt **Ansicht| Dimension** die gewünschte Blickrichtung.

Sie machen sich die Arbeit einfacher, wenn Sie nur bei wenigen Fenstern häufig die Blickrichtung ändern. Nutzen Sie lieber aus, dass Sie jedem der benennbaren Grafikfenster eine feste Blickrichtung zuordnen können und diese leicht mit den Short-Keys Alt+ 1..9 aufrufen können. Unter **Grafik|Eigenschaften** können Sie Blickrichtung eines Fensters auch einfrieren, sodass sie nicht mehr versehentlich geändert werden kann.

Sie können die Blickrichtung eines Fensters nicht mehr ändern, wenn Sie einen oder beide Scrollbars fixiert haben.

2.2.9 Farbtiefe

Wenn Sie die 3D-Darstellung als gläserne Objekte wählen, sollte die Farbtiefe mindestens 16-bit (high color) betragen.

So ändern Sie die Farbtiefe:
1. Minimieren Sie alle Fenster.
2. Klicken Sie irgendwo auf das Desktop mit rechts.
3. Wählen Sie aus dem Kontextmenü **Eigenschaften.**
4. Wählen Sie die Registerkarte **Einstellungen.**
5. Jetzt können Sie im Feld **Farben** die gewünschte Farbtiefe einstellen. Die Einstellungsmöglichkeiten hängen von Ihrer Grafikkarte, Ihrem Monitor und der aktuellen Auflösung ab.
6. In den meisten Fällen muss Windows neu gestartet werden, damit die neuen Einstellungen wirksam werden.

2.2.10 Auflösung

Die Monitorauflösung sollten Sie bei der Arbeit mit TrySim so hoch wie möglich einstellen, damit Sie auch tatsächlich Maschine und Programm gleichzeitig beobachten können. Wir empfehlen mindestens 1280x1024.

So ändern Sie die Auflösung:
1. Minimieren Sie alle Fenster.
2. Klicken Sie irgendwo auf das Desktop mit rechts.
3. Wählen Sie aus dem Kontextmenü **Eigenschaften.**
4. Wählen Sie die Registerkarte **Einstellungen.**
5. Jetzt können Sie mit dem Schieberegler **Bildschirmbereich** die gewünschte Auflösung einstellen. Die Einstellungsmöglichkeiten hängen von Ihrer Grafikkarte und Ihrem Monitor ab.
6. In den meisten Fällen muss Windows neu gestartet werden, damit die neuen Einstellungen wirksam werden.

2.2.11 3D-Ansicht

Zur besseren Veranschaulichung der Anlage kann diese echt 3-dimensional dargestellt werden. Wählen Sie **Grafik|3D-Ansicht**.

Ausschnitt und Blickrichtung können Sie mittels der Tastatur oder der Maus einstellen.

Wenn Sie die Anlage vollkommen aus den Augen verloren haben, können Sie unter **Ansicht|Ausgangsposition** einen Standard-Blickpunkt wiederherstellen.

Um häufiges Navigieren zwischen verschiedenen interessanten Ansichten zu vermeiden, können Sie Blickwinkel speichern und laden.

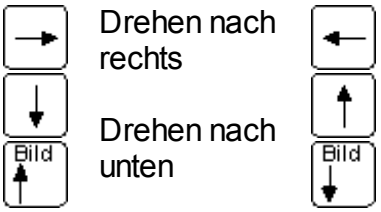
Unter **Ansicht|Linien/Gefüllt** können Sie wählen, ob die Elemente als Drahtmodelle oder als gläserne Objekte dargestellt werden sollen. Wenn Sie die Darstellung als gläserne Objekte wählen, sollte die Farbtiefe mindestens 16-bit (high color) betragen.

Während der Simulationslaufzeit sollten Sie das 3D-Fenster nicht zu groß machen, da sonst die Ausführungsgeschwindigkeit merklich gemindert wird. Die Liniendarstellung ist wesentlich schneller als die Gefüllte.

Auf den Editiermasken der Elemente können Sie einstellen, ob ein bestimmtes Element in der 3D-Ansicht dargestellt werden soll.

Um die 3D-Ansicht auszudrucken gibt es 2 Möglichkeiten:
1. Screenshot erstellen
2. wenn die 3D-Ansicht aktiv ist: Projekt|Drucken

3D-Tastaturbedienung



Drehen nach rechts

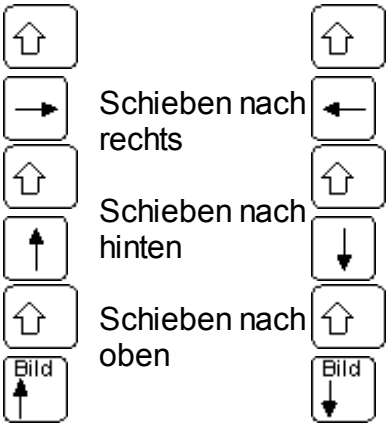
Drehen nach unten

Drehen nach links

Drehen nach oben

Drehen nach oben

Weg-Zoomen



Schieben nach links

Schieben nach rechts

Schieben nach vorne

Schieben nach hinten

Schieben nach unten

Schieben nach oben

3D-Mausbedienung

Wenn **keine** Maustaste gedrückt ist:
Mausrad vor / zurück : Weg-Zoomen / Heran-Zoomen

Wenn die **linke** Maustaste gedrückt ist:
Maus nach rechts / links : Drehen nach rechts / links
Maus nach oben / unten : Drehen nach oben / unten

Wenn die **rechte** Maustaste gedrückt ist:
Maus nach rechts / links : Verschieben nach rechts / links
Maus nach oben / unten : Verschieben nach hinten / vorne
Mausrad vor / zurück : Verschieben nach unten / oben

Wenn die  - Taste und die **linke** Maustaste gedrückt ist:
Maus nach oben / unten : Weg-Zoomen / Heran-Zoomen

Wenn die  - Taste und die **rechte** Maustaste gedrückt ist:
Maus nach oben / unten : Verschieben nach oben / unten

Mit **Ansicht|Ausgangsposition** bekommen Sie die Anlage immer wieder in Sicht.

Um einen Sie interessierenden Punkt der Anlage optimal anzupeilen, gehen Sie so vor:
- blicken Sie direkt von oben auf die Anlage, indem Sie die Maus mit links gedrückt zu sich heran ziehen
- verschieben Sie das Element mit der rechten Maustaste in die Mitte (dort wo das kleine Quadrat ab und zu blinkt)
- blicken sie horizontal über die Anlage, indem Sie die Maus mit links gedrückt von sich wegschieben
- verschieben Sie die Anlage mit rechts gedrückt und Rollrad nach oben/unten, bis das interessierende Element in der Mitte ist
- durch zoomen und Drehungen finden Sie jetzt die beste Ansicht

Sie können das in einer 2D-Ansicht markierte Element schnell in die Mitte bekommen, in dem Sie “In Bildmitte anzeigen” (“Focus”, Symbol ) wählen.

2.2.12 Grafik-Filter

Für jedes Element kann über die Registerkarte "Darstellung" auf den Editiermasken festgelegt werden, in welchem der 16 benennbaren Grafik-Fenster es angezeigt werden soll. Die Fenster können über **Grafik|Eigenschaften** so konfiguriert werden, dass sie von vorneherein nur Elemente eines bestimmten Typs aufnehmen.

Dadurch ist es z.B. möglich, ein Fenster “Pult” zu nennen und nur Bedienungselemente darauf anzeigen zu lassen oder bei einer mehrstöckige Anlage pro Etage ein Fenster zu reservieren.

Mit dem Button “Offene An” wird das Element in allen Fenstern angezeigt, die jetzt gerade geöffnet sind.

Sie können den Filter auch für mehrere markierte Elemente gleichzeitig editieren. Die Checkboxen der Fenster, die einige der Elemente anzeigen sollen und andere wiederum nicht, werden dann grau dargestellt.

2.2.13 Eigenschaften des 3D-Fensters

***

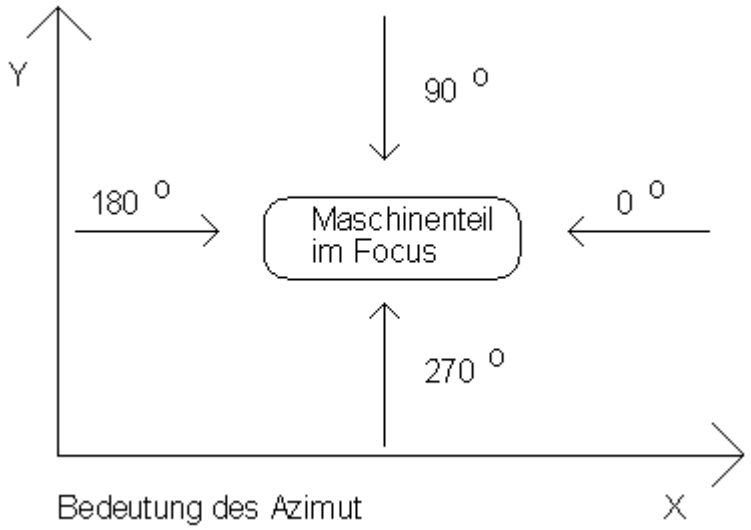
Auf dieser Maske wird die Position der 3D-Kamera beschrieben. Die Beschreibung erfolgt zweistufig und ist nicht ganz einfach zu verstehen. Nur dann, wenn Sie mit der Maus nicht die gewünschte Kameraposition einstellen können (dies ist manchmal schwierig, wenn Sie eine sehr große Anlage haben), sollten Sie sich mit dieser Beschreibung befassen. Versuchen Sie vorher auch den Menüpunkt (bei aktiviertem 3D-Fenster) **Ansicht|Ausgangsposition**. Der Attraktor bietet eine einfache Möglichkeit, einen bestimmten Teil der Maschine im 3D-Fenster darzustellen.

Focus: Hiermit geben Sie an, wohin die Kamera schauen soll. Ein Attraktor macht

nicht mehr, als diesen Punkt auf seine eigene Position zu setzen.

Abstand und Position der Kamera : Wenn Sie den Focus richtig gesetzt haben, müssen Sie noch den Abstand der Kamera von diesem Punkt festlegen. Danach sollte der Sie interessierende Anlagenteil zumindest erkennbar im 3D-Fenster sichtbar sein. Die Einstellung der Richtung, aus der der Anlagenteil betrachtet werden soll, ist dann am einfachsten mit der Maus (linke Taste gedrückt halten) einzustellen.

Die Position der Kamera wird neben dem Abstand vom Focus durch zwei Winkel angegeben (die, wie gesagt einfacher mit der Maus vorzugeben sind):
1.) Elongation : Hiermit wird angegeben, wie weit die Kamera von oben schaut. Ist die Elongation 0 Grad, so schaut die Kamera genau waagerecht auf den Focus. Wäre die Elongation 90 Grad (es können aber max. 80 Grad eingestellt werden), so würde die Kamera genau von oben auf den Focus schauen. Dieser Wert kann mit der Maus durch Bewegung nach oben oder unten bei gedrückter linker Taste verändert werden.
2.) Azimut : Hiermit wird angegeben, ob die Kamera aus Norden, Westen, Süden oder Osten auf den Focus schauen soll. Wenn die Kamera entgegen der X-Achse schaut, ist die Elongation 0. Schaut sie entgegen der Y-Achse, ist die Elongation 90 Grad und so geht es weiter: schaut die Kamera in Richtung der X-Achse, ist die Elongation 180 Grad und schaut sie in Richtung der Y-Achse, so ist die Elongation 270 Grad. Das folgende Bild veranschaulicht diese Zuordnung:



Dieser Wert kann mit der Maus durch Bewegung nach rechts oder links bei gedrückter linker Taste verändert werden.

2.2.14 3D-Blickwinkel speichern/laden

Zur Speicherung eines interessanten 3D-Blickwinkels wählen Sie bei aktivem 3D-Fenster den Menüpunkt **Ansicht|3D-Blickwinkel.** Klicken Sie sofort auf den Button “Neu”, Sie werden dann aufgefordert, einen Namen für den Blickwinkel einzugeben. Der Button ist nicht mehr anwählbar, wenn Sie bereits einen Blickwinkel aus der Liste gewählt haben.

Zum Laden eines gespeicherten Blickwinkels wählen Sie ihn durch Doppelklick aus der Liste oder markieren ihn und klicken auf “OK”. Sie können auch mit den Pfeil-Tasten durch die Liste wandern, die 3D-Ansicht wird dann sofort aktualisiert. Wenn Sie den gewünschten Blickwinkel gefunden haben, bestätigen Sie mit “Return”.

Es ist nicht möglich, einen einmal gespeicherten Blickwinkel nachträglich zu ändern. Die Fenstergröße und Position wird nicht abgespeichert.

Sie können die gespeicherten Blickwinkel auch von der SPS aus aktivieren. Laden Sie dazu die Nummer des Eintrages, den Sie aktivieren wollen, in den Akku 1 und rufen die FUNC 110 auf. Wenn Sie neue Blickwinkel abspeichern oder löschen, müssen Sie die Nummern im SPS-Programm anpassen. Sie sollten die FUNC 110 nicht zyklisch aufrufen, sondern nur, wenn wirklich ein neuer Blickwinkel angezeigt werden soll, denn sie ist sehr zeitaufwendig. Die FUNC 111 arbeitet genauso, zusätzlich jedoch muss im Akku 2 die Zeit (in 1/10 sec) stehen, die zum Wechsel von einer Ansicht zur nächsten benötigt werden soll. Diese Angabe bezieht sich auf die wirkliche Zeit, nicht auf die virtuelle Simulationszeit.

Sie können wechselnde oder sich bewegende Blickwinkel auch mittels Attraktoren vorgeben.

2.2.15 Vergrößern und verkleinern (Zoom)

Diese Funktion ist nicht bei der 3D-Ansicht verfügbar.

Klicken Sie auf die Lupen-Symbole  und  oder wählen Sie die gewünschte Operation aus dem Menü "Ansicht".

Je genauer der Sie interessierende Bereich beim Vergrößern in der Mitte des Fensters liegt, desto öfter können Sie auf  drücken, ohne dass der Bildausschnitt auswandert.

Der Menüpunkt **Ansicht|Stark vergrößern** eignet sich zum schnellen Heranzoomen eines bestimmten Bereiches. Nachdem Sie den Menüpunkt angewählt haben, wird der Mauszeiger zu einer Lupe, mit der Sie den zu vergrößernden Bereich anklicken. Mit **Ansicht|Stark verkleinern** machen Sie diese Vergrößerung wieder rückgängig.

Sie können den Zoom nicht mehr ändern, wenn Sie einen oder beide Scrollbars

fixiert haben.

2.2.16 Adressentabelle

Anlage|Adressentabelle

In der Adressentabelle sind alle von der Anlage verwendeten SPS-Adressen zusammengefasst. Bitte verwechseln Sie die Adressentabelle nicht mit der Symboltabelle, mit der Sie SPS-Adressen Symbole zuweisen können.

Sie können die Elementnamen und die Adressen ändern. Unter Extras|Optionen|SPS-Editor|Umverdrahten ist einstellbar, ob die Adressen im SPS-Programm automatisch geändert werden sollen.

Die Tabelle kann nach Elementen, Adressen und Element-Typen sortiert werden. Klicken Sie dazu auf die entsprechende Spaltenüberschrift. Wenn Sie links klicken, wird automatisch aufsteigend sortiert, wenn Sie rechts klicken, können Sie wählen, ob auf- oder absteigend sortiert werden soll.

Aus der Adressentabelle gelangen Sie direkt zur Editiermaske eines Elementes, wenn Sie aus dem Kontextmenü “Editieren” wählen.

Wenn Sie die Verwendung eines Operanden in der SPS überprüfen wollen, klicken Sie rechts und wählen aus dem Kontextmenü “Querverweise”. Die Querverweisliste wird dann an der richtigen Stelle aufgeschlagen und von dort kommen Sie mit einem Doppelklick zu der gewünschten Programmstelle.

Wenn Sie das Symbol oder den Kommentar zum Operanden editieren wollen, klicken Sie rechts und wählen aus dem Kontextmenü “Symboltabelle”.

Sie erreichen die Adressentabelle auch über den kleinen Button “Adr” auf jedem Interface-Panel.

Sie drucken die aktive Tabelle mit **Projekt|Drucken.**

In der Spalte für den Operanden können “E”, “A” und “M” sowie der Punkt auch über den Nummern-Block eingegeben werden.

2.2.17 Elementbaum

Sie erreichen diese Darstellung aller in der Anlage vorhandenen Elemente über **Anlage|Elementbaum** oder über das Baumsymbol auf jeder Editiermaske. Der Elementbaum ist sehr nützlich für die Arbeit mit großen Anlagen, Sie sollten ihn also häufig verwenden.

Im Elementbaum werden die Vater-Kind-Beziehungen dargestellt. Sie können daher schnell erkennen, woran ein bestimmtes Element befestigt ist. Wenn Sie ein Element mit der linken Maustaste packen und auf ein anderes ziehen, wird es an

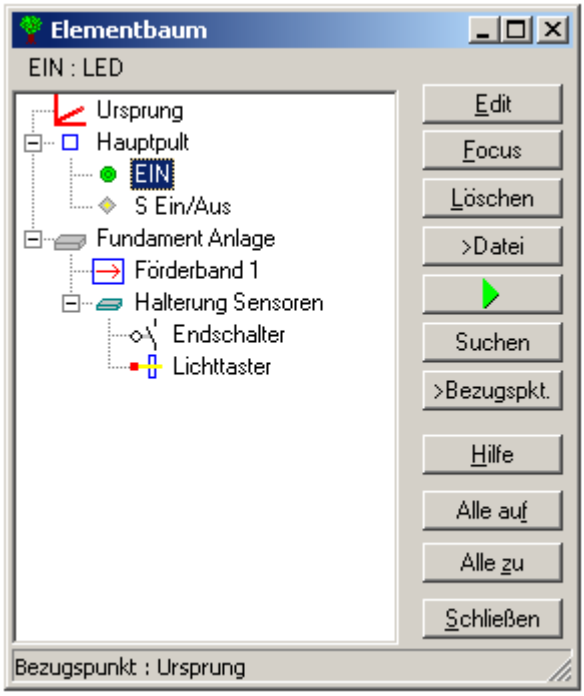
diesem befestigt, d.h., es bekommt einen neuen Vater.

Im Elementbaum können Elemente auch umbenannt werden. Klicken Sie dazu auf den Namen eines bereits markierten Elementes oder betätigen Sie die Funktionstaste F2.

Mit dem Button “Focus” wird das aktuelle Grafik-Fenster so verschoben, dass das markierte Element in der Mitte angezeigt wird. Diese Funktion steht auch unter **Bearbeiten|In Bildmitte anzeigen** zur Verfügung und sie funktioniert auch für die 3D-Grafik.

Der Elementbaum kann nicht ausgedruckt werden (siehe jedoch: Screenshot), aber er kann als Datei abgespeichert werden “>Datei”.

Mit dem Button “>Bezugspunkt” legen Sie den Beginn des Koordinatensystems auf das aktuelle Element. Dieses hat dann die Position (0,0,0) und alle anderen Positionen werden darauf bezogen. Bei großen Anlage erspart dies eine Menge Kopfrechnen. Durch einen Klick auf den Ursprung wird dieser wieder zum Bezugspunkt.



Anmerkung: Im Elementbaum ist nicht zu erkennen, dass alle Elemente den Ursprung als (Groß-) Vater haben. Dies ist kein Versehen, sondern wegen der Platzersparnis so von uns gemacht. Wenn wir den Ursprung richtig dargestellt hätten, wäre eine ganze Baum-Ebene vergeudet gewesen, was bei dem immer knappen Bildschirm nun wirklich nicht sein muss.

2.2.18 Gruppen

Gruppen sind nützlich zur Verwaltung größerer Anlagen. Sie können eine beliebige Auswahl von Elementen zu einer benannten Gruppe zusammenfassen. Alle Elemente einer Gruppe können dann schnell markiert und/oder editiert werden. Die Mitgliedschaft in einer Gruppe hat für ein Element keinerlei funktionelle Bedeutung - eine Gruppe besteht einfach aus einer Auflistung aller Elemente, die Mitglied sind, mehr nicht.

Typische Anwendungen sind:

1. Beim Bau einer mehrstöckigen Anlage verdecken die oberen Stockwerke die unteren. Für jedes Stockwerk wird eine Gruppe gebildet, dann lassen sich leicht alle Stockwerke mit Ausnahme des gerade Bearbeiteten unsichtbar schalten. Für diesen Zweck eignen sich auch die Streifen oder die Grafik-Filter.

2. Bei einer Anlage mit vielen gleichartigen Antrieben soll die minimal notwendige Geschwindigkeit ermittelt werden. Alle Antriebe werden zu einer Gruppe zusammengefasst, dann lassen sich leicht verschiedene Geschwindigkeiten einstellen, um die Wirkung auf den Durchsatz zu ermitteln.

Für viele Einsatzbereiche von Gruppen wie 2. ist es erforderlich, dass die Mitglieder möglichst gleichartig sind. Ein einziger Leuchtmelder in einer Gruppe von Förderbändern macht das Editieren von deren Antrieben unmöglich, weil er nicht über einen solchen verfügt.

Die Verwaltung der Gruppen geschieht zweistufig:
1. Im Gruppen-Manager können Gruppen erzeugt bzw. gelöscht werden, hier sind auch die Bedienelemente, um alle Elemente einer Gruppe gleichzeitig zu selektieren und/oder zu editieren.
2. Im Gruppen-Editor wird festgelegt, welche Elemente eine Gruppe enthalten soll.

Gruppen können nicht ausgedruckt werden.

Gruppen-Manager

Sie erreichen den Gruppen-Manager über **Anlage|Gruppen**.

Elemente | Select:
Wählen Sie eine Gruppe aus der Liste. Wenn Sie jetzt den Button “Select” betätigen, werden alle Elemente der Gruppe in den Grafik-Fenstern markiert und der Gruppenmanager wird geschlossen.

Elemente | Edit:
Wählen Sie eine Gruppe aus der Liste. Wenn Sie jetzt den Button “Edit” betätigen, wird eine Elemente - Editiermaske aufgerufen, auf der nur solche Eigenschaften editiert werden können, die in allen Elementen der Gruppe vorhanden sind. Je gleichartiger die Elemente einer Gruppe sind, desto mehr Eigenschaften können Sie verändern. Wenn Sie z.B. die Geschwindigkeit von Antrieben ändern wollen, dürfen in der entsprechenden Gruppe nur Elemente vorhanden sein, die auch

tatsächlich über einen Antrieb verfügen.

Gruppe | Neu:
Nach Eingabe des Namens der neuen Gruppe wird der Gruppen-Editor aufgerufen. Wenn in den Grafikfenstern Elemente markiert waren, werden diese automatisch in die neue Gruppe aufgenommen.

Gruppe | Edit:
Hiermit wird der Gruppen-Editor aufgerufen, um Elemente zu der Gruppe hinzuzufügen oder aus ihr zu entfernen.

Gruppe | Löschen:
Die Gruppe wird ohne Rückfrage gelöscht. Die in der Gruppe enthaltenen Elemente werden selbstverständlich nicht gelöscht.

Gruppen-Editor

Sie erreichen den Gruppen-Editor indirekt über **Anlage|Gruppen | unter der Überschrift Gruppe auf Edit** klicken.

Button ">":
Das im Elementbaum auf der linken Seite markierte Element wird zur Gruppe hinzugefügt.
Entsprechende Tastaturbedienung: ENTER

Button ">>":
Das im Elementbaum auf der linken Seite markierte Element und alle seine Kinder und Kindeskinder werden zur Gruppe hinzugefügt. Wenn der Ursprung markiert ist, werden **alle** Elemente außer dem Ursprung selbst der Gruppe hinzugefügt.
Entsprechende Tastaturbedienung: Shift ENTER

Button "Markierte >":
Alle in Grafikfenstern markierte Elemente werden zur Gruppe hinzugefügt.

Button "<":
Alle auf der rechten bzw. linken Seite markierten Elemente werden aus der Gruppe entfernt. Auf der rechten Seite ist auch eine Mehrfachauswahl möglich.
Entsprechende Tastaturbedienung: ENTF .

Button "<<":
Alle auf der rechten bzw. linken Seite markierten Elemente werden mit ihren Kindern aus der Gruppe entfernt. Auf der rechten Seite ist auch eine Mehrfachauswahl möglich.
Entsprechende Tastaturbedienung: Shift ENTF.

Weitere Funktionen:

- Sowohl im linken als auch im rechten Fenster steht ein Kontextmenü zur Verfügung.

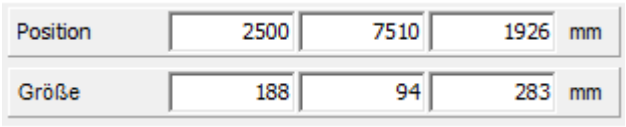
- Elemente können editiert werden, links nur über das Kontexmenü, rechts auch mittels ENTER oder Doppelklick.

SemiDynamiks können Sie nur einer Gruppe hinzufügen, indem Sie sie vor dem Aufruf des Gruppen-Editors in einem Grafik-Fenster markieren und dann den Butten “Markierte >>“ betätigen, denn sie sind im Elementbaum nicht aufgeführt.

2.2.19 Positionieren von Elementen

Die einfachste Art Elemente zu positionieren ist mit der Maus. Bei größeren Anlagen reicht das allerdings meist nur zur Grobpositionierung aus. Beachten Sie bei der Mauspositionierung bitte das Raster.

Genauer ist die Angabe der Koordinaten auf der Editiermaske.



Nützlich ist hierbei, dass der Bezugspunkt geändert werden kann. 

Die Verschiebung eines Elementes um einen definierten Betrag erreichen Sie dadurch, dass Sie den Verschiebeweg mit einem vorangestellten + in ein Positionsfeld eingeben:

Achtung!

Wenn Sie um einen negativen Wert verschieben wollen, müssen Sie z.B. “**+-**25” eingeben. Der Grund dafür ist, dass eine negative Zahl einen absoluten Wert bedeuten kann und von TrySim auch so interpretiert wird. Gerade bei veränderlichem Bezugspunkt kommt das sogar häufig vor. Nur anhand des + Zeichens (das man ja normalerweise nicht eingibt) erkennt TrySim, dass Sie eine Verschiebung meinen:

Verschiebung von mehreren Elementen

Bevor Sie mehrere Elemente verschieben, sollten Sie prüfen, ob es nicht günstiger ist, diese zunächst an einem gemeinsamen Vater zu befestigen. Dann brauchen Sie nur noch diesen zu verschieben. Mit dem Wachsen der Anlage werden Sie um diese Art der Strukturierung ohnehin nicht herumkommen.

2.2.20 Bezugspunkt

Jedes Element in TrySim hat eine Position, die durch drei Koordinaten (X, Y, Z) angegeben wird. Intern werden diese Werte in Mikrometer gespeichert und verarbeitet, zu sehen sind auf den Editiermasken meist nur mm.

Die Position wird intern immer mit dem Abstand zum “Ursprung” gemessen. Dieses Element, das Sie nur im Elementbaum sehen, hat per Definitionen die Koordinaten:
X = 0,000 mm
Y = 0,000 mm
Z = 0,000 mm

Gerade bei großen Anlagen werden die Zahlen für einen menschlichen Nutzer dann aber
schnell unübersichtlich. Wer sieht schon auf den ersten Blick, dass der Abstand zweier
Kästen mit den Positionen X=650021 und X=649001 recht genau 1 m ist?

Darum gibt es in TrySim die Möglichkeit, ständig wechselnd einen Bezugspunkt festzulegen. Diese Festlegung hat nichts mit der internen Speicherung zu tun und beeinflusst in keiner Weise die Simulation. Es werden nur alle angezeigten Daten umgerechnet, damit sie für Sie leichter lesbar sind.

Wenn Sie im obigen Beispiel den Kasten_1 als Bezugspunkt wählen, dann hat er die X-Position “0”. Das kann man sich leicht merken.
Für den Kasten_2 wird dann die X-Position “1020” angezeigt, also ungefähr 1 m.

Manchmal ist das Ende eines Elementes interessanter als der Anfang. Darum kann man mit einem Symbol in der Arbeitsleiste von “links unten” nach “rechts oben” umschalten. Das ist sehr nützlich, wenn man eine Positionierungsaufgabe der Art “Diese Lichtschranke sitzt 50 mm vor dem Ende des Förderbandes” zu lösen hat.

2.2.21 Anlagensimulation

Die Anlage bauen Sie mit dem Grafik-Editor. Sie wählen die benötigten Elemente aus einer Liste, plazieren Sie im Grafikfenster und editieren ihre Eigenschaften.

Dann starten Sie die Simulation mit dem Symbol  . Während die Simulation läuft (grüner Punkt unten links), können Sie nicht editieren. Das Stoppen der Anlage mit dem Symbol  ist gleichbedeutend mit dem Umschalten in den Editier-Modus. Sie können einen Simulationslauf beliebig oft zum Editieren stoppen, nach dem Start wird die Simulation dort wieder aufgenommen, wo sie unterbrochen wurde.

2.2.22 Finden von Elementen

Mit wachsender Größe der Anlage wird es immer schwieriger, ein bestimmtes Element schnell zu finden. Es gibt daher in TrySim eine Vielzahl von Mechanismen, Elemente zu markieren, ihre Editiermaske aufzurufen oder ihren SPS-Anschluss zu überprüfen. Das mag anfangs etwas verwirrend wirken, aber nach einer Weile werden Sie es zu schätzen wissen, dass Sie ein Element auch mit unvollständiger Information identifizieren können.

Wenn Sie nur wissen, an welcher Stelle im Programm das gesuchte Element verwendet wird, klicken Sie auf den Operanden mit rechts und wählen Adressentabelle. Über das Kontextmenü gelangen Sie direkt zur Editiermaske des Elementes. Beachten Sie, dass sich die Markierungen in den Grafik-Fenstern ändern, wenn Sie mit der Maus oder der Tastatur die Zeile in der Tabelle wechseln. Beachten Sie auch, dass Sie die Tabelle nach Elementname, Adresse und Elementtyp sortieren können.

Wenn Sie nur wissen, an welchem Ort sich das Element ungefähr befindet, dann vergrößern Sie diesen Bereich mittels **Ansicht|Stark vergrößern**. Ist das Element immer noch nicht zu identifizieren, weil es von vielen anderen verdeckt wird, dann verwenden Sie den Streifen, um alle anderen Bereiche auszublenden. Berücksichtigen Sie dabei auch, dass die Anzeige des gesuchten Elements mittels eines  Grafikfilters ausgeblendet sein könnte.

Wenn Sie nur wissen, woran Sie das Element befestigt haben oder welches andere Element an ihm befestigt ist, wählen Sie den Elementbaum. Bei Elementen ohne SPS-Anschluß, die zusätzlich in allen Richtungen unsichtbar gemacht worden sind und nicht in einer Gruppe enthalten sind, ist dies die einzige Möglichkeit, wieder Kontakt aufzunehmen.

Wenn Sie nur wissen, welcher Gruppe das Element angehört, öffnen Sie den Gruppeneditor und klicken Sie das Element an. Beachten Sie, dass das Element dann in den Grafik-Fenstern markiert wird und, wenn der Elementbaum offen ist, auch dort. Über das Kontextmenü gelangen Sie direkt zur Editiermaske des Elementes.

Elemente, die sich viel in der Anlage bewegen, die Sie aber dennoch mit starker Vergrößerung betrachten müssen, sollten Sie mit einem Attraktor versehen. So können Sie blitzschnell auf den entsprechenden Bereich fokussieren. Da die Attraktoren aktiviert und deaktiviert werden können, z.B. mittels eines Stufenschalters, können Sie auch zwischen verschiedenen Attraktoren umschalten.

2.2.23 Strukturieren großer Anlagen

Wenn Sie eine große Anlage erstellen, ist es unabdingbar, dass Sie diese gut strukturieren, die Arbeit damit wird sonst zur Qual. Das Prinzip der Strukturierung in TrySim ist das Gleiche, das Sie für die Verwaltung der Dateien auf der Festplatte Ihres Rechners anwenden: zusammengehörige Dateien werden in einem Ordner aufbewahrt und wenn darin zu viele Dateien enthalten sind, werden gegebenenfalls Unter-Ordner erstellt.

Ordner gibt es in TrySim nicht, stattdessen werden Kästen verwendet. Diese sind wie ein Fundament, an dem alle Elemente einer Funktionsgruppe befestigt werden. Auf der Editiermaske des Kastens können Sie auch “Fundament” anwählen, er wird dann hellgrau gefärbt (das können Sie später wieder ändern) und erscheint auch im Elementbaum gesondert.

Wenn zu viele Elemente an einem Fundament befestigt werden müssen, empfiehlt es sich, besonders wenn einige davon eigenständige Funktionsgruppen bilden, für diese Elemente einen weiteren, als Befestigung dienenden Kasten zu machen.

Bei Ihren ersten Anlagen werden Sie die Notwendigkeit der Strukturierung erst im Nachhinein erkennen. Gehen Sie in diesem Fall so vor: Erstellen Sie einen Kasten für die gewählte Funktionsgruppe, wählen Sie auf der Editiermaske "Fundament" (dadurch wird er hellgrau gefärbt) und positionieren Sie ihn am für die Funktionsgruppe logischen Bezugspunkt (nb.: schon das Nachdenken darüber, wo dieser ist, gibt einem häufig wertvolle Hilfen bei der späteren Inbetriebnahme). Wenn es sich bei der Funktionsgruppe um ein anreihbares Konstrukt handelt, z.B. Förderbandsegmente, ist es nützlich, das Fundament genauso groß wie die Anreihlänge zu machen - dann lassen sich weitere Segmente, die entweder durch "Kopieren und Einfügen" oder aus der Bibliothek erzeugt werden, bereits mit der Maus grob positionieren. Die Feinpositionierung wird dann durch die Eingabe der Koordinaten auf der Editiermaske des Fundamentes vorgenommen. In anderen Fällen ist es günstiger, das Fundament sehr klein zu machen. Es lohnt sich wirklich, einige Gedanken auf Position und Größe des "Fundamentes" zu verwenden. Für Fundamente, die diesen Namen nicht nur symbolisch tragen ist es z.B. absolut empfehlenswert, die Z-Koordinate immer auf "0" zu setzen.

Nachdem Sie das "Fundament" erzeugt und konfiguriert haben, befestigen Sie die dazugehörigen Elemente daran. Wenn Ihre (unstrukturierte) Anlage noch nicht zu groß ist, geht dies am einfachsten im Elementbaum: packen Sie das zu befestigende Element mit der linken Maustaste und schieben es auf das neue Fundament. Man spart sich Arbeit, wenn man zunächst die kleineren Funktionsgruppen zusammenfasst, da diese dann als ganzes der nächst Höheren zugeordnet werden können. Wenn Ihrer Anlage bereits so groß ist, dass der Element"baum" (in diesem Fall mehr eine Stange) nicht mehr auf einen Bildschirm passt (der TrySim-Baum scrollt z.Z. noch nicht automatisch), ist es einfacher, die zusammengehörigen Elemente in einem Grafikfenster zu markieren und dann ihren Vater zu ändern. Schauen Sie sich beim Erstellen großer Anlagen gelegentlich den Elementbaum an und handeln Sie, solange dies noch mit sehr wenig Aufwand möglich ist!

Die Vorteile einer gut strukturierten Anlage erfahren Sie spätestens, wenn Sie ein Element suchen oder ein Grafikfenster so konfigurieren wollen, dass es genau das zeigt, was Sie im Moment sehen wollen. Für Letzteres sind besonders die unter dem Menü Bearbeiten vorhandenen Funktionen hilfreich. Z.B. lässt die Funktion "Kinder ausblenden", angewendet auf ein "Fundament", alle dazugehörigen Elemente verschwinden. Das Fundament selbst bleibt aber noch sichtbar, kann daher leicht markiert werden und mit "Mit Kindern einblenden" wird die Funktionsgruppe als Ganzes wieder sichtbar.

Folgende Themen sind ebenfalls interessant, wenn es um das Editieren großer Anlagen geht:
Gruppen
Benennbare Grafikfester
Grafik-Filter

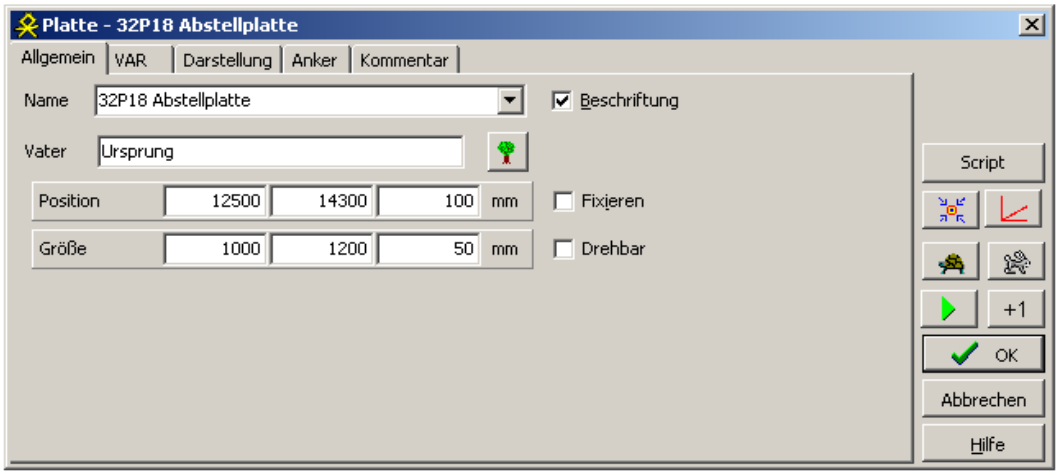
Fenster auf ein bewegtes Teile fokussieren
Bezugspunkt

2.3 Gemeinsame Eigenschaften von Elementen

2.3.1 Einführung

Fast alle Elemente haben die folgenden Eigenschaften:

Name
Vater
Position
Größe
Fixierbarkeit
Sichtbarkeit
Farbe
SPS-Anschluss
Kommentar
Befestigungspunkt (Anker)
Script



2.3.2 Name

Jedes Element hat einen Namen. Die Namen müssen nicht eindeutig sein. Ein neues Element bekommt vom System zunächst einen Namen, der aus dem Element-Typ und der ID gebildet wird. Der Name eines Elementes wird in der Grafik neben das Element geschrieben. Sie können die Position des Namen ändern, indem Sie mit der linken Maustaste auf den Namen klicken und die Maustaste festhalten. Dann schieben Sie den Namen an die gewünschte Position und lassen die Taste los.

Da bei Anlagen mit vielen Elementen die Namen oft mehr stören als nützen, können Sie die Anzeige des Namens durch Abwählen der Checkbox “Bezeichnung” in der Editiermaske unterdrücken. Der Name des Elementes ist dann in dem Hinweis

wiederzufinden, der erscheint, wenn Sie im Editiermodus mit der Maus auf das Element zeigen und ca. eine Sekunde warten. Auch wenn Sie den Namen nicht anzeigen lassen wollen, sollten Sie jedem Element einen klaren und eindeutigen Namen geben, damit Sie das Element im Elementbaum, in der Adressentabelle oder bei der Bildung von Gruppen leicht erkennen.

Taster und Leuchtmelder haben auch noch einen Namen2. Dieser Name (wenn Sie denn einen eingegeben haben) wird auf den Grafikfenstern angezeigt. Im Elementbaum und in der Adressentabelle wird aber weiterhin der normale Name angezeigt. Dies hat sich als nützlich für die Erstellung von Pulten erwiesen, auf denen man einen leicht verständlichen Text sehen möchte, während der eigentliche Name mit dem Betriebsmittelkennzeichen belegt werden kann.

2.3.3 Vater

Alle Elemente außer dem Ursprung haben einen Vater. Der Vater ist irgendein anderes Element, standardmäßig der Ursprung. Wenn der Vater bewegt wird, während des Editierens oder während der Laufzeit, wird das Element ebenfalls bewegt. Wenn der Vater gelöscht wird, wird das Element ebenfalls gelöscht.

Durch Zuordnen mehrerer Elemente zu einem gemeinsamen Vater lässt sich die Anlage strukturieren, was das Editieren besonders bei einer großen Anzahl von Elementen vereinfacht. Sie sollten unbedingt häufig im Elementbaum nachschauen und bei Unordnung dort aufräumen.

Den Vater eines Elementes können Sie auf folgende Arten festlegen:

1. Sie bestimmen ihn gleich bei der Erzeugung des Elementes, indem Sie das neue Element auf den Vater legen (den Mauszeiger so positionieren, dass der zukünftige Vater markiert wird).
2. Sie rufen Anlage|Elementbaum auf, packen das Element und ziehen es auf den neuen Vater. Diese Methode ist besonders geeignet, um Ordnung in eine Anlage zu schaffen, in der zu viele Elemente noch den Ursprung als Vater haben
3. Klicken Sie etwas länger mit der rechten Maustaste auf das Element. Aus dem dann erscheinenden Kontextmenü wählen Sie “Vater ändern”. Klicken Sie sodann mit der linken Maustaste auf den neuen Vater. Achten Sie darauf, erst zu klicken, wenn das Wort “Vater” am Mauszeiger unterstrichen wird.

Seit Version 3.0 können Sie den Vater eines Elementes nicht mehr ändern, indem Sie in der Editiermaske einen Vater aus der Liste auswählen. Das hat sich bei großen Anlagen als ein schwieriges und zu Fehlbedienung verleitendes Verfahren herausgestellt.

Wenn Sie mehrere Elemente zum Editieren ausgewählt haben und den Vater ändern, werden nur die Väter der Elemente geändert, deren alter Vater nicht in der Auswahl enthalten ist. Hierdurch wird vermieden, dass eine bereits aufgebaute Struktur von Vater-Kind-Beziehungen zerstört wird.

Wenn Sie ein neues Element erzeugen und es mit der Maus so nah an ein bereits vorhandenes Element plazieren, bis dieses die schwarzen Markierungsquadrate zeigt, wird es automatisch als Vater des neuen Elementes gewählt. Wenn Sie z.B. neue Bedienelemente zu einem als “Pult” bezeichneten Kasten hinzufügen, sollten Sie das Element kurz in der Nähe des Randes des Kastens absetzen (nur dort sind Kästen markierbar) und erst dann auf seine endgültige Position schieben, so ersparen Sie sich die nachträgliche Festlegung des Vaters.

Väter dienen jedoch nicht nur dazu, die Kinder zu bewegen, sondern auch dazu, die Anlage logisch zu strukturieren. Weisen Sie möglichst alle Elemente, die zu einem Anlagenteil gehören, einem gemeinsamen Vater zu. Es vereinfacht das Editieren und erleichtert das Finden von Elementen im Elementbaum.

Kind

Kind ist ein Element, das bei jeder Bewegung des Hotspots seines Vaters auch bewegt wird. Beachten Sie: das Löschen des Vaters bedeutet auch das Löschen der Kinder.

Wenn Sie wissen möchten, welche Kinder ein Element hat, betrachten Sie am besten den Baum, den Sie über den Menüpunkt Anlage|Elementbaum erreichen.

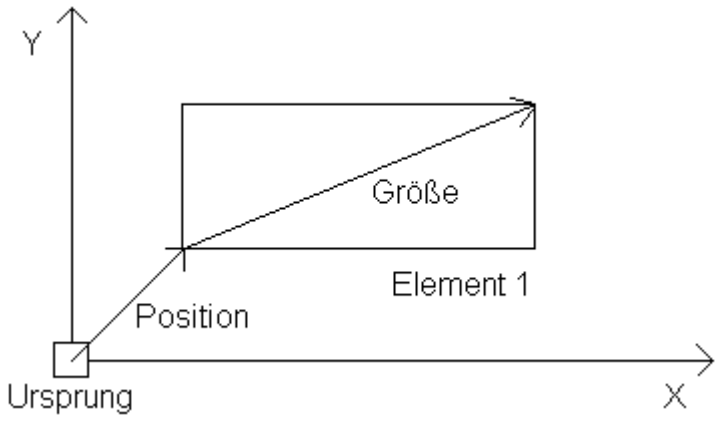
2.3.4 Position

Alle quaderförmigen Elemente haben eine Position (Ausnahmen sind die Linienförmigen wie Linearbeweger und Lichtschranke). Die Position bezeichnet die dem Ursprung am nächsten liegende Ecke. Sie wird normalerweise in absoluten Koordinaten bezüglich des Ursprungs angegeben, dies können Sie jedoch mit dem Kreuz ändern.

Sie können die Position eines Elementes im Editiermodus durch Eingabe der neuen Koordinaten in der Editiermaske ändern, oder indem Sie das Element mit der linken Maustaste packen und an den gewünschten Ort schieben. Wenn das Element Kinder hat, folgen diese der Bewegung.

Sie können auch mehrere bereits markierte Elemente gleichzeitig mit der Maus verschieben. Es darf dann jedoch kein einziges fixiertes Element in der Selektion sein, dessen Vater nicht ebenfalls in der Selektion ist.

Die Position wird auch implizit durch Änderung der Größe mit der Maus geändert, wenn die zur Größenänderung ausgewählte Ecke die Positionsecke ist, oder direkt mit dieser verbunden ist.



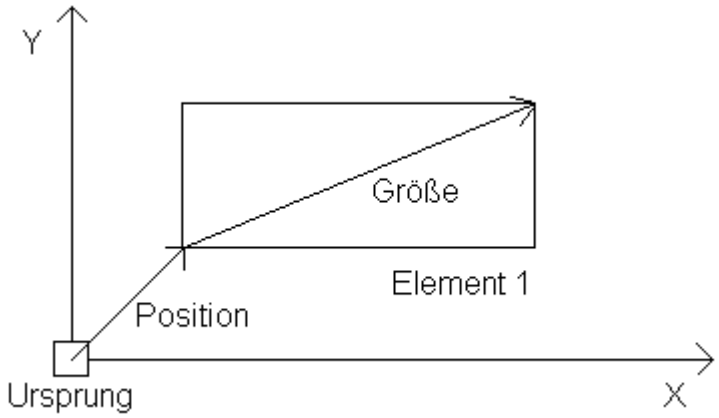
Die Darstellung im Bild ist der Klarheit halber nur zweidimensional.

2.3.5 Größe

Alle quaderförmigen Elemente haben eine Größe. Die Größe ist der Vektor von der als Position festgelegten Ecke zu der diagonal gegenüberliegenden Ecke.

Sie ändern die Größe entweder durch Eingabe der Koordinaten in der Editiermaske oder mit der Maus. Zur Änderung mit der Maus müssen Sie das Element zunächst durch Klicken mit der linken Taste markieren und dann den Mauszeiger so nahe eines der schwarzen Markierungsquadrate positionieren, dass sich der Mauszeiger in einen Doppelpfeil verwandelt. Dann packen Sie die Ecke und ziehen das Element auf die gewünschte Größe.

Bei der Änderung der Größe mit der Maus kann sich Position des Elementes ändern.



Die Darstellung im Bild ist der Klarheit halber nur zweidimensional.

Sie können die Größe von statischen Elementen auch vom SPS-Programm aus mittels eines POKEs ändern.

2.3.6 Fixierbarkeit

Wenn Sie die Position eines Elementes endgültig festgelegt haben, können Sie durch Anwählen der Checkbox “Fixieren” das Element davor schützen, mit der Maus verschoben zu werden. Diese Fixierung gilt nur für die unmittelbare Verschiebung des Elements. Wenn Sie den (nicht fixierten) Vater verschieben, folgt das Element selbstverständlich. Auch bleibt es Ihnen unbenommen, die Position des Elementes durch Eingabe neuer Koordinaten zu ändern.

Sie sollten Elemente möglichst frühzeitig fixieren, denn es geschieht sonst leicht, dass ein Element versehentlich verschoben wird. Daher sind im Kontextmenü jedes Elementes die Punkte “Kinder fixieren” (der eigentlich “Mit Kindern und Kindeskindern fixieren” heißen müsste) und “Kinder lösen”. Durch “Kinder fixieren” werden das Element selbst sowie alle daran befestigten Elemente fixiert. Durch “Kinder lösen” wird nur die Fixierung der unmittelbar an dem Element befestigten Kinder aufgehoben. Das Element selbst und die Kindeskinder bleiben fixiert. Diese Asymmetrie ist Absicht von uns und kein Versehen.

Die Fixierung betrifft nicht die Größenveränderung mit der Maus, da dies kaum versehentlich möglich ist. Aus dem gleichen Grund können fixierte Elemente dennoch via Pfeiltasten verschoben werden (seit Version 4.0).

Beachten Sie, dass Sie auch den Ausschnitt eines Fensters fixieren können, dies hat mit der Fixierung einzelner Elemente jedoch nichts zu tun.

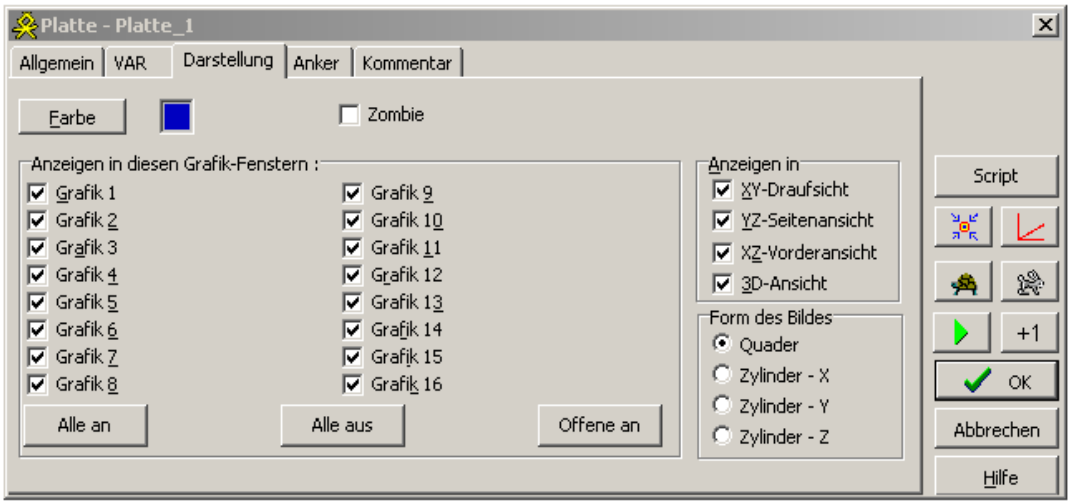
2.3.7 Sichtbarkeit

Zur übersichtlichen Darstellung einer dreidimensionalen Anlage können Sie für jedes Element festlegen, aus welcher Richtung es sichtbar ist. Standardmäßig sind die Elemente aus allen Raumrichtungen sichtbar. Sie können die Sichtbarkeit durch Abwählen der Checkboxen “Anzeigen in XY, ZY, XZ und 3D-Ansicht” abschalten.

Außerdem kann für jedes Element über den  Grafik-Filter festgelegt werden, in welchem der 16 benennbaren Grafik-Fenster es angezeigt werden soll. Die Fenster können über **Grafik|Eigenschaften** so konfiguriert werden, dass sie von vorneherein nur Elemente eines bestimmten Typs aufnehmen.

Mit der Editierhilfe Streifen können Sie während des Editierens vorübergehend bestimmen, welche Teile Ihrer Anlage Sie sehen möchten.

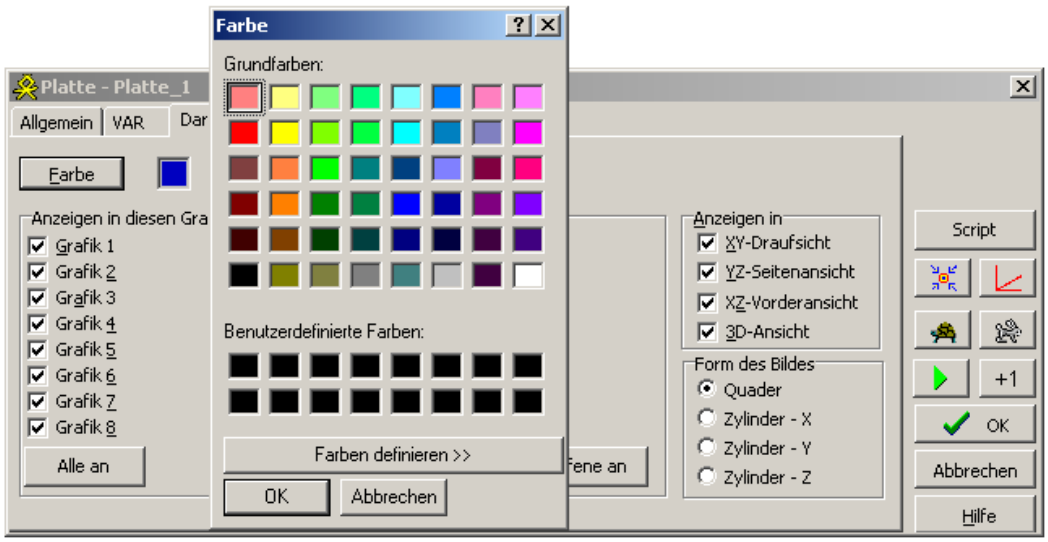
Beachten Sie in diesem Zusammenhang auch die Möglichkeit benannte Gruppen zu bilden.



2.3.8 Farbe

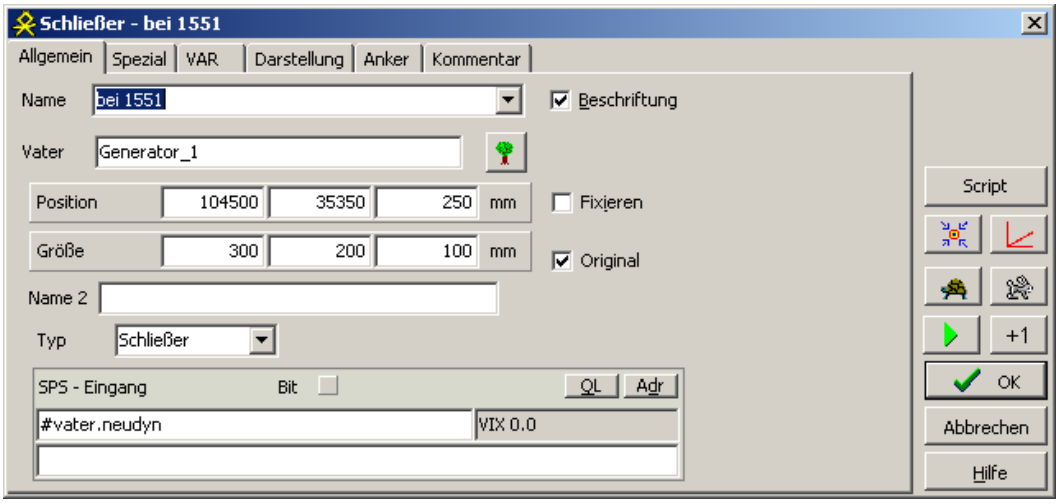
Die Darstellungsfarbe jedes Elementes können Sie frei wählen. Standardmäßig erhält jedes Element eine von seinem Typ abhängige Farbe. Die Farbe “Weiß” können Sie nicht wählen.

Bei den Leuchtmeldern ist die Farbauswahl auf die Grundfarben beschränkt.



2.3.9 SPS-Anschluss

* _ ***



Die meisten Elemente in TrySim müssen Sie an ein Bit oder mehrere Bits der SPS anschließen. Dazu gibt es auf der Editiermaske ein Interface-Panel in das Sie den Operanden eintragen. Wenn Sie eine Symboltabelle angelegt haben, können Sie stattdessen auch das Symbol angeben. Sie können die Elemente an jedes Bit aus dem Bereich Eingänge, Ausgänge, Merker und Datenbits anschließen.

Ob das Bit gelesen oder beschrieben wird, hängt allein von der Funktion des Elementes ab. Lesende Elemente, wie z.B. Leuchtmelder, fragen den Zustand des Bits nach Ende des SPS-Zyklus ab. Schreibende Elemente, wie z.B. Schalter, setzen den Zustand des Bits vor Beginn des SPS-Zyklus.

Wenn Sie eine Symboltabelle angelegt haben, wird auch der Kommentar zum Symbol in der Editiermaske angezeigt. Sie können den Kommentar dort ändern, die Änderung wird in die Symboltabelle übernommen. Das Symbol wird zwar angezeigt, aus Konsistenzgründen können Sie es jedoch nicht ändern, falls Sie dies wünschen, müssen Sie es in der Symboltabelle machen.

Sie können sich alle Verbindungen von der Anlage zur SPS übersichtlich in der Adressentabelle anzeigen lassen.

Wenn Sie die Verwendung eines SPS-Anschlusses im SPS-Programm überprüfen wollen, klicken Sie auf das Symbol QL und gelangen damit zur Querverweisliste.

2.3.10 Kommentar

Zu jedem Element können Sie einen Kommentar eingeben. Notieren Sie hier alle Besonderheiten, die bei der Installation oder bei der Erstellung der Anwender-Dokumentation berücksichtigt werden müssen.

Um den Kommentar aufzurufen, klicken Sie auf die Registerkarte "Kommentar" auf der Editiermaske des Elements. Wenn Sie mehrere Elemente markiert haben, kann der Kommentar nicht editiert werden.

Über Quick Kom | Script kann der Element-Kommentar ausgedruckt werden.

2.3.11 Anker

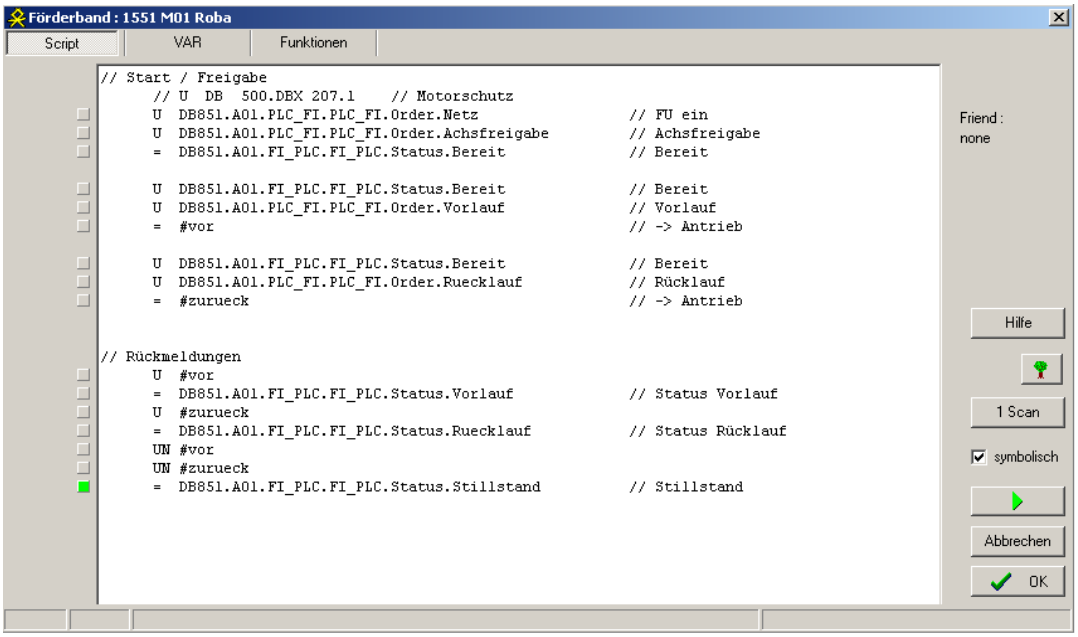
Die Einstellbarkeit des Befestigungspunktes ist bei erstmaligem Aufbau einer Anlage nicht wesentlich und hat keinen Einfluss auf die Simulation.

Wenn man die Größe eines Elementes ändern will, kann es sehr nützlich sein, für die Kinder festzulegen, an welchem Punkt genau sie befestigt sind. Wenn z.B. eine Lichtschranke am rechten Ende eines Förderbandes befestigt ist (also nicht an der Positionsecke), ist es lästig, wenn man nach einer Größenänderung des Bandes die Lichtschranke nachführen muss.

Für jede der Raumrichtungen können drei Werte festgelegt werden:
X : links (Standard), mitte, rechts
Y : vorne (Standard), mitte, hinten
Z : unten (Standard), mitte, oben

Über das Hauptmenü Extras|Quick Anker können Sie sich schnell den Befestigungspunkt anschauen. Änderungen müssen mit dem Button “!” bestätigt werden !

2.3.12 Script



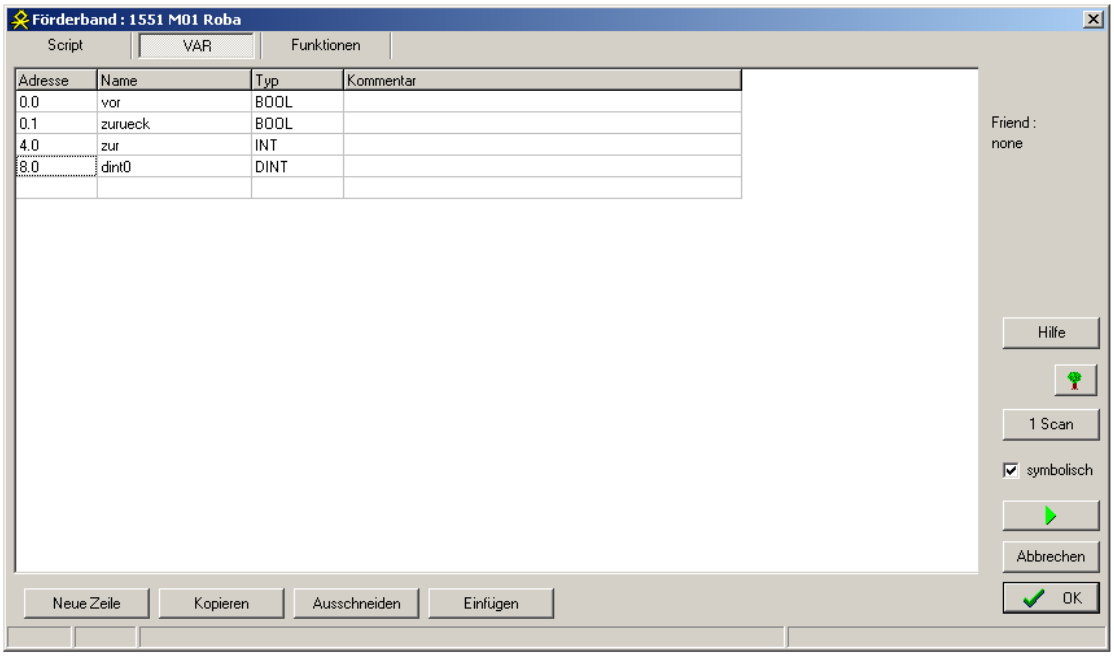
In dem Script kann für jedes Element ein kurzes Stück AWL-Code geschrieben werden, um die Eigenschaften zu modifizieren oder außerhalb der SPS befindliche Hardware nach zu bilden. Als Beispiel sei ein Förderband genommen, das nicht laufen kann, wenn der Motorschutzschalter gefallen ist.

Im Script würde dann stehen:
U “Q Motor” // Motorschutzschalter (Eingang)
U “A Motor” // Ausgang Motor start
= “#bool0”

An der Stelle in der Antriebsmaske, an der vorher nur “A Motor” stand, steht jetzt #bool0.

Das #bool0 ist ein Bit aus dem jedem Element einzeln zugeordneten Instanzdatenbaustein. Alle statischen Variablen dieses IDB sind nur innerhalb des Elementes bekannt und behalten von einem Simulationsschritt zum nächsten ihren Wert, es sei denn, sie stehen in einem Interface-Panel eines schreibenden Elementes.

Die Deklaration der statischen Variablen kann mit dem Button **VAR** angezeigt und editiert werden:



Sie können dort neue Variablen deklarieren (und auch den Namen der vordefinierten #bool0, #int0 und #dint0 ändern). Es sind die Datentypen BOOL, BYTE, INT, WORD, DINT, DWORD und REAL erlaubt. Es sind alle Adressen bis 20.0 erlaubt. Die Adressvergabe wird von TrySim nicht so streng überwacht wie bei Fbs, es können sogar überschneidende Adressen eingegeben werden. Sich überschneidende Adressen können nützlich sein, wenn z.B. auf Words auch bitweise zugegriffen werden soll. Die freie Adressvergabe erfordert allerdings auch mehr Aufmerksamkeit bei der Erstellung.

Am linken Rand werden die Werte der Operanden angezeigt. Bitte beachten Sie, dass dort nicht das **V**er**K**nüpfungs**E**rgebnis und nicht die Akku-Inhalte dargestellt sind. Durch einen Klick auf die LED oder die Zahl können sie diese ändern.

Der Nutzen dieser Scripts zeigt sich besonders bei der Wiederverwendung von Anlagen für das nächste Projekt. Fast alles, was sonst im OB2 und OB3 untergebracht war, ist jetzt direkt im Element und wird mit diesem übernommen. Außerdem ist transparenter geworden, was das Verhalten eines Elementes außerhalb des SPS-Programms zusätzlich beeinflusst.

Nützlich ist das Script auch, wenn man gar kein “richtiges” SPS-Programm schreiben möchte, sondern nur zu Demonstrationszwecken eine gesteuerte Simulation benötigt.

Auf die statischen Variablen des übergeordneten Elements kann z.B. mit #vater**.** bool0 zugegriffen werden (nur symbolisch). Dies hat sich als nützlich für die Kommunikation von Elementen untereinander erwiesen, da nun dafür keine absoluten Adressen (z.B. Merker oder Datenbausteine) mehr vergeben werden müssen.

Ein Zugriff auf eine freiere Weise (dann vermutlich über z.B. #friend.bool0) ist in

Vorbereitung. Ebenso wird mit der Syntax z.B. #element.position_x einfacher Zugriff auf interne Daten möglich sein, auf die bisher nur mit den Peek- und Poke-Elementen zugegriffen werden konnte.

Anregungen aus der Praxis zum Script sind uns sehr willkommen.

2.4 Statische Elemente der Simulation

2.4.1 Übersicht

Wenn wir Elemente “statisch” nennen, bedeutet dies nicht, dass sie sich nicht bewegen können, sondern dass sie während der Simulationslaufzeit weder erzeugt noch vernichtet werden können.

In der Hilfe zu den Elementen ist in der ersten Zeile markiert, wie schwierig dieses Element zu verwenden ist. Es bedeutet:
* : einfach, auch von Anfängern gut zu beherrschen
** : mittel
*** : schwierig, nur für erfahrene Anwender empfohlen
**** : sehr schwierig, diese Elemente befinden sich häufig noch im Test-Stadium.

Sensoren
Endschalter Lichtschranke Spion Barcode-Scanner Impulsgeber Ultraschall Thermometer Thermostat

Aktoren
Förderband Freibeweglicher Punkt Generator Linearbeweger Kette Haken Vernichter Gelenk Dreher Bogenförderband Drehbares Förderband Drehtisch Heizung Mangel Kontinuierlicher Generator

Bedienung
LED Digital-Anzeige Taster/Schalter Schieberegler Textanzeige String-Anzeige Oszillograph Stufenschalter MeisterSchalter

Flüssigkeiten
Allgemein Behälter Rohr Pumpe Ventil Flansch Niveau-Schalter Füllstandsmesser Durchflussmesser Drucksensor Rückschlagventil Analysator Reaktor Fluidor

Sonstige
Kasten Platte Stange Schieber Verschmelzer Teiler Säge Keil Presse Klinke Automatikhaken Sticker Dyn-Konverter Ausrichter Thermische Masse RFID-Chip&Antenne

Simulationshilfen (auch Registerkarte Sonstige)
Streifen Attraktor Kreuz Poke Peek Speed-Trigger Hintergrund

2.4.2 Sensoren

Lichtschranke

*

Die Lichtschranken in TrySim werden nur von dynamischen Elementen bedeckt. Da von diesen dynamischen Elementen sehr viele in der Anlage vorhanden sein können, ist die Überprüfung, ob die Lichtschranken von einem davon bedeckt wird sehr, sehr zeitintensiv. Standardmäßig sind die Lichtschranken daher nur in einem Punkt in der Mitte des Strahls empfindlich. Wenn Sie eine Empfindlichkeit über die ganz Länge benötigen, können Sie im Feld “Realitaetsgrad” die Anzahl der überwachten Punkte erhöhen.

Die Bezeichnungen “Endschalter” und “Lichtschranke” sollten in TrySim nicht wörtlich genommen werden. Verwenden Sie Endschalter immer dann, wenn Sie Teile der Anlage detektieren wollen und Lichtschranken, wenn Sie dynamische Elemente erfassen wollen.

Lichtschranken können Sie als Schließer oder Öffner konfigurieren. Wenn eine “1” an die SPS gemeldet wird, ist das kleine Quadrat an dem einen Ende rot. Sie können eine Lichtschranke auch als Schalter konfigurieren, dann ändert sie bei jeder Betätigung einmal den Zustand. Im Beispiel “Weiche” ist eine Lichtschranke so eingesetzt.

Der häufigste Fehler bei der Anwendung von Lichtschranken liegt darin, dass sie bei der Erzeugung zunächst auf dem Boden liegt (was man in der XY-Aufsicht ja nicht bemerkt) und die Dynamiks über die Lichtschranke hinweg fahren. Mit der Editierhilfe Kreuz kann dieser Fehler vermieden werden.

Um die Orientierung einer Lichtschranke mit der Maus zu ändern, müssen Sie mit der Maus so auf ein Ende zeigen, dass sich der Maus-Zeiger in ein Kreuz verwandelt.

Die Lichtschranke kann auch den von einem kontinuierlichen Generator erzeugten Strang detektieren.

Lichttaster

*

Dies ist eine Weiterentwicklung der Lichtschranke. Der Lichttaster wird allerdings von TrySim immer entlang der X-, Y- oder Z-Achse ausgerichtet. Er erfasst dann sicher alle Dynamiks, die seinen Strahl kreuzen.

Solange Ihre Sensoren nur rechts/links, vorne/hinten und oben/unten erfassen müssen, ist der Lichttaster einfacher anzuwenden. Wenn Sie allerdings unbedingt schräg gucken müssen, dann sollten Sie doch lieber eine Lichtschranke verwenden.

Der Lichttaster hat sich z.B. als hilfreich erwiesen, wenn sehr dünne Dynamiks erfasst werden müssen, weil die Lichtschranke standardmäßig nur an einem Punkt in der Mitte des Strahls empfindlich ist. Wenn diese dünnen Dynamiks (z.B. Platten)

sich dann außerhalb des einzig empfindlichen Punkt einer Lichtschranke befinden, werden sie auf einmal nicht mehr erkannt. Wenn man Lichttaster einsetzt, kann dieses Problem nicht auftreten.

Bei großen Anlagen (>500 E/As) mit vielen Dynamiks kann der Austausch von Lichtschranken gegen Lichttaster signifikant zur Reduzierung der Rechenzeit beitragen. Bitte kontaktieren Sie uns aber vor solchen Umbauten. www.trysim.de.

Die vereinfachte Rechenvorschrift des Lichttasters funktioniert bei der Erfassung von drehbaren Dynamiks leider nicht. Darum kann man auch beim Lichttaster die Anzahl der Punkte eingeben, die entlang des Strahls überwacht werden. Diese Zahl, die Sie von 1 - 19 einstellen können hat aber _nur_ Einfluss auf die Erfassung von drehbaren Dynamiks. Wenn Sie in ihrer Anlage keine drehbaren Dynamiks haben, ignorieren Sie diesen Parameter bitte.

ACHTUNG! Wenn andererseits der Lichttaster selbst gedreht wird, wird er nicht mehr funktionieren. In diesem Fall verwenden Sie bitte die Lichtschranke.

Endschalter

*

Mit Endschaltern erfassen Sie die Position eines bestimmten Elementes, das Master genannt wird und das Sie in der Editiermaske auswählen. Meistens wird dies ein Kasten sein, es kann aber auch jedes andere bewegte Element sein. Wählen Sie jedoch nicht einen Linearbeweger als Master, sondern irgendeinen von ihm bewegten Kasten.(Es gibt auch Linearbeweger mit integrierten Endschaltern).

Im Gegensatz zu echten Endschaltern können die Endschalter in TrySim jede beliebige Größe haben. Langgestreckte Endschalter können daher z.B. die Funktion einer Sicherheitsleiste übernehmen.

Der Erfassungsbereich ist gleich der Größe des Endschalters. Die Größe können Sie nicht mit der Maus, sondern nur durch Eingabe in der Editiermaske ändern.

Wenn Ihr Endschalter von mehr als einem Element bedeckt werden soll, können Sie die Endschalter-Nase verwenden.

Mit dem Feld “Typ” können Sie einen Endschalter als Schließer oder Öffner konfigurieren.
Die ebenfalls vorhandene Auswahlmöglichkeit “Schalter”, bei der der Endschalter bei jeder Bedeckung seinen Zustand ändert, ist nur für die Simulation von Spezialschaltern wie dem Kreuzhebel-Schalter sinnvoll.
Wenn der Endschalter eine “1” an die SPS meldet, wird er rot gefärbt.

Wenn Sie dynamische Elemente detektieren wollen, so verwenden Sie die Lichtschranke. Wenn der Sensor sowohl von Dynamiks als auch von einem Maschinenteil bedeckt wird (kommt bei Hubtischen häufiger vor), dann müssen Sie sowohl eine Lichtschranke, als auch einen Endschalter verwenden. Setzen Sie diese nicht auf die gleiche Adresse, sondern programmieren Sie z.B. im OB2:

U Lichtschranke
S Endschalter

Beachten Sie, dass dort steht: “S Endschalter”. Die Zuweisung “= Endschalter” funktioniert nicht (warum nicht?).

Endschalter können drehbar gemacht werden. Wählen Sie dazu auf der Editiermaske die Checkbox “Drehbar” an. Bitte beachten Sie, dass die drehbaren Endschalter ihren Master z.Z. nur erkennen, wenn entweder eine Ecke des Endschalters im Master ist, oder eine Ecke des Masters im Endschalter ist. Lesen Sie auch Allgemeines über drehbare Elemente .

Endschalternase

**

Normalerweise achtet ein Endschalter auf genau ein Element. Manchmal benötigt man jedoch einen Endschalter, der auf viele Elemente achtet. Dazu gibt es die Nase: sie bedeckt jeden Endschalter mit dem sie überlappt, auch wenn sie nicht als Master angegeben worden ist.

Die Knoten einer Kette haben die Nasen-Eigenschaft etwas eingeschränkt: sie sind scharf auf jeden Endschalter, der ihre Kette als Master hat.

Nasen können drehbar gemacht werden. Wählen Sie dazu auf der Editiermaske die Checkbox “Drehbar” an. Lesen Sie auch allgemeines über drehbare Elemente.

Spion

**

Den Spion verwenden Sie, um die Position beweglicher Elemente zu erfassen. In der Realität setzen Sie dafür Drehimpulsgeber, Absolutwertgeber, Potentiometer usw. ein. Alle diese Bauteile haben letztendlich die Aufgabe, irgendwo in der SPS die Position eines anderen Bauteiles zu hinterlegen. Genau dies tut der Spion.

Der Spion erfasst den Abstand von seiner eigenen Position zu der seines Masters in den drei Raumrichtungen. Für jede Raumrichtung müssen Sie ein DInt in der SPS angeben. Wenn Sie nur eine Koordinate benötigen, geben Sie den anderen einfach eine MD mit einer hohen Nummer, die Sie sonst nicht gebrauchen, z.B. MD 17000.

Da Positionserfassungen in der Wirklichkeit verschiedene Auflösungen haben, können Sie im Feld “Auflösung” vorgeben, um wie viele Einheiten sich der SPS-Wert ändern soll, wenn sich der Master um 1 mm verschoben hat.

Den Nullpunkt des Spions stellen Sie folgendermaßen ein: Sie bringen das detektierte Element an eine Position, von der Sie wissen, welcher Wert in der SPS angezeigt werden soll. Wenn dieses Element an einem Linearbeweger hängt, können Sie es mit dem Schieberegler auf dessen Editiermaske verschieben. Dann geben Sie in den Feldern “Anzeige” den Wert ein, der in die SPS übertragen werden soll. Das ist alles.

Wenn Sie die Position des Spions selbst erfassen wollen, z.B. weil Sie ihn aus Gründen der Übersichtlichkeit direkt an dem bewegten Element befestigt haben, behalten Sie den Ursprung als Master bei. In diesem Fall sollten Sie für die Auflösung einen negativen Wert eingeben, sonst zählt der Spion falsch herum.

Zur Erfassung der Position eines Linearbewegers ist es häufig günstiger den Linearbeweger mit Sensoren zu verwenden.

Ultraschallsensor

**

Mit einem Ultraschall-Sensor können Sie die Entfernung zu dynamischen Elementen bestimmen, nur auf diese reagiert der Sensor.

Die Entfernung zum Objekt wird in den von Ihnen auf der Editiermaske gewählten Einheiten (vorgegeben sind 1 Imp/mm) in ein SPS-Word geschrieben. Im Feld “Punkte” müssen Sie angeben, wie viele Punkte auf seiner Länge der Sensor prüfen soll, um überhaupt ein Dynamik zu finden. Diesen Wert müssen Sie an die Größe der Dynamiks anpassen. Ist er zu klein, kann es vorkommen, dass der Sensor ein Dynamik übersieht, ist er zu groß, wird viel Rechenzeit verbraucht.

Mit der Checkbox “Max-Wert, wenn nichts gefunden” können Sie einstellen, ob der Sensor eine “0” oder den maximalen Wert an die SPS melden soll, wenn er kein Dynamik gefunden hat.

Um die Orientierung eines Ultraschall-Sensors mit der Maus zu ändern, müssen Sie mit der Maus so auf ein Ende zeigen, dass sich der Maus-Zeiger in ein Kreuz verwandelt.

Wenn Sie nur einen Kontakt benötigen, der bei einer bestimmten Entfernung schaltet, programmieren Sie z.B. im OB2 :

```
L EW 512 // hier die richtige Adresse einsetzen
L 500    // hier den Schaltabstand einsetzen
>I       // je nach gewünschter Funktionalität auch <I
= E 4.6  // hier den richtigen Eingang einsetzen
```

Impulsgeber für Förderband/Kette

**

Mit diesem Element können Sie feststellen, wie weit ein Förderband, eine Kette oder eine Mangel etwas transportiert hat, oder wieviel Material aus einem kontinuierlichen Generator gekommen ist. Dazu müssen Sie das Förderband/die Kette/den Generator als Master angeben und ein Word in der SPS bestimmen. Der Impulsgeber zählt vorwärts und rückwärts, er nimmt dabei nur inkrementelle Änderungen an dem SPS-Word vor, d.h. Sie können den Wert jederzeit vom SPS-Programm aus ändern, der Impulsgeber zählt dann einfach von dem neuen Wert aus weiter. Wie bei den anderen Positionserfassungen können Sie die Auflösung

vorgeben. Einen Synchronisationsmechanismus haben wir nicht vorgesehen, da Sie ja den gemeldeten Wert von der SPS aus beeinflussen können.

Sie können diesen Impulsgeber nur an die oben genannten Elemente anschließen. Wenn Sie ihn an ein Diagonalförderband anschließen, beachten Sie bitte, dass die gemeldete Bewegung um einen Faktor von Wurzel 2 (ca. 1,4) zu klein ist. Dies können Sie aber leicht korrigieren, indem Sie die Auflösung entsprechend anpassen.

Wird der Impulsgeber an einen kontinuierlichen Generator angeschlossen, dann erfasst er nur dann die Bewegung des Strangs, wenn dieser seine linke untere Ecke bedeckt.

Wenn Sie gar keine Zahl benötigen, sondern nur einen blinkenden Eingang, dann verwenden Sie dazu einfach das niederwertigste Bit. Wenn Sie zwei um 90° versetzte Impulsspuren benötigen, nehmen Sie für Spur A das Bit1 und für Spur B Bit1 XOR Bit0. Z.B. so:

U Bit1
= SpurA
X Bit1
X Bit0
= SpurB

Natürlich müssen Sie die Auflösung dann so gering einstellen, dass sich von Zyklus zu Zyklus der Wert um höchstens 1 ändert. Gegebenenfalls müssen Sie die Simulationsrate anpassen oder ein anderes Paar von benachbarten Bits mit höherer Bitnummer verwenden.

Barcode-Leser/Schreiber

**

Der Barcode - Leser liest den Barcode von Dynamiks, der bei der Erzeugung durch den Generator festgelegt worden ist. Dieser Code wird in das angegeben Word der SPS geschrieben. Wird der Barcode-Leser von keinem Dynamik bedeckt, wird der Wert ‘0’ geschrieben.

Bei normalen Dynamiks gilt der Leser als bedeckt, wenn er mit dem Dynamik irgendwo überlappt. Für die Drehbaren gilt aber immer noch, dass sich die Positions-Ecke (links unten) innerhalb des Dynamiks befinden muss.

Wie beim Endschalter können Sie die Größe des Barcode-Lesers nur durch Eintippen neuer Werte in der Editiermaske ändern. Wird der Barcode-Leser von mehr als einem Dynamik bedeckt, ist nicht festgelegt, welchen Wert er liest.

Der Barcode-Leser kann auch als Barcode-Schreiber konfiguriert werden. Jedes Dynamik das ihn bedeckt, versieht er mit dem Code der im SPS-Word steht. Wenn dort eine Null steht, wird nichts geschrieben!

Der Barcode-Leser kann auch die Farbe der Dynamiks erkennen und auch verändern. Die Zahlen im SPS-Word sind seit der Version V2.9.60 ein

komprimiertes RGB-Format, das sich am besten mit der Digitalanzeige im Hex-Word Format anzeigen lässt.
Rot : 0 0 0 F
Grün : 0 0 F 0
Blau : 0 F 0 0
Schwarz : 0 0 0 1 !

--- #element Funktionen ---
#element.dyn_barcode_w (read/write)
#element.dyn_color_w (read/write)
#element.dyn_color (read/write), dword, normales RGB-Format

Laufrad

****

Mit diesem Element können Sie die Bewegung von Dynamiks detektieren, es erfasst aber auch die Bewegung eines Strangs, der von einem kontinuierlichen Generator erzeugt wird. Auf der Editiermaske können Sie verschiedene Datenformate und die Auflösung vorgeben.

Wichtig für die Anwendung ist, dass die Positionsecke (auf dem Bildschirm immer links unten) innerhalb des Dynamiks ist.

Das Format “Impulse” und der Z-Impuls sind bislang noch nicht implementiert. Zurzeit kann das Laufrad wechselnde Bewegungsrichtungen noch nicht unterscheiden: es zählt immer vorwärts.

2.4.3 Aktoren

Generator

*

Den Generator benötigen Sie, um das Material (dynamische Elemente, Dynamiks) zu erzeugen, das die Maschine bearbeitet. Er wird an ein Bit der SPS angeschlossen. Bei jeder steigenden Flanke dieses Bits wird ein Dynamik erzeugt. Steht das Bit dauernd an, wird in regelmäßigen Abständen, die Sie im Feld “Verzögerung” einstellen können, ein neues Dynamik erzeugt.

Die Dynamiks sind immer quaderförmig, können aber jede beliebige Größe haben. Die Größe des Dynamiks geben Sie in der Maske ein, die vom Button “Dynamiks” aufgerufen wird. Es ist am einfachsten, dort “wie Generator” anzuklicken, dann können Sie die Größe der erzeugten Dynamiks auch nachträglich durch Ziehen der Begrenzung des Generators anpassen.

Die Größe der Dynamiks wird zufällig um die in den Feldern unter der Größe stehenden Werte größer oder kleiner gemacht, um so die Widrigkeiten des wirklichen Lebens besser zu simulieren. Wenn die Dynamiks immer exakt gleich groß sein sollen, müssen Sie dort Null eintragen. Beachten Sie in diesem

Zusammenhang auch die Zufallszahlen.

Um Dynamiks zu identifizieren, können Sie in der Dynamikmaske ein Word der SPS angeben. Jedesmal, wenn ein Dynamik erzeugt wird, wird es mit dem Wert markiert, der in diesem Moment in dem entsprechenden Word steht. Mit dem Barcode-Leser können Sie diesen Wert anschließend an anderer Stelle in der Anlage wieder auslesen und die Programmbearbeitung davon abhängig machen, z.B. Dynamiks mit gerader Nummer nach rechts, die mit ungerader nach links.

Sie können auch drehbare Dynamiks erzeugen zu lassen. Die nur für diese Art von Dynamiks möglichen Optionen Center-Punkte, Kanten-Punkte und X-Y Drehungen sind im Abschnitt über die drehbaren Dynamiks erklärt.

Normalerweise sind die Dynamiks auf eine vereinfachte Weise schwerkraftabhängig, d.h. wenn nicht auf einer Unterlage stehen, fallen sie mit einer konstanten Geschwindigkeit. Diese Geschwindigkeit können Sie einstellen. Normalerweise gibt der Vorgabewert 400 mm/s gute Ergebnisse. Falls Sie Dynamiks aber auf bewegliche Unterlagen stellen, die sich schneller nach unten bewegen, sollten Sie diesen Wert anpassen. Eventuell muss dann aber auch die "Plattformdicke" unter **Ansicht|Optionen|Anlage** erhöht werden, sonst fallen die Dynamiks durch Förderbänder u.ä. durch. Wenn Sie die Schwerkraftabhängigkeit ganz abwählen, müssen Sie z.B. mit einem Haken oder einem Schieber selbst dafür sorgen, dass das Dynamik den Generator verlässt.

Wenn Sie die Dynamiks füllbar machen, verhalten sich diese wie ein Behälter.

In der Standardeinstellung sind die erzeugten Dynamiks in jeder Grafik und aus jeder Richtung sichtbar. Sie können dieses Verhalten über die Eigenschaften der Grafikfenster ändern. Sie können aber auch die Checkbox "Sichtbar wie Generator" aktivieren. Dann folgt die Sichtbarkeit der Dynamiks den gleichen Regeln wie für den Generator, d.h. die Richtungen aus denen das Dynamik zu sehen ist und der Grafikfilter werden vom Generator übernommen. Beachten Sie bitte, dass diese Übernahme zum Zeitpunkt der Erzeugung des Dynamiks erfolgt. Auf ein bereits erzeugtes Dynamik hat der Generator keinen Zugriff mehr.

Bitte beachten Sie, dass erst dann ein neues Dynamik erzeugt wird, wenn das vorherige den Generator verlassen hat. Gegebenenfalls müssen Sie die Größe des Generators verkleinern.

Häufiger Anfängerfehler: Ein Generator wird in der X-Y-Aufsicht plaziert und er scheint dann keine Dynamiks zu produzieren. TrySim simuliert die Welt dreidimensional! Wenn ein Generator, der auf dem Hallenboden steht, ein Dynamik erzeugt, dann steht dieses auch auf dem Hallenboden und hat keinen Anlass, sich in irgendeine Richtung zu bewegen. Das neue Dynamik wird nicht einmal sichtbar, außer man schaut ganz genau hin und erkennt, dass die Farbe des Generators von violett nach schwarz gewechselt hat.
Damit die Dynamiks aus dem Generator herausfallen, wie es in Beispiel 1 zu sehen ist, muss man den Generator über dem Hallenboden plazieren. Das geht nur in einer Z-Y Seitenansicht oder einer X-Z Vorderansicht. Und auch nur in diesen Ansichten

kann man das Herausfallen eines Dynamiks aus dem Generator beobachten.

Kontinuierlicher Generator

****

Hiermit erzeugen Sie einen kontinuierlichen Strang von Material. Die Erzeugungsgeschwindigkeit wird vom Antrieb vorgegeben, für diesen Generator stehen die gleichen Antriebe wie für das Förderband zur Verfügung. Auf der Editiermaske wählen Sie die Förderrichtung (+X, -X, +Y, -Y, +Z, -Z). Der Querschnitt des Strangs wird durch die Abmessungen des Generators vorgegeben.

Mit der Säge können Sie Stücke des Strangs abtrennen. Diese Stücke werden dann zu normalen Dynamiks, deren Eigenschaften Sie auf der Editiermaske des kontinuierlichen Generators unter “Dynamiks” einstellen können. Sie können nur quer zum Strang schneiden, nicht entlang seiner Bewegungsrichtung.

Damit ein Strang erzeugt wird, muss das Bit “Aktiv” gesetzt sein, oder die entsprechende Checkbox “Immer aktiv” auf der Editiermaske muss angeklickt sein. Wenn der kontinuierliche Generator nicht mehr aktiv ist, wird der bestehende Strang abgetrennt und in ein normales Dynamik verwandelt. Wird der Generator wieder aktiv, wird ein neuer Strang erzeugt.

Das Vorhandensein des Strang können Sie mit der Lichtschranke detektieren. Seine Bewegung können Sie mit dem Impulsgeber für Förderband und Kette erfassen. Dieser wirkt dann wie ein Laufrad: nur wenn der Strang den Impulsgeber (genauer: dessen Ecke links unten) bedeckt, werden die Impulse weiter gezählt. Als Master des Impulsgebers muss der Generator angegeben werden. Bitte beachten Sie, dass dieses Laufrad nicht mehr weiter zählt, wenn der Strang abgeschnitten und in ein Dynamik verwandelt worden ist (z.B. durch eine Säge oder durch deaktivieren des Generators).

Dieses Element befindet sich noch im Versuchsstadium. Anregungen aus der Praxis zur Verbesserung sind willkommen.

· Kontinuierliche Dynamiks / Strang

****

Diese werden von einem kontinuierlichen Generator erzeugt. Bitte lesen Sie zunächst dort.

Die Bewegung eines Strangs können Sie auch mit dem Laufrad erfassen. Dies hat gegenüber einer Erfassung über den Generator den Vorteil, dass die Bewegung auch dann weiterhin erfasst wird, wenn der Strang abgeschnitten wird, sei es durch Ausschalten des Generators oder durch eine Säge.

Vernichter

*

Das von einem oder mehreren Generatoren erzeugte Material wird in Wirklichkeit die Anlage ja irgendwann verlassen, entweder wird es abtransportiert oder von anderen Anlagen weiter bearbeitet. Für die von TrySim simulierte Anlage ist das Material dann auf jeden Fall nicht mehr wichtig und sollte darum “vernichtet” werden, da es sonst unnötigerweise Rechnerkapazität bindet.

Der Vernichter ist im Prinzip ein Förderband, aber er vernichtet jedes Dynamik, sowie alle Ecken auf ihm stehen oder in der Luft hängen. Erfahrungsgemäß schaffen es am Anfang eines Projektes nur wenig Dynamiks bis zum Vernichter, sodass nach einer Weile alles mit Dynamiks voll liegt. Dann können Sie mittels des Menübefehls **Anlage|Dynamiks löschen** alle Dynamiks auf einmal entfernen, oder durch Markieren mit der Maus und der Taste **Entf** einzelne Dynamiks löschen.

Drehbare Dynamiks werden vernichtet, wenn mindestens 3 Punkte auf dem Vernichter stehen. Dabei zählen evtl. Center- und Kanten-Punkte mit.

Für frei bewegliche Elemente, die immer innerhalb der Anlage bleiben, die aber sonst die Eigenschaften von Dynamiks haben, gibt es die speicherbaren Dynamiks. Diese werden nicht vom Vernichter vernichtet.

Linearbeweger

*

Siehe Linearbeweger mit Sensoren.

Linearbeweger mit Sensoren

*_**

Diese Elemente benötigen Sie, um Ihre Maschine in Bewegung zu versetzen. In der Wirklichkeit entsprechen sie allen Bauteilen, die eine gerade Bewegung ausführen. Das können hydraulische/pneumatische Zylinder, Zahnstangenantriebe, piezo-elektrische Stellglieder, auf Schienen rollende Wagen und alles möglich sonst sein.

Ein Linearbeweger in TrySim ist eine Linie, die irgendwie im Raum orientiert sein kann, entlang derer die Bewegung erfolgt. Wenn die von Ihnen gewünschte Bewegung nicht entlang einer Linie verläuft, müssen Sie entweder zwei oder mehr Linearbeweger miteinander koppeln oder, falls auch dies nicht möglich ist, ein Gelenk, den FreePoint oder die Kette verwenden.

Der Linearbeweger hat einen ausgezeichneten Punkt, der Hotspot genannt wird. Dieser Punkt kann entlang des Linearbewegers verschoben werden, jedoch nicht über die Enden hinaus. Sie können andere Elemente an dem Hotspot befestigen, indem Sie den Linearbeweger als ihren Vater angeben. Alle diese Elemente bewegen sich dann genau wie der Hotspot.

Die Geschwindigkeit oder die Position des Hotspots wird vom Antrieb vorgegeben. Der Antrieb entspricht also einem Motor, einem Ventil oder auch einer

Steuerungselektronik. Sie wählen den Antrieb aus, indem Sie ihn in der Editiermaske aus der Liste “Typ" anklicken. Mit dem Button “Antrieb” rufen Sie die Editiermaske des Antriebs auf.
Die möglichen Antriebe des Linearbewegers stehen weiter unten in diesem Kapitel.

Auf der Editiermaske wird die Position des Hotspots als Zahl und durch einen Schieberegler angezeigt. Während der Bauzeit kann sie dort geändert werden, um Endschalter zu justieren u.ä.

Es gibt auch einen Linearbeweger mit integrierten Endschaltern und Positionsgeber .

Um die Orientierung eines Linearbewegers mit der Maus zu ändern, müssen Sie mit der Maus so auf ein Ende zeigen, dass sich der Maus-Zeiger in ein Kreuz verwandelt.

Sie können einen Linearbeweger auch verwenden, um ein Gelenk, ein Ventil oder eine Heizung anzusteuern.

Der Linearbeweger ist mit integrierten Endschaltern und einem Positionsgeber ausgestattet. Die Sensoren können unter den Registerkarten "Endschalter" und "Position" auf der Editiermaske des Linearbewegers konfiguriert werden.

Die Synchronisierung des Positionsgebers geschieht folgendermaßen:

1. Bringen Sie den Linearbeweger an eine Lage, von der Sie den anzuzeigenden Wert des Positionsgebers kennen. Dazu können Sie u.a. den Schieber “HotSpot” auf dem Linearbeweger-Formular verschieben.

2. Öffnen Sie die Maske des Positionsgebers unter der Registerkarte "Position". Sie können entweder den Offset als Weg eingeben, oder den aktuell zu meldenden Geberwert im Feld "Anzeige" eintragen.

Beachten Sie den Unterschied dieser Positionserfassung zu derjenigen mit dem Spion: Bei schräg im Raum laufenden Linearbewegern wird hier der zurückgelegte Weg ausgegeben, mit dem Spion hingegen würden Sie drei Raumkoordinaten erhalten.

Die integrierten Endschalter sprechen nur an, wenn der Hotspot des Linearbewegers ganz am Anschlag ist. Falls der Hotspot dort nicht stehen bleibt (das kommt häufig vor, wenn der Linearbeweger von einen Gelenk angetrieben wird), führt dies zu Fehlfunktionen. In solchen Fällen sollten Sie einen normalen Linearbeweger und normale Endschalter verwenden. Diese können Sie so justieren, dass sie in jedem Fall ansprechen.

Die Positionserfassung liefert den Wert als DINT. Das haben wir so gemacht, weil für viele Anwendungsfälle der Wertebereich des INTs (-32000 bis +32000) nicht ausreicht. Falls Sie im Programm nur ein INT benötigen, müssen Sie nur die letzten beiden Bytes des DINTs laden. Beispiel:

Beim Sensor geben Sie ED 1200 an. Dieses ED besteht aus den Bytes 1200, 1201, 1202 und 1203. Wenn Sie nur ein INT benötigen, weil Ihre Positionen garantiert kleiner als 32.000 Einheiten sind, schreiben Sie im SPS-Programm: L EW 1202.
Bitte beachten Sie dass die Bytes 1200 und 1201 nicht mehr verwendet werden dürfen, denn die Positionserfassung wird sie in jedem Simulationszyklus beschreiben.
Ganz nebenbei: Es ist schlau, immer DINTs zu verwenden. Moderne SPSen haben Speicherplatz ohne Ende und bei einem DINT braucht man fast nie darüber nachzudenken, ob der Wertebereich groß genug ist.

Bitte lesen Sie auch die Hilfe zum Impulsgeber für Förderband.

Förderband

*

Mit Transportbändern bewegen Sie dynamische Elemente. Sie verfügen wie die Linearbeweger über einen Antrieb. Steht ein dynamisches Element mit allen vier Ecken auf dem Transportband, wird es mit der vom Antrieb vorgegebenen Geschwindigkeit bewegt. Stehen eine oder mehrere Ecken nicht auf dem Band, wird es entsprechend langsamer bewegt und rot gefärbt, um auf diesen unnatürlichen Zustand hinzuweisen. Achten Sie daher darauf, dass mehrere nacheinander folgende Förderbänder bündig aneinander stoßen.

Wenn Sie ein Transportband an einem senkrecht stehenden Linearbeweger befestigen, erhalten Sie einen Fahrstuhl für Paletten. Befestigen Sie es an einem auf dem Boden liegenden Linearbeweger so erhalten Sie einen Verfahrwagen.

Es sind nur Bewegungen in der X- und Y-Ebene und dort auch nur in 8 Richtungen möglich. Das Förderband transportiert die Dynamiks mit der von seinem Antrieb vorgegebenen Geschwindigkeit. Die Transportrichtung wird durch den Pfeil in der Editiermaske angezeigt, durch Klicken ändern Sie die Richtung.

Das Diagonalförderband transportiert die Dynamiks nicht entlang der X- oder Y-Richtung, sondern diagonal.

Gelenk

***

Das Gelenk kann verwendet werden, um Drehbewegungen um beliebig orientierte Achsen zu simulieren. Es hat die Gestalt eines Linearbewegers. Durch die Lage der Linie im Raum legen Sie die Drehachse fest. Um diese Achse werden alle Elemente (auch deren Kinder) gedreht, die Sie an dem Gelenk befestigen.

Die zu drehenden Elemente sollten möglichst klein sein, da die meisten Elemente in der jetzigen Version noch nicht um sich selbst gedreht werden. Eine Ausnahme bilden die linienförmigen Elemente wie Lichtschranke, Linearbeweger und Stange sowie der Kasten, die Haken, die Endschalter, das drehbare Förderband und die

drehbaren Dynamiks.

Folgende Antriebe stehen für das Gelenk zur Verfügung:

1. Bit-Antrieb: Dieser Antrieb wird an eines oder mehrere Bits der SPS angeschlossen und erlaubt die Steuerung der Drehgeschwindigkeit in beide Richtung und mit bis zu zwei Geschwindigkeiten. Auch die Hoch- und Runterlauf-Rampen können eingestellt werden, ebenso wie eventuelle Begrenzungen, falls das Gelenk nicht kontinuierlich rundum laufen soll.

2. Servo-Antrieb: Hier geben Sie während der Erstellung der Simulation einige Drehwinkel vor, die dann während der Simulationslaufzeit angefahren werden. Dazu muss in der SPS nur die Nr. des Drehwinkels angegeben werden.

3. Linearbeweger als Antrieb: Hiermit können Sie eine Linear- in eine Drehbewegung übersetzen. Auch eine Ratschen-Funktion ist möglich.

4. Kurbel-Antrieb: Hiermit koppeln Sie das aktuelle Gelenk an ein anderes, bereits vorhandenes Gelenk. Das Übersetzungsverhältnis ist einstellbar.

5. Absolut-Real-Antrieb: Direkte Vorgabe des Drehwinkels. Die gleiche Funktion wird besser vom Servo-Antrieb mit Direkt-Vorgabe erfüllt.

6. Einfacher Frq-Umrichter: Dieser Antrieb entspricht einem frequenzgeregelten Motor oder dem Proportionalventil eines Hydraulikmotors. Er muss an ein Word (E, A-, M- oder Daten-) der SPS angeschlossen werden.
   Neben der Adresse müssen Sie noch angeben, bei welchem Wert des Words die Maximalgeschwindigkeit erreicht werden soll. Wenn das Word in der echten SPS einen Analogausgang ansteuern soll, der einem Frequenzumformer +/- 10V vorgeben soll, müssen Sie hier 27.648 eingeben.
   Auch für diesen Antrieb können Sie Rampen und Begrenzungen des Drehwinkels vorgeben.

Auf der Editiermaske des Gelenkes sind zwei Schieberegler vorhanden:
1.) Der Schieberegler “ohne” lösen der Achsmutter. Hiermit drehen Sie die Achse mitsamt den befestigten Elementen. Sie bewirken dadurch keine konstruktiven Veränderungen an der Maschine. Diese Dreh-Möglichkeit ist nur dazu geschaffen worden, um während des Editierens leicht die Bewegung der an dem Gelenk befestigten Elemente beobachten zu können und z.B. Endschalter zu justieren. Sobald Sie die Simulation wieder starten, springen die an dem Gelenk befestigten Elemente auf ihre ursprüngliche Position.

2.) Der Schieberegler “mit” lösen der Achsmutter bewirkt jedoch konstruktive Änderungen an der Maschine. Eine Verstellung dieses Reglers entspricht dem Verdrehen der an der Achse angeschraubten Teile nach Lösen der Achsmutter

Mit den Bild-Rauf/Runter-Tasten bewirken Sie eine Verstellung des Drehwinkels um jeweils 45°, eine Feinjustierung ist mittels der Pfeil Rechts/Links - Tasten möglich

Mit einem Peek können Sie den aktuellen Winkel des Gelenks in ein Word der SPS (Einheit 0,1 Grad) laden.

Die Beispiele “Robby1” und “Robby2” demonstrieren den Gebrauch von Gelenken.

--- #element.Funktionen ---
#element.winkel_ist (read, REAL, grad).

Haken

*

Mit dem Haken greifen Sie Dynamiks, wie mit einem Kran. Wie bei den anderen Elementen auch, sollten Sie den Namen nicht wörtlich nehmen. Verwenden Sie den Haken immer dann, wenn Sie Dynamiks transportieren, die nicht auf einer Unterlage stehen.

Ob gegriffen werden soll oder nicht, legen Sie durch ein Ausgangsbit der SPS fest. Wenn Sie das Bit anschalten, greift der Haken alle Dynamiks, mit denen er irgendwo überlappt. Beachten Sie bitte, dass ein Haken auch extern durch einen Ein-/Aus-Klinker betätigt werden kann.

Haken eignen sich auch gut als Ausstoßer und Stopper. Da die Simulation solcher Elemente extrem aufwendig ist (es müssten Gewichte, Reibungskoeffizienten, Festigkeiten usw. berücksichtigt werden), müssen Sie hier ein wenig nachhelfen und den Haken im geeigneten Moment an- und abschalten. Falls dazu auch SPS-Programm-Code notwendig ist, sollten Sie diesen im Script (eventuell auch im OB 2 oder OB 3) anordnen. In einfachen Fällen kann hierfür aber auch der Schieber von Nutzen sein

Haken können drehbar gemacht werden. Wählen Sie dazu auf der Editiermaske die Checkbox “Drehbar” an. Lesen Sie auch allgemeines über drehbare Elemente weiter unten in diesem Kapitel.

Freibeweglicher Punkt

***

Dieses Element ist für den Fall vorgesehen, dass Ihre Maschine eine Bewegung ausführt, die sich nicht in einfacher Weise aus Linearbewegungen zusammensetzt, z.B. eine Kreisbewegung. Weil es sich leichter schreibt, nennen wir es auch Freepoint. Sie können alle drei Koordinaten des Freepoints von der SPS aus vorgeben. Dazu müssen Sie dann einen Funktionsbaustein schreiben, der die gewünschte Bewegung berechnet.

Für jede der drei Raumrichtungen ist Freepoint ein DInt vorgesehen, in welches Sie von der SPS aus die Position in den von Ihnen für die Anlage gewählten Einheiten (standard: mm) hinterlegen müssen.

Wenn Sie die Positionen nicht als DInt, sondern als INT ausrechnen, müssen Sie darauf achten, dass der Wert nicht negativ wird, oder ihn mittels des Befehls ITD auf das DInt-Format erweitern.

Kette

***

Die Kette ist ein geschlossener Linienzug aus beliebig vielen Segmenten. Auf diesem Linienzug bewegen sich beliebig viele Knoten mit der vom Antrieb vorgegebene Geschwindigkeit im Kreis. Sie können die Position jedes Knotens auf der Kette einzeln justieren oder die Knoten automatisch gleichmäßig verteilen lassen.

Wenn Sie eine Kette erzeugen, hat sie zunächst 4 Segmente. Weitere Segmente erzeugen Sie durch Teilung bereits vorhandener. Markieren Sie dazu ein Segment und rufen mit Rechtsklick die Editiermaske der Kette auf. Wählen Sie darauf den Button “Segment teilen” unter der Registerkarte "Parameter". Die einzelnen Segmente können Sie wie einen Linearbeweger schieben und drehen. Zum drehen müssen Sie die Maus an das Ende des markierten Segmentes führen, und zwar so, dass sich der Mauszeiger in ein Kreuz verwandelt. Das ist leider, besonders bei Segmenten, die genau entlang einer Achse verlaufen, manchmal eine etwas fummelige Angelegenheit.

Die Knoten entstehen automatisch, wenn Sie ein Element an der Kette befestigen wollen. Die Position des Knotens wird so nahe wie möglich zu der Position des neu befestigten Elementes gewählt. Sie können den Knoten dann mit der Maus an die gewünschte Position schieben. Um Ecken herum lassen sich Knoten schlecht schieben, in diesen Fällen kann man die Editiermaske des Knotens öffnen, und seine Position auf der Kette mit dem Schieberegler “Offset” korrigieren. Vor dem Anbringen der Knoten sollte die Kette fixiert werden, um eine versehentliche Verschiebung der Segmente zu vermeiden.

Wenn Sie die Bewegung der Kette erfassen wollen, dann verwenden Sie bitte den Impulsgeber für Förderband/Kette.

Im Zusammenhang mit Ketten sind auch der Ein-/Aus-Klinker und die Endschalternase von Nutzen. Wenn Sie einen Endschalter auf eine Kette scharf machen, wird er von jedem Knoten betätigt, mit dem er überlappt.

Die Beispiele “Formel” und “Baumsäge” demonstrieren den Gebrauch der Kette.

In dieser Version werden die an den Knoten hängenden Teile bei Kurven nicht gedreht.

Das Editieren einer Kette ist leider noch verbesserungswürdig.

· Knoten

***

Die so genannten Elemente kommen nur als Teile einer Kette vor. Sie dienen als Befestigungspunkte für weitere Elemente Ihrer Simulation. Sie werden automatisch erzeugt, wenn Sie versuchen, ein Element an einer Kette zu befestigen. Sie können einen Knoten entlang der Kette mit der Maus verschieben oder auf der Editiermaske den Schieberegler “Offset” verstellen. Dies entspricht dem Abschrauben eines Hakens von einer wirklichen Kette und dem Wiederanbringen an anderer Stelle. Im Gegensatz dazu entspricht der Schieberegler auf der Editiermaske der Kette selber nur dem Laufenlassen des Antriebs.

Wenn Sie einzelne Knoten mit einem Endschalter erfassen wollen, müssen Sie den jeweiligen Knoten als Master des Endschalters angeben. Wenn jedoch alle Knoten einen Endschalter bedecken sollen, müssen Sie die Kette als Master angeben.

· Segment

***

Eine Kette besteht aus Segmenten. Die Segmente können Sie mit der Maus oder über die Editiermaske positionieren wie andere linienförmige Elemente auch. Bei Verschiebung eines Segmentes werden das vorangegangene und das folgende Segment so angepasst, dass die Kette wieder geschlossen ist.

Neue Segmente erzeugen Sie durch Teilung eines bereits vorhandenen. Markieren Sie dazu das zu teilende Segment, öffnen mit Rechtsklick die Editiermaske der Kette und wählen Sie den Button “Segment teilen” unter der Registerkarte "Parameter". Wenn Sie ein Segment löschen, werden das vorhergehende und das nachfolgende Segment so verbunden, dass sie sich in der Mitte des gelöschten Segmentes treffen. Das Löschen des vorletzten Segmentes (die Kette besteht dann nur noch aus zwei aufeinander liegenden Segmenten) ist gleichbedeutend mit dem Löschen der ganzen Kette.

Um zur Editiermaske eines Segmentes zu gelangen, markieren Sie es, öffnen mit Rechtsklick die Maske der Kette und wählen dann den Button “Segment editieren”.

Die Knoten, die auf einem Segment sitzen, gehören nicht zu diesem, sondern zu der übergeordneten Kette.

Sie können die Sichtbarkeit für jedes Segment einer Kette einzeln einstellen, diese Einstellung wird jedoch überschrieben, wenn Sie die Sichtbarkeit der ganzen Kette geändert wird. Wenn Sie ein Element in allen Richtungen unsichtbar gemacht haben, können Sie es nur noch bearbeiten, indem Sie es über den Elementbaum aufrufen.

Dreher

**

Dieses Elemente sollten Sie nicht mehr verwenden, das Gelenk ist viel variabler.

Der Dreher bewegt lediglich ein an ihm befestigtes Element auf einer Kreisbahn in der XY-Ebene mit der durch den Antrieb vorgegebenen Geschwindigkeit. An dem Dreher direkt sollten Sie nur ein Element befestigen.

Reiterbahn

****

Die Reiterbahn ist eine Fortentwicklung der Kette. Sie ist viel flexibler, aber auch schwerer einzusetzen. Anfänger sollten die Reiterbahn nicht verwenden.

Zum Betrieb einer Reiterbahn benötigt man drei Elemente:

1.) Segmente der Reiterbahn an sich. Jedes dieser Segmente hat einen eigenen Antrieb und bewegt alle daran befestigten Elemente mit der vom Antrieb vorgegebenen Geschwindigkeit in Richtung des Segmentes. Sie sollten jedoch nur "Reiter" an der Reiterbahn befestigen.
Anders als bei der Kette ist jedes Segment der Hängebahn vollkommen selbständig. Die Verbindung zwischen den Segmenten muss von Ihnen explizit durch Übergänge geschaffen werden. Die Reiterbahn hat nicht wie der Linearbeweger ein Ende. Alle daran befestigten Reiter werden solange in Richtung der Bahn bewegt, bis sie einen neuen Vater gefunden haben.

2.) Reiter. Dies sind Elemente, die sich nur wenig vom Kasten unterscheiden. Sie sollten Sie als Aufhänger für Haken o.ä. verwenden. Ein Reiter wird immer mit der Geschwindigkeit der Reiterbahn bewegt, dessen Sohn er gerade ist. Und hier kommen die Übergänge ins Spiel: diese können nämlich den Vater eines Reiters ändern. So kann ein Reiter auf seiner Rundreise von verschiedenen Segmenten der Reiterbahnen bewegt werden.

3.) Übergänge. Ein Übergang arbeitet folgendermaßen: in jedem Zyklus überprüft er: überlappe ich mit einem Reiter? Falls er einen solchen gefunden hat fragt er: Überlappe ich mit dem Ende oder dem Anfang einer Reiterbahn? Falls es auch eine solche Reiterbahn gibt, wird der Reiter von der alten Reiterbahn auf die neue umgehängt, d.h. der Vater des Reiters wird von der alten auf die neue Bahn geändert.
Wichtig bei der Plazierung eines Übergangs ist es, dass er mit der neuen Bahn überlappt, aber nicht mit der alten.

Weichen kann man z.B. dadurch realisieren, dass man zwei Reiterbahnsegmente an einem Linearbeweger befestigt. Von welchem Segment der Reiter übernommen wird, entscheidet sich dadurch, welches Segment gerade im Übergang ist.

Tipps, um mit den Reiterbahnen gut arbeiten zu können:

Ein Übergang ist nur für eine Richtung zu gebrauchen. Wenn die Reiter in beiden Richtungen über eine Übergangsstelle laufen sollen, dann benötigen Sie zwei Übergänge. Sie müssen aber gut darauf achten, dass der Reiter nie mit beiden Übergängen überlappt.

Mangel

***

Der Name dieses Elementes wird Sie vermutlich irritieren. Es ist wie so häufig in TrySim: wir müssen dem Kind einen Namen geben. Falls in Ihrer Firma Elemente mit der gleichen Funktion eingesetzt werden, werden sie bestimmt nicht “Mangel” heißen. In Amerika vielleicht schon, aber in Deutschland wird ein solches Gerät sicher mit “Doppel-Walzen-Linear-Förderer” oder “Walzen-Ausstoßer” bezeichnet werden.

Die Mangel besteht aus zwei Walzen, die jedes Dynamik, das die untere Walze berührt einsaugen, und so lange transportieren, bis es die untere Walze nicht mehr berührt. Die Fördergeschwindigkeit wird durch den Antrieb vorgegeben: für die Mangel sind die gleichen Antriebe verfügbar wie für das Förderband. Die Mangel zieht Dynamiks nur ein, wenn die obere Walze gesenkt ist. Welches die obere Walze ist, muss auf der Editiermaske der Mangel angegeben werden. Hier können nur Kästen gewählt werden. Wenn der als “Obere Walze” angegebene Kasten mit der Mangel überlappt, dann transportiert diese die Dynamiks, sonst nicht. Ist kein Element als obere Walze angegeben, werden die Dynamiks immer transportiert.

Mit der Option “Weiterfliegen nach Verlassen” behält das Dynamik seine Geschwindigkeit in der XY-Ebene bei, nachdem es nicht mehr mit der Mangel überlappt. Gestoppt wird das Dynamik, wenn es auf einem Förderband landet oder gegen einen Schieber stößt. Auf anderen Dynamiks oder einer Platte rutscht es weiter.

Die Option “Aushebbar” ist für Mangeln, die in der Auswurfhöhe verstellbar sind. Ohne diese Option werden die Dynamiks einfach geradeaus gefördert, das macht es einfacher sie zu positionieren. Mit der Option werden die Dynamiks bis zur Oberkante angehoben. Das ist manchmal nützlich, erfordert aber mehr Sorgfalt bei der Simulationserstellung.

Z.Z. kann die Mangel nur in X oder Y - Richtung transportieren, d.h. nicht diagonal und nicht senkrecht.

Drehtisch

***

Dies ist ein Element, das nur mit drehbaren Dynamiks funktioniert. Es bewirkt eine aktive Drehung der Dynamiks, die auf ihm stehen. Die Drehgeschwindigkeit wird vom Antrieb vorgegeben. Die dort angegebene Geschwindigkeit in mm/s bedeutet die Geschwindigkeit für Dynamiks, die ganz außen auf dem Drehtisch stehen.

Wenn Sie die Dynamiks nicht mit einem Haken oder einem Automatik-Haken auf den Drehtisch setzen, sollten Sie einen senkrecht stehenden Linearbeweger verwenden, um den Drehtisch von unten durch das Förderband, auf dem das Dynamik steht, zu heben. Dabei können Sie ausnutzen, dass sich die Elemente in TrySim durchdringen können.

Der Drehtisch selbst kann nicht gedreht werden.

Der Drehtisch wird zwar als Achteck dargestellt, intern ist er aber quaderförmig wie alle anderen Elemente. Er wird also auch Dynamiks bewegen, die sich in den abgeschnittenen, nicht sichtbaren, Ecken befinden.

Das Beispiel “Drehungen” demonstriert den Gebrauch dieses Elementes.

Andere Möglichkeiten Dynamiks zu drehen ergeben sich durch die Verwendung eines Gelenks in Verbindung mit einem Haken, einer drehbaren Platte oder einem drehbaren Förderband.

Bogen-Förderband

***

Dies ist ein Element, das nur mit drehbaren Dynamiks funktioniert. Die Bewegungsrichtung geben Sie auf der Editiermaske vor. Mit der Checkbox “Revers” wird die Förderrichtung umgekehrt. Das Bogenförderband bewegt die Ecken aller drehbaren Dynamiks zwar exakt im Kreisbogen (das können Sie mit einem sehr kleinen Dynamik ausprobieren), aber durch das unvermeidliche Rutschen beim Übergang von und nach geraden Förderbändern sind die Dynamiks nach der Drehung meistens nicht mehr rechtwinklig ausgerichtet. Durch Anpassung der Geschwindigkeit (höher) lässt sich dieser Effekt weitgehend kompensieren. Eine verbleibende Drehung der Dynamiks können Sie mit dem Ausrichter korrigieren.

Wenn das Band um weniger als 90 Grad fördern soll, können Sie ein drehbares Förderband nach dem gewünschten Drehwinkel etwas höher anordnen. Dann laufen die Dynamiks auf diesem weiter und vollenden die 90 Grad nicht.

Das Bogenförderband ist immer noch nicht ganz ausgereift. Vor allem die Darstellung als Quader ist in diesem Fall besonders bei Bändern mit großem Radius sehr störend. Das Förderband selbst kann nicht gedreht werden.

Das Beispiel “Drehungen” demonstriert den Gebrauch dieses Elementes.

Drehbares Förderband

***

Dies ist ein Förderband, das Sie in alle Richtungen drehen können. Gut funktioniert es nur mit drehbaren Dynamiks.

Sie können das drehbare Förderband an einem Gelenk befestigen, um so einen Schwenktisch zu erhalten oder, wenn Sie das Gelenk in der XY-Ebene ausrichten, eine Weiche, die nach oben oder unten führt. Sie können das Förderband auch im Editiermodus drehen, indem Sie lange mit Rechts auf es klicken und dann aus dem Kontextmenü “Drehen” wählen.

Mit der Maus können Sie die Größe des Bandes nur ändern, wenn es sich in der Ausgangsposition befindet..

Das drehbare Förderband braucht recht viel Rechenleistung.

Die Beispiele “Beispiel1” und “Drehungen” demonstrieren den Gebrauch dieses Elementes.

2.4.4 Bedienung

Taster/Schalter

*

Durch Taster und Schalter können Sie während der Simulationslaufzeit das Geschehen beeinflussen. Ihre Funktion ist genau wie die echter Taster/Schalter mit der Ausnahme, dass Sie echte Taster immer an einen Eingang der SPS anschließen müssen, während Sie in TrySim auch Ausgänge und Merker direkt beeinflussen können. Wenn Sie z.B. während des Bauens Ihrer Anlage einen Motor sofort testen wollen, erzeugen Sie einfach einen Taster und schalten damit den Ausgang, an dem der Motor hängt.

Schalter/Schließer/Öffner

Diese drei Elemente sind in Wirklichkeit ein und das selbe Element, es wird in der Auswahlliste nur in drei Ausführungen angezeigt, um Ihnen das nachträgliche Umkonfigurieren zu ersparen.
Der Schalter ändert den Zustand des Bits jedesmal, wenn Sie auf ihn klicken.
Der Schließer setzt den Zustand des Bits auf “1” solange Sie auf ihn klicken, sonst setzt er den Zustand des Bits auf “0”. Klicken Sie nicht zu kurz, sonst erkennt der Schließer den Klick manchmal nicht.
Der Öffner setzt den Zustand des Bits auf “0” solange Sie auf ihn klicken, sonst setzt er den Zustand des Bits auf “1”. Öffner lassen sich gut verwenden um einen Sicherungsfall zu simulieren.
Sie können die Elemente jederzeit umkonfigurieren, indem Sie das Feld “Typ” der Editiermaske entsprechend ändern.

Bitte beachten Sie, dass Sie mit einem Schalter/Schließer/Öffner nicht ein auch vom SPS-Programm gesetztes Bit überschreiben können. Falls sich dieses Element nicht wie erwartet verhält, sollten Sie mittels der Querverweisliste überprüfen, ob das zu steuernde Bit auch im SPS-Programm modifiziert wird. Am Schnellsten ist die QL aufzurufen, wenn Sie auf der Editiermaske des Schalters auf den Button “QL” klicken. In der QL sind alle Stellen, an denen das Bit modifiziert wird, durch einen roten Punkt gekennzeichnet.

LED

*

Die Leuchter funktionieren genau wie echte Leuchtmelder, sie brennen nur nicht so schnell durch. Sie können Leuchter nicht nur an Ausgänge, sondern auch an Eingänge und Merker anschließen und überall an der Maschine plazieren. Dies ist

sehr nützlich, um verborgene Informationen übersichtlicher als in der Status-Variablen-Liste anzuzeigen. Mit der Checkbox **Original** auf der Editiermaske kann eingestellt werden, ob der Melder bei Zoom-Änderungen seine Größe beibehält.

Blinker
Diese Sonderform des Leuchtmelders blinkt mit einer festen Frequenz von 1 Hz. Verwenden Sie ihn, wenn das Blinken eines Leuchtmelders, den Sie von der SPS aus blinken lassen wegen der in TrySim variablen Simulationsgeschwindigkeit schlecht zu sehen ist. Der Blinker ändert alle 0,5 s seinen Zustand, wenn entweder das an ihn angeschlossene Bit "1" ist, oder wenn in den letzten 0,5 sec eine Zustandsänderung des Bits aufgetreten ist. Wenn das Bit "0" ist und in den letzten 0,5 sec keine Zustandsänderung aufgetreten ist, wird der Blinker ausgeschaltet. Siehe auch: Taktmerker.

Digitalanzeige

*
Hiermit können Sie jedes beliebige Byte, Word oder DWord der SPS in verschiedenen Formaten anzeigen lassen.

Die Digitalanzeige ist zugleich auch Digitaleingabe: durch einen Doppelklick während der Simulationslaufzeit schalten Sie in den Eingabemodus und geben die gewünschte Zahl ein.

Eine andere Möglichkeit, analoge Werte darzustellen, besteht in der Verwendung des Schiebereglers oder des Oszillographen.

Falls Sie die Größe des anzuzeigenden Wertes (Byte, Word oder DWord) ändern wollen, so wählen Sie bitte zunächst die gewünschte Größe und geben dann erst den neuen Operanden ein. Das ist lästig, aber wir sind noch nicht dazu gekommen dies komfortabler zu machen.

· Datenformate für Digitalanzeige

Wenn Sie keinen speziellen Zweck verfolgen, sollten Sie bevorzugt INT, DINT oder REAL verwenden.

BIT, BOOL: der Zustand des Bits wird als 0 (aus, falsch) oder 1 (ein, wahr) dargestellt. Im Allgemeinen ist ein Leuchtmelder zur Darstellung eines Bits besser geeignet.

Unsigned BYTE (1Byte) : Der Bereich geht von 0 bis 255
Nützliche SPS-Befehle: INC und DEC

Signed BYTE (1Byte): Der Bereich geht von -128 bis +128. In FUP, KOP und AWL hat dieser Datentyp keine Entsprechung.

Hex - BYTE (1Byte): Der Bereich geht von 00 bis FF, erlaubt sind die Ziffern von 0

bis 9 sowie die Buchstaben von A bis F. Wenn Sie sich auf die Ziffern beschränken, können Sie diesen Typ auch als 2 stelligen BCD verwenden.

BYTE als Bitmuster (1Byte): Der Bereich geht von 0000 0000 bis 1111 1111. Erlaubt sind nur die Ziffern 0 und 1. Bei der Eingabe können Sie Leerzeichen und "_" eingeben, diese werden anschließend jedoch nicht dargestellt.

WORD (2Byte): Der Bereich geht von 0 bis 65.535.

INT (2Byte): Der Bereich geht von -32.768 bis +32.767.
Nützliche SPS-Befehle: +I, -I, *I und /I

Hex WORD (2Byte): Der Bereich geht von 0000 bis FFFF. Erlaubt sind die Ziffern von 0 bis 9 sowie die Buchstaben von A bis F. Wenn Sie sich auf die Ziffern beschränken, können Sie diesen Typ auch als 4 stelligen BCD ohne Vorzeichen verwenden

3 stellige BCD mit Vorzeichen (2Byte): Der Bereich geht von -999 bis +999. Die Zahl wird als negativ interpretiert, wenn die vier höchstwertigen Bits gleich 1 sind.
Nützliche SPS-Befehle: BTI und ITB. Mit BCD-Zahlen kann man nicht rechnen

WORD als Bitmuster (2Byte): Der Bereich geht von 0000 0000 0000 0000 bis 1111 1111 1111 1111. Erlaubt sind nur die Ziffern 0 und 1. Bei der Eingabe können Sie Leerzeichen und "_" eingeben, diese werden anschließend jedoch nicht dargestellt.

DWORD (4Byte): Der Bereich geht von 0 bis 4.294.967.295.

DINT (4Byte): Der Bereich geht von –2.147.483.648 bis 2.147.483.647.
Nützliche SPS-Befehle: +D, -D, *D und /D

Hex - DWORD : Der Bereich geht von 0000 0000 bis FFFF FFFF. Erlaubt sind die Ziffern von 0 bis 9 sowie die Buchstaben von A bis F. Bei der Eingabe können Sie Leerzeichen und "_" verwenden, diese werden anschließend jedoch nicht dargestellt.

7-stellige BCD mit Vorzeichen (4Byte): Der Bereich geht von -9.999.999 bis +9.999.999. Die Zahl wird als negativ interpretiert, wenn die vier höchstwertigen Bits gleich 1 sind.
Nützliche SPS-Befehle: BTD und DTB. Mit BCD-Zahlen kann man nicht rechnen.

REAL (4Bytes): Fließkommazahlen
Nützliche SPS-Befehle: +R, -R, *R, /R, DTR und RND

String-Anzeige

**

Mit der String-Anzeige können Sie eine Zeile des Datentyps STRING der SPS ausgeben lassen. Geben Sie als SPS-Byte das erste Byte des Strings an.

Verwechseln Sie die String-Anzeige bitte nicht mit der Textanzeige, die eine ganz andere Funktion hat. Auch die ganz normale DigitalAnzeige hat nichts mit der String-Anzeige zu tun.

Wir haben diese Anzeige nur gemacht für den in einfachen S7-Programmen fast nie verwendeten Datentyp STRING. Man kann sie natürlich auch für die allgemeine Darstellung von Zeichenketten verwenden, aber dafür müssen Sie sich wenigstens kurz über diesen Datentyp informiert haben.

Schieberegler

*

Dies ist ein mausbetätigtes Eingabegerät, dessen Einstellung in eine beliebige Byte, Word oder DWord Variable der SPS in verschiedenen Formaten geschrieben wird.

Sie können den Schieberegler, nachdem Sie ihn angeklickt haben, auch mit der Tastatur bedienen:
Mit den Bild-Rauf/Runter-Tasten bewirken Sie eine Verstellung um jeweils einen der angezeigten Teilstriche. Eine Feinjustierung ist mittels der Pfeil Rechts/Links-Tasten möglich.

Steht der Schieberegler senkrecht, ist die Minimum-Stellung unten, liegt er waagerecht, ist sie links.

Der Schieberegler lässt sich auch als Anzeige verwenden - transferieren Sie dazu einfach einen Wert in die angeschlossene SPS-Variable.

Eine andere Möglichkeit, analoge Werte vorzugeben ist die Digitalanzeige, die sich auch als Digitaleingabe verwenden lässt.

Falls Sie die Größe des einzugebenden Wertes (Byte, Word oder DWord) ändern wollen, so wählen Sie bitte zunächst die gewünschte Größe und geben dann erst den neuen Operanden ein. Das ist lästig, aber wir sind noch nicht dazu gekommen dies komfortabler zu machen.

· Datenformate für Schieberegler

Wenn Sie keinen speziellen Zweck verfolgen, sollten Sie bevorzugt INT, DINT oder REAL verwenden.

Unsigned BYTE (1Byte) : Der Bereich geht von 0 bis 255
Nützliche SPS-Befehle: INC und DEC

Signed BYTE (1Byte): Der Bereich geht von -128 bis +128. In FUP, KOP und AWL hat dieser Datentyp keine Entsprechung.

2 stellige BCD (1Byte): Die Position des Schiebereglers wird in eine zweistellige

BCD-Zahl von 0 - 99 übersetzt. Wenn Sie diesen Wert mit der Digitalanzeige darstellen lassen wollen, können Sie dort das Format Hex-Byte verwenden.

WORD (2Byte): Der Bereich geht von 0 bis 65.535.

INT (2Byte): Der Bereich geht von -32.768 bis +32.767.
Nützliche SPS-Befehle: +I, -I, *I und /I

3 stellige BCD mit Vorzeichen (2Byte): Der Bereich geht von -999 bis +999. Die Zahl wird als negativ interpretiert, wenn die vier höchstwertigen Bits gleich 1 sind.
Nützliche SPS-Befehle: BTI und ITB. Mit BCD-Zahlen kann man nicht rechnen

DWORD (4Byte): Der Bereich geht von 0 bis 4.294.967.295.

DINT (4Byte): Der Bereich geht von –2.147.483.648 bis +2.147.483.647.
Nützliche SPS-Befehle: +D, -D, *D und /D

7-stellige BCD mit Vorzeichen (4Byte): Der Bereich geht von -9.999.999 bis +9.999.999. Die Zahl wird als negativ interpretiert, wenn die vier höchstwertigen Bits gleich 1 sind.
Nützliche SPS-Befehle: BTD und DTB. Mit BCD-Zahlen kann man nicht rechnen.

REAL (4Bytes): Fließkommazahlen
Nützliche SPS-Befehle: +R, -R, *R, /R, DTR und RND

Textanzeige

**

Die Textanzeige entspricht einem einfachen Operator-Panel, mit dem Sie Fehlermeldungen ausgeben können.

Auf der Editiermaske befindet sich eine Tabelle zur Eingabe von Störungsmerkern und der dazugehörigen Meldung. Wenn der Merker in der Symboltabelle eingetragen ist, wird als Meldung der Kommentar vorgeschlagen, er kann aber geändert werden. Während der Laufzeit erscheint die Textanzeige als einzeiliges Feld, auf dem immer die zuletzt aufgetretene Störung angezeigt wird. Durch Klick auf das Feld erscheint eine größere Maske, auf der alle zur Zeit anstehenden Meldungen sichtbar sind. Ein Button, dem zuvor ein SPS-Bit zugewiesen werden muss, dient als “Störung quittieren”-Taster.

Verwechseln Sie die Text-Anzeige nicht mit der String-Anzeige, die eine ganz andere Funktion hat.

Oszillograph

*

Dies ist ein Element, mit dem Sie den zeitlichen Verlauf von Bits und analogen Werten (16 Bit) darstellen können. Für die Darstellung der analogen Werte müssen

Sie den Minimal- und den Maximalwert geeignet anpassen. Wenn der Minimalwert kleiner als 0 ist, wird das SPS-Word als Integer interpretiert und der Maximalwert entsprechend auf 32.767 begrenzt.

Wenn Sie DINTs oder REALs darstellen wollen, müssen Sie sie im SPS-Programm in Wörter umrechnen. Dazu können Sie die FC51 oder FC52 (DINT und REAL) im Verzeichnis IECFuncs verwenden. Plazieren Sie solche Programmteile, die nur in der Simulation benötigt werden vorzugsweise im OB 2 oder OB 3.

Mit der Einstellung Time/Div können Sie die Darstellung in der Zeitachse strecken oder stauchen. Normalerweise ist keine Veränderung dieses Wertes notwendig. Verwenden Sie diese Einstellung zusammen mit Veränderung der Größe des Oszillographen, um eine Ihren Gegebenheiten angepasste Darstellung zu erreichen.

Mit der Einstellung "Rate" können Sie die Aufzeichnungsrate der Daten vorgeben. Eine Zeit, die kürzer als die Simulationsrate ist, hat keine Wirkung. Je größer Sie diesen Wert stellen, desto geringer ist die Auflösung für kurze Ereignisse, aber desto weiter zurück in die Vergangenheit reicht die Aufzeichnung. Beachten Sie den Unterschied zur Einstellung Time/Div, hierdurch wird nur die Darstellung der bereits aufgezeichneten Werte auf dem Bildschirm beeinflusst.

Bislang gibt es noch keine Triggermöglichkeiten oder weitere Einstellungen, beachten Sie aber in diesem Zusammenhang den Speed-Trigger.

Zum Löschen einer Zeile in der Tabelle der anzuzeigenden Operanden betätigen Sie “Strg+Entf.”

Stufenschalter

***

Dieses Element müssen Sie mindestens paarweise einsetzen. Alle Stufenschalter, die aneinander befestigt sind, bilden eine Einheit, von denen jeweils nur einer aktiv sein kann. Wenn Sie während der Laufzeit einen davon anklicken, werden alle anderen deaktiviert (bei Öffnern bedeutet aktiviert hier, dass das Bit “0” ist).

Es empfiehlt sich, den Kontakttyp aller Stufenschalter auf “Schliesser” zu belassen. Gerade noch überschaubar bleibt es, wenn alle den Kontakttyp “Öffner” bekommen. Sie können aber jede Kombination wählen und auch den Typ “Schalter” einstellen, eine sinnvolle Anwendung dafür ist uns jedoch noch nicht eingefallen.

Auf der Editiermaske können Sie einstellen, wie das Schaltverhalten sein soll:

- Instantan: für die SPS wird von einem Zyklus zum nächsten umgeschaltet
- Durch Null: Für mindestens einen Zyklus sind alle Schalter deaktiv
- Überschneidend: Erst wenn der neue Schalter aktiv ist, wird der alte deaktivert.

Wir empfehlen dringend, das Schaltverhalten für alle Stufenschalter einer Einheit identisch einzustellen. Welche Auswirkungen die Einstellung des Schaltverhaltens hat sehen Sie am besten, wenn Sie den Oszillograph verwenden, darauf eine Time/Div von 0,2 wählen und extreme Zeitlupe einstellen.

Mit welchen anderen Stufenschaltern ein bestimmter verbunden ist, sehen Sie am besten im Elementbaum. Beachten Sie, dass nur solche Schalter als aneinander befestigt gelten, die einen gemeinsamen (Groß-)Vater haben, der ebenfalls Stufenschalter ist. Zur Übersichtlichkeit ist es zu empfehlen, im Elementbaum alle Stufenschalter, die zu einer Einheit gehören, direkt an diesen ausgezeichneten Schalter zu hängen (man erkennt ihn leicht daran, dass sein Vater kein Stufenschalter ist). Das hört sich komplizierter an, als es ist.

Bei der Verwendung von Stufenschaltern sollten Sie unbedingt das Schaltverhalten Ihres tatsächlich eingesetzten Schalters beachten. Stellen Sie sich Fragen wie: Hat er überschneidende Kontakte? Welche Zwischenzustände können auftreten? Prüfen Sie vorher, wie sich Ihr Programm beim Umschalten verhält, das ist eine ganz eklige Fehlerquelle.

Meisterschalter

**

Der Meisterschalter hat das Aussehen eines Schiebereglers mit bis zu 16 Schaltstellungen. Auf der Editiermaske können Sie bis zu 16 Bits angegeben und die Betätigung dieser Bits für jede Schaltstellung frei vorgeben. Geben Sie auf der Editiermaske zunächst an, wie viele Schaltstellungen Sie wünschen und welche Bits angeschlossen werden sollen. Betätigten Sie sodann den Button “Kontakte”: es erscheint eine Matrix der Größe Stellungsanzahl x Bit-Anzahl. Klicken Sie in dieser Matrix an, welches Bit bei welcher Schaltstellung betätigt werden soll.

Wenn der Meisterschalter senkrecht steht, ist die Schaltstellung “1” unten, liegt er waagerecht, ist die Schaltstellung “1” links.

Wenn Sie ein neues Bit in die Tabelle eingetragen haben, denken Sie daran, dass Sie mit dem Cursor die neue Zeile verlassen müssen (z.B. mit der “Pfeil runter”-Taste), damit der Operand übernommen wird.

Wenn Sie die Schaltstellung vom SPS-Programm aus manipulieren wollen, müssen Sie einen Word-Poke verwenden.

· Kontakt-Editor des Meisterschalters

**

Auf dieser Maske legen Sie fest, welche Bits bei welcher Schaltstellung des Meisterschalters betätigt werden sollen.

Bevor Sie diese Maske ausfüllen, sollten Sie auf der Editiermaske des Meisterschalters festlegen, wie viele Schaltstellungen er haben soll und welche Bits an ihn angeschlossen werden sollen. Diese Maske wird danach die passende Anzahl von Feldern haben.

Wenn Sie ein neues Bit in die Tabelle eingetragen haben, denken Sie daran, dass

Sie mit dem Cursor die neue Zeile verlassen müssen (z.B. mit der “Pfeil runter”-Taste) damit der Operand übernommen wird.

2.4.5 Flüssigkeiten

Allgemein

**

Ab Version 2.3 gibt es Elemente zur Simulation von Flüssigkeiten. Wir haben uns bemüht, das Verhalten der Flüssigkeiten möglichst realistisch zu gestalten, haben dabei aber z.T. sehr starke Vereinfachungen vornehmen müssen. Unser Augenmerk hat darauf gelegen, dem SPS-Programmierer die Möglichkeit zu geben, Programme für Anlagen mit Behältern, Rohren, Pumpen und Ventilen zu testen, es ist nicht unser Ziel gewesen, Hilfen für die Auslegung solcher Anlagen zu geben.

Die Flüssigkeiten in TrySim bestehen aus einer Mischung von bis zu 16 Medien. Jedes Medium hat einen Namen, eine Dichte und eine Viskosität. Die resultierenden Eigenschaften der Mischung werden durch eine nach Anteil gewichtete Mittelwertbildung der einzelnen Medien bestimmt. Falls dies eine zu starke Vereinfachung ist, haben Sie die Möglichkeit, durch einen Reaktor Medien ineinander umzuwandeln und so die gewünschten Eigenschaften herbeizuführen.

Den Bau einer Anlage beginnen Sie am Besten mit einem Behälter. Auf dessen Editiermaske klicken Sie die Checkbox “Quelle” an, dann ist er immer gefüllt, egal wie viel Flüssigkeit Sie entnehmen. Dann installieren Sie weitere Behälter und verbinden Sie miteinander durch Pumpen, Ventile und Rohre. Der Einfachheit halber sind die Pumpen und Ventile bereits mit einem geraden Rohr verbunden. Nur dann, wenn Sie um Ecken herum wollen, müssen Sie explizit ein Rohr legen. Die Enden dieser Elemente werden entweder an einen Behälter angeschlossen, oder mittels eines Flansches miteinander verbunden. An einen Flansch können auch mehrere Rohre angeschlossen werden. Die geometrischen Verhältnisse müssen Sie berücksichtigen, d.h., wenn Sie einem halbvollen Behälter mittels einer Pumpe Flüssigkeit entnehmen wollen, muss die Pumpe an der unteren Hälfte des Behälters angeschlossen werden, sonst pumpt sie nur Luft.

Mittels des Füllstandssensors, des Niveauschalters, des Durchflussmessers, des Drucksensors und des Analysators erhält Ihre SPS die notwendigen Information zur Steuerung der Anlage.

Der Fluidor wandelt Dynamiks in Flüssigkeit um, löst sie sozusagen auf.

Behälter

**

Die Behälter in TrySim sind quaderförmig. Ihr Fassungsvermögen bestimmt sich aus den geometrischen Abmessungen. In jedem Behälter befindet sich eine Mischung aus verschiedenen Medien, deren Menge Sie mit dem Füllstandssensor und dem Niveau-Schalter und deren Zusammensetzung Sie mit dem Analysator detektieren

können. Den Druck in einer bestimmten Höhe können Sie mit dem Drucksensor erfassen.
Auf der Editiermaske wird der Füllstand in % durch einen Schieberegler angegeben und Sie können ihn dort auch verändern. Wenn Sie eine unerschöpfliche Quelle, z.B. einen Wasserhahn benötigen, klicken Sie die Checkbox “Quelle” an, dann wird der Behälter immer gefüllt sein, egal wieviel Flüssigkeit Sie ihm entnehmen. Wenn Sie einen Abfluss benötigen, klicken Sie die Checkbox “Senke” an, dann ist der Behälter immer leer, egal wieviel Flüssigkeit Sie in ihn hinein pumpen.
Die aktuell im Behälter befindliche Mischung wird auf der Editiermaske durch vier Prozentzahlen angegeben. Wenn Sie die komplette Mischung, die aus bis zu 16 verschiedenen Medien bestehen kann, sehen wollen, klicken Sie auf den Button “Alle Medien”. Die Reihenfolge, in der die Anteile dargestellt werden sollen, können Sie für jeden Behälter einzeln festlegen.Wählen Sie dazu das für jeden Platz gewünschte Medium durch Klicken auf den nach unten weisenden Pfeil neben dem Feld der Medien-Beschreibung.
Ein neues Medium erzeugen Sie, indem Sie entweder aus der Auswahlliste “Neues Medium” wählen, oder indem Sie den Button Medien-Manager betätigen und “Neu” anklicken.

Wenn Sie mehrere Behälter gleichzeitig editieren, können Sie die Reihenfolge der Medien und die Mischungszusammensetzung nicht verändern.

Damit Sie den Flüssigkeitsspiegel in den Grafiken sehen können, müssen Sie die “XZ-Vorderansicht” oder “YZ-Seitenansicht” wählen. Es ist eine häufige Frage an die Hotline: “Warum sehe ich den Flüssigkeitsspiegel nicht?”.

Rohr

**

Ein Rohr besteht aus beliebig vielen Segmenten, die miteinander verbunden sind. Wenn Sie ein Rohr erzeugt haben, besteht es zunächst nur aus einem Segment. Weitere Segmente erzeugen Sie, indem Sie auf der Editiermaske “Segment teilen” wählen. Die einzelnen Segmente können Sie als Ganzes mit der Maus verschieben, indem Sie sie in der Mitte packen, oder Sie können die Orientierung eines Segments ändern, indem Sie den Mauszeiger so nah an ein Ende positionieren, dass er sich in ein Kreuz verwandelt.

Die Nennweite des Rohres sollten Sie auf der Maske der erweiterten Rohreigenschaften einstellen.

Die Enden des Rohres können Sie an einen Behälter oder an einen Flansch anschließen. Der Anschluss geschieht einfach dadurch, dass das Ende im Behälter oder Flansch positioniert wird. Dabei brauchen Sie nicht allzu genau vorzugehen, beim Starten der Simulation werden nicht angeschlossene Rohrenden erkannt und nach Rückfrage an den nächstgelegensten Punkt angeschlossen. Denken Sie aber an die Dreidimensionalität der in TrySim simulierten Welt! Das Kreuz und der Streifen sind nützliche Hilfen beim Bauen der Anlage.

An das Rohr können Sie einen Durchflussmesser oder einen Analysator

anschließen.

Rohre und alle rohrähnlichen Elemente können auch zum Befüllen von füllbaren Dynamiks verwendet werden. In diesem Fall sollten Sie die Option “Warnen, wenn nicht angeschlossen” auf der Maske “Erweitert” abschalten.

Wenn ein Rohrende nicht angeschlossen ist, verschwindet die Flüssigkeit, die dort heraus läuft. Mindestens ein Rohrende muss aber an einen Topf oder einen Flansch angeschlossen sein, sonst wird das Rohr bei der Simulation nicht bearbeitet.

· Erweiterte Rohr-Eigenschaften

**

Nennweite

Die Elemente Rohr, Pumpe, Ventil und Rückschlagventil haben eine Länge, die durch ihre geometrischen Abmessungen bestimmt ist, sowie eine Nennweite, die Sie auf der Editiermaske festlegen müssen. Diese beiden Werte werden, zusammen mit der Viskosität, der Dichte und der Strömungsgeschwindigkeit, verwendet, um den Strömungswiderstand zu berechnen oder, genauer gesagt, grob abzuschätzen. Er hängt nämlich noch von vielen weiteren Faktoren ab und die alle zu erfassen würde den Rahmen von TrySim sprengen. Wir hätten auf die Berechnung des Strömungswiderstandes gerne ganz verzichtet, aber es hat sich gezeigt, dass ein ausreichend realistisches Verhalten der Flüssigkeiten ohne ihn nicht zu erreichen ist. Daher ist es **wichtig**, dass Sie die **Nennweite** aller o.g. Elemente zumindest größenordnungsmäßig Ihrer Anlage **anpassen**. Wenn Sie z.B. zwei Behälter mit je 1 Liter Inhalt durch ein 100er-Rohr verbinden, werden Sie keine vernünftigen Ergebnisse erzielen.
Unter **Ansicht|Optionen|Anlage** können Sie den Wert einstellen, der für neue Elemente verwendet wird.

Anzahl Abschnitte

Innerhalb von TrySim wird jedes Rohr behandelt, als sei es aus mehreren (mindestens 2) Abschnitten zusammengesetzt. Für viele Anwendungen wird es reichen, die Voreinstellung von 2 Abschnitten beizubehalten, wenn Sie jedoch z.B. einen Saugheber* simulieren wollen, dann müssen Sie diese Anzahl erhöhen. Eine weitere Notwendigkeit zur Erhöhung der Anzahl Abschnitte ist gegeben, wenn es für Ihre Anwendung wichtig ist, dass die Rohre ein endliches Volumen haben und dass sich die Zusammensetzung der Flüssigkeit entlang des Rohres ändern kann. Wählen Sie die Anzahl jedoch nicht höher als notwendig, denn dies geht zu Lasten der Simulationsgeschwindigkeit.

Warnen, wenn nicht angeschlossen

Normalerweise wird beim Starten der Simulation gewarnt, wenn ein Rohrende nicht an einen Behälter oder Flansch angeschlossen ist. Wenn Sie aber Dynamiks befüllen wollen, die ja i.A. erst während der Simulation zur Füllstelle kommen, ist diese Kontrolle sehr lästig. Durch das Abwählen dieser Checkbox wird die Kontrolle abgeschaltet.

* Gemeint ist hier das Verfahren, wenn man beispielsweise mit einem Schlauch das Wasser aus einem Aquarium ablässt, nachdem man kurz angesaugt hat.

Pumpe

**

Pumpen in TrySim werden durch den maximalen Förderdruck und die Förderleistung beschrieben. Der maximale Förderdruck wird auf der Editiermaske eingegeben, die Förderleistung wird durch den Antrieb bestimmt. Bislang stehen zwei Antriebe zur Verfügung: die Bit- und die Word-Pumpe. Der in der Auswahlliste erscheinende Antrieb “Linearbeweger” ist eigentlich für das Ventil gedacht, aber er funktioniert auch bei der Pumpe und es kann damit eine Pumpe mit variabler Leistung und Motor-Poti simuliert werden.

Eine Pumpe ist automatisch mit einem (nicht knickbaren) Rohr verbunden, dessen Eigenschaften Sie über den Button Erweitert editieren können. Zumindest die Nennweite des Rohrs sollten Sie an die tatsächlichen Verhältnisse anpassen.

Wie die Enden des Rohrs angeschlossen werden und welche Sensoren angeschlossen werden können, steht in der Beschreibung des Rohrs.

Wenn Sie eine negative Leistung angeben oder bei der Word-Pumpe von der SPS aus einen negativen Sollwert vorgeben, wird die Förderrichtung umgekehrt.

Die Pumpe fördert immer die durch den Antrieb vorgegebene Menge, soweit das bei dem eingestellten maximalen Druck möglich ist. Ist der Widerstand der nachfolgenden Elemente zu groß, wird nur soviel Flüssigkeit gefördert, wie es beim maximalen Druck möglich ist. Den maximalen Druck können Sie auch mittels eines Poke vom SPS-Programm modifizieren. Schreiben Sie dazu in das Word des Pokes den Druck in kPa (ca 0,01 bar), für einen max. Druck von 8 bar müssen Sie also den Wert 800 poken.

Ventil

**

Ein Ventil begrenzt die hindurchfließende Menge. Wie groß diese Menge ist, wird wie bei der Pumpe durch den Antrieb festgelegt. Viele reale Ventile oder Schieber werden durch einen Strahlantrieb betätigt. Dementsprechend können Sie als Antrieb auch einen Linearbeweger (den Sie natürlich vorher erzeugen müssen) wählen. Je nach Stellung des Hot-Spots des Linearbewegers wird das Ventil von 0 - 100 % auf- und zugefahren.

Auch beim Ventil sollten Sie unter den erweiterten Rohreigenschaften die Nennweite entsprechend den Verhältnissen einstellen.

Bitte beachten Sie, dass mit der hier beschriebenen Ventilart nicht solche pneumatischen oder hydraulischen gemeint sind, die zur Ansteuerung von Antrieben verwendet werden.

Überdruckventil

**

Dieses Element verhält sich wie ein nicht knickbares Rohr, das bei einem auf der Editiermaske einstellbarem Druck durchlässig wird.

Auf der Editiermaske können Sie einstellen:

1. “Max. Druck”
Das Ventil versucht, diesen Druck zwischen den beiden Enden in etwa auf diesen Wert zu begrenzen. Das Überdruckventil verhält sich in beiden Richtungen gleich.

2. “Fluss”
Hier müssen Sie eingeben, mit welchem Fluss das Ventil zu rechnen hat. Geben Sie lieber zu viel als zu wenig an. Der TrySim-Kernel braucht nur eine Idee davon, wieviel Flüssigkeit über das Überdruckventil abfließen wird (es gibt TrySim-Anwender, die in Kubikmetern rechnen und andere, die in Kubikmillimetern rechnen).

3. “Kennlinie”
Hiermit können Sie einstellen, wie genau die von Ihnen gewünschte Druckbegrenzung arbeiten soll. Für Standardanwendungen gilt:

1 - 5 Ungenauer Druck, aber gutmütig bei der Simulation.

6 - 14 Der Druck wird recht genau eingehalten, aber bei komplexeren Anlagen kann die Simulation sehr langsam werden oder sogar abbrechen.

15 - 30 In diesem Bereich wird der Druck exakt eingehalten, aber wenn die anderen Elemente nicht realistisch konfiguriert sind, wird die Simulation garantiert abbrechen.

Die Erweiterten Rohreigenschaften sind beim Überdruckventil nicht nötig und darum nicht verfügbar.

Rückschlagventil

**

Dieses Element verhält sich wie ein nicht knickbares Rohr, das nur in einer Richtung durchlässig ist.

Auch beim Rückschlagventil sollten Sie unter den erweiterten Rohreigenschaften die Nennweite entsprechend den Verhältnissen einstellen.

Flansch

**

Der Flansch dient zur Verbindung von zwei oder mehreren Rohrenden. Dabei ist es gleichgültig, ob dies Enden von einem normalen Rohr, einer Pumpe oder einem anderen rohrähnlichen Element sind. Sie schließen ein Rohr an einen Flansch an,

indem Sie das Ende einfach innerhalb des Flansches positionieren. Dabei brauchen Sie nicht übermäßig genau zu sein, beim Starten der Simulation werden alle Rohrenden überprüft und gegebenenfalls nach Rückfrage an den nächsten Flansch angeschlossen.

Bei der internen Berechnung der Flüsse durch einen Flansch wird dieser eher wie ein Druckausgleichsbehälter betrachtet, der auch ein gewisses Volumen zwischenspeichern kann. Der auf der Editiermaske ziemlich vage als “Faktor” bezeichnete Wert gibt an, wie “hart” dieser Druckausgleichsbehälter ist. Normalerweise können Sie diesen Wert auf “1” stehen lassen. Für manche Anwendungen (besonders wenn mit hohen Drücken gearbeitet wird) muss der Wert jedoch erhöht werden, um ein schnelles Reagieren auf das Schalten von Ventilen und Pumpen zu erreichen. Wenn Sie ihn aber zu groß machen, wird die Simulation instabil, alle Flüsse und Drücke zeigen dann ein chaotisches Verhalten. Natürlich würden wir einen angemessenen Wert für diesen Faktor gerne aus den Daten der an den Flansch angeschlossenen Rohre, Pumpen und Ventile berechnen, das ist uns aber noch nicht gelungen und darum müssen Sie diesen Wert ggfs. selbst anpassen.

Durchflussmesser

**

Den Durchflussmesser müssen Sie einem Rohr oder einem rohrähnlichen Element zuordnen, indem Sie ihn auf der Linie positionieren. Dabei brauchen Sie nicht übermäßig genau zu sein, beim Starten der Simulation werden alle Durchflussmesser überprüft und ggfs. nach Rückfrage auf das nächstgelegene Rohr verschoben.

Der Durchflussmesser kann den Fluß pro Zeit und den integrierten Fluß bestimmen. Wenn er den integrierten Fluß bestimmt, wird das SPS-Word nur incrementell verändert, d.h., Sie können den Wert von der SPS aus überschreiben, um ihn z.B. zurückzusetzen.

Falls Sie Flansche verwenden, müssen Sie evtl. den dort beschriebenen Faktor erhöhen, um eine befriedigende Durchflussmessung durchführen zu können.

Füllstandssensor

**

Den Füllstandssensor müssen Sie innerhalb eines Behälters positionieren. Er schreibt den Füllstand in den auf der Editiermaske vorgebbaren Einheiten und Auflösungen in ein Word der SPS.

Wenn Sie die Einheit kg wählen, wird auch die Dichte der Mischung in dem Behälter berücksichtigt, d.h., der Sensor verhält sich wie eine Waage.

Sie können mit diesem Element den Füllstand von füllbaren Dynamiks erfassen. In diesem Fall sollten Sie die Option “Warnen, wenn nicht angeschlossen” abschalten.

Drucksensor

**

Mit dem Drucksensor können Sie den Druck in einem Behälter oder einem rohrähnlichen Element bestimmen.

Wenn der Drucksensor in einem Behälter angeordnet ist, bestimmt er den Druck auf der Höhe seiner unteren Fläche.

Wenn der Drucksensor auf einem Rohr liegt, wird der Druck durch lineare Interpolation zwischen dem Druck am Rohranfang und am Rohrende bestimmt. Falls es sich um ein in Z-Richtung geknicktes Rohr handelt und der Schweredruck eine signifikante Rolle spielt, ist dies eine grobe Vereinfachung und die Ergebnisse sollten mit Vorsicht betrachtet werden.

Wenn der Drucksensor auf einer Pumpe, einem Ventil oder einem Rückschlagventil liegt, wird der Druck am näher liegenden Ende an die SPS gemeldet.

Falls Sie Flansche verwenden, müssen Sie evtl. den dort beschriebenen Faktor erhöhen, um eine befriedigende Druckmessung durchführen zu können.

Analysator

**

Mit dem Analysator erfassen Sie den Anteil eines Mediums an der Mischung in einem Behälter, einem Rohr oder einem rohrähnlichen Element.

Welches Medium der Analysator erfassen soll, ist auf seiner Editiermaske einstellbar. Die Auflösung beträgt vorgabemäßig 1%, sie ist jedoch ebenfalls einstellbar.

Es ist nicht wesentlich, wo Sie den Analysator innerhalb eines Behälters positionieren, d.h., er muss nicht innerhalb der Flüssigkeit sein. Wenn Sie die Flüssigkeit in einem Rohr analysieren wollen, muss dieses durch den Analysator hindurchführen. Wenn ein Analysator weder einem Rohr noch einem Behälter zugeordnet ist, wird er beim Starten der Simulation nach Rückfrage auf das nächstgelegene geeignete Element verschoben.

Sie können mit diesem Element auch die Flüssigkeit in füllbaren Dynamiks analysieren. In diesem Fall sollten Sie die Option “Warnen, wenn nicht zugeordnet” abschalten.

Niveau-Schalter

**

Der Niveau-Schalter wird innerhalb eines Behälters positioniert. Er wird betätigt, wenn die Flüssigkeit seinen unteren Rand erreicht.

Sie können mit diesem Element auch das Niveau in füllbaren Dynamiks überprüfen. In diesem Fall sollten Sie die Option “Warnen, wenn nicht angeschlossen” abschalten.

Fluidor

**

Der Fluidor verwandelt Dynamiks in Flüssigkeiten. Sie müssen ihn in einem Behälter positionieren. Wenn ein Dynamik mit dem Fluidor überlappt, wird es in Flüssigkeit im Behälter umgewandelt. In welches Medium umgewandelt wird, können Sie auf der Editiermaske vorgeben. Sie können ebenfalls einen Umrechnungsfaktor vorgeben, falls das Dynamik dichter oder weniger dicht ist als die Flüssigkeit.

Ein Element mit der umgekehrten Funktionalität gibt es bislang noch nicht, ließe sich aber erstellen.

2.4.6 Sonstige

Säge

***

Mit der Säge können Sie zwei Dinge tun:
1. Abschneiden eines Stückes des vom kontinuierlichen Generator erzeugen Strangs.
2. Abschneiden eines Stückes von einem Dynamik (nicht von Drehbaren).

Abschneiden des Strangs

Sie können nur quer zur Bewegungsrichtung des Stranges schneiden. Das abgeschnittene Stück wird zu einem Dynamik, dessen Eigenschaften auf der Editiermaske des kontinuierlichen Generators unter “Dynamiks” festgelegt werden können.

Damit die Säge überhaupt eine Wirkung auf den Strang hat, müssen zwei Bedingungen erfüllt sein: 1.) muss sie mit dem Strang überlappen und 2.) muss das SPS-Bit der Säge auf “1” sein (oder es muss die Checkbox “Immer 1” angeklickt sein). Nur wenn beide Bedingungen zutreffen, wird die Säge im Folgenden “aktiv” genannt.

Um zu schneiden, müssen Sie die Säge durch den Strang hindurch bewegen. Der Punkt, mit dem die Säge tatsächlich schneidet, liegt genau in ihrer Mitte. Ein Schnitt wird begonnen, wenn die Säge das erste mal aktiv wird. Das Stück wird abgetrennt, wenn der erste und letzte Schnittpunkt auf gegenüberliegenden Seiten des Stranges liegen. Ein Schnitt gilt intern jedoch erst dann als vollständig beendet, wenn die Säge nicht mehr aktiv ist. Das Stück kann dann schon längst abgeschnitten sein, aber ein neuer Schnitt wird nur begonnen, nachdem die Säge zumindest kurzzeitig nicht aktiv war. Wenn die Säge inaktiv wird, bevor der Schnitt vollendet wurde, wird er verworfen, d.h. Sie können nicht absetzen und einen begonnenen Schnitt später

fortsetzen.

Einen Strang kann man mit zwei Orientierungen durchschneiden. Einen nach Osten laufenden Strang kann man z.B. von oben nach unten oder von Süd nach Nord durchschneiden. Mit welcher Orientierung geschnitten wird, wird in TrySim dadurch festgelegt, in welcher Richtung der Strang das erste Mal von der Säge vollständig durchsetzt wird. Um von oben nach unten zu schneiden, müssen Sie die Säge also etwas breiter als den Strang machen. Um von Süd nach Nord zu schneiden reicht es, wenn die Säge etwas dicker ist als der Strang.

Auch falls Sie einen schiefen Schnitt machen, werden sowohl das entstehende Dynamik als auch der Rest vom Strang rechteckig werden.

Abschneiden eines Stückes von einem Dynamik
Dies ist im Wesentlichen wie beim Abschneiden des Stranges. Lediglich für die Schnittrichtung und Orientierung gilt eine etwas andere Regel, weil bei einem Dynamik kein Generator vorhanden ist, der eine Richtung auszeichnet.
Diese Regel ist am Einfachsten zu beschreiben, wenn Sie die Säge wie einen Stab formen und sich vorstellen, es sei das Blatt einer Stichsäge: Auf der Seite, die zuerst einen über die gesamte Breite gehenden Anschnitt erhält, beginnt der Schnitt. Er endet erst, wenn das Sägeblatt auf der gegenüberliegenden Seite ausgetreten ist. Die Orientierung des Schnittes ergibt sich dann von selbst.

Dieses Element befindet sich noch im Versuchsstadium. Anregungen aus der Praxis zur Verbesserung sind willkommen.

Um ein Dynamik (keinen Strang) in gleiche Teile zu teilen, können Sie auch den Teiler verwenden.

Um von einem Block ein oder mehreren Platten abzuschneiden, verwenden Sie den Abschneider.

Teiler

***

Wenn das Bit des Teilers eine steigende Flanke hat, teilt er alle Dynamiks, mit denen er überlappt, in die auf der Editiermaske angegebenen Anzahl von kleineren Dynamiks. Die Anzahl der Schnitte pro Richtung können Sie einzeln vorgeben.

Nützlich sind Teiler z.B. wenn viele gleichartige Dynamiks gleichzeitig erzeugt werden müssen. Anstatt für die Lieferung von 12 gleichen Kartons 12 Generatoren zu erstellen, reicht einer, der das Volumen erzeugt und ein Teiler, der es in die gewünschten Portionen aufteilt.

Das mit der steigenden Flanke ist natürlich notwendig, damit der Teiler die neuen Dynamiks (zumindest die, mit denen er nach der Teilung noch überlappt) nicht in Quarks zersägt.

Diese Operation kann z.B. mit dem Verschmelzer wieder rückgängig gemacht werden. Zum separieren der Teile können Sie u.a. den Keil verwenden. Zum Herausgreifen einzelner Teile können Sie den Haken oder den Schieber verwenden.

Bei Verwendung des Teilers mit drehbaren Dynamiks muss bei Teilung in Z-Richtung daran gedacht werden, dass diese bislang noch nicht aufeinander stehen können.

Das Beispiel “Baumsäge” demonstriert den Gebrauch eines Teilers.

Abschneider

****

Der Abschneider ist aus folgender Situation geboren worden: Eine große Anzahl von Platten wird zu einem Stapel aufgeschichtet und dann transportiert. Während des Transports ist es vollkommen gleichgültig, dass der Stapel aus einzelnen Platten besteht. Er wird daher vorzugsweise durch ein großes kompaktes Dynamik dargestellt. Die Stapelung einzelner Platten macht in TrySim immer noch Schwierigkeiten und ist extrem rechenzeitaufwändig. Bei der Stapelbildung kann ein Verschmelzer gute Dienste leisten. An der Verwendungsstelle müssen aber wieder einzelne Platten zur Verfügung stehen. Dazu dient der Abschneider. Er schneidet eine oder mehrere Platten von vorgegebener Dicke ab, die dann abgeschoben, abgehoben oder sonstwie bearbeitet werden können.

Aktivierungsbit
Es ist wohl am Günstigsten, den Abschneider immer aktiv zu lassen. Bereits abgeschnittene Platten bearbeitet der Abschneider, anders als der Teiler, nicht mehr. Wir haben dieses Bit dennoch für unvorhergesehene Fälle beibehalten.

Plattendicke
Die Dicke der Platte kann von Ihnen entweder fest vorgegeben werden oder sie kann über ein SPS-Word variabel eingestellt werden. Über das Editierfeld “Auflösung” können Sie bestimmen, wie die Informationen von der SPS interpretiert werden sollen.

Um eine durch Fehlbedienung entstehenden Plattendicke < 1 mm zu vermeiden, haben wir die Dicke auf diesen Wert begrenzt. Bei kleineren Werten werden sonst Hunderte von Platten abgeschnitten, mit denen TrySim leider nicht klarkommt. Wenn Sie dünnere Platten brauchen, wenden Sie sich bitte an uns. Wir werden dann eine Lösung finden.

Abschneiderichtung
Sie können vorgeben, von welcher der sechs Seiten des Dynamiks abgeschnitten werden soll. Der Abschneider lässt sich bei X - Y - Richtungen dann auch prima als Zerschredderer von Platten verwenden.

Verschmelzer

***

Wenn das Bit des Verschmelzers auf “1” ist, fügt er alle Dynamiks, mit denen er überlappt, zu einem Block zusammen. Der Block wird so groß gemacht, dass er alle verschmolzenen Dynamiks umfasst. Das führt dazu, dass der entstehende Block größer als die Summe aller Dynamiks sein kann, wenn diese vorher nicht quaderförmig angeordnet waren.

Diese Operation kann nicht rückgängig gemacht werden, denn sie wird intern so ausgeführt, dass eines der Dynamiks auf die neue Größe aufgeblasen wird, während die anderen gelöscht werden. Sie können aber das neue große Dynamik mit dem Teiler wieder in kleinere Teile zerlegen.

Bitte lesen Sie aber auch die Info über den Abschneider, damit können ebenfalls einzelne Platten wieder abgeschnitten werden.

Das Beispiel “Palettierer” demonstriert den Gebrauch eines Verschmelzers.

Keil

**

Der Keil schafft einen Zwischenraum zwischen zwei nahe beieinander stehenden Dynamiks. Damit er seine ,,Keil-Wirkung" entfalten kann, müssen Sie ihn länglich formen.

Der Keil ist häufig nützlich nach der Anwendung eines Teilers oder einer Säge.

Wenn Sie ein Gatter aus Keilen machen, dann muss der Abstand zwischen zwei Keilen größer sein, als die Teile, die Sie trennen wollen. Wenn die Teile stecken bleiben, sollten Sie die entsprechende Stelle ganz nah heranzoomen. Die begrenzte Auflösung der Monitore kann da leicht täuschen!

Die Beispiele “Baumsäge” und “Weiche” demonstrieren den Gebrauch von Keilen.

Der Keil ist sicher noch verbesserungswürdig, mit den drehbaren Dynamiks funktioniert er fast gar nicht.

Schieber

**

Mit diesem Element können Sie Dynamiks schieben und stoppen. Es ist ein passives Element ohne SPS-Anschluss und daher einfacher einzusetzen als der Haken, der dafür flexibler ist.

Der Schieber bewegt Dynamiks nur in XY-Richtung. Dies können Sie ausnutzen, indem Sie ihn nach oben oder nach unten wegziehen.

In manchen Fällen ist ein Keil besser geeignet, um Dynamiks passiv zu

manipulieren.

Die Option “Nur am Rand” hat folgende Funktion: Nur wenn eine Fläche des Dynamiks im Schieber ist, wirkt dieser. Ohne diese Option versucht der Schieber einfach auf dem kürzesten Weg das Dynamik aus sich herauszubekommen. Das ist häufig auch so gewünscht. Es gibt aber auch Anwendungen, bei denen ein Abschiebefinger lose auf ein Dynamik aufgesetzt wird, dann bis zur Kante geschoben wird (wo er in der Realität herunterklappt) und erst auf dem Rückweg das Dynamik bewegen soll. Das lässt sich zwar in TrySim auch genau so nachbilden, ist aber mit vermeidbarem Aufwand verbunden. Mit dieser Option kann der Abschiebefinger einfach in das Dynamik einsinken. Erst wenn er den Rand erreicht hat, wird er aktiv und kann dann auf dem Rückweg das Dynamik schieben.

Die Option “Schneller Schieber” ist hilfreich, wenn viele Dynamiks als Gruppe geschoben werden sollen. Wenn die Verschiebung vergleichsweise langsam geschieht, ist das normalerweise kein Problem, aber bei schneller Bewegung und mehr als 5 Dynamiks hintereinander, springen diese unkontrolliert umher. Mit dieser Option können auch sehr viele Dynamiks schnell geschoben werden, allerdings muss vorher entschieden werden, in welche Richtung. Auch mit dieser Option darf die Bewegung des Schiebers nur so schnell sein, dass in einem Simulationschritt (typisch 100 ms) nicht mehr als eine Dynamik-Größe zurückgelegt wird.

Das Beispiel “AssemblierStrecke”, das Beispiel “Palettierer” sowie das Beispiel “Sortierstrecke” demonstrieren die Verwendung von Schieber und Haken.

Der Schieber funktioniert eingeschränkt auch mit drehbaren Dynamiks, dabei müssen Sie jedoch berücksichtigen, dass diese Elemente (noch) hauptsächlich durch ihre Ecken definiert sind. Achten Sie bitte darauf, dass der Schieber möglichst viele Ecken schiebt. Evtl. müssen Sie auch die Center-Punkte und die Kanten-Punkte aktivieren, oder die Drehbarkeit der Dynamiks auf die XY-Ebene begrenzen.
Wann immer möglich sollten Sie auf drehbare Dynamiks verzichten. Während die normalen Dynamiks sich ohne viele Kenntnisse intuitiv bewegen lassen, braucht man für die Manipulation der Drehbaren viel Erfahrung.

Dynamik-Konverter, Ausrichter

***

Die ab Version 2.1 vorhandenen drehbaren Dynamiks haben viele der Eigenschaften der normalen Dynamiks noch nicht. Außerdem sind sie deutlich rechenzeitintensiver. Daher haben wir dieses Element geschaffen, um die beiden Typen ineinander umzuwandeln, je nachdem, welche Eigenschaften der Dynamiks gerade benötigt werden. Der Konverter kann außerdem das Verhalten der Dynamiks ändern.

Drehbare in normale Dynamiks wandeln
Auf der Editiermaske können Sie die Konvertierungsrichtung einstellen. Es werden alle Dynamiks konvertiert, die die linke untere Ecke des Konverters enthalten. Wenn ein nicht rechtwinklig stehendes drehbares Dynamik konvertiert werden soll, wird

das normale Dynamik erzeugt, das mit der geringsten Drehung zu erhalten ist. Um ein widernatürliches Springen von sehr schräg stehenden drehbaren Dynamiks zu vermeiden, können Sie auf der Editiermaske einen maximalen Winkel eingeben. Steht das Dynamik schräger als dieser Winkel, wird es nicht konvertiert.

Normale in drehbare wandeln
Wenn Sie normale in drehbare Dynamiks konvertieren, können Sie wie bei der Erzeugung durch einen Generator angeben, ob die Flächen und/oder Kantenpunkte des drehbaren Dynamiks ebenfalls berücksichtigt werden sollen, oder ob nur Drehungen in der XY-Ebene möglich sein sollen.

Ausrichten
In der Wirklichkeit werden rutschende, gedrehte Dynamiks (z.B. auf einem Bogenförderband) durch Führungen nach der Drehung wieder rechtwinklig zur Förderrichtung ausgerichtet. Die Nachbildung solcher Führungen ist sehr aufwendig, sowohl was den Rechenaufwand betrifft, als auch was den Aufwand zur Erstellung der Simulation betrifft. Da der Dynamik-Konverter bereits die Fähigkeit hat, schräg stehende Dynamiks in rechtwinklig stehende zu verwandeln, kann man ihn auch dazu verwenden, etwas schräg stehende drehbare Dynamiks nur rechtwinklig auszurichten. Das Beispiel “Drehungen” demonstriert diese Verwendung.

Eigenschaften ändern
Diese Optionen sind ursprünglich zur Geschwindigkeitssteigerungen bei Anlagen mit sehr vielen Dynamiks geschaffen worden. Wenn Sie z.B. 500 Dynamiks auf einer Kühlstrecke laufen haben, mit denen dort aber nichts passiert, außer, dass sie gefördert werden, dann ist es eine Vergeudung von Rechenzeit, in jedem Simulationsschritt zu prüfen, ob sie von den in der Anlage vorhandenen Lichtschranken bedeckt werden. Ähnliches gilt für die Haken, die prüfen müssen, ob sie mit einem dieser Dynamiks überlappen. Auch das Zeichnen so vieler Elemente benötigt viel Zeit, insbesondere in der 3D-Ansicht. Ganz besonders viel Rechenzeit wird für die Kollisionsüberwachung der Dynamiks untereinander benötigt, denn jedes der Dynamiks muss überprüfen, ob es nicht mit irgendeinem anderen kollidiert. Wenn Sie viele Dynamiks haben, können Sie mittels des Dyn-Konverters folgende Eigenschaften ab- und mit einem weiteren Dyn-Konverter wieder einschalten:

Sichtbarkeit: Wenn ein Dynamik unsichtbar gemacht worden ist, wird es in keinem Fenster mehr angezeigt. Falls Sie den Überblick verloren haben, wieviele von den jetzt unsichtbaren Dynamiks noch in der Anlage sind, können Sie sie mit Anlage| Dynamiks|Alle Sichtbar machen wieder anzeigen. Diese Operation kann aber nicht rückgängig gemacht werden.

Kollisionsüberwachung: Mit der Option “Inaktiv machen” schalten Sie die Fähigkeit der Dynamiks ab, andere zu schieben oder auf ihnen zu stehen.

Haken deaktivieren: Wenn diese Option angewählt ist, kann das Dynamik nicht mehr von Haken und Automatikhaken gegriffen werden.

Sensoren deaktivieren: Hierdurch wird das Dynamik für Lichtschranken, Ultraschall-

Abstandsmesser und Barcode-Leser unsichtbar.

Presse

**

Mit der Presse können Sie Dynamiks verkleinern. Die Presswirkung ist immer in der Richtung der kleinsten Ausdehnung der Presse, d.h., Sie müssen die anfänglich kubische Presse in eine flache Form bringen. Sie lässt sich auch als Hobel verwenden.

Lassen Sie nach Möglichkeit kein Dynamik von oben auf eine Presse fallen, denn sonst wird es so weit zusammengepresst, dass man es nicht mehr sehen kann.

Das Beispiel “Assemblierstrecke” demonstriert den Gebrauch der Presse.

Automatikhaken

**

Diese aus der Krantechnik bekannte Sonderform eines Kranhakens benötigt keine externe Ansteuerung, um ein Dynamik zu greifen und wieder loszulassen. Er klinkt sich automatisch ein- und aus. Beim Senken auf ein Dynamik klinkt er sich ein. Wenn das Dynamik nach dem Transport wieder abgestellt wird, klinkt der Haken sich wieder aus. Die Funktion ist wie bei einem Kugelschreiber: Beim ersten Drücken kommt die Mine heraus, beim zweiten Drücken verschwindet sie wieder.

Ein Bit mit der Funktion “Schlaffseil-Relais” meldet an die SPS, wenn beim nächsten Aufwärtsfahren der Greifzustand umgeschaltet würde. Bitte beachten Sie, dass das Bit nicht anzeigt, ob ein Dynamik am Haken hängt, es zeigt nur an, dass das Seil schlaff ist, d.h. dass nicht einmal mehr der Haken daran hängt. Außerdem gibt es ein Bit mit der Funktion “Last hängt am Haken”.

Sticker

***

Der Sticker klebt, wenn aktiviert, an dem ersten Element fest, welches bei ihm vorbeikommt. Auf der Editiermaske können Sie einstellen, ob er an Dynamiks, SemiDynamiks oder Statiks kleben soll. Es ist auch jede Kombination möglich.

Wenn das SPS-Bit, an das der Sticker angeschlossen ist, ausgeschaltet wird, gibt es zwei Möglichkeiten zwischen denen Sie auf der Editiermaske wählen können: 1. Der Sticker bleibt an der aktuellen Position stehen. 2. Der Sticker springt an den Ort zurück, an dem er aufgepickt wurde. Wenn der Sticker an einem Dynamik klebt, welches vernichtet wird, springt er in jedem Fall an die Ausgangsposition zurück.

Auf der Editiermaske wird angegeben, woran der Sticker zur Zeit hängt. Wenn dort “frei” steht, wartet der Sticker auf ein Element, an dem er hängt. Sie können die Bindung eines Stickers an sein Element durch den Button “-> frei” lösen.

In Kombination mit einem Attraktor lässt sich z.B. der Weg eines Dynamiks durch die Anlage verfolgen, dies wird auch im “Beispiel1” demonstriert.

Einen Sticker können Sie auch mit der Klinke aktivieren und deaktivieren.

Klinke

***

Mit diesem Element können Sie einen Haken oder einen Sticker ein- und ausschalten.

Normalerweise wird ein Haken direkt über ein SPS-Bit betätigt. Wenn Sie aber sehr viele Haken haben (z.B. an einer Kette), die gar keine eigene Identität mehr haben, sondern sich nur noch durch den Ort auszeichnen, an dem sie sich befinden, dann ist es sehr aufwendig, das entsprechende SPS-Bit zu finden. Der Klinker löst dieses Problem, indem er den Zustand seines eigenen Bits auf jeden Haken überträgt, mit dem er überlappt.

Die Bits der Haken müssen natürlich alle frei sein, d.h., sie dürfen nicht von der SPS angesteuert werden. Am günstigsten ist es, sie zu hohen Adressen zu verschieben (z.B. A 10000.0, --.1, --.2 usw.). Auch müssen die Bits der Haken alle verschieden sein, sonst werden ja alle Haken gemeinsam an- bzw. ausgeschaltet.

Das Beispiel “Baumsäge” demonstriert den Gebrauch von Klinken: die Haken werden von der ersten Klinke eingeschaltet, wenn die Lichtschranke bedeckt wird (E0.1, klicken Sie auf der Editiermaske der Klinke auf “Adr” um dies zu überprüfen). Ausgeschaltet werden die Haken von der zweiten Klinke.

RFID Antenne & DatenChip

**

Diese beiden Elemente werden gemeinsam eingesetzt.

Der Datenchip ist passiv, er dient nur als Speicher für bis zu 10 000 Bytes Daten. Diese Daten werden auch mit der Anlage gespeichert.

Die Antenne kommuniziert mit jedem Datenchip, mit dem sie überlappt.

Ob sie lesen oder schreiben soll, wird durch zwei entsprechende SPS-Bits vorgegeben. Die Startadresse des Ziel- bzw. Quell-Bereiches wird ebenfalls durch zwei Bits vorgegeben.

Die Anzahl der zu übertragenden Bytes wird auf der Editiermaske der Antenne fest vorgegeben.

Wenn die Antenne lesen soll, aber mit keinem Datenchip überlappt, dann bleibt der Zielbereich unverändert.

Die Daten selbst, die auf den Editiermasken von Antenne und Chip als Char oder als HexBytes gezeigt werden, können dort nicht editiert werden.

Die Antenne kann auch so konfiguriert werden, dass sie jedes nicht drehbare Dynamik als Datenchip betrachtet.

Die in Dynamiks abgelegten Daten werden z.Z. auch bei den speicherbaren Dynamiks beim Verlassen von TrySim nicht auf Platte gespeichert. Auch überleben sie die Wandlung in ein drehbares Dynamik nicht. Beides lässt sich aber bei Bedarf nachrüsten.

Kasten

*

Dies ist das einfachste Element. Es besteht aus einem Quader, dessen Größe und Farbe Sie vorgeben können. Wie alle weiteren Elemente auch, hat der Kasten einen Vater. Der Vater ist irgendein anderes Element, an dem der Kasten befestigt ist. Wird der Vater bewegt, folgt ihm der Kasten.

Kästen dienen hauptsächlich dazu, eine Anlage graphisch und logisch zu strukturieren. Sie erzeugen z.B. einen Kasten und nennen ihn “Pult”, wenn Sie für alle Taster und Leuchter dieses Pult als Vater angeben, dann können Sie später alle diese Elemente durch eine Mausbewegung verschieben.

Kästen können sich gegenseitig durchdringen.
Kästen können Endschalter bedecken. Dazu müssen sie in der Editiermaske des Endschalters als Master gewählt werden.

Wenn Sie eine Abstellfläche für Dynamiks benötigen, dann verwenden Sie eine Platte.

Kästen können drehbar gemacht werden. Wählen Sie dazu auf der Editiermaske die Checkbox “Drehbar” an. Lesen Sie auch allgemeines über drehbare Elemente .

Kästen können nicht von Förderbändern oder Haken bewegt werden, dies geht nur mit den von einem Generator erzeugten Dynamiks.

Platte

*

Die Platte gleicht in allem dem Kasten, allerdings kann man auf ihr Dynamiks (dynamische Elemente) abstellen. Dabei reicht es aus, wenn nur eine Ecke des Dynamiks auf der Platte steht. Das sieht zwar nicht aus wie in der Realität, ist aber zum Simulieren sehr nützlich. Wenn eine Platte erzeugt wird, hat sie tatsächlich Plattenform, aber das sollte Sie nicht davon abhalten, eine Stange oder einen Block aus ihr zu formen.

Achten Sie darauf, dass mindestens eine **Ecke** des Dynamiks auf der Platte steht, sonst fällt das Dynamik durch die Platte durch, oder sie stülpt sich darüber.

Die Platte kann drehbar gemacht werden. Wählen Sie dazu auf der Editiermaske die Checkbox “Drehbar” an. Lesen Sie auch Allgemeines über drehbare Elemente .

Der Parameter mit dem ominösen Namen “GlitschFaktor” bestimmt, wie fest Dynamiks auf der Platte stehen. Ein Wert von 100 war lange voreingestellt und für Anwendungsfälle vorgesehen, bei denen die Dynamiks auch wenn die Platte bewegt wird, fest auf ihr stehen bleiben. Wenn aber mehrere Dynamiks über die Platte geschoben werden, muss dieser Wert verringert werden, sonst springen die Dynamiks übereinander, was nur in manchen Fällen gewünscht ist.
Das Verhalten von auf Platten rutschenden Dynamiks kann auch durch den Einsatz eines virtuellen Hakens gesteuert werden.

Richtig überzeugend ist das Verhalten von vielen Dynamiks auf einer Platte aber insgesamt noch nicht. Wenn Sie in Ihrer Anwendung damit Probleme haben, wenden Sie sich bitte an uns. Wir werden Abhilfe finden, wenn auch vielleicht nur für diesen speziellen Fall.

Stange

**

Mit der Stange können Sie zwei beliebige statische Element miteinander verbinden. Eines der Elemente ist der Vater, das andere müssen Sie als Master angeben.

Standardmäßig werden die Mitten beider Elemente verbunden, Sie können jedoch durch Abwahl der Checkbox “Mitte” auch vorgeben, dass die Ecken, die dem Ursprung (unten links) am nächsten liegen, miteinander verbunden werden.

Die Stange ist eine reine Darstellungshilfe, sie hat keinerlei Einfluss auf die Simulation. Es ist also nicht möglich, mithilfe der Stange ein Element zu ziehen oder zu schieben, die Länge der Stange passt sich automatisch dem Abstand der beiden Elemente an.

Reiter

Siehe Reiterbahn.

Reaktor

***

Der Reaktor kann verwendet werden, um Medien in einem Behälter in andere Medien umzuwandeln. Auf der Editiermaske geben Sie ein oder zwei Ausgangsprodukte und ein oder zwei Endprodukte an. Den Anteil des jeweils ersten Mediums an der Reaktion können Sie eingeben, der Anteil des anderen Mediums wird automatisch auf 100 % ergänzt.

Die Geschwindigkeit der Reaktion geben Sie in Milliliter pro Sekunde, Liter pro

Minute oder Kubikmeter pro Stunde an. Falls “Konzentrationsabhängig” angewählt ist, wird die Reaktionsgeschwindigkeit verringert, wenn die Ausgangsprodukte nicht in ausreichender Konzentration vorliegen.
Beispiel:
Sie haben als Ausgangsprodukte Medium 1 mit 60% und Medium 2 mit 40 % sowie eine Reaktionsgeschwindigkeit von 100 l/min gewählt. In dem Behälter sei der Anteil des Mediums 1 30 % und der des Mediums 2 nur 10%. Dann wird die Reaktion nur mit der Geschwindigkeit 100 l/min * 30/60 * 10/40 = 12,5 l/min ablaufen.
Falls “Konzentrationsabhängig” nicht angewählt ist, läuft die Reaktion mit der vorgegebenen Geschwindigkeit ab, solange beide Ausgangsmedien in ausreichender Menge vorhanden sind.

Sie können einstellen, ob die Reaktion in jedem Fall stattfinden soll oder nur dann, wenn sich der Reaktor tatsächlich in der Flüssigkeit befindet.

Eine mit der Reaktion verbundene Volumenänderung kann ebenfalls berücksichtigt werden. Wenn z.B. 6 Einheiten von A und 4 Einheiten von B nur 9 Einheiten von C ergeben, geben Sie 90% in das entsprechende Feld ein.

Sie können mit diesem Element auch eine Reaktion in füllbaren Dynamiks durchführen. In diesem Fall sollten Sie die Option “Warnen, wenn nicht angeschlossen” abschalten.

Peek

****

Dies sind Spezial-Elemente, die keine Entsprechung in der Wirklichkeit haben, sondern dazu dienen, TrySim flexibler zu machen. Peeks haben die umgekehrte Funktion wie Pokes und werden genauso gehandhabt. Sie können damit Eigenschaften der Elemente wie Position und Größe in ein SPS-Bit/Byte/Word/ DWord kopieren lassen, um sie im SPS-Programm oder an anderen Stellen der Anlage auszuwerten.

Um einen Peek zu verwenden, gehen Sie folgendermaßen vor:

- Erzeugen Sie einen Peek (Registerkarte: Sonstiges)
- Befestigen Sie ihn an dem Element, dessen Eigenschaften Sie interessieren.
- Wählen Sie auf der Editiermaske des Peeks den Datentyp der interessierenden Größe aus.
- Wählen Sie auf der Editiermaske des Peeks unter “Wert” die gewünschte Eigenschaft aus.

· Liste der peekbaren Eigenschaften

Die Liste wird z.Z. noch erweitert (auch kurzfristig auf Anfrage!)

Bit-Peek

Alle Elemente mit Antrieb: Antrieb läuft (Geschwindigkeit <> 0)

Word-Peek (vorzeichenlose 16 Bit-Zahl)

Alle Elemente: Position X,Y,Z und Größe X,Y,Z
bzw. Von X,Y,Z und Nach X,Y,Z

Gelenk : aktueller Winkel (die Einheit ist 0,1 Grad)

Servo-Antrieb : Position (in den voreingestellten Einheiten)

Poke

***
—

ACHTUNG! Wenn Ihre Projekte nicht von älteren TrySim-Versionen geladen werden müssen (dies ist hauptsächlich in Schulen der Fall) verwenden Sie bitte das Element, das einfach nur mit “Poke” bezeichnet ist. Die mit “Poke Word” und “Poke DWord” bezeichneten Elemente sind nur noch für eine Übergangszeit wegen der Kompatibilität vorhanden. Sie werden in einer späteren Version nicht mehr in der Element-Liste auftreten. Falls Projekte diese Elemente dann noch enthalten, werden sie beim Laden automatisch in das neue Element umgewandelt.

Pokes sind Spezial-Elemente, die keine Entsprechung in der Wirklichkeit haben, sondern dazu dienen, TrySim flexibler zu machen.

Mit den Pokes können Sie viele Eigenschaften der Elemente von der SPS aus beeinflussen. Sie geben ein Word/DWord in der SPS vorund die Zahl die darin steht, wird dem Vater des Pokes übergeben, als ob Sie sie ihn der Editiermaske eingetippt hätten. Welches Feld Sie poken wollen, geben Sie durch die Auswahlliste “Feld” vor.

Die Poke-Elemente übergeben einen Wert nur dann ihrem Vater, wenn sich dieser in der SPS **ändert**. Damit dies in der SPS leichter zu programmieren ist, können sie die Checkbox “Reset nach poke zu:” anklicken und einen entsprechenden Reset-Wert angeben. Nach jedem Transfer in das Word/DWord wird jetzt der Poke ausgeführt und der Wert des Words/DWords danach auf den Reset-Wert zurückgesetzt. Diesen Wert können Sie vollkommen willkürlich wählen, achten Sie nur darauf, dass er außerhalb des Bereiches liegt, in dem Sie poken wollen. Wenn Sie diese Option wählen, sollten Sie sicherstellen, dass nur dann ein Wert in das Word/DWord transferiert wird, wenn Sie die Eigenschaft des Elementes tatsächlich ändern wollen, denn jeder Transfer in das Word/DWord bedeutet ja jetzt eine Änderung (außer natürlich, sie transferieren den Reset-Wert). Sie dürfen diese Option nicht verwenden, wenn das Projekt auch mit TrySim-Versionen kleiner als V2.9.10 geöffnet werden soll.

Beachten Sie, dass Sie bei Verwendung von Pokes die Maßeinheiten (mm,cm, usw...) nicht mehr ändern dürfen (oder Sie müssen den gepoketen Wert im SPS-

Programm entsprechend anpassen).

Die gepoketen Werte werden weitgehend ungeprüft dem Element übergeben, daher setzt die Verwendung von Pokes eine entsprechende Sorgfalt voraus, sonst werden Sie unerwartete Ergebnisse erhalten.

Die meisten Eigenschaften werden als vorzeichenlose 16-Bit-Zahl erwartet (Word-Poke), einige wenige (z.B. der Hotspot von Ketten) werden als vorzeichenlose 32-Bit-Zahl erwartet (DWord-Poke), das Element müssen Sie also entsprechend auswählen.

Sie können für jedes Element erkennen, dass einige seiner Eigenschaften gepoket werden: Öffnen Sie seine Editiermaske, der Baum trägt giftige Äpfel:  . Klicken Sie auf das Symbol, dann können Sie im Elementbaum erkennen, welche Pokes  unter den Kindern dieses Elementes sind.

Problem: Die Werte, die gepoked werden, dürfen nicht unitialisiert sein. Das kann manchmal sehr lästig sein und ggfs. sollten Sie uns fragen, was der beste Weg ist, dieses Problem zu umgehen.

Das Beispiel “Formel1” demonstriert die Verwendung von Pokes.

· Liste der pokebaren Eigenschaften

Die Liste wird z.Z. noch erweitert (auch kurzfristig auf Anfrage!)

Word-Poke (vorzeichenlose 16 Bit-Zahl)

Alle Elemente: Position X,Y,Z und Größe X,Y,Z
bzw. Von X,Y,Z und Nach X,Y,Z

Generator: Dynamikgröße X,Y,Z
Meisterschalter: Schaltstellung
Servo-Antrieb: Max-Speed und Rampe
Bit-Antrieb des Gelenks: Geschwindigkeiten und Rampen
Reset-Button der thermischen Masse (1 = Reset)

DWord-Poke (vorzeichenlose 32-Bit-Zahl)

Kette: Hotspot

Knoten: Hotspot - Offset

2.4.7 Thermische Elemente

Thermische Masse

***

Die thermische Masse besteht intern aus einer rechteckigen Anordnung von Quadern, innerhalb derer die Temperatur als konstant angesehen wird. Die Anzahl der Teilung können Sie auf der Editiermaske vorgegeben. Der Wärmeaustausch zwischen den einzelnen Quadern erfolgt mit der ebenfalls festzulegenden “Inneren Leitfähigkeit”. Der Wärmeaustausch mit der Umgebung erfolgt mit der “Äußeren Leitfähigkeit”. Sie können der thermischen Masse durch eine Heizung Energie zuführen, oder sie damit kühlen, indem Sie für die Heizleistung einen negativen Wert angegeben. Mit dem Thermometer erfassen Sie die Temperatur.

Das für die thermische Masse verwendete Modell ist das Einfachste und liefert einigermaßen realistische Ergebnisse. Es berücksichtigt nur die Wärmekapazität und die Wärmeleitung. Da der Energietransport in der Wirklichkeit zum größten Teil durch Konvektion erfolgt, die zu simulieren außerordentlich aufwendig wäre, muss bei der Leitfähigkeit ein effektiver Wert vorgegeben werden, der nur durch Probieren ermittelt werden kann. Es ist, wie so häufig in TrySim, nicht unser Ziel gewesen, physikalische Vorgänge im Detail nachzubilden, sondern sie auf das für den SPS-Programmierer Wesentliche zu reduzieren.

Mit dem Button “Reset” auf der Editiermaske setzen Sie die thermische Masse auf die Umgebungstemperatur zurück. Sie können den Reset auch durch die SPS mittels eines Word-Pokes anstoßen. Wenn der Wert 1 gepoked wird, wird die thermische Masse zurückgesetzt. Wenn Sie auf der Editiermaske des Pokes “Reset nach poke” anwählen und den Resetwert von Null beibehalten, brauchen Sie für den Reset nur das LSBit des Wortes auf “1” zu setzen, z.B. vom MW 1200 das Bit M 1201.0.

Heizung

**

Mit der Heizung können Sie einer thermischen Masse an einer bestimmten Stelle Energie zuführen oder entnehmen. (Anmerkung: in V2.6 ist die Temperatur der Flüssigkeiten noch konstant.)

Die Heizung verfügt über einen Antrieb, durch den die Heizleistung bestimmt wird. Eine negative Heizleistung entspricht einer Kühlung. Zur Zeit gibt es 3 Antriebe für die Heizung: die Bit-Heizung, die nur an- oder auszuschalten ist, die Bit-Word-Heizung, deren Heizleistung zusätzlich von der SPS aus vorgegeben werden kann und der Antrieb über einen Linearbeweger. Mit diesem kann z.B. eine mechanisch betätigte Kühlklappe simuliert werden oder es kann der bei einem Ofen auftretenden Energieverlust beim Öffnen der Türen nachgebildet werden.

Die Heizleistung des Antriebes wird noch mit dem Wirkungsgrad gewichtet, der durch eine Kennlinie von der Temperatur abhängig gemacht werden kann. Dadurch kann berücksichtigt werden, dass eine 1000°C heiße Flamme einem ebenso heißen Werkstück keine Energie zuführen kann.

Kennlinie

***

Das Element Kennlinie dient zum Umsetzen einer Größe in eine andere. Bislang ist dies nur für den Wirkungsgrad der Heizung in Abhängigkeit von der Temperatur möglich, aber die Ausweitung auf andere Anwendungen ist vorgesehen.

Eine Kennlinie erzeugen Sie entweder genau wie ein normales Element, oder Sie wählen aus der Liste eines Elementes, das Kennlinien verwalten kann “Neue KennLinie”. Sie können die Editiermaske der Kennlinie auch aufrufen, indem Sie auf das Kennlinienfeld eines solchen Elementes doppelklicken.

Geben Sie beim Erstellen einer Kennlinie zunächst die Anzahl der gewünschten Stützpunkte sowie die XMin- und XMax-Werte ein. Mit dem Schieberegler über der Grafik wählen Sie dann die zu editierenden Punkte an und geben in das Feld “Y-” den entsprechenden Wert ein. Wenn Sie die Eingabe mit “Enter” bestätigen, springt der Schieberegler automatisch auf den nächsten Stützpunkt.

Dieses Element ist noch nicht vollständig fertig, erfüllt aber prinzipiell seine Aufgabe.

Thermometer

**

Mit dem Thermometer bestimmen Sie die Temperatur an einer bestimmten Stelle innerhalb einer thermischen Masse. (Anmerkung.: in V2.6 ist die Temperatur der Flüssigkeiten noch konstant.)

Bei der Einheit können Sie auf der Editiermaske zwischen verschiedenen Möglichkeiten wählen.

Wenn sich das Thermometer nicht innerhalb einer thermischen Masse befindet, liefert es keinen Wert, d.h., es lässt das angeschlossene SPS-Word unbeeinflusst.

Wenn Sie nur die Überwachung auf eine feste Grenztemperatur benötigen, können Sie auch den Thermostat verwenden.

Thermostat

**

Mit dem Thermostat können Sie, ebenso wie mit dem Thermometer, die Temperatur an einer bestimmten Stelle innerhalb einer thermischen Masse bestimmen. Allerdings liefert der Thermostat nur einen binären Wert “Temperatur überschritten”.

Die Temperatur müssen Sie auf der Editiermaske einstellen. Auch die Hysterese geben Sie dort ein. Der Thermostat schaltet ein, wenn die Grenztemperatur überschritten ist, er schaltet wieder aus, wenn die Grenztemperatur abzüglich der

Hysterese unterschritten wird.

Standardmäßig ist der Thermostat auf “Öffner” eingestellt, d.h., er liefert “0” wenn die Grenztemperatur überschritten wurde. Sie können ihn aber auf Schließer umkonfigurieren.

Wenn sich der Thermostat nicht innerhalb einer thermischen Masse befindet, liefert er keinen Wert, d.h., er lässt das angeschlossene SPS-Bit unbeeinflusst.

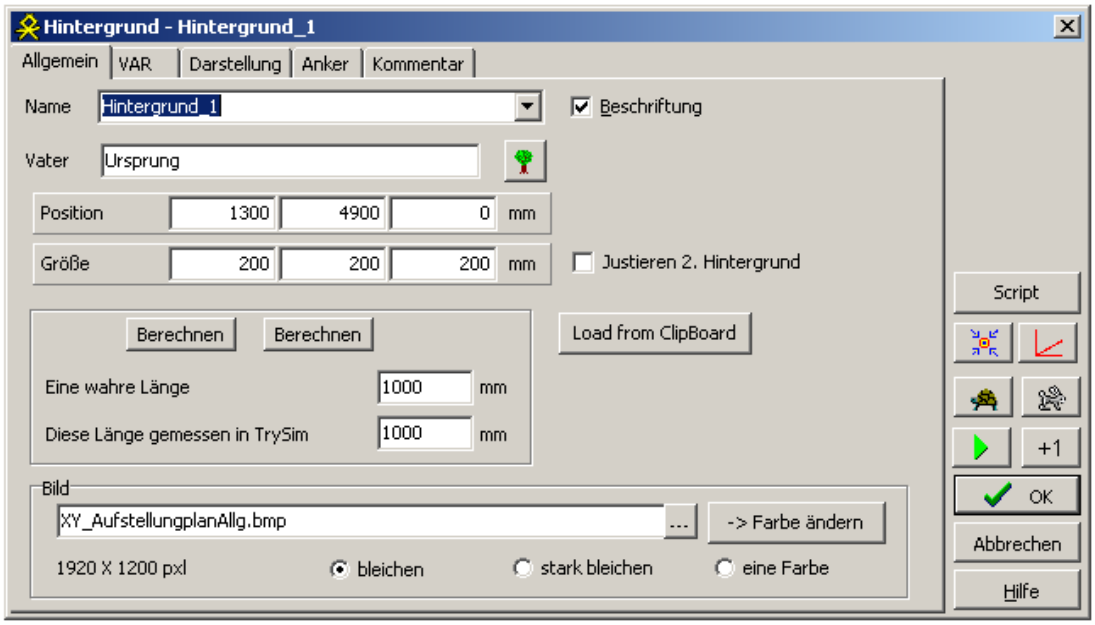
2.4.8 Simulationshilfen

Hintergrund

**

Sie können den Hintergrund der **2D**-Fenster mit einer beliebigen Grafik hinterlegen. Dies kann sehr hilfreich seien, wenn man dazu bei einer größeren Anlage den Aufstellungsplan nimmt.

Die Grafik erzeugen Sie mit einem Grafikprogramm Ihrer Wahl. Wir haben in TrySim nur die Änderung der Farbe als Bearbeitung vorgesehen. Sie müssen aber wissen, welche Breite und Länge ihre Grafik in der **wirklichen Anlage** darstellt. Genau diese Werte (in mm) geben Sie auf der Editiermaske ein. Sie können auch die unten beschriebene Rechenhilfe verwenden. Die extern erzeugte Grafik können Sie als *.bmp-Datei in das Projekverzeichnis kopieren. Wenn Sie eine Grafik in die Zwischenablage gelegt haben, z.B. mit der Schnappschuss-Funktion im Adobe-Reader, können Sie sie mittels “Load from ClipBoard” direkt in TrySim einfügen.



Rechenhilfe für Größe:

- Ungefähr werden Sie die wahre Größe der Anlage kennen. Diese Werte geben

Sie zunächst für die beiden entsprechenden Koordinaten ein und schließen Sie die Maske.

- Dann suchen Sie auf dem Bild ein möglichst großes Bauteil, dessen Größe sie genau kennen, z.B. durch eine Bemaßung.
- Für dieses Bauteil bestimmen Sie dann eine Länge, so wie sie in TrySim zur Zeit gemessen wird. Am Einfachsten geht dies mittels eines temporären Kastens, den Sie darüber legen. Die Größe des Kastens wird z.B. in der Statusleiste von TrySim angezeigt.
- Öffnen Sie wieder die Editiermaske und geben die wahre Länge und die soeben gemessene Länge ein.
- Dann klicken Sie auf denjenigen “Berechnen”-Knopf, der zur gemessenen Richtung passt.
- Jetzt wird zuerst diese Größe angepasst und dann die andere, sodass das Breite/ Länge-Verhältnis in TrySim demjenigen der Bitmap entspricht.
- Die Anlagengröße (weißer Bereich) können Sie unter Extras|Optionen|Anlage einstellen.

Der Hintergrund ist absichtlich mit der Maus nicht zu verändern. Um die Größe und Position zu ändern, erreichen Sie die Editiermaske über den Elementbaum, oder über die ENTER-Taste, während der Hintergrund noch markiert ist.

Änderung der Farbe:
Es hat sich herausgestellt, dass während des Konstruierens der Anlage die Hintergrundgrafik möglichst blass sein sollte, da die eigentlichen Funktionselemente sonst nicht mehr gut zu sehen sind.
Die Änderung der Farbe kann mittels des Buttons “-> Farbe ändern” geschehen. Jedes Pixel im Bild, das nicht weiß ist, wird dadurch entweder gebleicht oder auf die gewählte Farbe gesetzt. Sie können auch wiederholt bleichen. Das Bild auf der Festplatte wird dabei bleibend verändert. Für andersartige Bearbeitung der Grafik verwenden Sie bitte einen Grafikeditor Ihrer Wahl.
Wenn es für Demonstrationszwecke geraten erscheint, kann man die Grafik jederzeit wechseln. Bewahren Sie dazu eine Kopie der originalgefärbten Bitmap auf.

Sie können auch mehrere Hintergründe, z.B. an bewegten Elementen befestigten. Wir raten jedoch vor einem allzu häufigen Gebrauch ab, da mit hohem Rechenzeitbedarf zu rechnen ist.

Der Hintergrund kann in der 3D-Darstellung zwar gezeigt werden, aber nur als Kasten/Fläche, das Bild ist darauf nicht zu sehen. Manchmal ist er dort als Orientierungshilfe nützlich. Sie können die Farbe anpassen, die 2D-Darstellung bleibt davon unberührt.

Wichtig! Falls es Probleme mit der Rechenzeit während der Simulation gibt, sind die Hintergründe die ersten Elemente, die sie ausblenden sollten.

Streifen

***

Mit dem Streifen können Sie während des Editierens Teile der Anlage ein- und ausblenden.

Der Einsatz des Streifens lohnt sich nur bei großen Anlagen.

Nur die Elemente, die sich innerhalb des Streifens befinden, werden dargestellt. Sie können den Streifen aber auch invertieren, dann werden nur die Elemente angezeigt, die sich außerhalb befinden.

Wenn Sie einen Streifen erzeugen, ist er zunächst inaktiv. Nachdem Sie ihn auf die von Ihnen gewünschte Größe und Position gebracht haben, müssen Sie ihn aktivieren. Öffnen Sie dazu die Editiermaske und klicken Sie auf die Check-Box “Aktiv” oder Doppelklicken Sie bei stehender Anlage auf den Rand des Streifens.

Sie können die Wirksamkeit eines Streifens auf bestimmte Blickrichtungen begrenzen.

Wenn Sie mehrere Streifen haben, werden alle Elemente dargestellt, die innerhalb mindestens eines Streifens liegen (sinngemäß bei invertierten Streifen). Mehrere Streifen werden immer gleichzeitig aktiviert und deaktiviert, d.h., wenn Sie einen Streifen aktivieren, werden alle anderen automatisch auch aktiv.

Die Streifen sind als reine Editierhilfe entworfen und sind nicht laufzeitoptimiert. Wenn Sie also eine große Anlage mit hohem Zeitrafferfaktor simulieren wollen, sollten Sie alle Streifen löschen (deaktivieren reicht nicht). Während der Laufzeit funktionieren Streifen nicht befriedigend !

Wenn Sie neue Elemente erzeugen, während ein Streifen aktiv ist, sollten Sie unbedingt das Kreuz innerhalb des Streifens positionieren, sonst liegen die neuen Elemente in dem unsichtbaren Raum.

Streifen sind (wenn man sich an sie gewöhnt hat) relativ einfach in der Handhabung, beim Bauen sehr großer Anlagen sollten Sie aber auch die Verwendung von Gruppen erwägen.

Kreuz

*

Das Editieren der Anlage ist nur in den 2D-Fenstern möglich. Wenn Sie dort ein neues Element plazieren, können Sie entlang der Blickrichtung nicht erkennen, wo das Element landet. Mit dem Kreuz können Sie bei der Erzeugung neuer Elemente bestimmen, welchen Wert die unsichtbare Koordinate erhalten soll. Dadurch wird vermieden, dass alle neuen Elemente auf dem Boden liegen, bzw. an der Wand kleben.

Bevor Sie einen neuen Anlagenteil bauen, sollten Sie das Kreuz in mindestens zwei

verschiedenen Ansichten in der Nähe der Baustelle positionieren.

Es kann immer nur ein Kreuz zugleich geben. Wenn Sie ein neues erzeugen, wird ein bereits bestehendes automatisch gelöscht. Falls Sie an dem bestehenden Kreuz Elemente befestigt hatten (was nicht empfohlen wird) werden diese an dem Vater des Kreuzes (meistens der Ursprung) befestigt.

Auf jeden Fall sollten Sie das Kreuz verwenden, wenn Sie neue Elemente innerhalb eines Streifens erzeugen wollen. Sonst verschwinden die Elemente scheinbar unmittelbar nach der Erzeugung, weil sie entlang der unsichtbaren Koordinate außerhalb des Streifens landen.

Auch ohne das Kreuz können Sie die unsichtbare Koordinate in etwa vorgeben, wenn Sie neue Elemente gleich bei der Erzeugung einem Vater zuordnen.

Das Kreuz kann auch als Bezugspunkt verwendet werden, um die Koordinaten anderer Elemente relativ zu einem Punkt in der Anlage einzugeben. Wählen Sie dazu die Checkbox “Als Bezugspunkt” an. Seit V3.0 ist das für alle Elemente möglich, kann allerdings nur im Elementbaum aktiviert werden.

Anders als bei allen anderen (nicht drehbaren) Elementen bezeichnet die “Position” des Kreuzes nicht die Ecke links unten, sondern die Mitte (dort wo sich die Linien kreuzen).

Wenn die CheckBox “Folge 3D-Center” aktiviert ist, wandert das Kreuz in den 2D-Fenstern so, dass es im 3D-Center in der Mitte dargestellt wird. Das kann z.B. dafür nützlich sein, ein Element zu identifizieren, welches in der 3D-Grafik angezeigt wird, obwohl dies nicht gewünscht ist. Oder einfach nur, um zu erkennen, wo der Blickpunkt in der 3D-Grafik ist, um den sich alles dreht.

Attraktor

***

Der Attraktor führt seinen Namen, weil er die Blicke auf sich lenkt. Die Position, an der sich der Attraktor befindet, wird im Mittelpunkt der Grafikfenster angezeigt, auch wenn sich der Attraktor bewegt. Dadurch ist es möglich, eine stärkere Vergrößerung einzustellen und dennoch im entscheidenden Moment den richtigen Bildausschnitt zu haben. In der 3D-Grafik kann der Attraktor verwendet werden, um mit wenig Aufwand eine Art Film zur Demonstration der Anlagenfunktion zu erstellen.

Unter **Ansicht|Attraktor** müssen Sie jedes Fenster, das nach Attraktoren Ausschau halten soll, einzelnd freigeben. Wenn nämlich alle Fenster den Attraktoren folgen würden, wäre kein vernünftiges Arbeiten mehr möglich.

Attraktoren können an ein SPS-Bit angeschlossen werden, welches sie aktiviert. Mehrere Attraktoren können so verwendet werden, um zwischen verschiedenen Bildausschnitten hin und her zu schalten. Wenn Sie mehr als einen Attraktor verwenden, müssen Sie darauf achten, dass immer nur einer aktiv ist, es ist sonst unbestimmt, welcher Attraktor sich letztlich durchsetzt. Wenn Sie die Attraktoren

über Taster aktivieren wollen, sollten Sie dazu einen Stufenschalter verwenden. Vergessen Sie nicht das Kästchen “Immer 1” abzuwählen.

Ein Attraktor kann immer nur in den Ansichten aktiviert werden, die auf der Editiermaske unter “Anzeigen in” angewählt sind.

Mit einem Sticker können Sie einen Attraktor auch an ein Dynamik binden und so seinen Weg durch die Anlage verfolgen. Das “Beispiel1” demonstriert diese Verwendung.

SpeedTrigger

**

Mithilfe dieses Elementes können Sie die Simulationsgeschwindigkeit automatisch ändern lassen. Sie geben ein Bit der SPS vor und können dadurch Zeitlupe bzw. Zeitraffer aktivieren.

Es gibt vier Triggermöglichkeiten:
1. Steigende Flanke: Hierbei wird die Geschwindigkeit einmalig bei einer steigenden Flanke des Bits auf den eingestellten Wert gesetzt.
2. Fallende Flanke: wie 1.)
3. “Dauernd 1”: Hier wird die Geschwindigkeit bei einer steigenden Flanke auf den eingestellten Wert gesetzt. Wenn die Geschwindigkeit bis zu der fallenden Flanke nicht geändert wird (durch einen weiteren Speed-Trigger oder über Kaninchen/ Schildkröte), wird beim Verschwinden des Signals die alte Geschwindigkeit wieder eingestellt.
4. “Dauernd 0”: wie 3.)

Sie können auch einstellen, dass die Simulation beim Eintreten der Triggerbedingung ganz gestoppt wird.

Das Beispiel “Oszi” demonstriert den Gebrauch eines Speed-Triggers.

2.4.9 Master-Liste

Einige Elemente benötigen außer dem Vater noch ein weiteres Element (Master), um ihre Funktion zu erfüllen.

Siehe:
Endschalter
Spion
Stange

2.5 Allgemeines über drehbare Elemente

***

Die meisten Elemente in TrySim sind noch nicht drehbar, wir arbeiten jedoch daran, sie nach und nach drehbar zu machen. Die Einbeziehung von Drehungen macht das

Editieren und Darstellen sehr viel schwieriger. Sie sollten Drehungen nur verwenden, wenn es unbedingt notwendig ist und wenn Sie bereits einige Erfahrung mit TrySim gesammelt haben. Da die Drehungen außerdem sehr viel Rechenzeit verbrauchen, ist ein schneller Rechner die Voraussetzung.

Im Editiermodus können Sie ein drehbares Elemente drehen, indem Sie lange mit der rechten Maustaste darauf klicken, dann erscheint das Kontext-Menü und Sie wählen den Punkt “Drehen”. Das kleine Fenster, das dann erscheint, hat drei Schieberegler zum Drehen um alle Achsen. Mit dem Button “Standard” können Sie die ungedrehte Position wieder herstellen.

Die Größe eines drehbaren Elementes können Sie mit der Maus nur verändern, wenn es rechtwinklig ausgerichtet ist. Der Grund hierfür ist, dass es andernfalls keine eindeutige Zuordnung einer zweidimensionalen Mausbewegung zu einer dreidimensionalen Größenänderung gibt.

Anders als bei den nicht-drehbaren Elementen, bei denen die “Position” die Ecke links unten bezeichnet, wird bei den Drehbaren die Mitte des Elementes auf der Editiermaske angezeigt, da dies der einzige Punkt ist, der bei Drehungen unverändert bleibt.

Einige Elemente sind von vorneherein drehbar:
Lichtschranke, Linearbeweger, Gelenk, drehbares Förderband

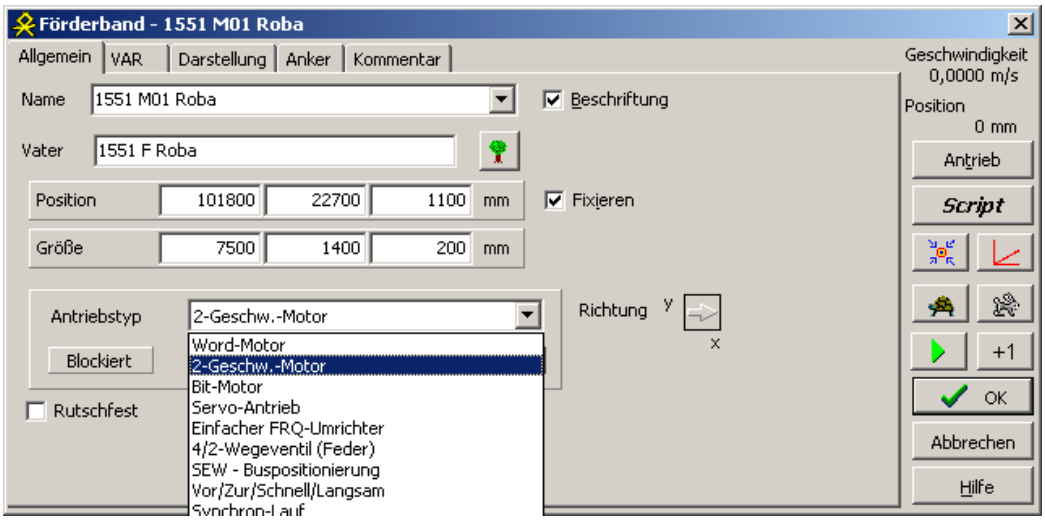
Einige Elemente können auf der Editiermaske mittels der Checkbox “Drehbar” drehbar gemacht werden:
Kasten, Platte, Endschalter, Endschalternase, Haken
Achtung! Wenn Sie eines dieser Elemente einmal drehbar gemacht haben, können Sie es nicht mehr in ein Nicht-drehbares zurückverwandeln.

2.5.1 Markieren von drehbaren Elementen

Zum Markieren von Elementen, die bereits gedreht worden sind, müssen Sie auf den Rand des (nicht sichtbaren) Rechtecks klicken, welches das Element umrahmt. Am Einfachsten geht dies, wenn Sie knapp innerhalb einer Ecke des Elementes klicken.

2.6 Antriebe

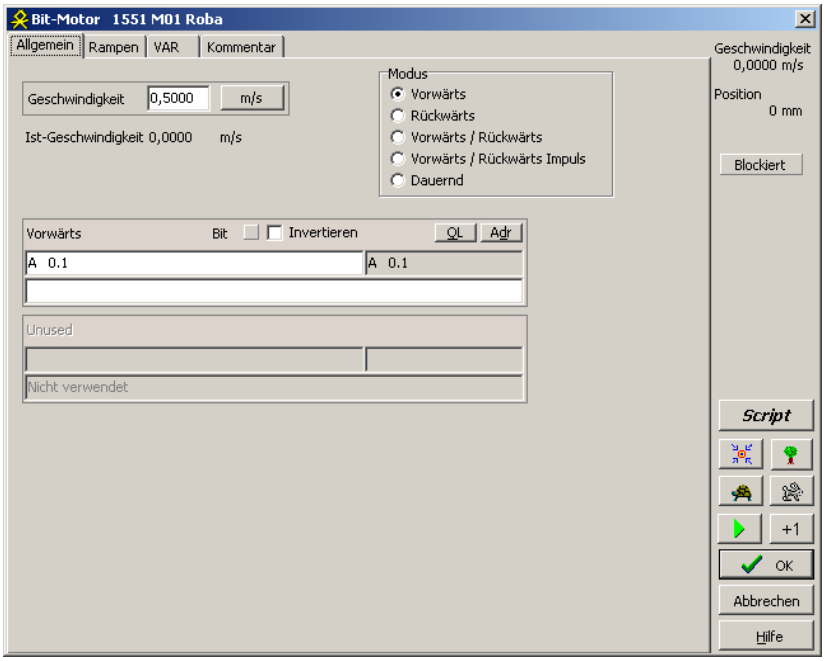
*
–

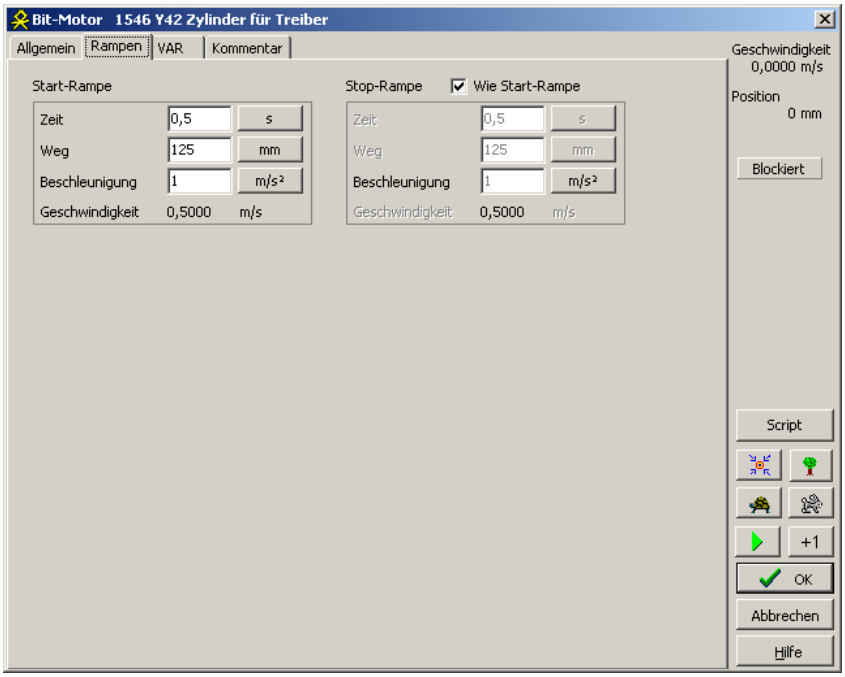


Der Antrieb kommt nie alleine vor, sondern ist immer Bestandteil eines anderen Elementes, wie Linearbeweger und Förderbänder. Ein Antrieb übersetzt Ausgänge der SPS in Geschwindigkeiten oder Positionen.

Es gibt verschieden Arten von Antrieben:

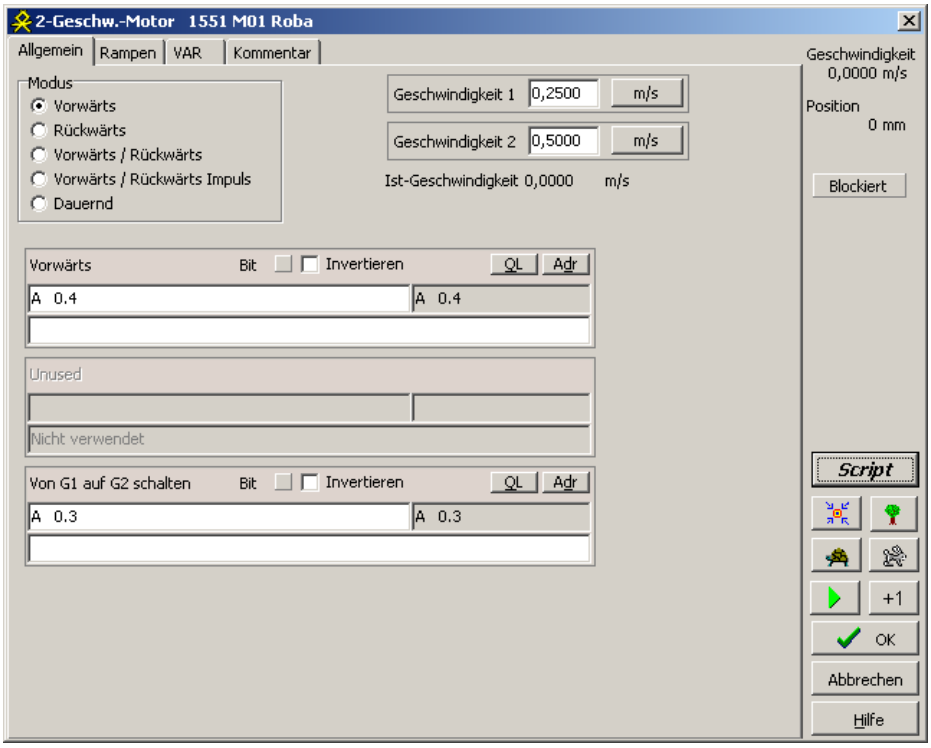
- **Bit-Motor**





Dieser Antrieb kann mit einer Geschwindigkeit vor- und zurück fahren. Sie können entweder ein oder zwei SPS-Ausgänge angeben. Außerdem können Sie Vorwärts/Rückwärts mit Impulsverhalten wählen. In diesem Fall verhält sich der Antrieb wie ein 5/2-Wegeventil mit Impulsverhalten.

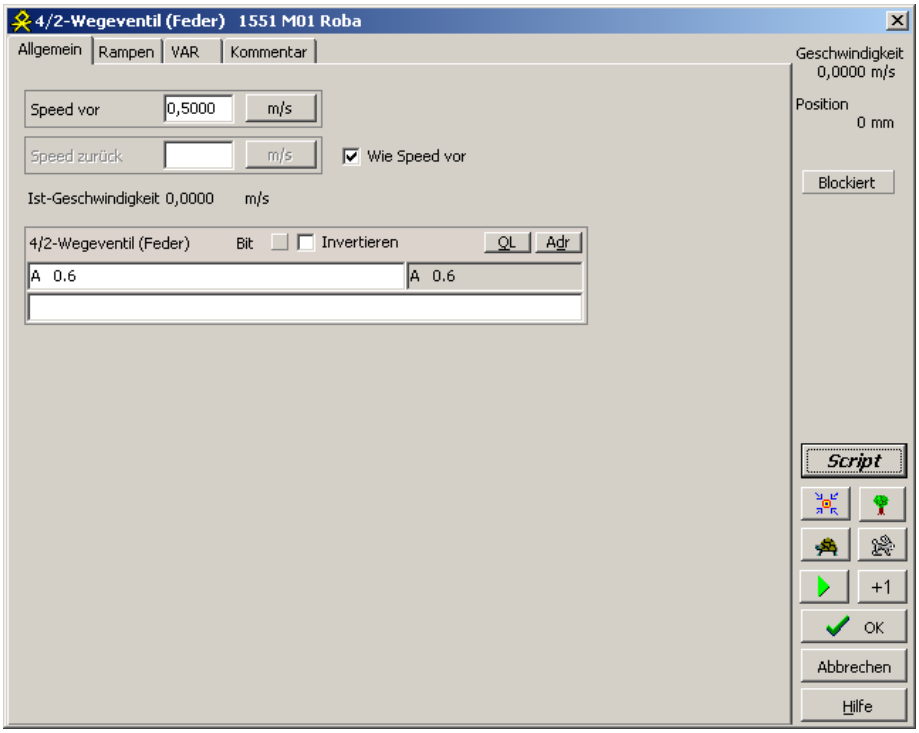
- **Bit-Motor mit zwei Geschwindigkeiten**



Hier können Sie noch ein weiteres Bit "schnell" angeben sowie die

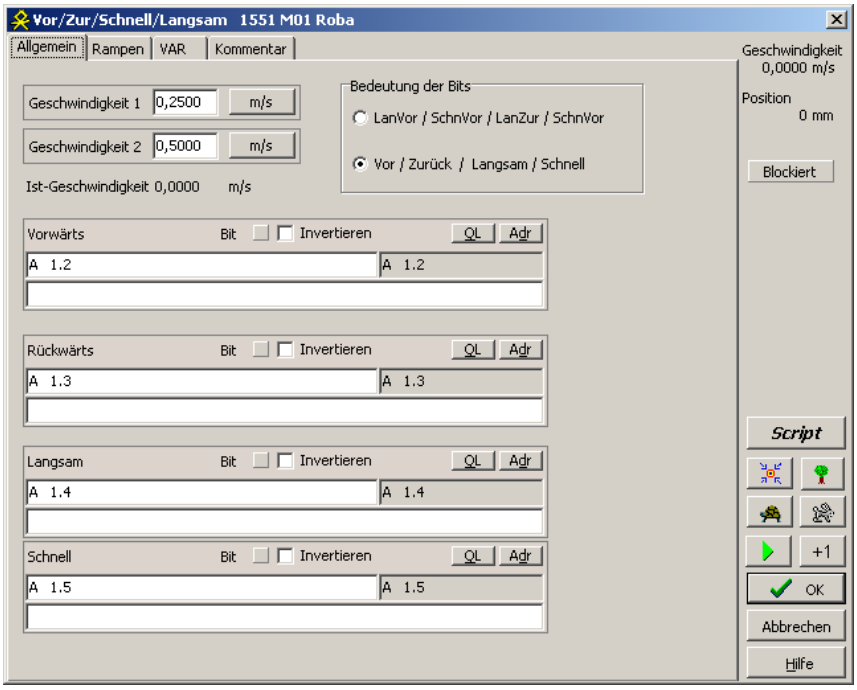
entsprechende Geschwindigkeit.

- **4/2 - Wegeventil (federrückgestellt)**



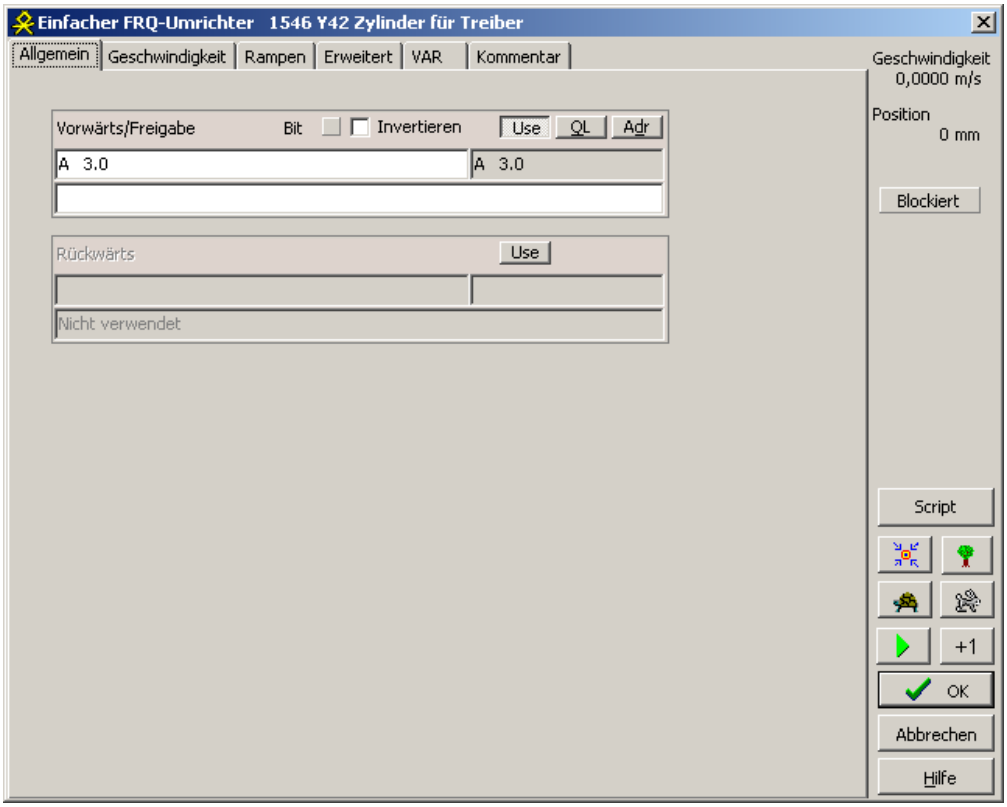
Hier brauchen Sie nur einen SPS-Ausgang angeben, ist er gesetzt, fährt der Antrieb vor, ist er nicht gesetzt, fährt der Antrieb zurück.

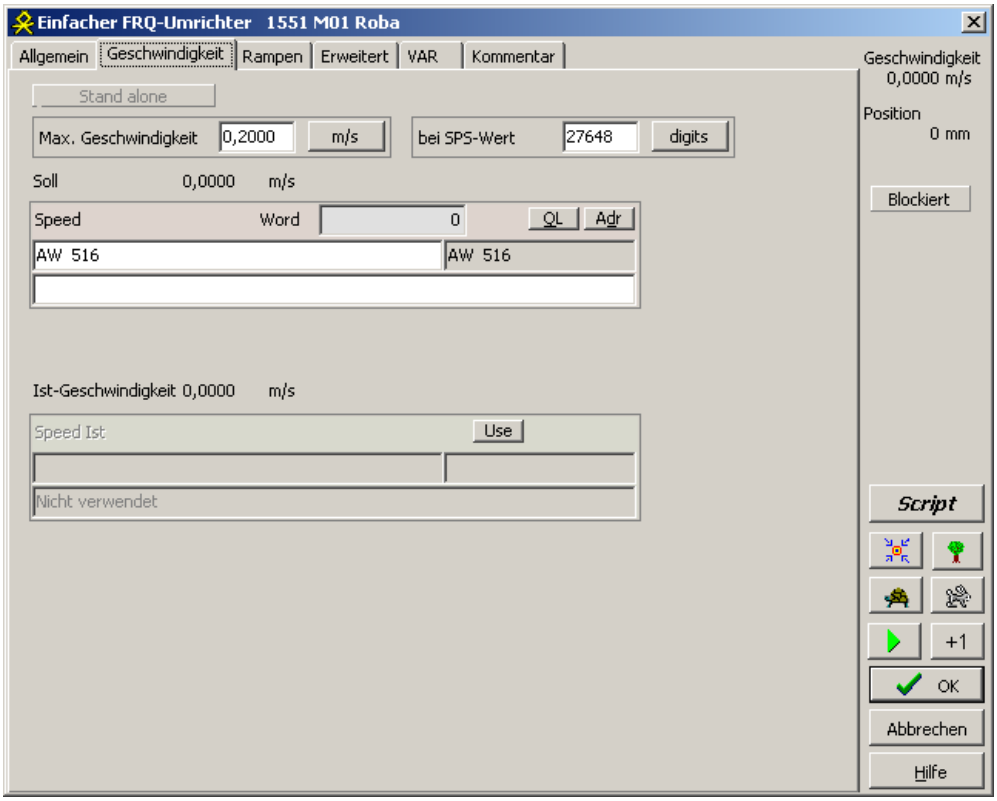
- **Vor/Zurück - Schnell/Langsam**



Hier werden Richtung und Geschwindigkeit über vier Bits vorgegeben.

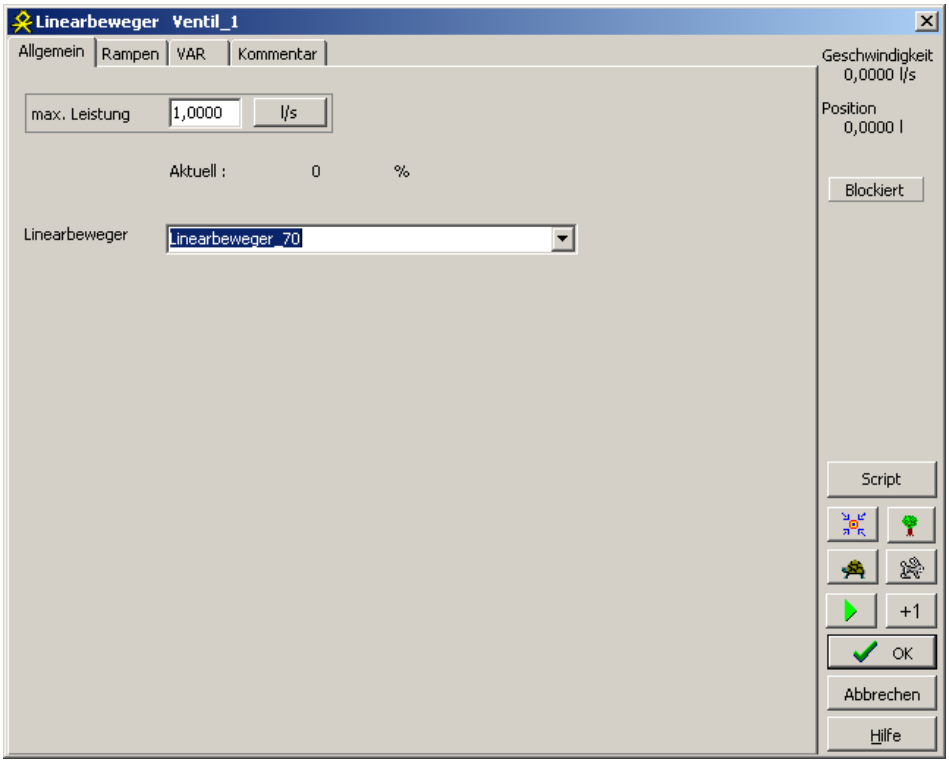
- **Einfacher Frequenzumrichter** mit variabler Geschwindigkeit



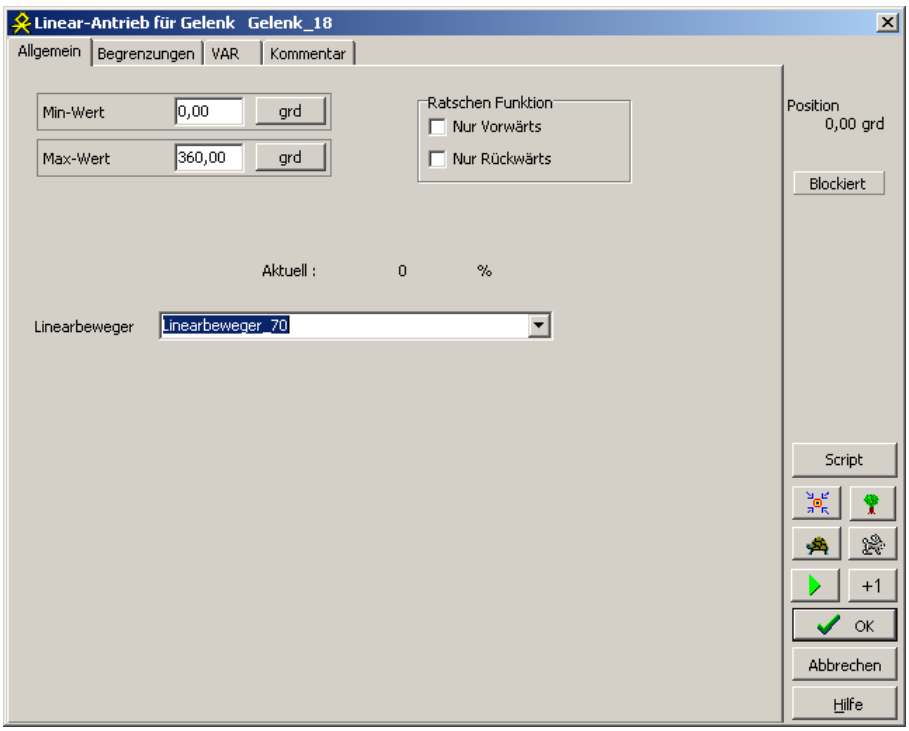


- [Ventil - Antrieb über Linearbeweger]

–

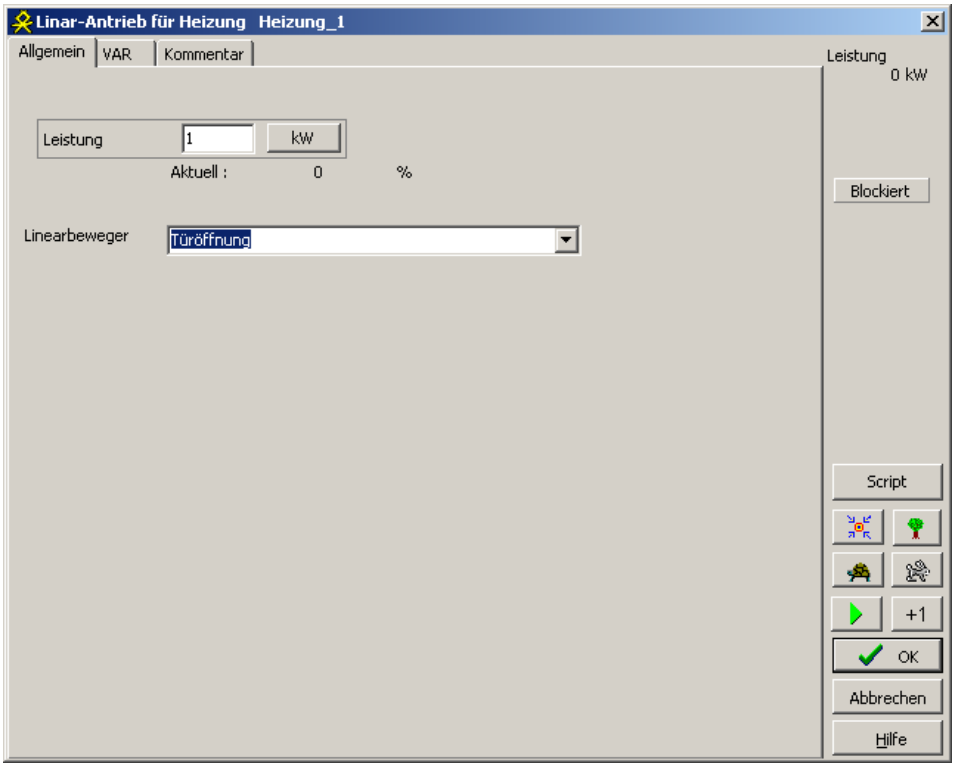


- Gelenk - Antrieb über Linearbeweger (Hebel)

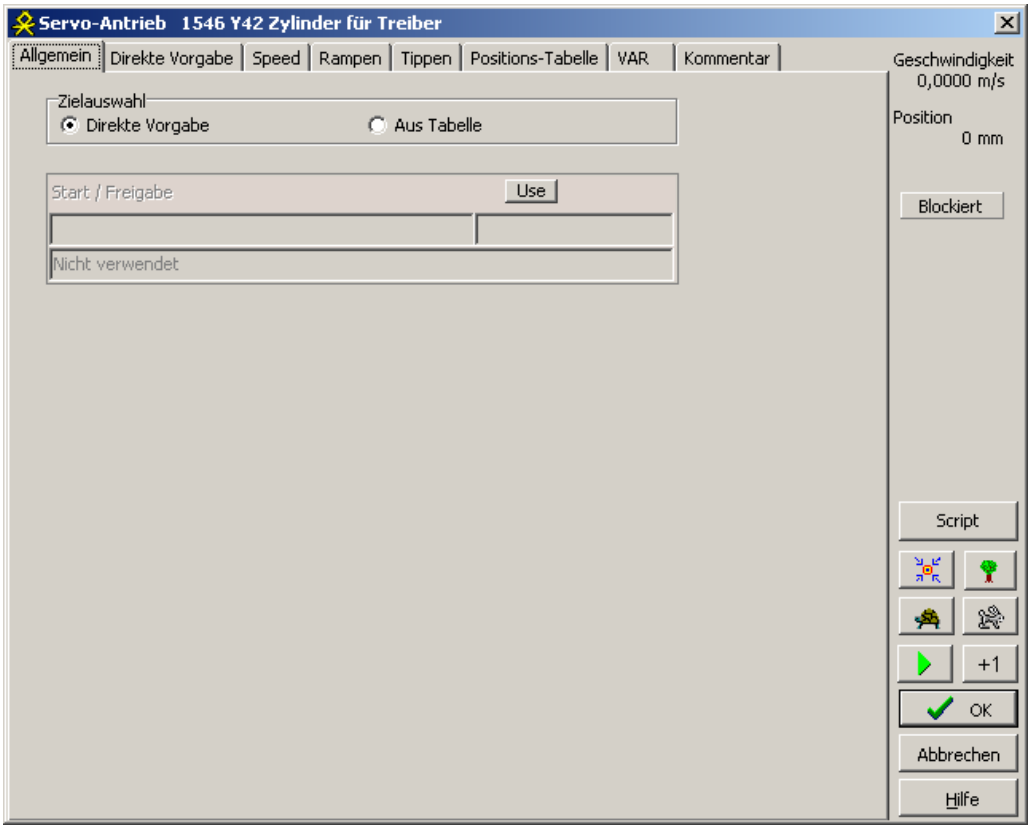


- Heiz- /Kühlantrieb über Linearbeweger

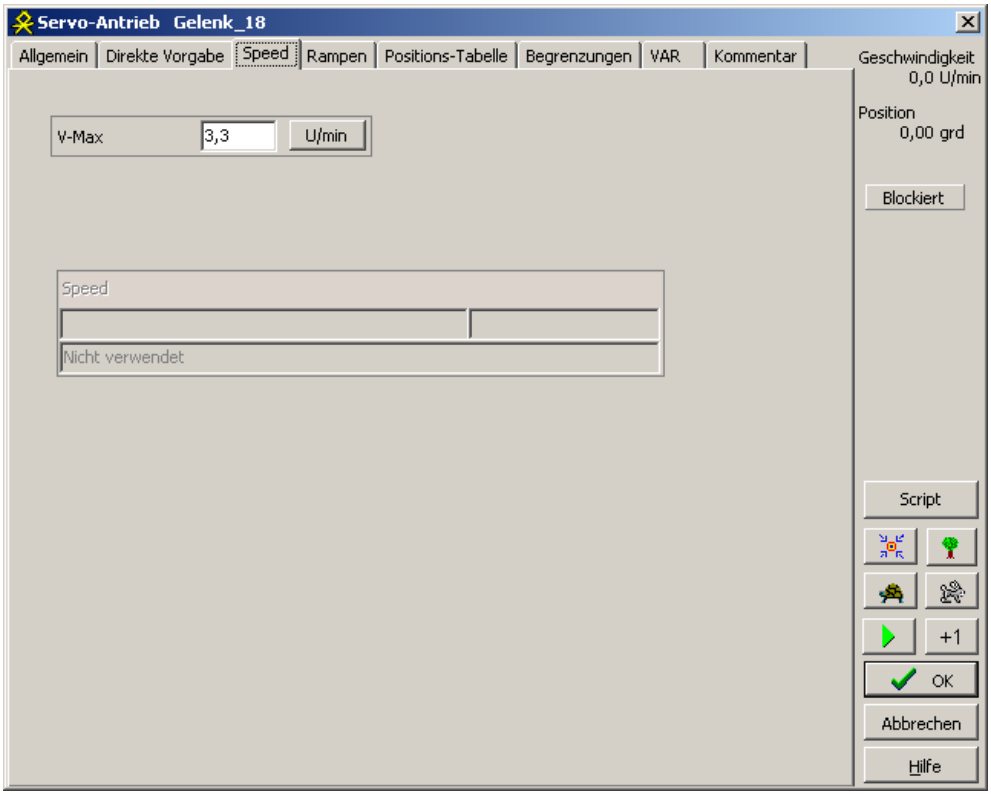
—



- **Servo-Antrieb** für Linearbeweger



- **Servo-Antrieb** für Gelenk



- **Absolut-Real-Antrieb** für Gelenk (nicht mehr unterstützt, bitte verwenden Sie den Servo-Antrieb)

2.6.1 Einfacher Frequenzumrichter

**

Dieser Antrieb entspricht einem frequenzgeregelten Motor oder einem Proportionalventil. Er muss mindestens an ein Word (E, A-, M- oder Daten-) der SPS angeschlossen werden. Neben der Adresse müssen Sie noch angeben, bei welchem Wert des Words die Maximalgeschwindigkeit erreicht werden soll. Wenn das Word in der echten SPS einen Analogausgang ansteuern soll, der einem Frequenzumformer +/- 10V vorgeben soll, müssen Sie hier 27648 eingeben.

Die Start- und Stoprampen können Sie entweder fest vorgeben oder von der SPS aus modifizieren. Die Startrampe wird immer dann verwendet, wenn der Antrieb freigegeben ist (auch wenn die Geschwindigkeit verringert wird). Die Stoprampe wird nur dann verwendet, wenn der Antrieb nicht freigegeben ist.

Sie können ein Freigabe-Bit festlegen. Wenn dieses Bit verwendet werden soll, ist der Antrieb nur freigegeben, wenn es aktiv ist (Ausnahme: wenn das Rückwärts-Bit verwendet werden soll, reicht es, wenn nur dieses gesetzt ist). Rückwärts läuft der

Antrieb auch, wenn ein negativer Sollwert vorgegeben ist.

Sie können weiterhin ein Rückwärts-Bit festlegen. Wenn dieses Bit gesetzt ist, wird der Sollwert negiert, d.h. mithilfe dieses Bits können Sie den Antrieb auch nur mit positivem Sollwert vor- und zurückfahren lassen.

Es gibt eine Vielzahl von Frequenzumformern am Markt und jeder wird etwas anders angesteuert. Wir haben versucht, diesen Antrieb so zu gestalten, dass er möglichst vielen Anforderungen gerecht wird. Eventuell werden Sie aber im OB3 ein paar Zeilen Code schreiben müssen, um exakt das Verhalten des von Ihnen eingesetzten Umrichters zu erreichen.

Dieser Antrieb wird auch dann verwendet, wenn keiner der anderen Antriebe geeignet erscheint, z.B., wenn man einen Motor mit 3 Geschwindigkeiten hat. Die Ansteuerung der Geschwindigkeit müssen Sie dann selbst programmieren, vorzugsweise im OB3.

Der einfache Frequenzumrichter bietet auch die Möglichkeit, dass ein Antrieb einem anderen genau folgt (Master-Slave-Betrieb). Wählen Sie dazu den Button “Erweitert”. Auf der nun erscheinenden Editier-Maske können Sie das fremde Element auswählen, dem der Antrieb folgen soll. Sie müssen auch ein Freilaufbit angeben. Wenn dieses Bit gesetzt ist, folgt der Antrieb nicht seinem Master, sondern verhält sich ganz wie ein normaler Frequenzumrichter. Falls Ihr Antrieb nie freilaufend sein soll, müssen Sie hier ein Bit angeben, dass “immer 0” ist. Wir empfehlen dafür das Bit M 0.0. (und M 0.1 für “immer 1”). Auch ein als Slave konfigurierter Antrieb muss freigegeben sein, damit er läuft. D.h. entweder Sie deaktivieren die Checkbox “Vorwärts-Eingang verwenden” oder sie müssen dafür sorgen, dass das angegebene Bit “1” ist, wenn der Antrieb laufen soll.

Der Word-Motor ist, ähnlich dem einfachen Frequenzumrichter, aber nicht so flexibel. Er ist nur noch aus Kompatibiltätsgründen vorhanden und sollte nach Möglichkeit nicht mehr benutzt werden.

2.6.2 Ventil - Antrieb über Linearbeweger

**

Bei dieser Form des Antriebes steuern Sie ein Element nicht direkt über die SPS, sondern indirekt über einen Linearbeweger, den Sie bereits erzeugt haben müssen. Der auf der Editiermaske anzugebende Maximalwert wird in Abhängigkeit der Stellung des Hotspots des Linearbewegers von 0 - 100 % variiert. Die Länge und Orientierung des Linearbewegers sind dabei gleichgültig. Die eine Endlage des Linearbewegers bedeutet immer 0 %, die andere immer 100 %. Sie können auch einen Linearbeweger mit integrierten Sensoren verwenden.

Bislang steht dieser Antrieb nur für das Ventil, das Gelenk, die Heizung und eher versehentlich für die Pumpe zur Verfügung.

2.6.3 Gelenk - Antrieb über Linearbeweger

***

Bei dieser Form des Antriebes steuern Sie ein Element nicht direkt über die SPS, sondern indirekt über einen Linearbeweger, den Sie bereits erzeugt haben müssen. Der vom Antrieb ausgegebene absolute Drehwinkel wird in Abhängigkeit der Stellung des Hotspots des Linearbewegers zwischen dem Minimalwert und dem Maximalwert variiert. Die Länge und Orientierung des Linearbewegers ist dabei gleichgültig, die eine Endlage des Linearbewegers entspricht immer dem Minimalwert, die andere immer dem Maximalwert. Sie können auch einen Linearbeweger mit integrierten Sensoren verwenden.

Mit den Ratschen-Funktionen “Nur vorwärts” bzw. “Nur rückwärts” können Sie einstellen, dass das Gelenk nur bewegt wird, wenn sich der Linearbeweger in die entsprechende Richtung bewegt.

Bislang steht dieser Antrieb nur für das Ventil, das Gelenk, die Heizung und eher versehentlich für die Pumpe zur Verfügung.

2.6.4 Heiz- /Kühlantrieb über Linearbeweger

**

Bei dieser Form des Antriebes steuern Sie ein Element nicht direkt über die SPS, sondern indirekt über einen Linearbeweger, den Sie bereits erzeugt haben müssen. Die von der Heizung ausgegebene Heizleistung wird in Abhängigkeit der Stellung des Hotspots des Linearbewegers zwischen dem Minmalwert und dem Maximalwert variiert. Die Länge und Orientierung des Linearbewegers ist dabei gleichgültig, die eine Endlage des Linearbewegers entspricht immer dem Minimalwert, die andere immer dem Maximalwert. Sie können auch einen Linearbeweger mit integrierten Sensoren verwenden.

Bislang steht dieser Antrieb nur für das Ventil, das Gelenk, die Heizung und eher versehentlich für die Pumpe zur Verfügung.

2.6.5 Servo-Antrieb für Linearbeweger

**

Bei dieser Form des Antriebs für den Linearbeweger können Sie auf der Editiermaske beliebig viele Positionen angeben, die dann während der Simulationszeit von der SPS aus über ihre Nummer angewählt werden.

Die Nummer des Ziels geben Sie über ein Word der SPS vor. Wenn Bit-codiert angeklickt ist, wird als Nummer die Position des niederwertigsten gesetzten Bits im Word verwendet. Das neue Ziel wird übernommen, wenn das Bit “Start/Freigabe” eine **steigende Flanke** hat oder sofort, falls die CheckBox “Immer 1” angeklickt ist. Auch wenn Sie ein Bit angegeben, können Sie konfigurieren, dass jede Sollwertänderung sofort (auch ohne steigende Flanke) übernommen wird: aktivieren Sie dazu die Checkbox “Sofort”.

Der Servoantrieb fährt nur, wenn das Bit “Start/Freigabe” ansteht, oder die

CheckBox “Immer 1” angeklickt ist.

Über das Bit “Position erreicht” meldet der Servoantrieb, wenn er den von Ihnen vorgegebenen Abstand zur Zielposition unterschritten hat.

Die Position wird mit der vorgegebenen Maximalgeschwindigkeit über Start- und Stoprampen angefahren. Die Rampensteilheit geben Sie nicht über eine Zeit vor, sondern über die Entfernung nach der der Antrieb die Maximalgeschwindigkeit erreicht haben soll (beim Starten) bzw. bei welcher Entfernung zum Ziel mit dem Bremsen begonnen werden soll (beim Stoppen). Wenn Sie die Rampe zu klein machen, wird der Antrieb nicht mehr kontrolliert stoppen können. Das ist in Wirklichkeit nicht viel anders und wir haben uns daher keine Mühe geben, diesen Effekt zu beseitigen. Sie können sowohl die Geschwindigkeit als auch die Rampe vom SPS-Programm mittels eines Pokes modifizieren.

Mit “Shift+Einfg” und “Strg+Einfg” können Sie neue Zeilen in die Tabelle der Positionen einfügen. Mit “Shift+Entf”, “Strg+Entf” und “Strg+X” können Sie Zeilen löschen. Denken Sie daran, die Ziel-Nummern im SPS-Programm anzupassen, wenn Sie durch Einfügen oder Löschen die Nummerierung der Positionen ändern.

Mit dem Button “Teach” können Sie die aktuelle Position des Linearbewegers übernehmen. Stellen Sie diese Position vorher mit dem Schieberegler “Hotspot” auf der Editiermaske des Linearbewegers ein.

Sie können die Sollposition des Hotspots auch direkt aus der SPS vorgeben. Siehe hierzu den Eintrag Direkte Vorgabe beim Servo-Antrieb weiter unten in diesem Kapitel.

2.6.6 Servo-Antrieb für Gelenk

***

Bei dieser Form des Antriebs für das Gelenk können Sie auf der Editiermaske beliebig viele Drehwinkel angeben, die dann während der Simulationszeit von der SPS aus über ihre Nummer angewählt werden.

Die Nummer des Ziels geben Sie über ein Word der SPS vor. Wenn Bit-codiert angeklickt ist, wird als Nummer die Position des niederwertigsten gesetzten Bits im Word verwendet. Das neue Ziel wird entweder übernommen, wenn das Bit “Start/ Freigabe” eine **steigende Flanke** hat oder sofort, falls die CheckBox “Immer 1” angeklickt ist.

Der Servoantrieb dreht nur, wenn das Bit “Start/Freigabe” ansteht, oder die CheckBox “Immer 1” angeklickt ist.

Über das Bit “Position erreicht” meldet der Servoantrieb, wenn er den von Ihnen vorgegebenen Abstand zum Zielwinkel unterschritten hat.

Der Zielwinkel wird mit der vorgegebenen Maximaldrehgeschwindigkeit über Start-

und Stoprampen angefahren. Die Rampensteilheit geben Sie nicht über eine Zeit vor, sondern über den Winkel, nach der der Antrieb die Maximalgeschwindigkeit erreicht haben soll (beim Starten) bzw. bei welchem Winkelabstand zum Ziel mit dem Bremsen begonnen werden soll (beim Stoppen). Wenn Sie die Rampe zu klein machen, wird der Antrieb nicht mehr kontrolliert stoppen können. Das ist in Wirklichkeit nicht viel anders und wir haben uns daher keine Mühe geben, diesen Effekt zu beseitigen.

Sie können sowohl die Geschwindigkeit als auch die Rampe vom SPS-Programm mittels eines Pokes modifizieren. Falls Sie als Einheit “Grad” gewählt haben, müssen Sie beide Größen in 0,1 Grad / s bzw. 0,1 Grad poken (z.B. für Geschwindigkeit 20 Grad/s müssen Sie in das SPS-Word des Pokes 200 transferieren). Falls Sie als Einheit “rad” gewählt haben, müssen Sie in 0,001 rad / s bzw. 0,001 rad poken.

Mit “Shift+Einfg” und “Strg+Einfg” können Sie neue Zeilen in die Tabelle der Positionen einfügen. Mit “Shift+Entf”, “Strg+Entf” und “Strg+X” können Sie Zeilen löschen.
Denken Sie daran, die Ziel-Nummern im SPS - Programm anzupassen, wenn Sie durch Einfügen oder Löschen die Nummerierung der Positionen ändern.

Mit dem Button “Teach” können Sie die aktuelle Position des Gelenkes übernehmen. Stellen Sie diese Position vorher mit dem Schieberegler “Winkel ohne Lösen der Achsmutter” auf der Editiermaske des Gelenkes ein.

Sie können den Sollwinkel auch direkt aus der SPS vorgeben. Siehe hierzu den Eintrag Direkte Vorgabe beim Servo-Antrieb weiter unten in diesem Kapitel.

2.6.7 Direkte Vorgabe beim Servo-Antrieb

**

Beim Servo-Antrieb können Sie die Position (oder Drehwinkel) auch direkt über das SPS-Wort vorgeben. Aktivieren Sie dazu die Checkbox “Direkte Vorgabe”. In das SPS-Word müssen Sie jetzt im Programm ein INT-Wert transferieren. Dabei gelten für den Linearbeweger und das Gelenk etwas unterschiedliche Regeln:

Beim Linearbeweger geben Sie die Position in den global eingestellten Einheiten ein (default ist mm, einstellbar unter Ansicht|Optionen|Anlage). Der Bezugspunkt ist der Anfang des Linearbewegers.

Beim Gelenk können Sie die Einheit “Grad” oder “rad” unmittelbar auf der Editiermaske des Servoantriebes einstellen. Wenn Sie als Einheit “Grad” gewählt haben, übergeben Sie den Soll-Winkel in 0,1 Grad. Wenn Sie den Servo-Antrieb z.B an Word AW 50 angeschlossen haben und einen Winkel von 72 Grad wünschen, programmieren Sie :

```
L    720
T  AW 50
```

Wenn Sie als Einheit “rad” eingestellt haben, müssen Sie den Winkel in 0,001 Radian vorgeben.

Die anderen Einstellungen auf der Editiermaske des Servoantriebes behalten ihre Funktion.

Beim Gelenk entspricht diese Betriebsart des Servoantriebes etwa dem alten Absolut-Real-Antrieb, allerdings mit mehr Komfort.

Wenn der Wertebereich eines INTs für Ihren Zweck nicht ausreicht, können Sie die Checkbox “DINT” aktivieren. Jetzt muss der Sollwert als DINT angegeben werden, dessen Wertebereich sollte für alle realistischen Anwendungsfälle ausreichen.

2.6.8 Word-Motor

**

Dieser Antrieb sollte nur noch von Lehrern eingesetzt werden, deren Schüler eine Version von Trysim kleiner als V2.9.12 verwenden. Professionelle Anwender sollten den neuen Antrieb Einfacher Frq-Umrichter einsetzen, der eine erweiterte Funktionalität hat.

Dieser Motor entspricht einem frequenzgeregelten Motor oder einem Proportionalventil. Er muss an ein Word (E, A-, M- oder Daten-) der SPS angeschlossen werden. Neben der Adresse müssen Sie noch angeben, bei welchem Wert des Words die Maximalgeschwindigkeit erreicht werden soll. Wenn das Word in der echten SPS einen Analogausgang ansteuern soll, der einem Frequenzumformer +/- 10V vorgeben soll, müssen Sie hier 27648 eingeben.

Der Word-Motor wird auch dann verwendet, wenn keiner der anderen Antriebe geeignet erscheint, z.B. wenn man einen Motor mit 3 Geschwindigkeiten hat. Die Ansteuerung der Geschwindigkeit müssen Sie dann selbst programmieren, vorzugsweise im OB3.

2.6.9 Word - Antrieb für Pumpe und Ventil

**

Diesen Antrieb können Sie ein- und ausschalten und außerdem noch die Förderleistung (bzw. max. Durchfluss beim Ventil) vorgeben.

Wenn dieser Antrieb für die Pumpe verwendet wird, führt eine negative Förderleistung zu einer Umkehrung der Förderleistung. Beim Ventil hat das Vorzeichen keine Bedeutung.

Absolut Real-Antrieb

***

Bevor Sie diesen Antrieb verwenden, sollten Sie prüfen, ob der Servo-Antrieb mit Direkt-Vorgabe nicht besser geeignet ist.

Dieser Antrieb liefert keine Geschwindigkeit, sondern eine absolute Position an sein

Element. Die Zahl muss als REAL in einem DWord der SPS vorliegen. Zurzeit steht dieser Antrieb nur für das Gelenk zur Verfügung. Ob Sie den Wert in Grad oder im Bogenmaß vorgeben wollen, können Sie auf der Editiermaske einstellen.

Falls in Ihrem SPS-Programm nicht der absolute Drehwinkel gebildet wird, sondern Sie diesen (nur für die Simulation) aus anderen Daten bestimmen, sollten Sie den Code zur Umrechnung im OB3 plazieren.

2.6.10 1 Bit - Antrieb für Pumpe und Ventil

*

Dies ist ein recht einfacher Antrieb. Wenn das Bit gesetzt ist, wird über die Hochlauframpe die Förderleistung angefahren, wird es wieder ausgeschaltet, wird sie über die Runterlauframpe wieder auf 0 gefahren.

Wenn dieser Antrieb für die Pumpe verwendet wird, führt eine negative Förderleistung zu einer Umkehrung der Förderleistung. Beim Ventil hat das Vorzeichen keine Bedeutung.

2.6.11 Antriebe des Linearbewegers

*

- Bit-Motor
  Dieser Antrieb kann mit einer Geschwindigkeit vor- und zurück fahren. Sie können entweder ein oder zwei SPS-Ausgänge angeben. Außerdem können Sie Vorwärts/Rückwärts mit Impulsverhalten wählen. In diesem Fall verhält sich der Antrieb wie ein 5/2-Wegeventil mit Impulsverhalten.

- Bit-Motor mit zwei Geschwindigkeiten
  Hier können Sie noch ein weiteres Bit “schnell” angeben und entsprechend eine weitere Geschwindigkeit.

- 4/2 - Wegeventil
  Hier brauchen Sie nur einen SPS-Ausgang angeben, ist er gesetzt, fährt der Antrieb vor, ist er nicht gesetzt, fährt der Antrieb zurück.

- Einfacher Frequenzumformer
  Dieser Motor entspricht einem frequenzgeregelten Motor oder einem Proportionalventil. Er muss an ein Word (E, A-, M- oder Daten-) der SPS angeschlossen werden.
  Neben der Adresse müssen Sie noch angeben, bei welchem Wert des Words die Maximalgeschwindigkeit erreicht werden soll. Wenn das Word in der echten SPS einen Analogausgang ansteuern soll, der einem Frequenzumformer +/- 10V vorgeben soll, müssen Sie hier 27648 eingeben.

- Servo-Antrieb für Linearbeweger
  Hier geben Sie während der Erstellung der Simulation einige Positionen vor, die dann von der SPS aus über ihre Nummer ausgewählt werden und automatisch

angefahren werden. Sie können die Sollposition auch direkt vorgeben.

- Kurbelantrieb
  Hier wird ein Gelenk / eine Achse verwendet, um den Linearbeweger wie ein Pleulstange anzutreiben.

2.6.12 Bit - Motor für Gelenk

**

Bei dieser Form des Antriebs für das Gelenk müssen Sie eine Geschwindigkeit in U/min vorgeben. Zur Auswahl der aktuellen Geschwindigkeit stehen verschiedene Möglichkeiten zu Verfügung, die im Wesentlichen denen des Linearbewegers entsprechen. Bitte beachten Sie, dass sich bei der Auswahl des Antriebtyps (oben links in der Maske) die Anzahl und die Bedeutung der Bits, die zur Steuerung benötigt werden, ändern. Genauere Informationen zu den einzelnen Alternativen finden Sie im nachfolgenden Kapiteleintrag.

Sie können zwei Arten von Rampen vorgeben: Start-Rampe und Stop-Rampe. Die angegebene Zeit bezieht sich immer auf die Zeit, die der Antrieb benötigt, um von Null auf die “Geschwindigkeit Normal” zu kommen, bzw. von dort auf Null. Die Start-Rampe wird immer dann verwendet, wenn die Soll-Geschwindigkeit ungleich Null ist, sonst wird die Stop-Rampe verwendet. D.h. auch wenn Sie einen Antrieb von Schnell auf Langsam schalten, wird die Start-Rampe verwendet, obwohl der Motor langsamer wird. Die Stop-Rampe wird nur verwendet, wenn der Motor ausläuft. Falls in Ihrer echten Anlage ein Motor mit Bremse eingebaut ist, sollten Sie die Stop-Rampe auf einen sehr kleinen Wert einstellen. Wenn Ihr wirklicher Motor nur eine Geschwindigkeit hat, können Sie in diesem Fall das Bremsverhalten explizit dadurch modellieren, dass Sie einen 2-Geschwindigkeiten-Motor wählen, für die kleine Geschwindigkeit eine 0 vorgeben und die Stoprampe entsprechend dem Bremsverhalten Ihres Motors einstellen. Wenn Sie in Wirklichkeit einen Motor mit 2 Geschwindigkeiten einsetzen, dann haben Sie diese Möglichkeit mangels Parameter nicht und müssen stattdessen etwas improvisieren.
Eine Ausnahme zu dem eben Geschilderten bilden die Rampen beim 4/2-Wegeventil. Hier wird die Start-Rampe immer dann verwendet, wenn das Bit “1” ist und die Stop-Rampe, wenn es “0” ist. Hierdurch können Sie z.B. verschieden gedrosselte Anschlüsse eines doppelt wirkenden Zylinders nachbilden.

Für alle Antriebe können Sie auch Begrenzungen des Drehwinkels vorgeben. Dazu müssen Sie zunächst entscheiden, ob die Begrenzungswinkel in Grad oder in Rad angezeigt werden sollen. Der einfachste Weg, die Begrenzung vorzugeben, besteht darin, das Gelenk mittels des Schiebereglers “Winkel ohne Lösen der Achsmutter” in die gewünschte Stellung zu bringen und dann einen der Buttons “Teach Min” oder “Teach Max” zu betätigen.

Die beiden Geschwindigkeiten und die Rampen können über Word-Pokes auch während der Laufzeit von der SPS aus geändert werden. Die Geschwindigkeiten werden in 0,01 U/min gepoked, die Rampen in 0,001 s. Dabei ist es vielleicht etwas lästig, dass die Pokes immer um das Gelenk kreisen - blenden Sie sie einfach aus über Bearbeiten | Ausblenden..

Wichtig: Wenn Sie eine Begrenzung für einen zwei-Geschwindigkeiten-Antrieb festgelegt haben, müssen Sie natürlich, wie im echten Leben, selbst dafür sorgen, dass der Antrieb vorher auf langsam geschaltet wird. Sonst prallen Ihre Maschinenteile gegen den Anschlag. In TrySim wäre dies nicht weiter tragisch, in der Realität aber schon.

2.6.13 Bit-Antriebe des Gelenks

**

1.) Start-Stop
Dies ist ein Motor, der nur eine Geschwindigkeit hat und nur in eine Richtung laufen
kann. Er kann natürlich nicht sinnvoll mit einer Begrenzung eingesetzt werden.

2.) 4/2-Wegeventil
Dieser Motor steht nie still. Er läuft entweder vorwärts (wenn der Ausgang eine “1” führt) oder rückwärts (wenn der Ausgang eine “0” führt). In den meisten Fällen wird es sinnvoll sein, ihn mit Begrenzungen des Drehwinkels zu verwenden.

3.) Vor-Zurück
Dieser Motor läuft vorwärts, wenn das erste Bit gesetzt ist und rückwärts, wenn das zweite Bit gesetzt ist, aber immer mit der gleichen Geschwindigkeit. Ist kein Bit gesetzt, dann läuft er mit der Stop-Rampe aus. Er kann sowohl mit als auch ohne Begrenzung verwendet werden.

4.) Langsam / Schnell
Dieser Motor läuft nur vorwärts, dafür aber mit zwei Geschwindigkeiten. Wenn das “Schnell”-Bit gesetzt ist, läuft er schnell, unabhängig vom Zustand des “Langsam”-Bits. (Anmerkung: Das ist u.U. anders in Ihrer wirklichen Schaltung: dort kann es sein, dass der Motor nur schnell läuft, wenn beide Ausgänge gesetzt sind.)
Eine Angabe von Begrenzungen ist hier nicht sinnvoll, da man das Gelenk nie wieder zurückbekommen würde.

5.) Vor / Zurück / Schnell
Für die Ansteuerung dieses Motors benötigen Sie drei Bits. Die ersten beiden legen die Drehrichtung fest, mit dem dritten bestimmen die Geschwindigkeit. Ist das “Schnell”-Bit “0”, so dreht sich das Gelenk langsam, ist es “1” dreht sich das Gelenk entsprechend schnell. Dieser Antrieb lässt sich sowohl mit als auch ohne Begrenzungen verwenden. Schaltungstechnisch realisiert man einen solchen Antrieb, indem man das “Langsam”-Schütz über einen Öffner des “Schnell”-Schützes ansteuert.

6.) Vor / Zurück / Langsam / Schnell
Dieser Antrieb ist fast wie der vorhergehende (Nr.5), nur muss hier auch die kleine Geschwindigkeit explizit angesteuert werden. Schaltungstechnisch sieht dies so aus, dass man eine Wendeschützschaltung gefolgt von z.B. einer

Dahlander-Schaltung hatund dass jedes Schütz durch einen eigenen SPS-Ausgang angesteuert wird.

7.) Langsam Vor / Schnell Vor / Langsam Zurück / Schnell Zurück
Auch dieser Antrieb entspricht im Wesentlichen den Antrieben 5.) und 6.). Er benötigt 4 Bits zur Ansteuerung, deren Bedeutung selbsterklärend ist. Achten Sie darauf, dass jeweils nur eins der Bits aktiv ist. Es sind hauptsächlich Frequenzumrichter, die so angesteuert werden, obwohl dies natürlich auch mit konventioneller Schaltungstechnik realisiert werden könnte, was jedoch in Verbindung mit einer SPS eher unüblich ist.

2.6.14 Kurbel-Antrieb

***
Bei dieser Form des Antriebs müssen Sie zunächst ein Gelenk erzeugen und an die SPS anschließen.
Dieses Gelenk kann dann verwendet werden, um einen Linearbeweger oder ein weiteres Gelenk anzutreiben.

Wenn Sie einen Linearbeweger mit der Kurbel antreiben, wird der Hotspot bei jeder vollständigen Umdrehung des Gelenkes einmal vom einen Ende bis zum anderen und zurück gefahren.

Wenn Sie ein weiteres Gelenk mit der Kurbel antreiben, werden die beiden Gelenke synchron gekoppelt, als seien sie über Zahnräder und eine Kette miteinander verbunden.

Sie können das Übersetzungsverhältnis auch ändern. Wenn Sie in das entsprechende Feld eine Zahl größer als “1” eintragen, bewegt sich das Element schneller, tragen Sie eine Zahl kleiner als “1” ein, bewegt es sich langsamer.

2.7 Analogwert-Verarbeitung

Häufig werden wir gefragt, ob es in TrySim auch Anlogwert-Verarbeitung gibt. Dazu ist folgendes zu sagen: Der gesamte E-/A - Adressraum von je 64 KB steht in TrySim dauernd ohne irgendwelche Konfiguration zur Verfügung. Ob es sich dabei um binäre oder analoge Signale handelt, hängt allein davon ab, welche Elemente der Simulation Sie anschließen. Wenn Sie an ein EW z.B. einen Füllstandssensor anschließen, dann verhält sich dieses EW wie ein Analog-Eingang (n.b. Sie können im Programm auch PEW schreiben, zwischen EW und PEW wird in TrySim genauso wenig unterschieden wie zwischen AW und PAW).
TrySim ist dazu entwickelt worden, Programme in einer Umgebung zu testen, die sich wie eine echte Maschine verhält. Über welche Hardware die Informationen von den Sensoren zum Programm und vom Programm zu den Aktoren kommen, ist dabei nicht wichtig. Natürlich können auch bei der Auswahl, Installation und Konfiguration dieser Hardware viele Fehler gemacht werden, aber dies sind Fehler, die durch eine Simulation realistischerweise nicht im Vorfeld zu vermeiden sind.

Elemente, die an analoge Eingänge angeschlossen werden können sind z.Z.:
Schieberegler
Spion
Barcode-Scanner (wenn man denn einen Barcode analog nennen will)
Impulsgeber für Förderband und Kette
Ultraschall Entfernungssensor
Thermometer
Füllstandssensor
Durchflussmesser
Analysator
Linearbeweger mit integriertem Positionsgeber
Digital-Eingabe

Elemente, die an analoge Ausgänge angeschlossen werden können sind z.Z.:
Förderband mit Word-Motor
Freibeweglicher Punkt
Generator (der Barcode des Dynamiks kann vorgegeben werden)
Linearbeweger mit Word-Motor
Kette mit Word-Motor
Gelenk
Bogenförderband mit Word-Motor
Drehbares Förderband mit Word-Motor
Drehtisch mit Word-Motor
Dreher mit Word-Motor
Heizung mit regelbarer Leistung
Digital-Anzeige
Oszillograph
Pumpe mit Word-Antrieb
Ventil mit Word-Antrieb
Schieberegler als Analoganzeige verwendet

Wenn Sie weitere Elemente mit analogem Anschluss benötigen, sollten Sie mit uns Kontakt aufnehmen. Wir bemühen uns, Vorschläge für neue Elemente, solange sie in den Rahmen von TrySim passen, schnellstmöglich zu verwirklichen.

2.8 Dynamische Elemente der Simulation

2.8.1 Normale Dynamiks

Diese “Dynamiks” genannten Elemente entsprechen dem Material, welches die Anlage bearbeitet. Sie werden während der Laufzeit durch einen Generator erzeugt und können auch wieder verschwinden. Die Dynamiks können Sie auf verschiedene Arten von der Maschine bewegen und detektieren lassen.

In früheren Versionen von TrySim nannten wir die Dynamiks **Kisten**, aber dieser Name hat sich als äußerst unglücklich erwiesen, da zum einen ständig Verwechslungen mit dem statischen Element Kasten auftraten und zum anderen der Eindruck erweckt wurde, TrySim wäre nur zum Umherschieben von Kisten geeignet.

Ab Version 2.1 gibt es zwei Arten von Dynamiks:

1. Normale Dynamiks
2. Drehbare Dynamiks

Die drehbaren Dynamiks sind Elemente mit ganz anderen Eigenschaften als die Normalen, da es uns nicht gelungen ist, die Normalen einfach drehbar zu machen. In den folgenden Versionen wird das Verhalten der drehbaren Dynamiks aber immer mehr dem der Normalen angepasst werden. Wenn die heute schnellsten Rechner zum Standard geworden sind, wird es nur noch eine Sorte, nämlich die Drehbaren geben. Bis dahin lassen wir beide Sorten nebeneinander bestehen, denn die Drehbaren brauchen ungleich viel mehr Rechenleistung als die normalen Dynamiks. Es gibt aber mittels des Dyn-Konverters die Möglichkeit, beide Sorten auch während der Simulationslaufzeit ineinander umzuwandeln, so können Sie das Verhalten der Dynamiks der jeweiligen Situation anpassen.

Im Editiermodus können Sie die Dynamiks mit der Maus bewegen und so bestimmte Test-Situationen gezielt herbeiführen. Die drehbaren Dynamiks können Sie mit Schiebereglern um alle Achsen drehen, wählen Sie dazu den Punkt Drehen aus dem Kontextmenü.

Dynamiks können bei der Erzeugung mit einem Barcode versehen werden, der mit Hilfe des Barcode-Lesers detektiert werden kann.

Die “Einsinktiefe” (einstellbar in der Editiermaske der Dynamiks), dient dazu, das Verhalten der Dynamiks realistischer zu machen, wenn kleine Niveauunterschiede zwischen verschiedenen Bändern vorhanden sind. Sie sollte nicht größer als 10 mm eingestellt werden.

Auf der Editiermaske der Dynamiks können Sie auch einstellen, ob die Dynamiks gespeichert werden sollen. Diese speicherbaren Dynamiks (SemiDynamiks) werden nicht von Vernichtern zerstört. Sie sind nützlich, wenn Ihre Anlage zirkulierende Paletten, Wagen oder Traversen hat. Schimpfen Sie nicht auf TrySim, wenn Ihre Steuerung nach dem Umschalten auf speicherbare Dynamiks und anschließendem Neustart von TrySim nicht mehr funktioniert: in einer echten Anlage würden Sie nach dem Austausch einer defekten CPU vor dem gleichen Problem stehen.

Die normalen Dynamiks sind immer rechtwinklig ausgerichtet, es gibt jedoch auch die drehbaren Dynamiks, für die diese Einschränkung nicht gilt.

Die Lichtschranke detektiert Dynamiks, mit dem Ultraschall-Sensor lässt sich deren Position bestimmen. Mit dem Haken können Dynamiks gegriffen und mit dem Schieber sowie dem Keil in der XY-Ebene manipuliert werden. Mit dem Verschmelzer können Sie mehrere Dynamiks zu einem vereinigen. Der Teiler zersägt Dynamiks, während die Presse sie verkleinert. Mittels des Stickers können auch statische Elemente an einem Dynamik befestigt werden.

Wenn Sie auf der Dynamik-Maske des Generators die Option “Füllbar” gewählt

haben, verhalten sich die Dynamiks wie ein Behälter, d.h., Sie können die Dynamiks mit dem Rohr, dem Ventil und der Pumpe befüllen. Mit dem Füllstandssensor können Sie dann die eingefüllte Menge erfassen. Der Niveau-Schalter überprüft den Füllstand auf einen Grenzwert. Mit dem Analysator erfassen Sie die Zusammensetzung der Flüssigkeit in den Dynamiks und mit dem Reaktor können Sie schließllich die Zusammensetzung verändern.

Häufige Frage bei füllbaren Dynamiks: “Warum sehe ich den Füllstand nicht, wie z.B. im Beispiel Abfüllanlage?” Antwort: In der X-Y-Draufsicht kann man die Füllstände nicht sehen.

2.8.2 Drehbare Dynamiks

***

Im Gegensatz zu den normalen Dynamiks, die immer rechtwinklig ausgerichtet sind, können diese Dynamiks in allen Raumrichtungen gedreht werden.

Zur Erzeugung von drehbaren Dynamiks wählen Sie auf der Editiermaske des Generators unter “Dynamiks” die Checkbox “Drehbar” an, oder wandeln ein normales Dynamik mittels des Dyn-Konverters um.

Das Verhalten dieser Elemente ist (noch) sehr durch die Ecken bestimmt. Viele der Operationen, die mit normalen Dynamiks durchgeführt werden können, funktionieren bei den Drehbaren nur, wenn mindestens eine Ecke mit dem ausführenden Element überlappt. Sie können jedoch auch durch die Optionen “Center-Punkte” und “Kanten-Punkte” die Zahl der aktiven Punkte um die 6 Flächenmitten bzw. 12 Kantenmitten erhöhen. Da viele Anwendungen ausschließlich Drehungen in der XY-Ebene benötigen, wobei ein Kippen der Dynamiks eher störend ist, können Sie durch die Anwahl der Option “Nur in XY-Ebene” alle anderen Drehungen ausschließen.

Im Editiermodus können Sie die Dynamiks mit der Maus bewegen und so bestimmte Test-Situationen gezielt herbeiführen oder sie nach Anwahl des Punktes “Drehen” aus dem Kontextmenü mit drei Schiebereglern um alle Achsen drehen.

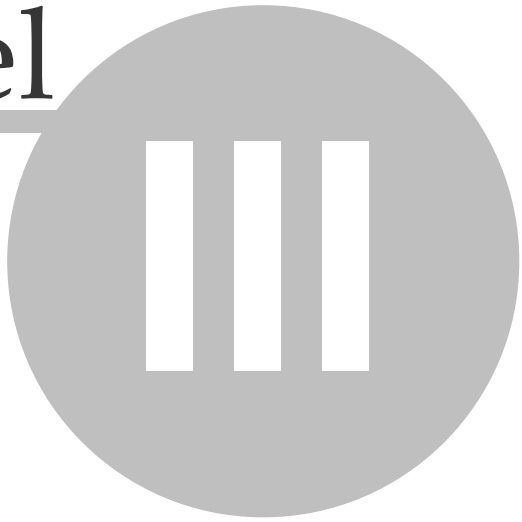
Die drehbaren Dynamiks können noch nicht gespeichert werden.

Bei den echten Drehungen gibt es immer noch einige Einschränkungen. So können Sie drehbare Dynamiks nicht aufeinander stapeln und auch nicht gegenseitig schieben lassen. Die Auswirkungen dieser Einschränkungen lassen sich etwas vermindern, wenn Sie den Dyn-Konverter verwenden, mit dem sich die beiden Arten von Dynamiks leicht ineinander umwandeln lassen. Da die Simulation von Drehungen einen hohen Rechenaufwand erfordert, lässt sich mit Hilfe dieses Elementes die maximale Simulationsgeschwindigkeit beim Verwenden sehr vieler Dynamiks erhöhen.

Die statischen Elemente Bogenförderband, Drehtisch, drehbares Förderband sind speziell für die drehbaren Dynamiks gemacht. Mithilfe des Ausrichters können versehentlich schräge Dynamiks wieder gerichtet werden.

SPS

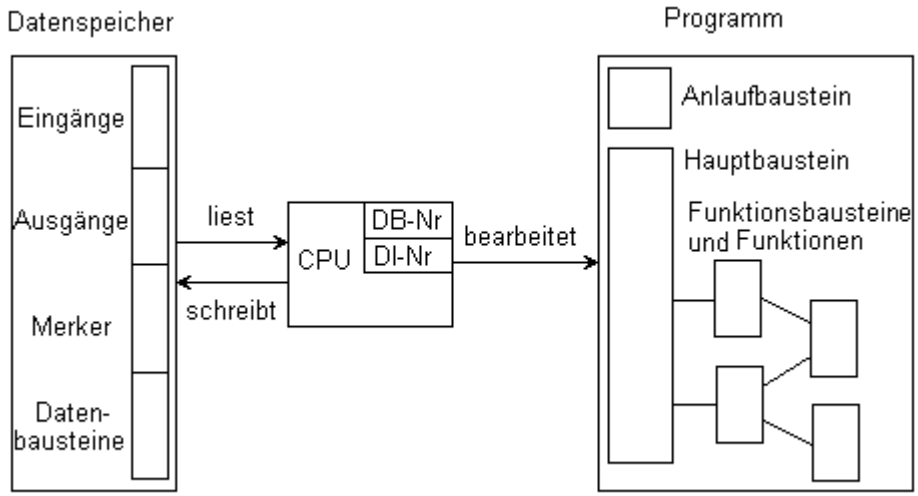
Kapitel



3 SPS

3.1 Einführung

Obwohl die SPS ein zentraler Bestandteil von TrySim ist, bekommen Sie sie nie zu sehen. Wenn Sie wissen möchten, wie eine SPS aussieht, schauen Sie in die Kataloge der Hersteller.
Der Aufbau der von TrySim simulierten SPS ist folgender:



Aufbau der SPS

Im weiteren Verlauf werden die einzelnen Teile kurz beschrieben. Dabei werden Grundkenntnisse über die Funktionsweise von Computern vorausgesetzt. Wir verweisen auf die reichhaltig vorhandene Fachliteratur. Wenn Sie gar nicht wissen, was eine SPS überhaupt ist, lesen Sie bitte im "Kapitel für Anfänger", zu finden unter "Beispielsitzung".

3.2 Datenspeicher

3.2.1 Über Datenspeicher

In einer wirklichen SPS unterscheidet man im Wesentlichen zwischen vier verschiedenen Bereichen des Datenspeichers. Die Eingänge werden zu Beginn jedes Bearbeitungszyklus auf den Wert gesetzt, den der angeschlossene Sensor liefert. Die CPU liest die Eingänge und bestimmt durch das Programm den Zustand der Ausgänge. Mit den Ausgängen werden dann am Ende des Zyklus die angeschlossenen Aktoren betätigt. Die Merker und die Datenbausteine werden verwendet, um Zwischenergebnisse und Parameter zu speichern.

Diese Struktur haben wir natürlich übernommen, im Gegensatz zu einer wirklichen SPS hat die Unterteilung in Eingänge, Ausgänge, Merker und Datenbausteinen in TrySim aber nur symbolischen Charakter: Sie können z.B. mit Eingängen Aktoren betätigen und den Zustand von Merkern durch Sensoren beeinflussen. Wenn Ihr Programm später auf einer wirklichen SPS laufen soll, werden Sie von diesen

Möglichkeiten selbstverständlich keinen Gebrauch machen, während des Testens jedoch sind sie von großem Wert. Nehmen wir an, Ihre Maschine stapelt Kisten und die Steuerung soll speichern, wie viele Kisten an einer bestimmten Stelle stehen. Irgendwo in Ihrem Programm sei ein Fehlerund die Zahl stimmt gelegentlich nicht mehr. Mit TrySim nehmen Sie einfach eine Digitalanzeige, schließen Sie an das Datenwort mit der Kistenzahl an und setzen sie direkt unter den Stapel. Jetzt sehen Sie sofort, wenn die Zahl nicht stimmt und können den Fehler schnell einkreisen.

3.2.2 Bezeichnung der Ein- und Ausgänge sowie der Merker

Diese Datenbereiche sind byteweise organisiert und bestehen jeweils aus 65536 Bytes.
Sie können die Daten in diesen Bereichen bit-, byte-,wort- und doppelwortweise ansprechen. Beispiele:

```
E  5.2      // Bit Nr 2 vom Eingangsbyte 5 (Bitnummern von 0 - 7)
AB 6        // Ausgangsbyte 6
MW 10       // Merkerwort 10, besteht aus MB10 (high,h) und MB 11 (low,l)
ED 2        // Eingangsdoppelwort 2 : EB 2 (hh), EB3 (hl), EB4(lh),EB5(ll)
```

Unter **Ansicht|Optionen|SPS-Editor** können Sie auch die IEC-Mnemonics einstellen, dann werden Eingänge nicht mit E, sondern mit I und Ausgänge nicht mit A, sondern mit Q bezeichnet.

3.2.3 Bezeichnung der Daten in Datenbausteinen

Es gibt bis zu 4095 Datenbausteine, die im Prinzip je 65536 Bytes enthalten können. Wieviel Sie davon benutzen können hängt allerdings vom Speicherausbau Ihres Computers und vor allem von der Größe des Datenspeichers Ihres Zielsystems ab. Die Datenbausteine müssen Sie erzeugen und dabei festlegen, welche Größe sie haben. Die Daten in den Datenbausteinen sind ebenfalls byteweise organisiert und werden genau wie die anderen Daten bezeichnet, jedoch mit dem Zusatz, in welchem Datenbaustein sie sich befinden. Beispiele:

DB17.DBX 6.2	Bit Nr. 2 im Datenbyte 6 des Datenbausteins 17
DB22.DBB 8	Byte 8 im Datenbaustein 22
DB39.DBW 11	Datenwort 11 im DB 39, bestehend aus DBB11 und DBB12
DB88.DBD 20	Datendoppelwort 20 im DB 88

Zur Vereinfachung der Schreibweise, wenn Sie nacheinander viele Daten des gleichen Datenbausteins ansprechen wollen, können Sie einen Datenbaustein aufschlagen. Bis Sie einen anderen Datenbaustein aufschlagen, können Sie dann die Bezeichnung des Datenbausteins weglassen. Diese Möglichkeit gibt es aber nur innerhalb des Programms. Wenn Sie Daten von der Anlagensimulation aus ansprechen wollen, müssen Sie immer den Datenbaustein angeben. Intern bedeutet das Aufschlagen eines Datenbausteines, dass seine Nummer in ein Register der CPU eingetragen wird. Im Schema des SPS-Aufbaus ist dieses Register durch den kleinen Kasten DB-Nr symbolisiert.

3.2.4 Instanzdatenbausteine

Das Konzept der Instanzdatenbausteine wird bei den Funktionsbausteinen ausführlich erklärt. Im Datenspeicher der SPS sind dies ganz normale Datenbausteine, deren Daten Sie wie gewohnt ansprechen könnten. Üblicherweise wird das aber nicht getan. In der CPU gibt es ein weiteres Register (im Bild DI-Nr), in dem die Nummer des gerade aufgeschlagenen Instanzdatenbausteins gespeichert wird. Wenn Sie Daten aus diesem Baustein ansprechen wollen, müssen Sie folgende Notation verwenden:

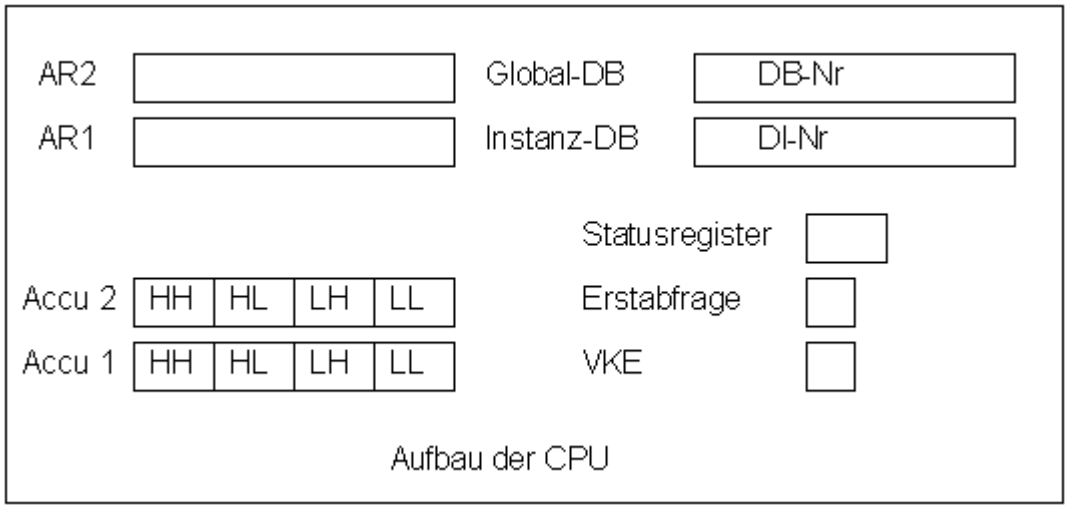
DIX 7.2	Bit Nr. 2 vom Byte 7	des DBs mit der Nummer im DI- Register
DIB 9	Byte 9	´´
DIW 10	Wort 10	´´
DID 16	Doppelwort 16	´´

Auch dies ist nur eine Möglichkeit die Daten anzusprechen, üblicherweise werden Daten aus Instanzdatenbausteinen über den Namen adressiert, der im Deklarationsteil des zugehörigen FB's festgelegt worden ist.

3.3 CPU

3.3.1 Einführung

Die CPU (Central Processing Unit) ist das Herz jedes Computers. Sie liest die Informationen des Datenspeichers, bearbeitet sie entsprechend den Anweisungen des Programmes und speichert die Ergebnisse wiederum im Datenspeicher ab. Zur Erfüllung dieser Aufgabe sind in der CPU Zwischenspeicher vorhanden:



Aufbau der CPU

3.3.2 Akkumulator 1

Dieser 32-bit breite Speicher ist die Drehscheibe für fast alle Informationen, die byte-, wort- oder doppelwortweise verarbeitet werden. Die meisten Befehle beziehen sich auf den Akku 1, entweder als Datenquelle oder als Datenziel, oder beides zugleich. Wenn Daten aus dem Speicher gelesen werden, gelangen sie zuerst in den Akku 1 und wenn Daten in den Speicher geschrieben werden sollen, werden sie hieraus entnommen. Wenn Berechnungen und Manipulationen vorgenommen werden, ist der Inhalt des Akkus 1 beteiligt und das Ergebnis wird ebenfalls hier abgelegt.

Der Akku 1 besteht, je nach Betrachtungsweise, aus einem Doppelwort, zwei Wörtern, vier Bytes oder 32 Bits. Die beiden Wörter werden durch den Zusatz H (high) und L (low) unterschieden, die vier Bytes durch anhängen HH (vom high-wort das high-byte), HL (vom high-wort das low-byte) und entsprechend LH und LL unterschieden. Beispiele:

Akku 1 H	Bits 31 - 16
Akku 1 LH	Bits 15 - 8

3.3.3 Akkumulator 2

Dieser genau wie Akku 1 aufgebaute Speicher dient als Zwischenablage für Operationen, die zwei Operanden benötigen. Er wird zumeist dadurch beschickt, dass ein neuer Wert in Akku 1 geladen wird. Der bisherige Inhalt von Akku 1 wird dadurch nach Akku 2 verschoben.

3.3.4 Akku 3 und 4

In den SPS der S7-400-Baureihe gibt es Akku 3 und Akku 4, die ebenfalls 4 Bytes groß sindund zur Speicherung von Zwischenergebnissen bei komplexen Berechnungen benutzt werden.

Operationen mit diesen Akkus:
ENT
LEAVE
PUSH
POP

S7-400 ist eingetragenes Warenzeichen der Siemens AG.

3.3.5 Verknüpfungsergebnis (VKE)

Das Verknüpfungsergebnis spielt für die Bit-Operationen die gleiche Rolle, wie der Akku 1 für die anderen Operationen. Für den Programmierer, der in FUP oder KOP programmiert, taucht es jedoch nie auf, es wird nur vom Compiler, der diese Sprachen in die Maschinensprache übersetzt verwendet.

3.3.6 Erstabfrage

Für dieses Bit gilt das gleiche wie für das VKE, wenn man in FUP oder KOP programmiert, braucht man sich nicht darum zu kümmern.

3.3.7 Globaldatenbaustein-Register

Den durch den in diesem Speicher bezeichneten DB verwendet die CPU immer dann, wenn ein Datenwort (-bit, -byte,-doppelwort) angesprochen wird, das ohne den Zusatz des Datenbausteins notiert worden ist. Dieses Register wird durch die Operation AUF(schlagen) DB beschickt.
Beim Aufruf eines neuen Bausteins bleibt das Register unverändert, sein Inhalt wird jedoch im DB-Stack gesichert und beim Rücksprung vom aufgerufen Bausteins wiederhergestellt. D.h. wenn in dem aufgerufenen Baustein ein anderer DB geöffnet worden ist, hat dies keine Auswirkung auf den aktuellen Baustein.

3.3.8 Instanzdatenbaustein-Register

Den durch den in diesem Speicher bezeichneten DB verwendet die CPU immer dann, wenn ein Instanzdatenwort (-bit, -byte, -doppelwort) angesprochen wird. Dieses Register wird normalerweise automatisch beim Aufruf eines Funktionsbausteines beschickt, kann jedoch auch durch die Operation AUF DI mit einem neue Wert geladen werden.
Beim Aufruf eines Bausteins wird der aktuelle Inhalt dieses Registers auf dem DB-Stack gespeichert und bei der Rückkehr vom aufgerufenen Baustein wiederhergestellt.

3.3.9 Adressregister 1 und 2

In der CPU gibt es zwei mit AR1 und AR2 bezeichnete Adressregister, die auf Adressen im E-, A- oder M-Bereich zeigen können oder eine unbestimmte Adresse enthalten, deren Bereich erst bei der Anwendung festgelegt wird. Die Adressregister werden für die register-indirekte Adressierung verwendet. Der Vorteil dieser Adressierungsart im Vergleich mit der speicher-indirekten Adressierung bei komplexen Speicheroperationen entsteht dadurch, dass es spezielle Befehle (+AR1,+AR2) gibt, mit denen die Adressregister leicht incrementiert oder decrementiert werden können, ohne die gerade bearbeiten Zahlen aus den Akkus 1 und 2 zu verdrängen.

Sie können diese Register aber auch als normale Rechenregister für Sonderaufgaben verwenden, das ist manchmal ganz nützlich.

3.3.10 Statusregister

Im Statusregister werden interne Bit-Zustände der CPU gespeichert, dazu gehören auch das VKE und das Erstabfragebit. Das Statusregister in TrySim ist keine exakte Nachbildung einer S7, da eine solche Nachbildung merklichen Einfluss auf die Simulationsgeschwindigkeit gehabt hätte und nur in wenigen Anwendungsfällen von Nutzen wäre. Der Befehl L STW (Lade Statuswort), kann in TrySim zwar programmiert werden, aber er wird nicht ausgeführt.

3.4 Programm

3.4.1 Einführung

Durch das Programm wird festgelegt, was die CPU machen soll. Um auch komplexe Aufgaben übersichtlich formulieren zu können, wird es in Organisationsbausteine, Funktionsbausteine und Funktionen unterteilt. Alle Bausteinarten verfügen über den gleichen Befehlssatz und können in allen drei Darstellungsarten (FUP, KOP und AWL) dargestellt werden.

3.4.2 Organisationsbausteine

Die Organisationsbausteine (OB) werden immer vom Betriebsystem aufgerufen. In einer wirklichen SPS gibt es eine Vielzahl von OB, die bei den verschiedensten Ereignissen aufgerufen werden. In TrySim werden nur 5 OB vom Betriebssystem aufgerufen: die OB 1, 2 und 3 sowie die OB 100 und OB 35.
Sie können auch weitere OB programmieren, müssen diese dann jedoch selbst aufrufen.

1 OB 1

Im OB 1 steht das Hauptprogramm. Dies ist der Baustein, der jedesmal aufgerufen wird, wenn die Simulation der Maschine abgeschlossen ist. Wenn der OB 1 bearbeitet worden ist, wird wieder die Simulation der Maschine gestartet. Üblicherweise werden im OB 1 hauptsächlich Funktionen und Funktionsbausteine aufgerufen, Sie können aber im OB 1 auch ganz normal programmieren. Kleine Programme, die keine Strukturierung benötigen, werden vollständig im OB 1 programmiert, der dann der einzige Baustein des Programms ist.

2 OB 100

Der OB 100 ist der Anlaufbaustein. Hier werden Initialisierungen vorgenommen und in einer wirklichen SPS Spezialbaugruppen vorbereitet. In TrySim wird der OB 100 dann bearbeitet, wenn Sie das Programm zum ersten Mal starten oder nachdem Sie die SPS mithilfe des Menüpunktes **SPS|Reset SPS** zurückgesetzt haben.

3 OB 2 und 3

Hier können Sie Operationen programmieren, die nur für die Simulation benötigt werden. Wenn Sie z.B. im eigentlichen Programm eine Einschaltüberwachung von Schützen programmiert haben, würde diese ohne weitere Vorkehrung ständig ansprechen. Um dies zu vermeiden, programmieren Sie im OB 2 oder OB 3, dass

der Schütz-Abfrage-Eingang immer dann gesetzt wird, wenn auch der Ausgang angesteuert ist.

Der OB 2 wird vor dem OB 1 aufgerufen, der OB 3 danach. Diese Bausteine brauchen Sie natürlich nicht mit Ihrem Programm nach STEP®7 zu exportieren.

4 OB 35

Der OB 35 wird in festen Abständen der virtuellen Zeit aufgerufen. Wie groß diese Abstände sind, können Sie unter **SPS|CPU-Eigenschaften|Weckalarme** einstellen.

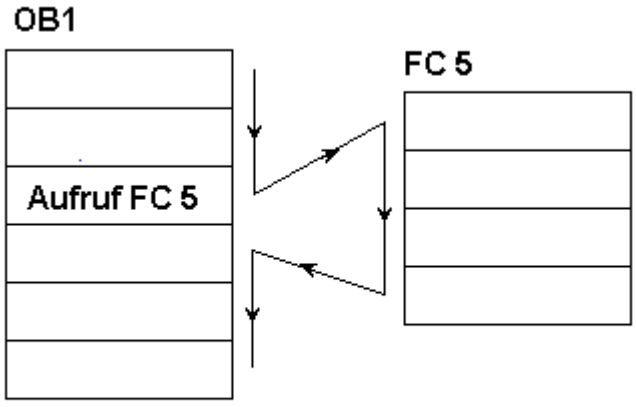
In Organisationsbausteinen können Sie wie in allen anderen Code-Bausteinen temporäre Variablen deklarieren.

STEP®7 ist eingetragenes Warenzeichen der Siemens AG.

3.4.3 Funktionen

1 Aufgabe

In Funktionen formulieren Sie Unter-Aufgaben des Programms. Wenn Sie im OB 1 oder einem anderen Baustein eine Funktion aufrufen, werden die Anweisungen der Funktion bearbeitet. Am Ende der Funktion wird die Bearbeitung im aufrufenden Baustein nach dem Aufruf fortgesetzt.



Bearbeitung des Programms bei Aufruf einer Funktion

Funktionen können auch geschachtelt aufgerufen werden, d.h. Sie können innerhalb einer Funktion eine weitere aufrufen. Funktionen, die häufig wiederkehrende Aufgaben zu erfüllen haben, können auch mehrfach aufgerufen werden.

2 Parameter

Funktionen können auch mit Parametern (Platzhaltern) versehen werden. Dadurch ist es möglich, die Funktion bei jedem Aufruf mit anderen Ausgangsdaten arbeiten zu lassen und von der Funktion Werte an den aufrufenden Baustein zurückzugeben. Innerhalb einer Funktion werden die Parameter Formal-Parameter genannt.

Bei der Programmierung der Funktion verwenden Sie die Parameter wie Ein- und Ausgänge, Merker oder Datenwörter. Welche Daten dann von der SPS tatsächlich verwendet werden legen Sie beim Aufruf der Funktion fest. Diese Daten heißen dann Aktual-Parameter.
Welche Parameter Ihre Funktion haben soll, legen Sie im Kopf der Funktion fest. Für jeden Parameter müssen Sie diese Eigenschaften festlegen (deklarieren), die im Weiteren erläutert werden:

1. Deklarationstyp	In,Out,InOut,Temp
2. Name	Dadurch bezeichnen Sie den Parameter innerhalb der Funktion
3. DatenTyp	Dadurch legen Sie u.a. die Größe des Parameters fest
4. Anfangswert	Hat bei Funktionen keine Bedeutung
5. Kommentar	Damit auch andere Ihr Programm verstehen können

3 Deklarationstyp

Bei Funktionen gibt es drei Arten von Parametern und eine Art von Variablen:

- In - Parameter
- Out - Parameter
- In - Out – Parameter
- Temporäre Variablen

In-Parameter werden beim Aufruf der Funktion mit dem Wert des Aktual-Parameters belegt. In-Parameter sollten innerhalb der Funktion nur gelesen werden. Sie können sie zwar auch auf In-Parameter schreibenund dann innerhalb der Funktion mit dem geänderten Wert arbeiten, aber diese Änderung wirkt sich nicht auf die Aktual-Parameter im aufrufenden Baustein aus.

Die an Out-Parameter angeschlossenen Operanden werden bei der Rückkehr zum aufrufenden Baustein mit dem in der Funktion ermittelten Wert belegt. Sie können innerhalb einer Funktion Out-Parameter auch lesen, wenn Sie dies jedoch tun, bevor Sie dem Parameter einen Wert zugewiesen haben, wird das Abfrageergebnis zufällig sein, da Out-Parameter beim Aufruf einer Funktion nicht mit dem Wert des angeschlossenen Operanden belegt werden.

In-Out-Parameter schließlich vereinigen das Verhalten der In- und Out-Parameter. Beim Aufruf einer Funktion werden sie mit dem Wert des angeschlossenen Operanden belegt. Sie können innerhalb der Funktion gelesen und verändert werden. Bei der Rückkehr zum aufrufenden Baustein wird der geänderte Wert in den angeschlossenen Operanden übertragen.

Das von TrySim verwendete Parametermodell unterscheidet sich in Details von dem einer S7. Beachten Sie dazu bitte das Kapitel Abweichend implementierte Eigenschaften.

Zur Speicherung von Zwischenergebnissen, die nur innerhalb der Funktion benötigt werden, können Sie die temporären Variablen verwenden.

4 Name

Die Parameter und die Variablen erhalten je einen Namen, den Sie beim Programmieren verwenden, um darauf zuzugreifen. Im Programmcode werden die Namen immer mit einem vorangestellten ‚#' (Doppelkreuz,Gartenzaun) gekennzeichnet. Bei eindeutigen Namen, die nicht auch in der Symboltabelle verwendet werden oder einem Schlüsselwort entsprechen, brauchen Sie das Zeichen ‚#' nicht einzugeben, lediglich wenn Verwechslungsmöglichkeiten bestehen ist die Eingabe erforderlich.

5 Datentyp

Außer der Festlegung, ob ein Parameter In-, Out- oder InOut- Eigenschaften haben soll, müssen Sie auch den Datentyp des Parameters bestimmen (deklarieren). Das sind z.B.

Typ	Größe	Beispiel für anschließbaren Aktualparameter
BOOL	1 Bit	M 1.2
BYTE	1 Byte	EB 2
CHAR	1 Byte	MB 5
INT	2 Bytes	AW 10
WORD	2 Bytes	DBW 10
REAL	4 Bytes	DBD 20

Siehe auch: Datentypen

6 Anfangswert

Diese Spalte ist nur bei den Funktionsbausteinen relevant.

7 Kommentar

Hiermit können Sie die Aufgabe der Parameter und Variablen erläutern, sodass auch Andere Ihr Programm verstehen können. Da Sie selbst in spätestens einem Jahr ebenfalls ein "Anderer" sind, sollten Sie von dieser Möglichkeit ausgiebig Gebrauch machen.

8 ENO-Ausgang

Wenn Sie eine Funktion in FUP oder KOP aufrufen, können Sie, in Abhängigkeit von Berechnungsergebnissen in der Funktion, andere Operationen (oder weitere Aufrufe) durchführen lassen, indem Sie sie an den ENO-Ausgang der Funktion anschließen. Die an diesen Ausgang angeschlossenen Operationen werden ausgeführt, wenn das BIE-Bit innerhalb der Funktion auf "1" gesetzt worden ist. Sie werden übersprungen, wenn das BIE-Bit "0" ist.

3.4.4 Funktionsbausteine

1 Aufgabe

Funktionsbausteine gleichen in vielem den Funktionen. Der Unterschied zwischen diesen beiden besteht darin, dass beim Aufruf eines Funktionsbausteines (fast) immer ein Datenbaustein angegeben wird, in dem die Daten stehen, mit denen der Baustein arbeiten soll. Dieser Datenbaustein wird Instanz-Datenbaustein genannt. Seine Struktur wird bei der Programmierung des FB's festgelegt und auf seine Daten kann innerhalb des FB's besonders komfortabel zugegriffen werden.

In Funktionsbausteinen gibt es eine weitere Art von Variablen: die statischen Variablen. Diese Variablen behalten von einem Aufruf des Funktionsbausteines zum Nächsten ihren Wert.

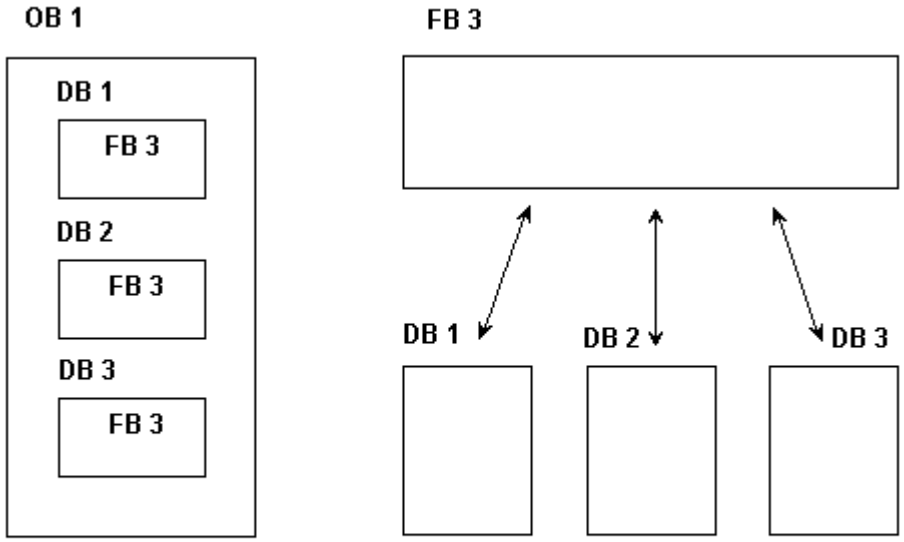
2 Instanz-Datenbaustein-Modell

Nachdem Sie die Parameter und Variablen eines Funktionsbaustein deklariert haben und den Baustein programmiert haben, müssen Sie ihn irgendwo aufrufen, sonst wird er ja nicht bearbeitet. Wenn Sie den Aufruf programmieren, müssen Sie eine Datenbaustein-Nummer angeben. Das Entwicklungssystem wird nun einen Datenbaustein mit dieser Nummer erzeugen und ihn genauso groß machen, dass alle Parameter und (statischen) Variablen darin Platz finden. Auch die Namen, Datentypen und die Kommentare werden aus dem Deklarationsteil des Funktionsbausteins übernommen. Die Werte der Datenwörter (-bits, -bytes, -doppelwörter) werden bei der Erzeugung auf die im Kopf des Funktionsbausteines angegebenen Anfangswerte gesetzt.

Bei der Programmbearbeitung wird dieser Datenbaustein automatisch aufgeschlagen, wenn der Funktionsbaustein aufgerufen wird. Er ist jetzt der Instanz-Datenbaustein. Die Werte der Aktual-Operanden die als In- oder In-Out-Parameter deklariert sind, werden in den Datenbaustein kopiert. Alle Schreib- und Lesezugriffe im Funktionsbaustein, die Sie unter Verwendung der Parameternamen programmiert haben, beziehen sich jetzt auf die Daten im Instanz-Datenbaustein. Bei der Rückkehr zum aufrufenden Baustein werden die In-Out- und die Out-Parameter vom Instanz-Datenbaustein in die angeschlossenen Operanden kopiert.

Der Nutzen dieses Verfahren wird dann ersichtlich, wenn man für den gleichen Funktionsbaustein mehrere Aufrufe programmiert und zwar mit verschiedenen Datenbausteinen. Jeder dieser DB wird dann nach der Maske des Funktionsbaustein-Kopfes erzeugt.

Ein Standardbeispiel für die Anwendung des Instanz-Datenbaustein-Modells ist ein Betriebszeitzähler für mehrere Motoren. Sie programmieren den Betriebszeitzähler in einem FB, in dessen Kopf Sie einen In-BOOL-Parameter mit der Bedeutung "Motor läuft" und eine Stat-INT-Variable mit der Bedeutung "Betriebsminuten" deklariert haben. Diesen FB rufen Sie dann für Motor 1 mit dem DB 1, für Motor 2 mit DB 2 usw. auf. Bei jeder Bearbeitung des FB werden dann automatisch die dem Motor zugeordneten Daten bearbeitet.



Der FB 3 wird jedesmal mit einem anderen Instanz-Datenbaustein aufgerufen

Bei jedem Aufruf greift der FB 3 nur auf die Daten des aktuellen Instanz-Datenbausteines zu

3 Temporäre Variablen

Die temporären Variablen in Funktionsbausteinen werden genau wie die von Funktionen auf dem Stack gespeichert, sie sind nicht Bestandteil des Instanz-Datenbausteins.

3.4.5 Statische Variablen

Zur Speicherung von Ergebnissen, die innerhalb eines FBs (nicht OB oder FC) von einem Zyklus zum nächsten ihren Wert beibehalten, können Sie statische Variablen deklarieren.

Die statischen Variablen werden wie die In-, Out- und In-Out-Parameter im Instanzdatenbaustein des FB gespeichert.

3.4.6 Temporäre Variablen

Zur Speicherung von Zwischenergebnissen, die nur innerhalb eines Blockes (OB, FC,FB) benötigt werden, können Sie temporäre Variablen deklarieren. Diese Zwischenspeicher haben beim Aufruf des Blocks keinen definierten Wert, Sie müssen daher darauf achten, dass Sie ihnen vor dem ersten Lese-Zugriff einen Wert zuweisen. Nachdem die Bearbeitung des Blocks abgeschlossen ist, stehen die hierin abgelegten Werte nicht mehr zur Verfügung.
In der STEP®5 – Welt wurde diese Aufgabe von einem “Schmiermerker” genannten Merkerbereich erfüllt, der bei Merkermangel von verschiedenen Programmteilen gleichzeitig benutzt wurde. Das Schöne an den temporären Variablen ist ihr Name. Da die Schmiermerker in verschiedenen Bedeutungen verwendet wurden, konnte man ihnen in der Zuordnungsliste nur den wenig aussagekräftigen Kommentar

"Schmiermerker" geben. Die temporären Variablen können jedoch ihres Inhaltes entsprechend bezeichnet werden.

Intern werden die temporären Variablen auf dem Stack gespeichert.

3.5 Externe (Soft) SPS

3.5.1 Allgemeines

Ein häufiger Einsatz von TrySim ist, nur die Anlage zu simulieren und das SPS-Programm auf einer realen (oder Soft-) SPS laufen zu lassen. In den allermeisten Fällen ist dies mit zusätzlichen Kosten verbunden, sei es für die notwendige Hardware, Lizenzgebühren für Treiber anderer Hersteller oder Anpassungen von TrySim an eine bestehende Konfiguration. Sprechen Sie mit uns, wenn Sie die TrySim - Anlage extern steuern wollen.
www.trysim.de

Mit TrySim können Sie keine externe SPS programmieren.

Über **MPI** können Sie TrySim problemlos an eine wirkliche S7 anbinden.

Die Anbindung über **TCP/IP** ist an jeden Partner möglich, den man entsprechend programmieren kann.

Die Anbindung an die S7-300/400 Simulation **S7-PLCSim** funktioniert recht gut.

Die Anbindung über **Profibus** funktioniert schnell und einfach, erfordert allerdings eine recht teure Schnittstellenkarte und ist in dieser Hilfe nicht dokumentiert.

Die Anbindung an **Allen-Bradley** RSLogix5000 ist über RSLinx möglich, aber in dieser Hilfe noch nicht dokumentiert.

Die Anbindung an SPS **beliebiger Hersteller** ist über die offene Schnittstelle möglich.

Die Anbindung zur **IBH softec SoftSPS** ist problemlos möglich.

Die Anbindung an **3S -CoDeSys** - IEC1131- SoftSPSen ist möglich, aber ebenfalls in dieser Hilfe noch nicht dokumentiert.

Falls Sie Interesse an einer dieser Anbindungen haben, wenden Sie sich bitte an uns.

Die Anbindung an Soft-SPS / SPS-Simulatoren anderer Hersteller ist geplant.

Für Anwender, die ein wenig unter Windows programmieren können, besteht jedoch außerdem die Möglichkeit, die Anbindung an TrySim selbst vorzunehmen, da die Schnittstelle von TrySim offengelegt ist. Im Prinzip kann mit jedem Compiler ein

Programm erstellt werden, das die TrySim-Anlage steuert.

Anmerkung.: o.g. Möglichkeiten stehen in der TrySim Lite Version nicht zur Verfügung.

3.5.2 Sicherheitshinweise

Mißachtung dieser Hinweise kann zu Sachschaden, Körperverletzung oder Tod führen.

Wenn TrySim über irgendeine Schnittstelle an eine externe SPS oder Soft-SPS angeschlossen wird, an die wiederum eine real existierende Maschine angeschlossen ist, kann das Verhalten dieser Maschine von TrySim aus beeinflusst werden. Die Anlagensimulation in TrySim verhält sich dann quasi selbst wie eine SPS, da sie in Abhängigkeit von den eingehenden Informationen die ausgehenden Informationen modifiziert. Das Verhalten dieser “Quasi-SPS” ist aber, obschon deterministisch, nur schwer mit ausreichender Genauigkeit vorher zu sagen. Das Verhalten einer SPS wird durch das explizit vorliegende Programm bestimmt, welches Schritt für Schritt analysiert werden kann. Das Verhalten der Anlagensimulation wird aber nur implizit durch die Anordnung und Funktion der Elemente bestimmt.

Aus diesem Grund empfehlen wir dringend, TrySim nicht zusammen mit Maschinen zu betreiben, von denen möglicherweise Gefahren für Menschen und Sachen ausgehen.

Wir weisen ausdrücklich darauf hin, dass wir keinerlei Haftung, aus welchem Rechtsgrund auch immer, für Schäden übernehmen, die aus einer Mißachtung dieser Empfehlung herrühren.

3.5.3 Offene Schnittstelle von TrySim

TrySim stellt anderen Anwendungen den gesamten E-, A- und M-Bereich zur Verfügung. Die Eingänge und Ausgänge (aus der Sicht der SPS) bestehen jeweils aus 64KB. Insgesamt besteht der Schnittstellenbereich aus 4 x 64K:
1. Besondere Dienste, 64K noch in Entwicklung begriffen
2. Ausgänge, 64K
3. Eingänge, 64K
4. Merker, 64K

Dieser Bereich wird von TrySim als “Mapped Object” eingerichtet. Alle anderen Anwendungen können durch den Aufruf von Windows-API-Funktionen auf diesen Bereich zugreifen. Dazu muss folgender Code programmiert werden:

C
Pascal (Delphi)

Schließen Sie in TrySim alle SPS-Fenster und wählen Sie **SPS|Externe SPS**. Sie werden gefragt, ob jetzt das Schnittstellen-Testprogramm gestartet werden soll. Dies ist ein sehr einfaches Programm, mit dem Sie Ausgänge setzen und Eingänge abfragen können. Unter **Optionen | Externe SPS** können sie stattdessen Ihren Schnittstellentreiber angeben.

Die Schnittstelle ist bei Win95/98, bedingt durch das Zugriffsverfahren, nicht besonders schnell (ca. 50 - 100 ms Antwortzeiten). Bei WinNT jedoch liegen die Antwortzeiten unter einer Millisekunde.

3.5.4 Schnittstelle, C-Code

```
handle hfile;  // handle auf das mapped file

hfile = OpenFileMapping(FILE_MAP_ALL_ACCESS,
                        FALSE,          // handle soll nicht geerbt werden
                        "TRYSIM_SCHNITTSTELLE");

// Bei manchen Compilern ist es notwendig, den Schnittstellennamen als PChar -
Variable zu übergeben.

// hier sollte überprüft werden, ob hfile <> NULL ist. Mit GetLastError lässt sich der
Fehlercode ermitteln

*byte EAdrSpace;  // pointer auf das EB 65535 !!!
*byte AAdrSpace; //  pointer auf das AB 65535 !!!
*byte MAdrSpace; //  pointer auf das MB 65535 !!!

AAdrSpace = MapViewOfFile(hfile,                    // Handle to mapping
object.
                          FILE_MAP_ALL_ACCESS, // Read/write permission
                          0,                   // Offset HiDWord
                          1 * 64 * 1024,       // Offset LoDWord
                          64 * 1024);          // Length

// hier sollte überprüft werden, ob AAdrSpace <> NULL ist.
```

```
EAdrSpace = MapViewOfFile(hfile,                    // Handle to mapping
object.
                          FILE_MAP_ALL_ACCESS, // Read/write permission
                          0,                        // Offset HiDWord
                          2 * 64 * 1024,            // Offset LoDWord
                          64 * 1024);               // Length

// hier sollte überprüft werden, ob EAdrSpace <> NULL ist.

MAdrSpace = MapViewOfFile(hfile,                    // Handle to mapping object.
                          FILE_MAP_ALL_ACCESS, // Read/write permission
                          0,                        // Offset HiDWord
                          3 * 64 * 1024,            // Offset LoDWord
                          64 * 1024);               // Length

// hier sollte überprüft werden, ob MAdrSpace <> NULL ist.

Die Reihenfolge der Bytes im Adressraum ist verkehrt herum !!!

//  Beim Beenden des Programms sollten Sie das mapped object wieder
freigeben:

    CloseHandle(hfile);
```

Siehe auch:
Schnittstelle

3.5.5 Schnittstelle, Pascal-Code

```
uses windows;

const  MaxAdr = $FFFF;

type   PAdrSpace = ^TAdrSpace;
       TAdrSpace = packed array [-MaxAdr..0] of Byte;

var EAdrSpace: PAdrSpace;
    AAdrSpace: PAdrSpace;
    MAdrSpace: PAdrSpace;


var hfile    : THandle;

{------------------------------------------------------------------------}
function InitSchnittstelle: boolean;
```

```
con Output  = 64 * 1024;
st   Offset
     InputOff = 64 * 1024 * 2;
     set
     Merker = 64 * 1024 * 3;
     Offset
     Length  = 64 * 1024;

var mappedname: PAnsiChar;

begin
 mappedname:= 'TRYSIM_SCHNITTSTELLE';
 hfile:= OpenFileMapping(FILE_MAP_ALL_ACCESS,
                         FALSE,
                         mappedname);

 if hfile = 0 then begin
   result:= false;
   EXIT;
 end;

AAdrSpace:=  MapViewOfFile(hfile,                    // Handle to mapping
object.
                           FILE_MAP_ALL_ACCESS,  // Read/write permission
                           0,                    // Offset HiDWord
                           OutputOffset,         // Offset LoDWord
                           Length);              // Size of View

 if AAdrSpace = nil then begin
  CloseSchnittstelle;
  result:= false;
  EXIT;
 end;

 EAdrSpace:=  MapViewOfFile(hfile,  // Handle to mapping object.
                            FILE_MAP_ALL_ACCESS,      // Read/write
permission
                            0,                        // Offset HiDWord
                            InputOffset,              // Offset LoDWord
                            Length);                  // Size of View

 if EAdrSpace = nil then begin
  CloseSchnittstelle;
  result:= false;
  EXIT;
 end;

MAdrSpace:=  MapViewOfFile(hfile,  // Handle to mapping object.
```

```
                                    FILE_MAP_ALL_ACCESS,      // Read/write
permission
                                    0,                        // Offset HiDWord
                                    MerkerOffset,              // Offset LoDWord
                                    Length);                  // Size of View

  if MAdrSpace = nil then begin
    CloseSchnittstelle;
    result:= false;
    EXIT;
  end;

  result:= true;
end; { InitSchnittstelle }
{-------------------------------------------------------------------------}
procedure CloseSchnittstelle;
begin
  CloseHandle(hfile);
  hfile:= 0;
  AAdrSpace:= nil;
  EAdrSpace:= nil;
  MAdrSpace:= nil;
end; { CloseSchnittstelle }
{-------------------------------------------------------------------------}
```

3.5.6 Anbindung an externe S7 über MPI

Diese Schnittstelle wird nicht mehr unterstützt. Bitte lesen Sie Anbindung an Externe SPS über MPI (Prodave).

MPI ist eingetragenes Warenzeichen der Firma Schildknecht.

3.5.7 SoftSPS nicht gefunden

Für die Verbindung zur IBH SoftSPS benötigt TrySim die dll “plc32.dll”. TrySim sucht diese dll in dem Pfad, den Sie unter **SPS|ExterneSPS|Schnittstellen-Parameter** eingeben können. (Achten Sie darauf, dass “IBH SoftSPS” angewählt ist, bevor Sie “Schnittstellen-Parameter” anklicken.)

Wenn Sie keine Ahnung haben, wo die dll sein könnte, verwenden Sie die Windows-Funktion **Start|Suchen** nach “plc32.dll”.

Wenn Sie die Demo-Version des S5-Simulators verwenden, heißt die dll “plc32dem.dll”.

Siehe auch:
Externe SPS

3.5.8 SoftSPS läuft noch nicht

Bei dem Versuch, Kontakt mit der IBH SoftSPS über die DLL “plc32.dll” aufzunehmen, ist ein Fehler aufgetreten.

Dafür sind zwei Gründe möglich:
1. Sie haben die IBH SoftSPS / Simulator noch nicht gestartet. Holen Sie das jetzt nach und wählen Sie dann erneut **SPS|Externe SPS**.
2. Wenn die IBH SoftSPS bereits läuft, haben Sie vermutlich mehr als eine Kopie der “plc32.dll” auf Ihrem Rechner. Es ist wichtig, dass TrySim genau die gleiche Kopie der dll verwendet wie die SoftSPS! Der Pfad, den Sie unter **SPS|ExterneSPS|Schnittstellen-Parameter** eingeben können, weist zu einer der Kopien, die **nicht** von der IBH SoftSPS verwendet wird. (Achten Sie darauf, dass “IBH SoftSPS” angewählt ist, bevor Sie “Schnittstellen-Parameter” anklicken.)
Über die Windows-Funktion **Start|Suchen** können Sie nach “plc32.dll” suchen. Schlimmstenfalls müssen Sie alle dort gefundenen Dateien ausprobieren.
Wenn Sie die SoftSPS über das Netzwerk gestartet haben, müssen Sie dem Pfad den kompletten Rechnernamen (‘\\ComputerX\C\.....’) voranstellen. Oder Sie verbinden das Netzlaufwerk mit einem Laufwerksbuchstaben über den Button “Netzwerk”.

Wenn Sie die Demo-Version des S5-Simulators verwenden, heißt die dll “plc32dem.dll”.

Siehe auch:
Externe SPS

3.5.9 Anbindung an externe S7 über MPI (Prodave)

Bitte lesen Sie unbedingt die Sicherheitshinweise bei externer (Soft-)SPS !

Um die TrySim - Anlage durch eine MPI angebundene S7 zu steuern, benötigen Sie einen MPI - Adapter:

- Entweder den USB-Adapter von Siemens
- oder eine MPI- ISA/PCI - Card von Siemens zum Einbau in den PC
- oder einen MPI-fähigen CP von Siemens
- oder einen MPI-Adapter von Deltalogic (bitte vorher Rücksprache halten)
- oder einen anderen MPI-Adapter, der sich von PRODAVE-MPI ansprechen lässt.

Falls Sie ein Programmiergerät einsetzen oder einen PC, der bereits über eine solche Schnittstelle verfügt, benötigen Sie keine weitere Hardware.

Weiterhin benötigen Sie eine Siemens-Lizenz für PRODAVE-MINI , die Sie bei uns

erhalten können. Wenn Sie diese Lizenz nicht bei uns beziehen, achten Sie darauf, dass Sie nicht nur die Lizenz, sondern auch die Daten bekommen, auf die sich die Lizenz bezieht.

Mit PRODAVE-MINI ist nur die Übertragung von Datenbausteinen möglich. Daher müssen Sie alle Ein-/Ausgänge der Simulation in DBs in der SPS übertragen. Im zu testenden SPS-Programm muss dann der Eingangs-DB im ersten NW des OB1 auf das Prozessabbild der Eingänge übertragen werden und im letzten NW muss das Prozessabbild der Ausgänge auf den Ausgangs-DB übertragen werden. Es ist also eine nur für die Simulation notwendige Veränderung des zu testenden Programms notwendig. Details dazu finden Sie im Verlauf dieses Kapitels unter "Festlegung eines Blockes der I/O - Konfiguration für MPI"

Es gibt auch eine Lösung, die ohne jede Modifikation des SPS-Programms auskommt, aber diese ist teuerer. Falls Sie daran Interesse haben, setzen Sie sich bitte mit uns in Verbindung.

Die kleineren S7-300-CPUs unterstützen nur eine MPI-Baudrate von 187 kB/sek. Damit ist selbst bei einem kleinen Prozessabbild nur eine Daten-Aktualisierungs-Rate mindestens 100 ms möglich. Die DP-CPUs erlauben auch eine Baudrate von 12 MB/sek. Damit wird die Aktualisierungsrate deutlich kleiner. Wirklich befriedigende Ergebnisse erhält man aber nur mit CPUs der S7-400-Reihe.

1.) Falls Sie auf Ihrem Rechner nicht bereits Software von Siemens installiert haben, die MPI verwendet, müssen Sie Prodave-Mini einmalig installieren, damit das Programm “PG-PC-Interface” verfügbar ist. Wenn dieses Programm bereits auf Ihrem Rechner verfügbar ist (z.B. wenn Sie dort einmal STEP7 installiert hatten), brauchen Sie Prodave nicht zu installieren. Das Installationsprogramm befindet sich auf der CD (vorausgesetzt, Sie haben eine Kopierlizenz bei uns erworben) unter d:\ProdaveMini\Disk1\setup.exe.

2.) Konfigurieren Sie Ihren Adapter/MPI-Card mittels des Programms “PG-PC-Interface” unter Start|Programme|Prodave_S7_Mini. Dazu können Sie einen neuen Zugangspunkt mit Namen “TRYSIMMPI” einrichten oder sie können einen bereits vorhandenen Zugangspunkt verwenden. Wichtig ist nur, dass der in TrySim eingegebene Zugangspunkt mit dem im "PG-PC-Interface” aktivierten übereinstimmt.

3.) Wählen Sie **SPS|Externe SPS**. Mit dem Button “I/O - Konfiguration” rufen Sie das Fenster zur Eingabe der Ein- und Ausgänge auf, die zur/von der SPS übertragen werden sollen.

4.) Legen Sie die Blöcke fest, die zur SPS übertragen werden sollen. (Auf der Maske steht, weil es intuitiver ist, “Eingänge”, aber Sie können hier genauso gut Ausgänge, Merker oder DB angeben). Alle hier angegebenen Blöcke werden nach der Bearbeitung der Anlagensimulation zur SPS übertragen. Beachten Sie, dass bei Prodave Mini alle Daten in Datenbausteine der SPS übertragen werden.

Wichtig!

Bei E-/A-/M- Bereichen beginnt der Zielbereich im DB in der SPS (gleichgültig was

die Start-Adresse ist) immer am DBB 0. Im D-Bereich hingegen werden die Daten in die gleichen Datenwörter übertragen, aus denen sie stammen.

5.) Legen Sie genauso die Blöcke fest, die von der SPS zu TrySim übertragen werden sollen.

6.) Da die Verbindung über MPI wegen des aufwendigen Protokollrahmens recht langsam ist, sollten Sie nur die Blöcke angeben, die tatsächlich benötigt werden. Da ein Großteil der Übertragungszeit auf den Protokollrahmen fällt, die für jeden Bereich einzeln anfällt, ist es günstiger, wenige, aber große Bereiche festzulegen, anstelle vieler Kleiner. Dies gilt auch dann, wenn dadurch nicht benötigte Daten übertragen werden.

7.) Wählen Sie “S7 über MPI”.

8.) Geben Sie in der IO-Liste unter “Node-Nr” die MPI-Adresse des Partners an und “Slot-Nr” den Steckplatz der CPU an. Die MPI-Adresse ist meistens 2, die Slot-Nr bei S7-300 immer 2, bei S7-400 meistens 3.

9.) Wenn es noch nicht getan haben, schalten Sie jetzt die SPS ein und stellen Sie sicher, dass das Kabel eingesteckt ist.

10.) Schließen Sie das “Externe SPS”-Fenster mit OK. Danach werden Sie aufgefordert, zu bestätigen, dass Sie die externe SPS anbinden wollen. Lesen Sie die Sicherheitshinweise und bestätigen danach mit OK.

11.) Der Aufbau der Verbindung dauert etwa 5 sec, dann schließt sich das “Externe SPS” - Fenster.

12.) Jetzt ist die TrySim-Anlage an die SPS angeschlossen, als sei sie eine wirkliche Anlage. Sie sollten die Simulationsgeschwindigkeit in TrySim unter **Anlage|Echte Zeit** auf Echtzeit einstellen, da der Ablauf der Zeiten in der SPS nicht beeinflussbar ist. Die Simulationswiederholrate sollten Sie unter **Ansicht|Optionen|Simulation** auf mindestens 100 ms einstellen, bei einer kürzeren Einstellung kann nicht mehr jeder Simulationschritt zur SPS übertragen werden.

S7-300, S7-400 und Prodave sind eingetragene Warenzeichen der Siemens AG. MPI ist eingetragenes Warenzeichen der Firma Schildknecht.

PRODAVE MPI mini von Siemens

Dies ist ein Treiber, der die Kommunikation mit einer S7 über MPI gestattet.
Auch wenn Sie auf dem TrySim-Rechner den Simatic-Manager installiert haben und eigentlich schon eine MPI-Verbindung zur SPS haben, benötigt TrySim diesen Treiber, um an der Kommunikation teilhaben zu dürfen.

Mit PRODAVE können nur Daten, keine Programme zwischen SPS und PC ausgetauscht werden.

Sie können diese Software von uns erwerben oder von Siemens direkt.

S7-300, S7-400, Simatic und Prodave sind eingetragene Warenzeichen der Siemens AG.
MPI ist eingetragenes Warenzeichen der Firma Schildknecht.

3.5.10 Schnittstelle zur SoftSPS von IBH softec

Um die TrySim - Anlage durch die SoftSPS oder die S5/S7 Simulatoren von IBH steuern zu lassen, gehen Sie wie folgt vor:

1. Wählen Sie **SPS|Externe SPS**. Mit dem Button “I/O - Konfiguration” rufen Sie das Fenster zur Eingabe der Ein- und Ausgänge auf, die zu/von der SPS übertragen werden sollen.

2. Legen Sie die Blöcke fest, die zur SPS übertragen werden sollen. (Auf der Maske steht, weil es intuitiver, ist “Eingänge”, aber Sie können hier genauso gut Ausgänge, Merker oder DB angeben). Alle hier angegebenen Blöcke werden nach der Bearbeitung der Anlagensimulation auf das Prozessabbild der SoftSPS kopiert.

3. Legen Sie genauso die Blöcke fest, die von der SoftSPS nach TrySim übertragen werden sollen.

4. Seien Sie bei der Festlegung der Blöcke ruhig großzügig, lediglich bei sehr langsamen Rechnern lohnt es sich, nur die tatsächlich benötigten Blöcke zu übertragen, sonst machen ein paar zig Bytes mehr oder weniger kaum einen Unterschied in der Bearbeitungsgeschwindigkeit.

5. Wenn Sie es noch nicht getan haben, starten Sie jetzt die IBH SoftSPS.

6. Wählen Sie “IBH SoftSPS”.

7. Geben Sie auf der Maske “Schnittstellen-Parameter” den Pfad zur SoftSPS ein. (Siehe auch: SoftSPS läuft noch nicht und IBH SoftSPS nicht gefunden).

8. Bestätigen Sie alle Masken mit “OK”, bis Sie wieder im Hauptmenü angelangt sind.

9. Jetzt ist die TrySim-Anlage an die IBH SoftSPS angeschlossen, als sei sie eine wirkliche Anlage. Sie sollten die Simulationsgeschwindigkeit in TrySim unter **Anlage|Echte Zeit** auf Echtzeit einstellen, da bislang keinerlei Synchronisation zwischen der IBH SoftSPS und TrySim vorhanden ist.

S7-300 und S7-400 sind eingetragene Warenzeichen der Siemens AG.

3.5.11 Anbindung an externe Allen-Bradley PLC

Bitte lesen Sie unbedingt die Sicherheitshinweise bei externer (Soft-)SPS !

Diese Anbindung ist nur möglich, wenn Sie die entsprechende Option erworben haben.

Um die TrySim - Anlage durch eine über RSLinx angebundene Allen-Bradley-PLC zu steuern, benötigen Sie die entsprechende Hard- und Software von Rockwell.

Auf der PLC-Seite benötigen Sie eine CNB (ControlNetBridge) oder eine Ethernet-Baugruppe.

TrySim spricht nur auf die auf Ihrem Rechner laufende RSLinx-Software an und hat nur wenig mit der Weiterleitung der Daten zur angepeilten CPU zu tun.

Zum Aufbau einer Verbindung müssen Sie aber in der IO-Konfiguration festlegen:

1. die Control-Net-Node-Nr. der von Ihnen eingesetzten CNB-Baugruppe: Diese wird über kleine Räder auf der CNB eingestellt, die man nur nach Ausbau der Baugruppe erreichen kann.

2. die Steckplatznummer (Slot-No) der CPU: Der erste Platz neben dem Netzteil hat die Nummer 0! Das ist anders als bei Siemens-SPSen und sollte daher für Umsteiger immer im Kopf behalten werden.

3. Die Tag-Names. Sie können von TrySim aus nur Controller-Tags ansprechen! Wenn Sie die Infos von TrySim in den Tags innerhalb Routinen benötigen, müssen Sie sie selbst (z.B. mittels Alias) dorthin transferieren.

RSLinx ist eingetragenes Warenzeichen der Rockwell Software Inc.

3.5.12 Festlegen der I/O - Konfiguration

Diese Maske muss nur beachtet werden, wenn Sie die TrySim-Anlage durch eine externe (Soft-) SPS steuern wollen.
Sie müssen sie nicht beachten, wenn die Anbindung über TCP/IP oder Profibus oder an S7-PLCSIM erfolgt.

Wenn Sie einen eigenen Schnittstellentreiber verwenden, müssen Sie keine Ein- und Ausgänge konfigurieren, da Ihnen dann der gesamte E-, A- und M-Bereich von je 65.536 Bytes ohne jede Konfiguration zur Verfügung gestellt wird.

Auf dieser Maske geben Sie an, welche Daten von TrySim zu einer externen SPS übertragen werden sollen und welche Daten von der externen SPS zu TrySim

übertragen werden sollen.
Für jede Übertragungsrichtung können Sie mehrere Blöcke angeben. Ein Block besteht jeweils aus der Bezeichnung des Operanden-Bereichs (Eingänge, Ausgänge, Merker, Datenbausteine), der Startadresse und einer Größe, die in Bytes angegeben wird. Bei Datenbausteinen müssen Sie natürlich noch die Nummer des Datenbausteins nennen.

Die Felder “TagName” und “Typ” sind nur notwendig, wenn Sie eine Anbindung an eine AllenBradley SPS oder eine CoDeSys SPS planen. Bei einer CoDeSys SPS muss bei “TagName” der Startoffset im Merkerbereich der SPS angegeben werden.

Die Felder “Node-Nr” und “Slot-Nr” sind nur für die AllenBradly-SPS und die MPI - Anbindung relevant. Bei MPI entspricht “Node-Nr” der MPI-Adresse, “Slot-Nr” dem Steckplatz der CPU.

Wenn Daten von mehreren Stellen aus verändert werden können, tauchen oft schwer zu findende Fehler auf. Wenn Sie also TrySim mit einer externen SPS betreiben und es treten anscheinend unerklärbare Fehler auf, schauen Sie sich noch einmal in Ruhe die I/O-Konfiguration an, gegebenenfalls auch das Gegenstück auf der SPS-Seite und prüfen Sie,
ob hier die Ursache für den Fehler liegt.

PLCSIM ist eingetragenes Warenzeichen der Siemens AG.

3.5.13 Festlegung eines Blockes der I/O - Konfiguration

Ein Block besteht jeweils aus:

- der Bezeichnung des Operanden-Bereichs (Eingänge, Ausgänge, Merker, Datenbausteine),
  diese bezieht sich **nur** auf den Adressbereich von TrySim
- bei Datenbausteinen aus der DB - Nummer
  diese Nr. bezeichnet **nur** eine Datenbaustein innerhalb von TrySim
- der Startadresse (in Bytes)
  wie bei den beiden Werten vorher, nur innerhalb von TrySim

- je nach SPS-System wird dann auch das Ziel angegeben, z.B. für S7 über MPI

- und einer Größe, die in Bytes angegeben wird

Bei der Eingabe des Operandenbereiches wird nur der erste Buchstabe berücksichtigt (E, A, M oder D), auf Groß- und Kleinschreibung brauchen Sie nicht zu achten.

3.5.14 Festlegung eines Blockes der I/O - Konfiguration für MPI

Ein Block besteht jeweils aus:

- der Bezeichnung des Operanden-Bereichs (Eingänge, Ausgänge, Merker, Datenbausteine),
  diese bezieht sich **nur** auf den Adressbereich von TrySim
- bei Datenbausteinen aus der DB - Nummer
  diese Nr. bezeichnet **nur** einen Datenbaustein innerhalb von TrySim
- der Startadresse (in Bytes)
  wie bei den beiden Werten vorher, nur innerhalb von TrySim

- der Datenbaustein-Nummer in der S7. Diese DB müssen Sie vor der ersten Verbindungsaufnahme eingerichtet haben.
- der Startadresse im DB in der S7. Diese sollte 0 sein, oder der Quelladresse entsprechen.
  es ist zwar auch jeder andere gerade Wert möglich, aber das führt zu unnötiger Verwirrung.

- und einer Größe, die in Bytes angegeben wird

Bei der Eingabe des Operandenbereiches wird nur der erste Buchstabe berücksichtigt (E, A, M oder D), auf Groß- und Kleinschreibung brauchen Sie nicht zu achten.

Wenn Sie die Anbindung über PRODAVE MPI MINI vornehmen, müssen Sie auch für die E- Bereiche jeweils einen Datenbaustein angeben, der in der SPS eingerichtet werden muss. In diesen DB wird der E-Bereich, beginnend ab der “StartAdresse in der SPS”, kopiert. Im ersten Netzwerk des OB 1 können Sie dann die SFC 20 verwenden, um die Daten auf das Prozessabbild der Eingänge zu kopieren.

Beispiel: Ihre SPS hat den Eingangsbereich EB20 - EB29. Sie richten in der SPS einen Datenbaustein DB 2 mit einer Größe von 10 Bytes ein. In TrySim erstellen Sie unter Externe SPS|IO-Konfiguration einen “Eingänge der SPS” - Block.
Bereich : E
DB-Nr. : 2
StartAdr: 20
.
Größe : 10.

Im ersten Netzwerk des OB1 programmieren Sie:

```
call SFC 20
  SRCBLK : P#DB2.DBX0.0 BYTE 10
  RET_VAL : #Ret_val // zuvor als Temp-Word deklarieren!
  DSTBLK : P#E20.0 BYTE 10
```

Mit den Ausgängen verfahren Sie sinngemäß. Die SFC 20 muss jetzt im letzten NW des OB1 aufgerufen werden. Für den Ausgangsbereich AB48 - AB63, der über den

DB3 transportiert werden soll würde das dann so aussehen:

call SFC 20
SRCBLK : P#A48.0 BYTE 16
RET_VAL : #Ret_val // zuvor als Temp-Word deklarieren!
DSTBLK : P#DB3.DBX0.0 BYTE 16

Und der zugehörige “Ausgänge der SPS” Block in TrySim würde sein:
Bereich : A
DB-Nr. : 3
StartAdr. : 48
Größe : 16.

Falls Sie auch Merker übertragen wollen, verfahren Sie genauso.

MPI ist eingetragenes Warenzeichen der Firma Schildknecht.

3.6 Referenz SPS

3.6.1 Datentypen

BOOL
BYTE
CHAR
WORD
INT
DWORD
DINT
POINTER
TIMER
COUNTER
ARRAY
STRUCT
S5TIME
TIME
DATE
DATE_AND_TIME
TIME_OF_DAY
ANY
BLOCK_FB
BLOCK_FC
BLOCK_DB
REAL
STRING
UDT

BOOL

Dieser Datentyp ist nur ein Bit groß. Die beiden Zustände werden neben “0” und “1” auch mit “falsch” und “wahr” oder “false” und “true” bezeichnet.

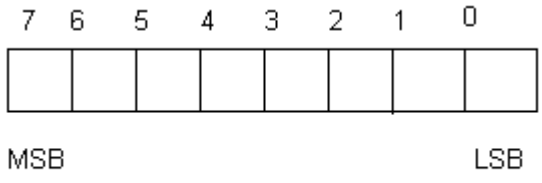
Beispiele für die Verwendung von BOOL-Typen:

```
U   E 5.2
=   #Lampe (Lampe ist als Out-BOOL deklariert)
S   M[MD8] (MD 8 ist mit P#4.2 geladen worden)
```

BYTE

Dieser Datentyp ist 8 bits groß. Der Wertebereich umfasst die Zahlen 0 – 255 bzw. $00 - $FF

oder $0 – (2^8 -1)$. Die Bits werden von 0 bis 7 numeriert, wobei Bit 7 das höchstwertige Bit ist (MSB most significant bit).



Numerierung der Bits im Byte

Beispiele für die Verwendung von BYTE-Typen

```
T   MB 177
L   #Byte2  (Byte2 ist als In-BYTE deklariert worden)
L   B#16#0 (“10” wird in Akku 1 geladen)
    A
```

CHAR

Dieser Datentyp ist ein Byte groß. Er dient zum speichern von ASCII-Zeichen. In TrySim ist er zuweisungskompatibel zum Typ BYTE, d.h. immer dort, wo Sie ein BYTE verwenden, können Sie auch ein CHAR verwenden und umgekehrt. Wenn Sie Ihr Programm nach STEP®7 exportieren wollen, dürfen Sie aber von dieser Kompatibilität keinen Gebrauch machen, da sie dort nicht besteht.

Beispiele für die Verwendung von CHAR-Typen

```
L   ‚T‘          (Anfangsbuchstabe von TrySim)
T   #Wort[3]   (#Wort ist als ARRAY[1..5] of CHAR deklariert)
```

Mittels der Operation L (Lade) können Sie bis zu vier Zeichen gleichzeitig in den Akku laden:

```
L   ‘afGh’
```

Wenn Sie aber einen CHAR-Parameter eines FB/FC beschalten wollen, darf die

Konstante nur ein Zeichen lang sein.

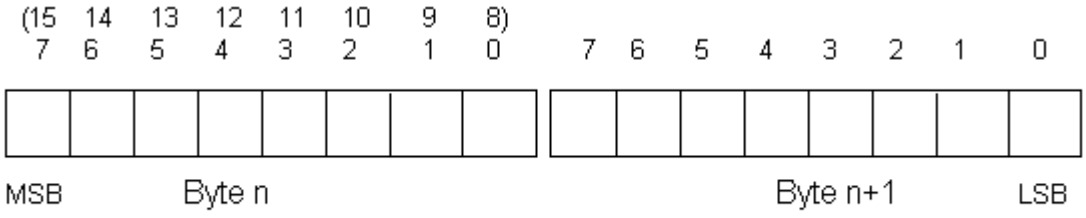
ASCII-Tabelle:

SP - 32	**0**- 48	**@** - 64	**P** - 80	` - 96	**p** - 112
! - 33	**1** - 49	**A** - 65	**Q** - 81	**a** - 97	**q** - 113
“ - 34	**2** - 50	**B** - 66	**R** - 82	**b** - 98	**r** - 114
# - 35	**3** - 51	**C** - 67	**S** - 83	**c** - 99	**s** - 115
$ - 36	**4** - 52	**D** - 68	**T** - 84	**d** - 100	**t** - 116
% - 37	**5** - 53	**E** - 69	**U** - 85	**e** - 101	**u** - 117
& - 38	**6** - 54	**F** - 70	**V** - 86	**f** - 102	**v** - 118
‘ - 39	**7** - 55	**G** - 71	**W** - 87	**g** - 103	**w** - 119
(- 40	**8** - 56	**H** - 72	**X** - 88	**h** - 104	**x** - 120
) - 41	**9** - 57	**I** - 73	**Y** - 89	**i** - 105	**y** - 121
***** - 42	**:** - 58	**J** - 74	**Z** - 90	**j** - 106	**z** - 122
+ - 43	**;** - 59	**K** - 75	**[** - 91	**k** - 107	**{** - 123
, - 44	**<** - 60	**L** - 76	**** - 92	**l** - 108	**|** - 124
- - 45	**=** - 61	**M** - 77	**]** - 93	**m** - 109	**}** - 125
. - 46	**>** - 62	**N** - 78	**^** - 94	**n** - 110	**~** - 126
/ - 47	**?** - 63	**O** - 79	**_** - 95	**o** - 111	- 127

STEP®7 ist eingetragenes Warenzeichen der Siemens AG.

WORD

Dieser Typ ist aus zwei Bytes zusammengesetzt. Das höherwertige Byte ist immer das mit der **niedrigeren** Adressnummer. Das ist für Programmierer aus dem PC-Bereich ungewohnt, denn dort ist es genau umgekehrt. Der Zahlenbereich des WORD umfasst die positiven Zahlen von 0 – 65535 bzw. $0000 - $FFFF



Numerierung der Bits im WORD

Die Numerierung der Bits im höherwertigen Byte von 8 – 16 ist etwas akademischer Natur, da Sie auch für den Zugriff auf diese Bits immer nur die Nummern 0 – 7 verwenden können.
Der Name eines Wortes bezieht sich immer auf das höherwertige Byte. AW 6 bedeutet Byte 6 und Byte 7.

Beispiele für die Verwendung von WORD-Typen:

```
T    MW 18
L    2#0000 0101 1111 0011
```

AUF DB[#DBau] (#Dbau ist als Temp-WORD deklariert)

INT

Der Datentyp INTEGER entspricht unseren normalen Zahlen mit Vorzeichen, sein Wertebereich reicht von –32.768 bis +32.767. Er belegt, genau wie das WORD, zwei Bytes.

Beispiele für die Verwendung von INT-Typen

T "Summand" ("Summand" ist in der Symboltabelle als INT deklariert worden)
L -5

DWORD

Dieser Datentyp besteht aus 4 Bytes (32 Bits). Sein Wertebereich umfasst die Zahlen

0 bis +4.294.967.295 bzw. $0000 0000 - $FFFF FFFF

Viel häufiger als zur Darstellung so großer Zahlen wird dieser Datentyp in der SPS-Programmierung zur Codierung von Bitmustern und als POINTER verwendet.

Beispiele für die Verwendung von DWORD-Typen:

```
L     MD 46624      (In TrySim geht der Merkerbereich bis ca. 65 000)
U     M[#Idx]       (#Idx ist als Temp-DWORD deklariert)
L     DW#16#0800 090A 040C 0E07
```

DINT

Der Datentyp DOUBLE INTEGER wird verwendet, wenn der Wertebereich des INT nicht mehr ausreicht. Er ist 4 Bytes groß und sein Wertbereich umfasst die Zahlen

–2.147.483.648 bis +2.147.483.647.

Bei der gleichzeitigen Verwendung von INT und DINT – Größen müssen Sie daran denken, dass INT-Zahlen aus der Ansicht eines DINT immer positiv sind. Vorzeichenbehaftete Zahlen sind immer dann negativ, wenn ihr höchstwertigstes Bit "1" ist. Wenn Sie eine Zahl aus einem Int (Word) laden, werden die beiden linken Bytes des Akku 1 mit "0" geladen. Führen Sie danach eine "D" Operation (+D, -D,.., >D,<D,..) mit diesem Wert durch, gilt diese Zahl immer als positiv, auch wenn sie eigentlich negativ ist.

POINTER

1.) Pointer sind 4 Bytes groß und werden verwendet, um auf Bits im E-, A-, M- oder D-Bereich zu zeigen. In ihm werden keine Prozessdaten gespeichert, sondern eine Adresse innerhalb der SPS. Das interne Format ist schwer zu lesen, aber wenn Sie die P# - Notation verwenden, lässt sich unmittelbar erkennen, auf welche Speicherstelle der Pointer zeigt:

P#5.7	zeigt auf das Bit 5.7 irgendeines Datenbereiches
P#E 4.5	zeigt auf das Bit E 4.5
P#A 3.1	zeigt auf das Bit A 3.1
P#M 12.0	zeigt auf das Bit M 12.0
P#D 3.2	diese Notation ist nicht erlaubt, auf Datenbits muss man anders zugreifen.

Ein Verwendungsbeispiel:

```
L P#6.8
T MD 10
....
....
U E[MD10]
```

2.) Der Datentyp POINTER im Deklarationsteil von FB/FC ist 6 Bytes groß, da hierbei auch evtl. noch eine DB-Nr gespeichert wird.

Intern wird im HSB eines Pointers zunächst gespeichert, auf welchen Datenbereich er zeigt:

```
E    81h   // Eingangsbereich
A    82h   // Ausgangsbereich
M    83h   // Merkerbereich
D    84h   // Global-DB
ID   85h   // Instanz-DB
L    86h   // Lokaldaten
FX   87h   // Parameterbereich einer Funktion (vorherige Lokaldaten)
```

In den nächsten 24 Bits kommt dann die Byte-Adresse, aber um 3 - Bits nach links verschoben. In den 3 Bits, die dadurch auf der rechten Seite frei werden, wird schließlich die Bit-Nr. gespeichert.

Zusammengesezte Datentypen werden als DB-Zeiger an Funktionen übergeben.

```
CALL "RotationControl" (struct_motor_data := "db_motor_data".struct_motor_4)
```

Der Zugriff auf die Datenelemente innerhalb der Funktion erfolgt so.

```
// DB-Register einstellen.
      L P##struct_motor_data
      LAR1
      L W[AR1, P#0.0]
      T #i_db_number
```

```
        AUF DB [#i_db_number]
// Bereichsinternen Zeiger auf die zu verarbeitende Datenstruktur in das
// Adressregister 1 laden.
        L D[AR1, P#2.0]
        LAR1
```

Nun kann auf die Elemente der Datenstruktur zugegriffen werden.

z.B.

```
        L W[AR1, P#2.0]
```

So können große Datenmengen in Datenbausteinen zusammengefasst werden. Dies erleichtert die
Konfiguration von Kommunikationsbausteinen, und die weitere Verarbeitung der Daten.
z.B. OPC-Server, Visualisierungen, Netzwerkübertragungen, Datenbanken u.s.w.

STEP®7 ist eingetragenes Warenzeichen der Siemens AG.

TIMER

Dieser Datentyp bezeichnet nicht wie viele der anderen Datentypen ein bestimmtes Format, in dem Werte in der SPS dargestellt werden, sondern er ist der Name einer der 2048 Zeitglieder, die in der SPS vorhanden sind. In der Symboltabelle wird er automatisch gewählt, wenn die Adresse mit “T” anfängt. Im Deklarationsteil von Funktionen und Funktionsbausteinen ist er nur als IN-Parameter erlaubt, innerhalb des Bausteines können Sie den Timer dann wie gewohnt verwenden, es ist nur jedesmal ein anderer Timer, der gestartet oder abgefragt wird, je nachdem, welchen Timer Sie beim Aufruf des Bausteins als Aktualparameter angegeben haben. Verwechseln Sie den Datentyp TIMER nicht mit S5TIME und TIME.

Beispiele für die Verwendung von TIMER

```
T 47    Mot1EinVerz    TIMER          Motor 1 einschaltverzögert
( Zeile in der Symboltabelle)

        SA      #Nachl                (#Nachl ist als In-TIMER deklariert)
```

COUNTER

Dieser Datentyp bezeichnet nicht wie viele der anderen Datentypen ein bestimmtes Format, in dem Werte in der SPS dargestellt werden, sondern er ist der Name einer der 2048 Zählglieder, die in der SPS vorhanden sind. In der Symboltabelle wird er automatisch gewählt, wenn die Adresse mit “Z” anfängt. Im Deklarationsteil von Funktionen und Funktionsbausteinen ist er nur als IN-Parameter erlaubt, innerhalb des Bausteines können Sie den Zähler dann wie gewohnt verwenden, es ist nur jedes Mal ein anderer Zähler, der hochgezählt oder abgefragt

wird, je nachdem, welchen Zähler Sie beim Aufruf des Bausteins als Aktualparameter angegeben haben.

Beispiele für die Verwendung von COUNTER

```
Z 31    KistImKarton          COUNTER    Kisten im Karton
( Zeile in der Symboltabelle)

        ZV      #Kisten                  (#Kisten ist als In-COUNTER deklariert)
```

Anmerkung: Vermeiden Sie Zähler, es arbeitet sich nicht schön mit ihnen.

ARRAY

In einem Array (Feld) speichern Sie viele gleichartige Werte. Jeder Wert bekommt eine Nummer (Index). Wenn Sie ein Array deklarieren, müssen Sie angeben, welchen Bereich diese Nummern haben sollen und um welche Art von Werten es sich handelt.

Wenn Sie einen bestimmten Wert des Arrays benötigen, laden Sie ihn mit dem Namen des Arrays und seiner Nummer:

```
        L       #Arr[5]          (#Arr ist als Stat-ARRAY of  INT deklariert)
```

Wenn Sie aber alle Daten eines Feldes zusammen einer Funktion übergeben wollen, z.B., damit diese die Daten sortiert, verwenden Sie nur den Namen des Feldes.

```
        CALL FC 27
          Feld: #Arr
(Feld ist in FC 27 als In-Out-ARRAY of  INT deklariert)
```

In TrySim gibt es Arrays nur von einfachen Werten. Arrays von Strukturen und mehrdimensionale Arrays, z.B. wie sie in STEP ®7 möglich sind, können Sie in TrySim noch nicht verwenden. Erlaubte Typen sind:

BOOL, BYTE, CHAR, INT, WORD, DATE, S5TIME, DWORD, DINT, REAL, TIME_OF_DAY, TIME

Eingeschränkt sind auch Arrays von UDT möglich.

STEP®7 ist eingetragenes Warenzeichen der Siemens AG.

STRUCT

In einer Struktur speichern Sie mehrere Werte, die eine verschiedene Bedeutung haben, aber dennoch zusammengehören. Wenn Sie den Auftrag erhalten hätten, das Gepäcksystem des neuen Denver Flughafens zu programmieren (was wir Ihnen nicht wünschen), hätten Sie für jedes eingehende Gepäckstück eine STRUCT folgenden Aufbaus angelegt:

```
    Koffer    STRUCT       Beschreibung eines Gepäckstücks
```

Besitzer	DWORD	Kennung des Besitzers
Länge	INT	Länge in mm
Höhe	INT	Höhe in mm
Dicke	INT	Dicke in mm
Gewicht	INT	Gewicht in kg
Ziel	DWORD	Internationale Kennung des Zielflughafens
	ENDSTRUCT	

Solange Sie das Gepäckstück als Ganzes behandeln, verwenden Sie einfach den Namen #Koffer. Wenn Sie jedoch wissen wollen, ob er in der Höhe durch eine bestimmte Durchfahrt passt, greifen Sie auf ein einzelnes Element der Struktur durch den Strukturnamen, gefolgt von einem "." und dem Namen des Elementes zu:

```
L     #Koffer.Höhe
```

Bei Strukturen gilt in TrySim ausnahmsweise die strenge Typ-Prüfung. Sie können beim Aufruf eines FB (FC), der eine STRUCT als Parameter erwartet, nur eine STRUCT mit exakt dem gleichen Aufbau übergeben. Auf die Namen der Elemente kommt es dabei nicht an, aber die Datentypen müssen übereinstimmen.

S5TIME

Dieser Datentyp ist ein Word groß und heißt darum so, weil er die angegebene Zeitdauer durch das gleiche 16-bit breite Bit-Muster darstellt, wie in der STEP ®5 – Welt eine mit KT eingeleitete Konstante. Wir sind etwas unglücklich darüber, dass dieses Datenformat, dem nach fast 30 Jahren moderner Computertechnik langsam die Daseinsberechtigung ausgeht, in die STEP ®7-Welt hinübergerettet wurde. Da wir jedoch auf Kompatibilität zu STEP ®7 achten mussten, haben wir es übernommen.

Eine als S5TIME eingegebene Konstante wird in eine dreistellige BCD-Zahl übersetzt. Weil zur Darstellung einer dreistelligen BCD-Zahl nur 12 Bits benötigt werden, bleiben in einem Word noch 4 Bits frei. Die beiden höherwertigen Bits davon sind in STEP ®5 für interne Informationen verwendet worden, sodass noch zwei Bits übrig blieben, um das "Zeitraster" zu speichern. Das "Zeitraster" codiert, in welcher Einheit die dreistellige BCD-Zahl ausgedrückt ist.

Dabei gilt folgende Festlegung

00	:	1/100 sec
01	:	1/10 sec
10	:	1 sec
11	:	10 sec

Als Folge davon lassen sich nur Zeiten von 0.01 sec bis 9990 sec in diesem Format ausdrücken.

Solange Sie das S5TIME-Format nur verwenden, um konkrete Zeitdauern beim Starten von Zeiten anzugeben, ist es recht komfortabel. Folgende Buchstaben können Sie zur Angabe von Zeitdauern verwenden:

H	hour	Stunde
M	minute	Minute
S	second	Sekunde
MS	milli second	Millisekunde

Eine typische S5TIME-Konstante sieht so aus:

S5T#2H	2 Stunden
S5T#20MS	20 Millisekunden
S5T#1M5S	1 Minute und 5 Sekunden
S5T#1H200S	1 Stunde und 200 Sekunden

Eine S5T#-Konstante wird vom Compiler immer in dem Zeitraster mit der höchsten Auflösung codiert. Dies müssen Sie berücksichtigen, wenn Sie den abgelaufenen Zeitwert als Dualzahl laden, denn hierbei ist das Zeitraster nicht mehr enthalten und kann nur durch das Wissen um das Zeitraster, mit dem die Zeit gestartet worden ist, rekonstruiert werden.

STEP®7 und STEP®5 sind eingetragene Warenzeichen der Siemens AG.

TIME

Dieser Datentyp so groß wie ein DINT (4 Bytes). Als TIME eingegebene Konstanten werden vom Compiler in Millisekunden umgerechnet und als Dual-Zahl gespeichert. Dieser Datentyp wird u.a. für die IEC-Zeiten verwendet. Versuchen Sie auf keinen Fall, ein S7-Zeitglied mit einer TIME-Konstante zu starten, verwenden Sie stattdessen S5TIME.

Folgende Buchstaben können Sie zur Angabe von Zeitdauern verwenden:

D	day	Tag
H	hour	Stunde
M	minute	Minute
S	second	Sekunde
MS	milli second	Millisekunde

Eine typische TIME-Konstante sieht so aus:

T#2D44MS	2 Tage und 44 Millisekunden
T#20MS	20 Millisekunden
T#1M5S	1 Minute und 5 Sekunden
T#1H242S5MS	1 Stunde und 242 Sekunden und 5 Millisekunden

DATE

Das Datenformat DATE ist 1 Word groß und gibt die Tage seit dem 1.1.1990 an (dualcodiert).

01.01.1990	D#1990-01-01	Wert: 0
24.12.1998	D#1998-12-24	Wert: 3279

DATE_AND_TIME

Dieser Datentyp ist 8 Bytes groß und kann daher nicht mehr mit “L” in den nur 4 Bytes großen Akku geladen werden, sondern nur als Parameter übergeben werden. Eine typische DATE_AND_TIME – Konstante sieht so aus:

DT#99-12-24-18:30:39.7 Bescherung 1999

Die Bedeutung der Bytes ist (BCD-codiert):
Byte 0 Jahr
Byte 1 Monat
Byte 2 Tag
Byte 3 Stunde
Byte 4 Minute
Byte 5 Sekunde
Byte 6 1/100 - Sekunden
Byte 7 (High-Nibble): 1/1000 - Sekunden

Im Low-Nibble von Byte 7 kann der Wochentag gespeichert werden, diese Funktion wird aber von TrySim nicht unterstützt.

Mit der IEC-Funktion FC 3 können Sie eine DT-Variable aus DATE und TIME_OF_DAY zusammensetzen.

Mit der Systemfunktion SFC1 können Sie das aktuelle Datum und die aktuelle Zeit in eine Variable des Typs DATE_AND_TIME übertragen.

Die IEC-Funktion FC6 extrahiert das Datum aus einer DT-Variable.
Die IEC-Funktion FC8 extrahiert die Tageszeit aus einer DT-Variable.
Die IEC-Funktion FC9 vergleicht zwei DTs auf gleich
Die IEC-Funktion FC12 vergleicht DT1 >= DT2
Die IEC-Funktion FC14 vergleicht DT1 > DT2
Die IEC-Funktion FC18 vergleicht DT1 <= DT2
Die IEC-Funktion FC23 vergleicht DT1 < DT2
Die IEC-Funktion FC28 vergleicht DT1 <> DT2

Bitte beachten Sie, dass DT-Parameter von Funktionen in TrySim ganz anders übergeben werden als in STEP®7. Vermeiden Sie daher Zugriffe auf DT-Parameter mittels der Adressregister.

TIME_OF_DAY

Dieser Datentyp ist 4 Bytes groß und wird intern durch die Anzahl der abgelaufenen Millisekunden des aktuellen Tages dargestellt (dualcodiert). Eine typische TIME_OF_DAY-Konstante sieht so aus:

TOD#23:12:04.090 Zeit, ins Bett zu gehen.

ANY

Dieser Datentyp ist ein Zeiger, der auf irgendeinen Datenbereich weist und gleichzeitig angibt, wie viele Daten welchen Typs dort zu finden sind.
Eine typische Notation ist:

P#M22.0 WORD 3

Dieser Ausdruck bezeichnet die Merkerwörter MW 22, MW 24 und MW 26.
Der Datentyp ANY hat daher einiges gemeinsam mit dem Typ ARRAY, während aber bei einem FB-Aufruf mit einem Array-Parameter tatsächlich alle Elemente des Arrays übergeben werden, werden beim Aufruf mit einem ANY-Parameter nur die 10 Bytes übergeben, die zur Beschreibung eines ANYs notwendig sind.

Sie können als Aktualparameter für einen ANY jedoch auch einfach ein Merkerwort, eine Struktur in einem DB oder die Lokaldaten angeben.

ANYs sind manchmal recht schwierig zu handhaben und wenn es nicht unbedingt sein muss, empfehlen wir Ihnen, sie in TrySim nicht zu verwenden. Wir müssen leider zugeben, dass wir die in STEP®7 gültige Syntax und Semantik der ANYs nicht vollständig begriffen haben. Sie scheint sich zu dem von STEP®7-Version zu - Version leicht zu ändern. Wenn Sie irgendwelche Probleme mit ANYs haben, informieren Sie uns bitte sofort, hier ist ein Fehler unsererseits so wahrscheinlich wie in kaum einem anderen Bereich von TrySim sonst.

Wenn Sie an einen als ANY-deklarierten Parameter einen Operanden anschließen, beachten Sie folgende Besonderheiten:

1. Sie können den Operanden wie oben beschrieben als Konstante schreiben:
z.B. P#DB100.DBX 3.0 S5TIME 3
Dann werden die 10 Bytes, die zur Beschreibung dieser Konstant notwendig sind, ohne Modifikationen an den aufgerufenen Baustein übergeben.

2. Sie können als Operanden auch Parameter und Variablen angeben, die im Bausteinkopf deklariert worden sind. Der Übersetzer bestimmt dann automatisch die richtige Adresse (Leider schreibt er sie dann auch so, darum erkennen Sie Ihren Operanden kaum wieder, das werden wir aber irgendwann ändern). Während der Laufzeit werden dann aber noch folgende Umsetzungen unternommen:
2.a Wenn es sich um einen Parameter oder eine statische Variable handelt, wird nicht die Kennung 85h (für Instanzdaten) übertragen, sondern die Kennung 84h (für Global - DB) und die aktuelle IDB - Nr. wird übergeben.
2.b Wenn es sich um eine lokale Variable handelt, wird nicht die Kennung 86h (für Lokaldaten), sondern die Kennung 87h (für vorherige Lokaldaten) übergeben.
2.c Parameter von Funktionen (FC) können Sie nicht an ein ANY-Parameter übergeben.

3. Wenn Sie jedoch einen als ANY-deklarierten Parameter oder Variable angeben, wird ausnahmsweise nicht die Adresse dieses Parameters übergeben, sondern es werden die 10 Bytes, die an dieser Adresse stehen, übergeben.

BLOCK_FB

Sie können einen Funktionsbaustein als Parameter übergeben, damit in dem aufgerufenen Baustein wechselnde Funktionen ausgeführt werden. Für den Aufruf müssen Sie dann eine der Operationen UC oder CC verwenden. Beachten Sie die dort beschriebenen Gefahren beim Aufruf von Funktionsbausteinen, die In/Out-Parameter haben oder statische Variablen verwenden.

Die Operation CALL ist für Aufrufe mit Angabe von Parametern reserviert. Parameter können aber beim Aufruf eines BLOCK_FB nicht angegeben werden, da ja zur Entwicklungszeit nicht bekannt ist, welcher FB aufgerufen wird.

BLOCK_FC

Sie können eine Funktion als Parameter übergeben, damit in dem aufgerufenen Baustein wechselnde Funktionen ausgeführt werden. Für den Aufruf müssen Sie dann eine der Operationen UC oder CC verwenden. Beachten Sie die dort beschriebenen Gefahren beim Aufruf von Funktionen die In/Out-Parameter haben.

Die Operation CALL ist für Aufrufe mit Angabe von Parametern reserviert. Parameter können aber beim Aufruf eines BLOCK_FC nicht angegeben werden, da ja zur Entwicklungszeit nicht bekannt ist, welche FC aufgerufen wird.

BLOCK_DB

Sie können einen Baustein als Parameter übergeben, damit in dem aufgerufenen Baustein wechselnde Funktionen ausgeführt werden.

REAL

Dieser Datentyp ist 4 Bytes groß. Das interne Format ist für Menschen kaum zu lesen. Der Zahlenbereich geht von ca. – 10 hoch 38 bis – 10 hoch – 38, Null und dann von 10 hoch –38 bis 10 hoch 38.
Bei der Eingabe von Real-Konstanten müssen Sie auf jeden Fall einen Punkt eingeben, also “8.0” und nicht “8”, sonst interpretiert das System Ihre Eingabe als Integer, was bei der Auswertung als REAL-Zahl zu absurden Ergebnissen führt.

RND	Wandelt REAL nach DINT
DTR	Wandelt DINT nach REAL

STRING

Strings sind ab Version 1.5 implementiert. Sie können Strings deklarieren, als Parameter von Funktionen verwenden und im- und exportieren. Auf die einzelnen Buchstaben greifen Sie mit der Notation Stringname[idx] zu.

Folgende Funktionen stehen im Verzeichnis IECfuncs für die Bearbeitung von Strings bislang zur Verfügung:

FC 21 Länge eines Strings ermitteln.

Weitere sind in Vorbereitung, bei Bedarf bitte nachfragen.

UDT

Vom Anwender definierter Datentyp. (**u**ser **d**efined **t**ype).

Wenn häufig große Strukturen benötigt werden, ist es lästig, sie in den Deklarationsteilen der Bausteine und in den DB jedesmal ausschreiben zu müssen. Daher kann man Strukturen als UDT definieren und diese danach wie die normalen Datentypen verwenden.

UDT werden wie Funktionsbausteinköpfe definiert, nur, dass dabei kein Deklarationstyp festgelegt werden muss.

UDT können auch als “Maske” zur Erstellung neuer Datenbausteine dienen.

Wenn Sie einen UDT in die Symboltabelle eingetragen haben, können Sie als Typ-Bezeichnung auch das Symbol verwenden.

3.6.2 Konstanten

Konstanten können in verschieden Darstellungen eingegeben werden:

Dual-Konstante, z.B. 2#1100. Jede Stelle steht für ein Bit und darf nur 0 oder 1 sein. diese Konstante kann für Bytes, Words und DWords verwendet werden. Natürlich muss die maximale Anzahl der Bits berücksichtigt werden.

Byte-Konstante hex, z.B. B#16#F5. Jede Stelle steht für ein Halb-Byte und darf nur 0..9, A..F sein. Der maximale Wert dieser Konstanten ist FF hex oder 255 dezimal.

Word-Konstante hex, z.B. W#16#6F34. Jede Stelle steht für ein Halb-Byte und darf nur 0..9, A..F sein. Der maximale Wert dieser Konstanten ist FFFF hex oder 65535 dezimal.

Word-Konstante als zwei dezimale Bytes, z.B. B#(152,43). Die beiden Zahlen geben die beiden Bytes des Words an und werden dezimal angegeben. Der maximale Wert dieser Konstanten ist B#(255,255).

DWord-Konstante hex, z.B. DW#6FA42322. Jede Stelle steht für ein Halb-Byte und darf nur 0..9,A..F sein. Der maximale Wert dieser Konstanten ist FFFF FFFF hex oder ca. 4 Milliarden dezimal.

Integer-Konstante, z.B. -5. Diese Konstante entspricht unseren ganzen Zahlen. Sie ist die Einzige, die ohne # eingegeben werden kann. Der Wertebereich geht von -32.768 bis +32.767.

Double Integer-Konstante, z.B. L#334124. Diese Konstante entspricht der Integer-Konstante mit einem erweitertem Wertebereich von -2.147.483.648 bis +2.147.483.647. Wenn Sie die Double Integer Operationen +D, -D, *D, /D, <D, >D usw. verwenden und eine negative Konstante laden, ist es _wichtig_, dass Sie das L# voranstellen.

Real Konstanten, z.B. 3.5 oder 5.342124e+002. Wenn Sie mit Realzahlen rechnen oder diese vergleichen und dabei Konstanten verwenden, ist es _wichtig_, dass Sie den '.' mit angeben. 5 und 5.0 sind etwas total Verschiedenes für eine SPS!

S5-Time-Konstanten, z.B. S5T#1H2M. Die Syntax ist unter S5Time erklärt. Diese Konstante benötigen Sie, um Timer zu starten. Verwenden Sie auf keinen Fall eine Konstante vom Typ TIME, z.B. T#200MS !

Zähler-Konstanten, z.B. C#59. Hiermit laden Sie die angegebene Zahl BCD - codiert. Der Wertebereich geht von C#0 bis C#999.

Time-Konstanten, z.B. T#5D2H5M. Hiermit laden Sie die angegebene Zeitdauer in ms als double integer (DINT).

3.7 Liste der Operationen

Siehe auch: Liste der Operationen (alphabetisch)

Bit-Operationen

U	U(	UN	UN(
O	O(	ON	ON(
X	X(	XN	XN(
)(Klammer zu)			
= (Zuweisung)	S	R	
SET	CLR	NOT	SAVE
FP	FN		

Operationen mit Zeiten

SE	SA	SI
SV	SS	ZW (Parameter)

Operationen mit Zählern

ZV	ZR	Setze Zähler	Rücksetze Zähler
FR			

Lade- und Transferoperationen

L	T	LC

Operationen mit Integers

+I	-I	*I	/I

MOD
= = I <>I >I >=I
<I <=I NEGI
+ (Plus)

Operationen mit Double-Integers
+D -D *D /D
MOD
= = D <>D >D >=D
<D <=D NEGD
+ (Plus)

Operationen mit Real-Zahlen
+R -R *R /R
= = R <>R >R >=R
<R <=R
ABS NEGR
SQRT SQR LN EXP

Sprung-Operationen
SPA SPBN SPMZ SPN
SPP SPPZ SPZ SPBNB
SPB SPBB SPBI SPBIN
LOOP SPL

Nicht implementierte Sprünge:
SPO SPU SPS

Schiebe- und Rotier-Operationen
SRD SSD
RLD RLDA SLD SLW
RRDA RRD SSI SRW

Logische Wort- und DWort-Operationen
UD UW OW OD
XOD XOW
INVI INVD

BCD-Operationen
ITB DTB BTD BTI

Andere Umwandlungen
DTR ITD TRUNC
RND RND+ RND-

Trigonometrische Operationen
SIN COS TAN

ASIN ACOS ATAN

Sonstige Operationen mit Akku 1
TAK TAW TAD
INC DEC + (Plus)

Operationen mit Akku 3 und 4
PUSH POP LEAVE ENT

Register-indirekte Adressierung
LAR1 oder 2 +AR1 oder 2 TAR1 oder 2 TAR

Baustein-Operationen
AUF BEA BEB
CALL CC UC
TDB

Null-Operationen
NOP 0 oder 1 BLD

Master Control Relais (nicht implementiert)
MCR( )MCR MCRA MCRD

3.7.1 Bit-Operationen

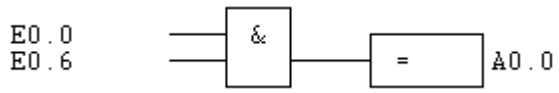
U

UND

AWL

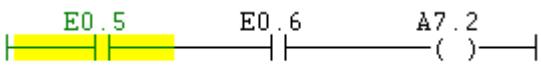
Verknüpft den Wert des Operanden und das VKE mit der Operation “Und” und speichert das Ergebnis im VKE. Wenn das VKE zuvor abgefragt worden ist, wird der Wert des Operanden ohne Verknüpfung in das VKE übertragen.

FUP



Der Ausgang des Und-Blockes ist “1”, wenn alle Eingänge “1” sind.
Erzeugt wird der Und-Block mittels des Symbols “&” oder über die Liste und Wahl von “Und”.

KOP



Hier wird die UND-Verknüpfung durch Hintereinanderschalten mehrerer Kontakte

gebildet. Die gesamte Kette leitet den Strom nur, wenn alle Kontakte geschlossen sind.

Erzeugt wird ein neuer Kontakt mittels des Symbols  oder über die Liste und Wahl von “Und”.

U(

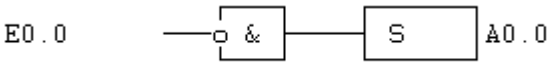
UND mit Verzweigung

Beginnt einen Klammer-Ausdruck, dessen Ergebnis nach der abschließenden “)” Klammer mit dem VKE “Und”- verknüpft wird. Die erste Operation nach “U(“ überträgt den Wert des Operanden ohne Verknüpfung ins VKE.

UN

UND NICHT

Verknüpft den negierten Wert des Operanden und das VKE mit der Operation “Und” und speichert das Ergebnis im VKE. Wenn das VKE zuvor abgefragt worden ist, wird der negierte Wert des Operanden ohne Verknüpfung in das VKE übertragen.



UN(

UND NICHT mit Verzweigung

Beginnt einen Klammer-Ausdruck, dessen negiertes Ergebnis nach der abschließenden “)” Klammer mit dem VKE “Und”-verknüpft wird. Die erste Operation nach “UN(“ überträgt den Wert des Operanden ohne Verknüpfung ins VKE.

O

ODER

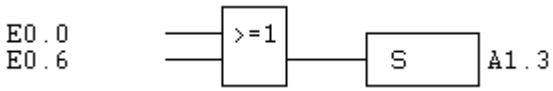
AWL

Beginnt eine “Oder”-Verknüpfung nach der Und-vor-Oder-Regel. Intern wird “O” ohne Operanden durch “O(“ ersetzt. Vor der nächsten, das VKE-auswertenden Operation, wird die “)” automatisch eingefügt.

ODER mit Operanden

Verknüpft den Wert des Operanden und das VKE mit der Operation “Oder” und speichert das Ergebnis im VKE. Wenn das VKE zuvor abgefragt worden ist, wird der Wert des Operanden ohne Verknüpfung in das VKE kopiert.

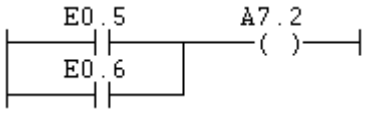
FUP



Der Ausgang des Oder-Blockes ist “1” wenn mindestens einer der Eingänge “1” ist.

Erzeugt wird der Oder-Block mittels des Symbols >= oder über die Liste und Wahl von “Oder”.

KOP



Hier wird die Oder-Verknüpfung durch Parallelschalten mehrerer Kontakte erreicht. Der gesamte Pfad ist leitend, wenn mindestens einer der Kontakte geschlossen ist.

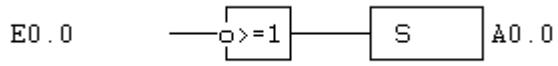
Erzeugt wird eine Oder-Verknüpfung am Einfachsten durch das Symbol (dabei darauf achten, wo der gelbe Cursor steht). Nachdem die gewünschten Kontakt mittels oder eingefügt sind, wird die Verzweigung mit wieder geschlossen. Vorher den gelben Cursor auf den Kontakt stellen, **hinter** dem die Verzweigung wieder geschlossen werden soll.

O(

ODER mit Verzweigung
Beginnt einen Klammer-Ausdruck, dessen Ergebnis nach der abschließenden “)”-Klammer mit dem VKE “Oder” verknüpft wird. Das Ergebnis wird im VKE gespeichert. Die erste Operation nach “O(“ überträgt den Wert des Operanden ohne Verknüpfung ins VKE.

ON

ODER NICHT
Verknüpft den negierten Wert des Operanden mit dem VKE und speichert das Ergebnis im VKE. Wenn das VKE zuvor abgefragt worden ist, wird der Wert des Operanden ohne Verknüpfung in das VKE kopiert.



ON(

ODER NICHT mit Verzweigung
Beginnt einen Klammer-Ausdruck, dessen negiertes Ergebnis nach der abschließenden “)”-Klammer mit dem VKE “Oder” verknüpft wird. Das Ergebnis wird im VKE gespeichert. Die erste Operation nach “O(“ überträgt den Wert des Operanden ohne Verknüpfung ins VKE.

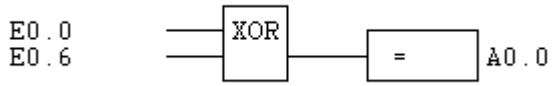
X

EXLUSIV ODER

Verknüpft den Wert des Operanden mit dem VKE nach der “Exklusiv Oder”-Regel und speichert das Ergebnis im VKE.

Wert des Operanden	Altes VKE	Neues VKE
0	0	0
1	0	1
0	1	1
1	1	0

Wenn das VKE zuvor abgefragt worden ist, wird der Wert des Operanden ohne Verknüpfung ins VKE übertragen.



X(

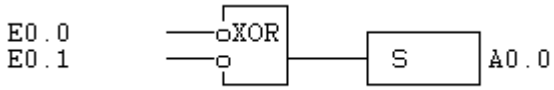
EXLUSIV ODER mit Verzweigung

Beginnt einen Klammerausdruck, dessen Ergebnis bei der abschließenden “(“-Klammer mit dem VKE “Exklusiv Oder” verknüpft wird. Die erste Operation nach “X(“ überträgt den Wert des Operanden ohne Verknüpfung ins VKE.

XN

EXLUSIV ODER NICHT

Verknüpft den negierten Wert des Operanden mit dem VKE nach der “Exklusiv Oder” Regel und speichert das Ergebnis im VKE. Wenn das VKE zuvor abgefragt worden ist, wird der Wert des Operanden ohne Verknüpfung ins VKE übertragen.



Diese Operation kann in KOP nicht programmiert werden.

XN(

EXLUSIV ODER NICHT mit Verzweigung

Beginnt eine Klammerausdruck, dessen negiertes Ergebnis bei der abschließenden “(“-Klammer mit dem VKE “Exklusiv Oder” verknüpft wird. Die erste Operation nach “XN(“ überträgt den Wert des Operanden ohne Verknüpfung ins VKE.

) (Klammer zu)

Mit der “)” wird der zuletzt programmierte, noch nicht abgeschlossene Klammerausdruck abgeschlossen und das Ergebnis des Klammerausdrucks über die Operation, die beim Öffnen des Klammerausdrucks angegeben worden ist, mit dem damaligen VKE verknüpft und als neues VKE gespeichert.

Die Operationen, die beim Öffnen eines Klammerausdruckes angegeben werden können, sind:

= (Zuweisung)

Operanden:

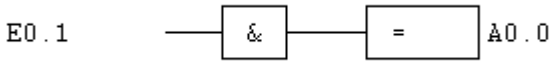
E, A, M
DBX, DIX,
BOOL-Parameter
BOOL-Variablen

AWL

Syntax: = A 5.2

Der Inhalt des VKE wird auf den angegebenen Bit-Operanden kopiert. Nach einer Zuweisung wird eine neue Verknüpfungskette angefangen, d.h. die erste Operation nach einem “=” liefert immer nur den Wert des Operanden, gleichgültig, ob die Operation “U”, “O” oder “X” heißt. Die Operationen “UN”, “ON” und “XN” dagegen beschicken das VKE mit dem negierten Wert des Operanden.

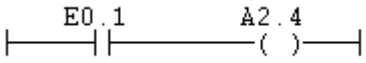
FUP



Das Ergebnis der vorangegangen Verknüpfung wird in den angegebenen Operanden kopiert. Der Zuweisungsblock wird meistens automatisch erzeugt, Sie können jedoch auch die Liste aufrufen und “= Zuweisung wählen”.

Sie können das “=“ -Zeichen mittels der Leertaste oder “Shift-PfeilRechts” nach “S” oder “R” ändern. Der gelbe Cursor muss dazu auf dem Block stehen.

KOP



In KOP wird die Zuweisung meistens “Spule” genannt. Die Spule wird eingeschaltet, wenn Strom durch sie fließt. Die Spule wird meistens automatisch erzeugt, Sie können jedoch auch die Liste aufrufen und “= Zuweisung wählen”.

Sie können das “-( )-“ Zeichen mittels der Leertaste oder “Shift-PfeilRechts” nach “-(S)-” oder “-(R)” ändern. Der gelbe Cursor muss dazu auf der Spule stehen.

S

Operanden:

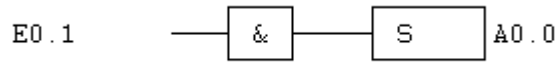
E, A, M
DBX, DIX,
BOOL-Parameter
BOOL-Variablen

AWL

```
Syntax:   U    E 0.1   // z.B.
          S    A 5.2
```

Wenn das VKE "1" ist, wird der angegebenen Bit-Operanden auf "1" gesetzt. Ist das VKE "0", passiert nichts. Nach dieser Operation wird eine neue Verknüpfungskette angefangen, d.h. die erste Operation nach einem "S" liefert immer nur den Wert des Operanden, gleichgültig, ob die Operation "U", "O" oder "X" heißt. Die Operationen "UN", "ON" und "XN" dagegen beschicken das VKE mit dem negierten Wert des Operanden.

FUP



Wenn Ergebnis der vorangegangen Verknüpfung "1" ist, wird der angegebene Operand auf "1" gesetzt. Wenn das Ergebnis der vorangegangen Verknüpfung "0" ist, passiert nichts. D.h., der Operand wird nicht wieder zurückgesetzt. Dies muss an anderer Stelle im Programm geschehen, z.B. durch die Operation R. Sie erzeugen diesen Block, indem Sie zunächst ein Zuweisungsblock erzeugen, den gelben Cursor auf ihn stellen und die Leertaste oder "Shift-PfeilRechts" betätigen.

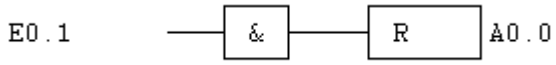
KOP

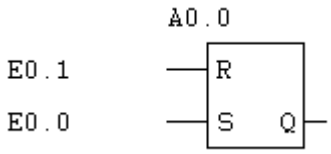
Hier wird die Spule nur eingeschaltet, wenn Strom fließt. Wenn kein Strom fließt, wird sie nicht wieder ausgeschaltet, das muss an anderer Stelle im Programm geschehen, z.B. durch die Operation R .

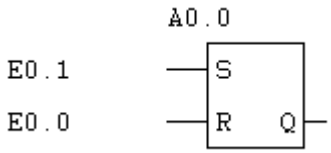
Sie erzeugen das "-(S)-" - Zeichen, indem Sie zunächst eine normale Spule herstellen und dann die Leertaste oder "Shift-PfeilRechts" betätigen. Der gelbe Cursor muss dazu auf der Spule stehen.

R

Wenn der Eingang "1" ist, wird der Ausgang zurückgesetzt, sonst passiert gar nichts.







SET

Das VKE wird einfach auf "1" gesetzt, mehr kann man dazu kaum sagen.

CLR

Dieser Befehl setzt einfach nur das VKE auf “0”, mehr kann man über ihn nicht sagen.

NOT

Wenn das VKE "1" ist, wird es zu "0"und wenn es "0" ist wird es zu "1".

Dieser Befehl wird vom System auch verwendet, um die Negation eines FUP - UND/ ODER - Blockes in AWL auszudrücken.

SAVE

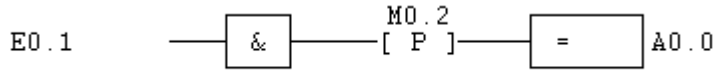
Das aktuelle VKE wird im BIE-Bit gespeichert. Das BIE-Bit wird beim Rücksprung aus einer Funktion oder einem Funktionsbaustein ausgewertet, in FUP und KOP ist das BIE-Bit der ENO - Ausgang.

FP

Wenn der Eingang eine steigende (**p**ositive) **F**lanke hat, wird der Ausgang für einen Zyklus auf “1” gesetzt. Unbedingt darauf achten, dass der Flankenmerker (im Bild M 0.2) nirgendwo (d.h. weder im Programm noch in der Anlage) anders verwendet wird !

Syntax:

```
U     E 0.1
FP    M 0.2
=     A 0.0
```



Anmerkung:
In STEP®7 FUP/KOP gibt es auch noch eine andere Darstellung der Flanke, die in AWL so notiert wird.

```
U     E 0.1
BLD   100
FP    M 0.2
=     A 0.0
```

Diese Darstellung wird von TrySim nicht unterstützt.

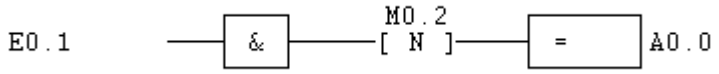
STEP®7 ist eingetragenes Warenzeichen der Siemens AG.

FN

Wenn der Eingang eine fallende (**n**egative) **F**lanke hat, wird der Ausgang für einen Zyklus auf “1” gesetzt. Unbedingt darauf achten, dass der Flankenmerker (im Bild M 0.2) nirgendwo (d.h. weder im Programm noch in der Anlage) anders verwendet wird !

Syntax:

```
U     E 0.1
FN    M 0.2
=     A 0.0
```



Anmerkung:
In STEP®7 FUP/KOP gibt es auch noch eine andere Darstellung der Flanke, die in AWL so notiert wird.

```
U     E 0.1
BLD   100
FN    M 0.2
=     A 0.0
```

Diese Darstellung wird von TrySim nicht unterstützt.

STEP®7 ist eingetragenes Warenzeichen der Siemens AG.

3.7.2 Operationen mit Zeiten

Es gibt in der SPS 512 “Zeiten”, die zum Verzögern von Ereignissen oder zur Impulserzeugung verwendet werden können. Die Zeiten sind numeriert von T0 bis T511, selbstverständlich kann ihnen über die Symboltabelle auch ein vernünftiger Name gegeben werden. Welche Funktion eine Zeit hat, ist nicht von vornherein festgelegt, sondern wird beim Starten der Zeit bestimmt. Folgende Funktionen sind möglich:
Einschaltverzögerung
Ausschaltverzögerung
Impuls
Verlängerter Impuls
Speichernde Einschaltverzögerung

Im Beispiel “Für_Anfänger” wird der Gebrauch von Zeiten demonstriert.

Wie die Zeitdauer einer Zeit eingegeben wird, steht weiter unten in diesem Kapitel.

Die Zeiten können auch speicher-indirekt oder register-indirekt angeben werden.

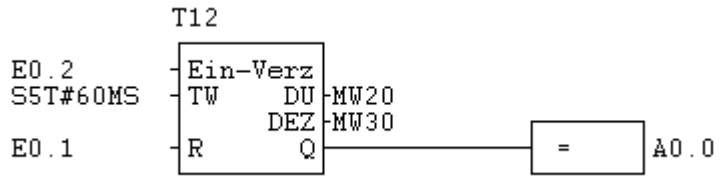
Operation	Operanden	Beschreibung
SE	T TIMER-Parameter	Startet eine Zeit als Einschaltverzögerung, wenn das VKE von “0” auf “1” wechselt.
SA	T TIMER-Parameter	Startet eine Zeit als Ausschaltverzögerung, wenn das VKE von “0” auf “1” wechselt.
SI	T TIMER-Parameter	Startet eine Zeit als Impuls, wenn das VKE von “0” auf“1” wechselt.
SV	T TIMER-Parameter	Startet eine Zeit als verlängerter Impuls, wenn das VKE von “0” auf “1” wechselt.
SS	T TIMER-Parameter	Startet eine Zeit als speichernde Einschaltverzögerung, wenn das VKE von “0” auf “1” wechselt.
R	T TIMER-Parameter	Wenn das VKE “1” ist, wird die Zeit zurückgesetzt. Solange das VKE “1” ist, ergibt eine Abfrage der Zeit auf jeden Fall “0” und ein Auslesen der abgelaufenen Zeit mit L Txx oder LC Txx liefert in jedem Fall “0”.
L	T TIMER-	Lädt den aktuellen Wert der Zeit in den Akku 1. Die Zeiten laufen von dem Wert, mit dem sie gestartet

Operation	Operanden	Beschreibung
	Parameter	worden sind, herunter bis “0”. Mit diesem Befehl wird der aktuelle Wert in Einheiten des Zeitrasters dualcodiert geladen. Bei der Auswertung der so geladenen Zahl muss daher berücksichtigt werden, mit welchem Zeitraster die Zeit gestartet worden ist.
LC	T TIMER- Parameter	Lädt den aktuellen Wert der Zeit BCD-codiert mit Zeitraster in den Akku 1. Ein so geladener Zeitwert kann direkt verwendet werden, um einen weitere Zeit zu starten.

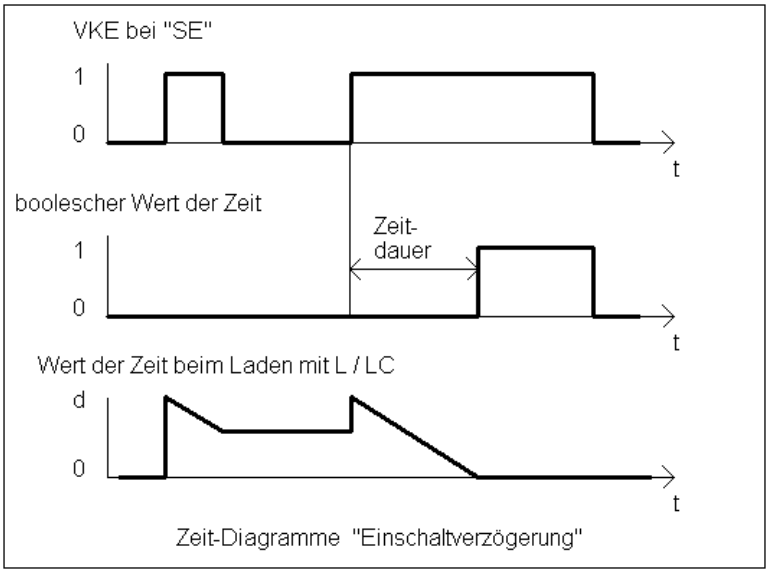
Die Funktion “FR”, freigeben einer Zeit ist in TrySim nicht implementiert.

SE

```
U E 0.2
SE T12
```



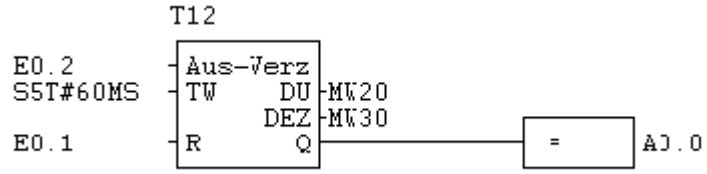
· Einschaltverzögerung



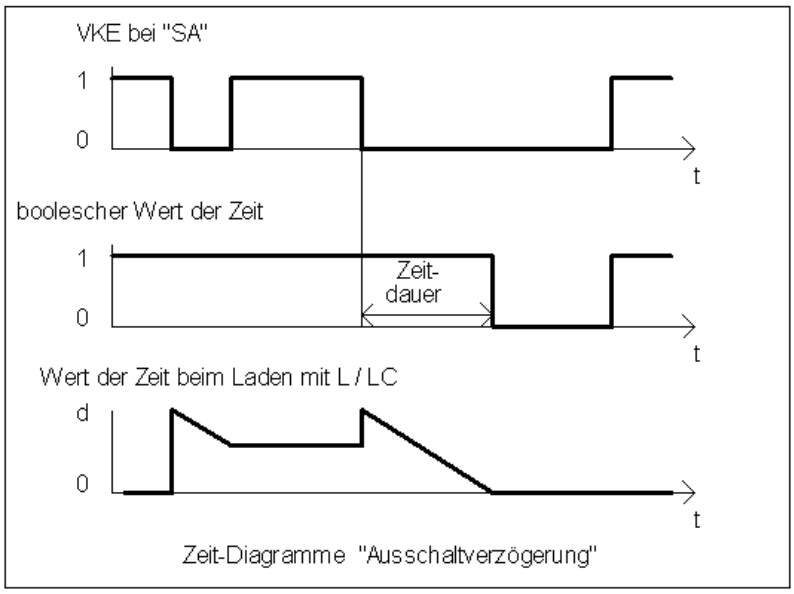
Zeit-Diagramme "Einschaltverzögerung"

SA

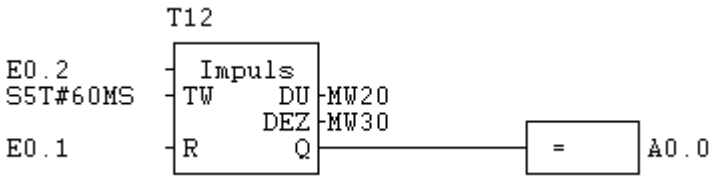
U E 0.2
SA T12



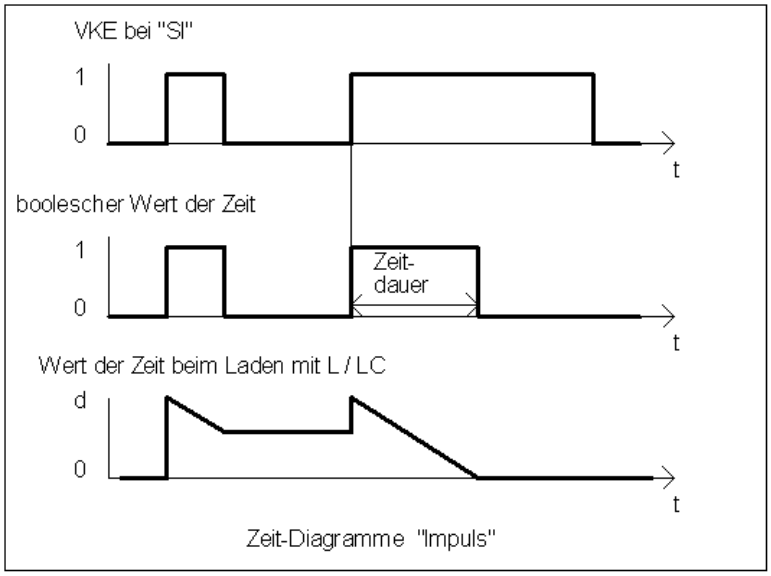
· Ausschaltverzögerung



SI



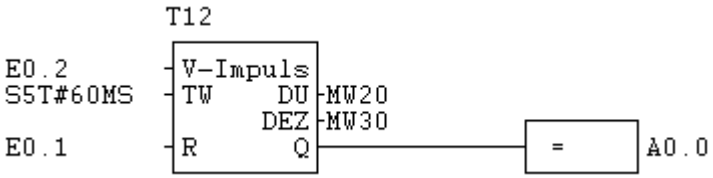
· Impuls



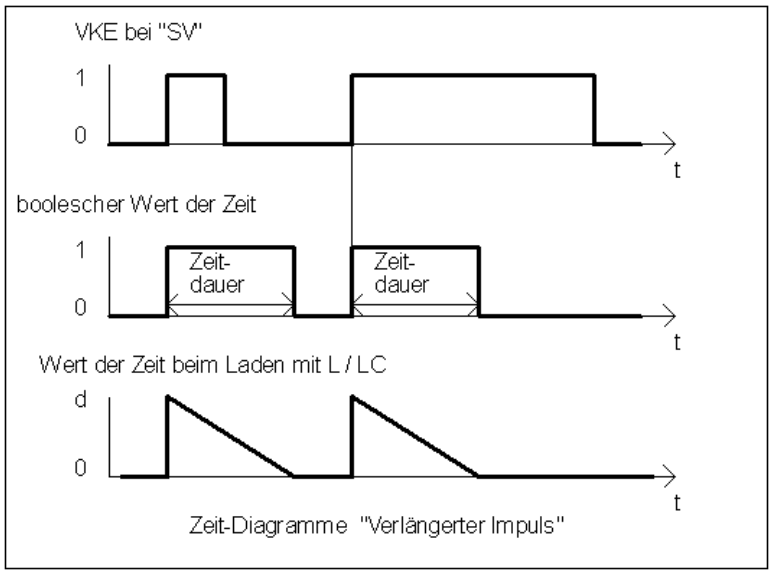
Zeit-Diagramme "Impuls"

SV

```
U  E 0.2
SV  T12
```



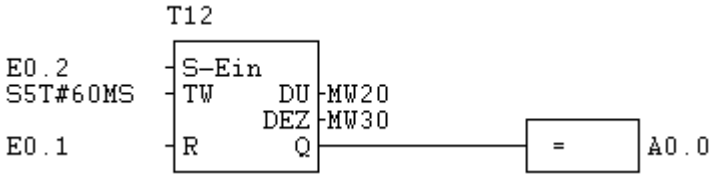
· Verlängerter Impuls



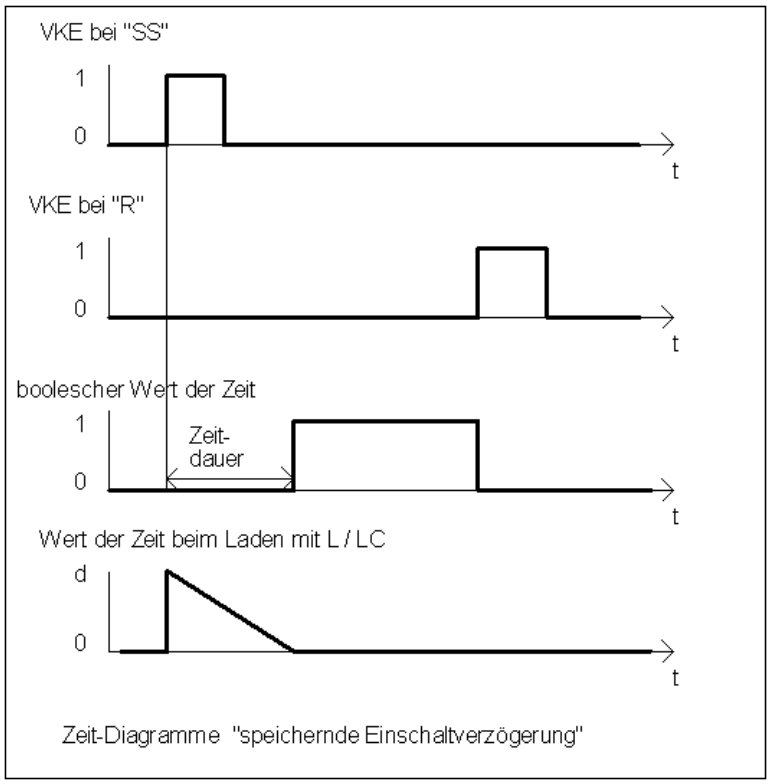
Zeit-Diagramme "Verlängerter Impuls"

SS

```
U  E 0.2
SS  T12
```



· **Speichernde Einschaltverzögerung**



ZW

Auf diesen Wert wird der Zähler gesetzt, wenn eine steigende Flanke am S-Eingang auftritt. Ganz wichtig ist, dass dieser Wert als BCD angegeben wird. Wenn man hier eine Konstante angeben will, sollte man die C#xyz - Notation verwenden. Wenn man eine vorher berechnete und abgespeicherte Zahl als ZW-Wert verwenden möchte, dann wird die Funktion ITB von Nutzem sein.

Ausgang DU einer Zeit

An diesen Ausgang der Zeit können Sie eine beliebige Word-Variable anschließen. In diese Variable wird dann kontinuierlich der aktuelle Wert der Zeit übertragen. Für einfache SPS-Programme wird dies nicht benötigt.
An diesem Ausgang wird der aktuelle Wert der Zeit dualcodiert in dem Zeitraster ausgeben, mit dem die Zeit gestartet wurde. Das Zeitraster ist im ausgegebenen Wert aber nicht mehr enthalten, daher muss man, um ihn interpretieren zu können, wissen, mit welchem Zeitraster die Zeit gestartet worden ist. Im Zweifelsfall ist es besser, den am DEZ-Ausgang ausgegebenen Wert zu verwenden und diesen dann in ms umzurechnen.

Ausgang DE einer Zeit

An diesen Ausgang der Zeit können Sie eine beliebige Word-Variable anschließen. In diese Variable wird dann kontinuierlich der aktuelle Wert der Zeit übertragen. Für einfache SPS-Programme wird dies nicht benötigt.
An diesem Ausgang wird der aktuelle Wert der Zeit BCD-codiert zusammen mit dem Zeitraster ausgeben, mit dem die Zeit gestartet wurde, Sie können ihn also direkt verwenden, um eine andere Zeit zu starten.

Wenn Sie die ausgelesene Zeitdauer in ms umrechnen wollen, können Sie z.B. folgenden AWL-Code programmieren:

```
      LC    Tx          // 0 < x < 511; LC = Lade codiert
      T     #ZeitW      // #ZeitW ist als Word-Variable deklariert
      L     W#16#0FFF   // Maske für BCD-codierten Roh-Wert
      UW                // most significant nibble ausblenden
      BTI               // BCD - INT Konvertierung
      T     #Roh        // #Roh ist als Int-Variable deklariert
      L     #ZeitW      // Zwischengespeicherter Wert
      SRW   12          // 12 Bits nach rechts schieben
      SPL   err         // Sprungliste
      SPA   R0          // Zeitraster 10 ms
      SPA   R1          // Zeitraster 100 ms
      SPA   R2          // Zeitraster 1 s
      SPA   R3          // Zeitraster 10 s
err : NOP   0
R0    L     10          // 10 ms der Wert
      SPA   Mul
R1    L     100         // 100 ms
      SPA   Mul
R2    L     1000        // 1.000 ms = 1 s
      SPA   Mul
R3    L     10000       // 10.000 ms = 10 s
Mul : L     #Roh
      *D                // Achtung! Das Ergebnis kann > 32767 sein
                                    // Jetzt liegt die Zeit
                        in ms dualcodiert als 32-Bit-Zahl im Akku 1
```

Zeiten in FUP/KOP einfügen

Positionieren Sie den gelben Cursor an der gewünschten Position.

Wählen Sie aus der Liste den gewünschten Zeit-Typ.

Beschriften Sie die Zeit über dem Block mit der Nummer der Zeit.

Legen Sie am Eingang oben links fest, wann die Zeit gestartet werden soll. Dort können Sie selbstverständlich auch einen ganzen Logik-Baum anbauen.

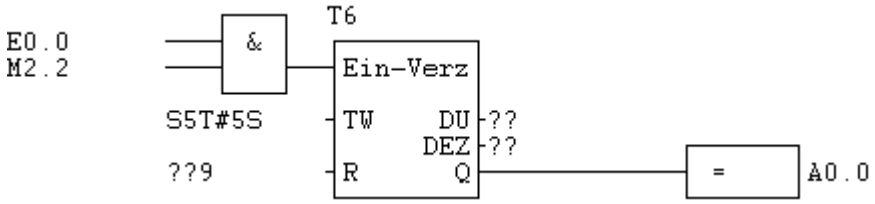
Legen Sie die Zeitdauer fest. Im einfachsten Fall ist dies eine Zeit-Konstante. Sie können hier aber auch jeden Word-Speicher angeben, müssen dann aber dafür sorgen, das dort eine gültige Zeit-Dauer steht.

Den Rücksetz-Eingang "R" sollten Sie nach Möglichkeit unbeschaltet lassen. Mit Ausnahme der speichernden Einschaltverzögerung wird er nicht unbedingt gebraucht und schadet i.A. mehr, als er nützt.

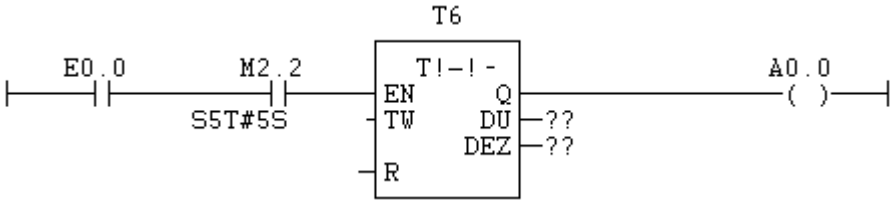
Auch die DU- und DEZ-Ausgänge werden i.A. nicht benötigt. Hier können Sie den Wert der ablaufenden Zeit in zwei Formaten in Word-Speicher übertragen lassen.

Wichtig ist jedoch der Q-Ausgang. Hier wird ausgegeben, ob die Zeit abgelaufen ist oder nicht. Wenn Sie die Zeit nicht nur in diesem Netzwerk verwenden, wird Ihr Programm leichter lesbar, wenn Sie hier eine boolesche Variable (Merker, Stat- oder Temp) anschließen und diese im weiteren Verlauf verwenden.

Beispiel einer Zeit in FUP:



Beispiel einer Zeit in KOP:



Und so sieht diese Zeit in AWL aus:

```
U  E   0.0
U  M   2.2
L   S5T#5S
SE T   6
NOP 0
NOP 0
NOP 0
U  T   6
=  A   0.0
```

Bezeichnung einer Zeit

Hier müssen Sie auswählen, welche Zeit Sie verwenden wollen. Die Bezeichnungen gehen von “T 0” bis “T 511”, beachten Sie jedoch, dass nicht jede CPU der S7-300/400-Reihe alle diese Zeiten hat.

Sie müssen auf jeden Fall darauf achten, dass Sie jede Zeit nur einmal verwenden. Die doppelte Verwendung von Zeiten ist in der SPS-Programmierung ein häufiger und schwer zu findender Fehler. Wenn zwei Menschen nur eine Eieruhr verwenden, um zu verschiedenen Zeiten Eier zu kochen, dann geht das häufig gut, aber wehe, wenn sie das fast zur gleichen Zeit versuchen. In der Querverweisliste können Sie kontrollieren, ob eine Zeit bereits verwendet wurde. Wenn Sie konsequent jede Zeit, die Sie verwenden, in die Symboltabelle eintragen, dann merken Sie sofort, wenn Sie eine Zeit versehentlich zweimal verwenden wollen, weil dann nämlich ein Symbol anstelle von z.B. T56 angezeigt wird.

S7-300 und S7-400 sind eingetragene Warenzeichen der Siemens AG.

Eingabe der Zeitdauer bei Verwendung von Zeiten

Die Dauer von Zeiten muss beim Starten im Akku 1 vorliegen, bzw. in FUP oder KOP am TW-Eingang angegeben werden und zwar in 4-stelligem BCD-Format, wobei die vierte Stelle (Tausender) das Zeitraster angibt:

0	0,01s
1	0,1 s
2	1 s
3	10 s

Das Laden des Akkus mit einer so codierten Zahl geschieht am einfachsten durch den Befehl "L S5T#". Die Zeitdauer wird dann im Format "*hHmMsSmsMS*" angegeben.
Beispiel:

S5T#1H20M	1 Stunden 20 Minuten
S5T#3M10S400MS	3 Minuten 10 Sekunden und 400 Millisekunden
S5T#5S	5 Sekunden

Das Zeitraster wird bei diesem Eingabeformat automatisch kleinstmöglich gewählt. Die maximale Zeit ist 999 * 10 s = 2 h 46 min 30 sec
Da die Zeitdauer, angegeben in Einheiten des Zeitrasters, nur bis 999 gehen kann, können bei einer zu genau angegebenen Zeit die niederwertigen Stellen verschwinden:

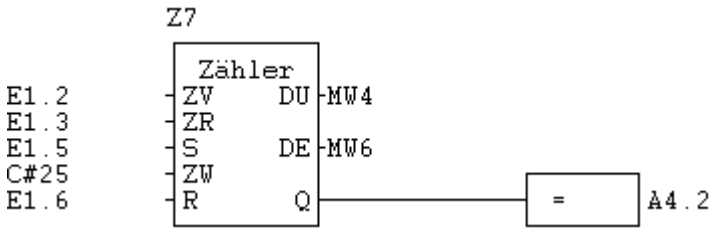
Aus S5T#2H500MS wird S5T#2H.

Das T#....- Format ist nicht zum Laden des Zeitwertes im richtigen Format geeignet, da so codierte Zeitdauern als Double-Integer in ms in den Akku geladen werden !

3.7.3 Operationen mit Zählern

Es gibt in der SPS 512 "Zähler", die zum Zählen von Ereignissen verwendet werden können. Die Zähler sind numeriert von Z0 bis Z511, selbstverständlich kann ihnen über die Symboltabelle auch ein vernünftiger Name gegeben werden.
Zähler können als Aufwärtszähler, Abwärtszähler und als Auf-und-Abwärtszähler verwendet werden. Der Zählbereich geht von 0 bis 999.

Beispiel eines Zählers in FUP:



Wenn am Eingang ZV eine steigende Flanke auftritt, wird der Zähl-Wert um 1 erhöht. Wenn am Eingang ZR eine steigende Flanke auftritt, wird der Zähl-Wert um 1 erniedrigt. Wenn am Eingang S eine steigende Flanke auftritt, wird der Zähl-Wert auf den am Eingang ZW anliegenden Wert gesetzt. Wenn am Eingang R eine “1” anliegt, wird der Zähler statisch auf 0 gesetzt, d.h., nicht nur bei einer Flanke. Am Ausgang DU wird der aktuelle Zähl-Wert dualcodiert in eine beliebige Word-Variable geschrieben. Am Ausgang DE wird der Zählwert BCD-codiert in eine beliebige Word-Variable geschrieben. In dieser Form kann er direkt wiederverwendet werden, um einem weiteren Zähler als ZW-Wert zu dienen. Der Ausgang **Q** führt eine “1”, wenn der Zählwert > 0 ist.
Negative Zählwerte oder Werte > 999 gibt es nicht. Falls beim Zählwert 0 eine Flanke am ZR-Eingang auftritt, geschieht nichts, Gleiches gilt, wenn bei Stand von 999 eine Flanke am ZV-Eingang auftritt.

Eingang ZW eines Zählers

Auf diesen Wert wird der Zähler gesetzt, wenn am S-Eingang eine steigende Flanke anliegt. Dieser Eingang muss nicht beschaltet werden, wenn Sie diese Funktion nicht benötigen. Der reine Abwärtszähler ist aber bei unbeschaltetem S- oder ZW-Eingang ziemlich nutzlos.
Der an diesem Eingang anliegende Wert muss BCD codiert sein. Dies ist am Einfachsten zu erreichen, wenn man ihn als C# - Konstante schreibt.
Selbstverständlich kann hier auch eine beliebige Word-Variable stehen. Der Zähler kann nur auf Werte von 0 bis 999 gesetzt werden.

Ausgang DU eines Zählers

An diesem Ausgang wird der aktuelle Zählwert dualcodiert in eine beliebige Word-Variable geschrieben. Wenn Sie diesen Wert nicht benötigen, können Sie den Ausgang auch unbeschaltet lassen. Der Wertebereich der hier ausgegebenen Zahl geht von 0 bis 999. Sie können diesen Wert nicht als ZW-Wert eines anderen Zählers verwenden, da dieser BCD-codiert sein muss. Verwenden Sie dazu den am DE-Ausgang ausgegebenen Wert.

Ausgang DE

An diesem Ausgang wird der aktuelle Zählwert BCD codiert in eine beliebige Word-Variable geschrieben. Wenn Sie diesen Wert nicht benötigen, können Sie den Ausgang auch unbeschaltet lassen. Der Wertebereich der hier ausgegebenen Zahl geht von 0 bis 999. Sie können diesen Wert als ZW-Wert eines anderen Zählers

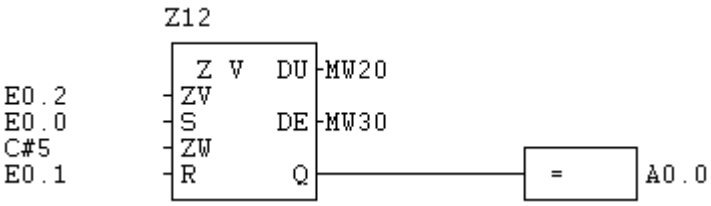
verwenden, jedoch nicht mit den +I, -I, *I und /I - Operationen weiterverarbeiten. Wenn Sie mit dem Zählwert rechnen wollen, sollten Sie den am DU Ausgang ausgegebenen dualcodierten Wert verwenden. Wenn Sie den am DE-Ausgang ausgegebenen Wert mit anderen Zahlen vergleichen wollen, müssen Sie darauf achten, dass diese ebenfalls BCD-codiert sind.

ZV

Bei einer steigenden Flanke des VKE bei der Bearbeitung dieses Befehls wird der Zählerwert um Eins erhöht. Wenn der Zählerwert bereits 999, ändert er sich nicht.

Syntax:

```
ZV      Z 38
ZV      Z[#Widx]        #Wldx ist Wort mit der Nummer des Zählers
ZV      #Count1         #Count1 ist COUNTER-In-Parameter
```

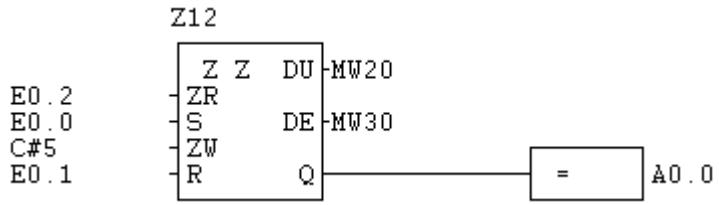


ZR

Bei einer steigenden Flanke des VKE bei der Bearbeitung dieses Befehls wird der Zählerwert um Eins erniedrigt. Wenn der Zählerwert bereits Null ist, ändert er sich nicht.

Syntax:

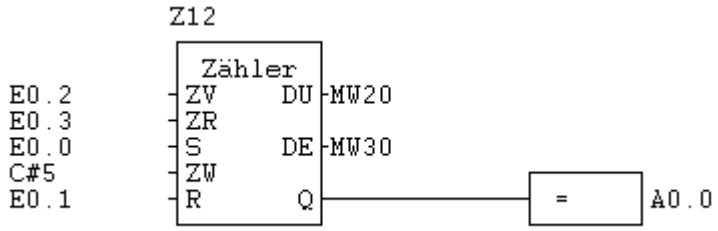
```
ZR      Z 38
ZR      Z[#Widx]        #Wldx ist Wort mit der Nummer des Zählers
ZR      #Count1         #Count1 ist COUNTER-In-Parameter
```



Setze Zähler

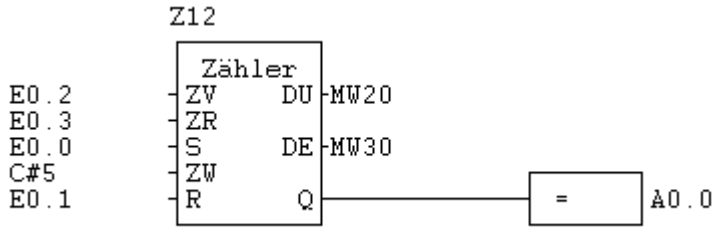
Wenn der S - Eingang eines Zählers von 0 nach 1 geht, wird der Zähler auf den am ZW-Eingang angegeben Wert gesetzt. Die Zahl am ZW-Eingang muss BCD-codiert sein, was automatisch geschieht, wenn sie als C#.. Konstante angegeben wird.

```
U  E 0.0
S  Z 12
```



Rücksetze Zähler

Der angegebene Zähler wird auf “0” gesetzt. Abfragen des Boolschen Zustandes des Zählers ergeben so lange “falsch”, bis der Zähler wieder hochgezählt wird.



FR

Die Funktionen Freigabe von Zeiten und Zählern sind in TrySim nicht implementiert.

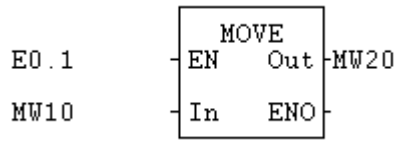
3.7.4 Lade- und Transferoperationen

L

Der angegebene Operand wird in Akku 1 geladen. Falls der Operand nicht 4 Bytes groß ist, werden die höherwertigen Bytes von Akku 1 mit “0” geladen. Vorsicht, diese Operation wird in jedem Fall ausgeführt, auch wenn eine vorhergehende Bit-Verknüpfung “0” ergeben hat.
Wie viele Bytes geladen werden, hängt von der Größe des Operanden ab.

```
L   DBB 2     // Der Inhalt von DBB2 wird nach des Akku 1LL kopiert
L   MW 5      // Der Inhalt von MW5 wird nach Akku 1L kopiert
L   AD 10     // Der Inhalt von AD10 wird nach Akku 1 kopiert
```

In FUP und KOP ist dieser Befehl nur mit einer Zielangabe verfügbar:



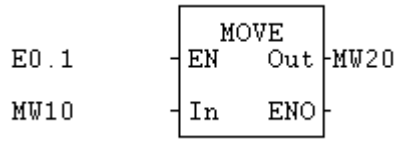
Die Bedeutung ist:
Wenn Eingang E0.1 gesetzt ist, dann lade den Inhalt von MW10 in den Akku1L und transferiere ihn dann in das MW20.

T

Transferieren
Die Zahl im Akku 1 wird in den angegeben Operanden kopiert. Vorsicht, diese Operation wird in jedem Fall ausgeführt, egal welchen Wert das VKE hat.
Wie viele Bytes kopiert werden, hängt von dem Operanden ab.

```
T   DBB 2     // Der Inhalt des Akku 1LL wird nach DBB 2 kopiert
T   MW 5      // Der Inhalt des Akku 1L wird nach MW 5 kopiert
T   AD 10     // Der Inhalt des Akku 1 wird nach AD 10 kopiert
```

In FUP und KOP ist dieser Befehl nur mit einer Quellenangabe verfügbar:



Die Bedeutung ist :
Wenn Eingang E0.1 gesetzt ist, dann lade MW10 in den Akku1L und transferiere diesen Wert anschließend in das MW20.

LC

Lade aktuellen Zählerwert oder Zeitwert als BCD in Akku 1.

LC T 5
LC Z 7

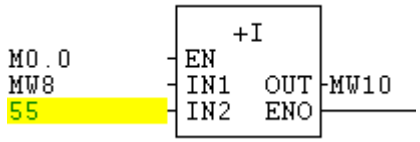
3.7.5 Operationen mit Integers

+I

Akku 1-L und Akku 2-L werden als Integers addiert und das Ergebnis in Akku 1 abgespeichert. Akku 2 bleibt unverändert. Wenn das Ergebnis größer als 32767 ist, wird das OV-Bit gesetzt.

```
L     2000
L    MW 12
+I
```

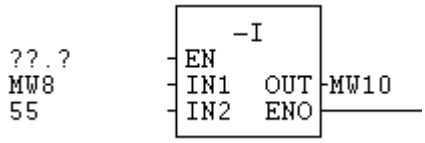
```
      T    MW 14    // in MW 14 steht jetzt 2000 + MW 12
```



-I

Akku 1 wird von Akku 2 subtrahiert und das Ergebnis im Akku 1 abgespeichert. Akku 2 bleibt unverändert. Wenn das Ergebnis kleiner als –32768 ist, wird das OV-Bit gesetzt.

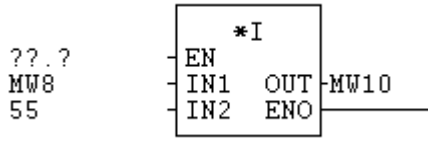
```
      L     2000
      L    MW 12
      -I
      T    MW 14    // in MW 14 steht jetzt 2000 - MW 12
```



*I

Akku 1 L und Akku 2 L werden miteinander multipliziert und das Ergebnis in Akku 1 abgespeichert. Wenn das Ergebnis größer als 32767 oder kleiner als -32768 ist, wird das OV-Bit gesetzt.

```
      L     2000
      L    MW 12
      *I
      T    MW 14    // in MW 14 steht jetzt 2000 * MW 12
```



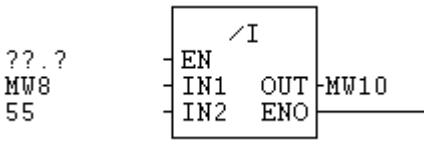
/I

Dividiere Akku 2 durch Akku 1 als Integer

Akku 2 wird durch Akku 1 dividiert und das Ergebnis in Akku 1 abgespeichert. Akku 2 bleibt unverändert. Der Rest der Division wird in Akku 1 – H gespeichert. Daher erscheint der Akku 1 im Beobachten-Modus nach einer Division meist als gigantisch

große Zahl. Solange die nachfolgenden Transfer-Operationen sich aber nur auf Words beziehen, ist das Ergebnis der Division dennoch korrekt. Der Rest der Division kann mit der Operation SRD 16 (schiebe um 16 Bits nach links) in den Word-Bereich des Akkus 1 geschoben werden.

```
      L          2000
      L          MW 12
      /I
      T          MW 14    // in MW 14 steht jetzt 2000 / MW 12
......SRD 16
      T          MW 16    // in MW 16 steht jetzt Rest der Divison
```



MOD

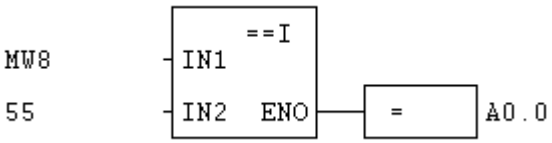
Akku 1 wird mit dem Rest der Doppelt-Integer-Divsion von Akku 2 durch Akku 1 geladen. Akku 2 bleibt unverändert.

Bei der Integer-Divsion wird diese Operation automatisch ausgeführt und das Ergebnis in den beiden höherwertigen Bytes von Akku 1 gespeichert.

==I

Die 16 niederwertigen Bits von Akku 1 und Akku 2 werden miteinander verglichen und wenn sie gleich sind, das VKE auf “1” gesetzt, andernfalls auf “0”.

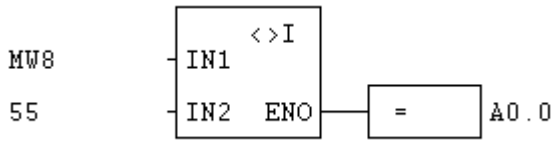
```
      L     2000
      L    MW 12
      ==I
      =   A 77.6    // wenn 2000 gleich MW 12 ist,
                          wird A77.6 auf 1 gesetzt,
                          sonst auf 0
```



<>I

Ist Integer in Akku 2 <> Akku 1 ? Wenn ja, setze VKE auf 1, sonst auf 0.

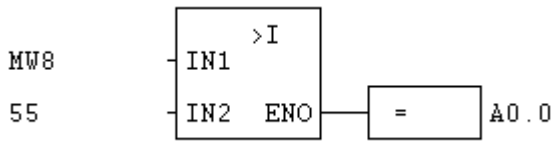
```
L     2000
L    MW 12
<>I
=    A 77.6    // wenn 2000 ungleich MW 12 ist,
                    wird A77.6 auf 1 gesetzt,
                    sonst auf 0
```



>I

Ist Integer in Akku 2 > Akku 1 ? Wenn ja, setze VKE auf 1, sonst auf 0.

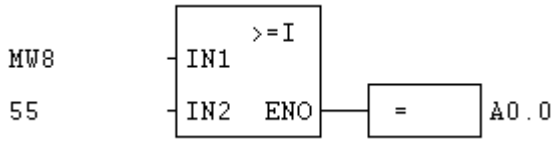
```
L     2000
L    MW 12
>I
=    A 77.6    // wenn 2000 größer als MW 12 ist,
                    wird A77.6 auf 1 gesetzt,
                    sonst auf 0
```



>=I

Ist Integer in Akku 2 >= Akku 1 ? Wenn ja, setze VKE auf 1, sonst auf 0.

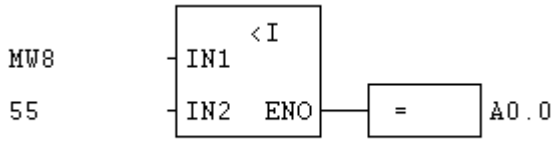
```
L     2000
L    MW 12
>=I
=    A 77.6    // wenn 2000 größer oder gleich MW 12 ist,
                    wird A77.6 auf 1 gesetzt,
                    sonst auf 0
```



<I

Ist Integer in Akku 2 < Akku 1 ? Wenn ja, setze VKE's auf 1, sonst auf 0.

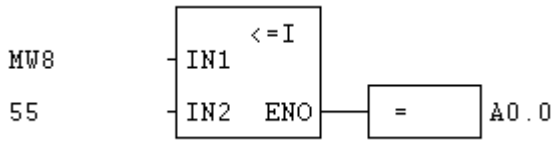
```
      L     2000
      L    MW 12
      <I
      =    A 77.6    // wenn 2000 kleiner MW 12 ist,
                        wird A77.6 auf 1 gesetzt,
                        sonst auf 0
```



<=I

Ist Integer in Akku 2 <= Akku 1 ? Wenn ja, setze VKE auf 1, sonst auf 0.

```
L     2000
      L    MW 12
      <=I
      =    A 77.6    // wenn 2000 kleiner oder gleich MW 12 ist,
                        wird A77.6 auf 1 gesetzt,
                        sonst auf 0
```



NEGI

Die Integerzahl in Akku1 wird mit -1 multipliziert.

+ (Plus)

Syntax: + 5

Zum Akku 1 wird die als Operand angegebene Konstante addiert. Die anderen Akkus und die Anzeigen werden nicht beeinflusst.

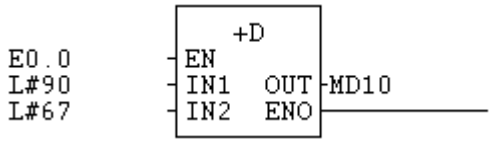
Achtung: TrySim ist beim Zulassen der erlaubten Konstanten großzügiger als STEP®7. “+ P#1.0” wird zum Beispiel akzeptiert, vom Simatic-Manager allerdings nicht. Dies kann zu Problemen beim Exportieren führen.

3.7.6 Operationen mit Double-Integers

+D

Akku 1 und Akku 2 werden addiert und das Ergebnis im Akku 1 abgespeichert. Solange die Werte in Akku 1 und Akku 2 positiv sind, ist es gleichgültig, ob sie aus einem Word oder aus einem DWord geladen wurden. Könnte jedoch einer der als Word geladenen Operanden negativ sein, wird “+D” zu einem unsinnigen Ergebnis führen. In diesem Fall muss der evtl. negative Operand vor der Addition mit der Operation ITD auf das DWord-Format erweitert werden.

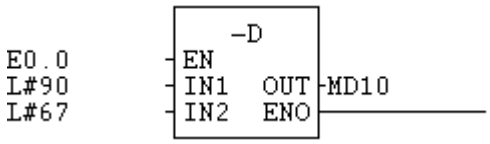
```
L  L#300000
L     MD 8
+D
T     MD 20
```



-D

Akku 1 wird von Akku 2 subtrahiert und das Ergebnis im Akku 1 abgespeichert. Solange die Werte in Akku 1 und Akku 2 positiv sind, ist es gleichgültig, ob sie aus einem Word oder aus einem DWord geladen wurden. Könnte jedoch einer der als Word geladenen Operanden negativ sein, wird “-D” zu einem unsinnigen Ergebnis führen. In diesem Fall muss der evtl. negative Operand vor der Subtraktion mit der Operation ITD auf das DWord-Format erweitert werden.

```
L  L#300000
L     MD 8
-D
T     MD 20   // in MD 20 steht jetzt 300000 - MD8
```

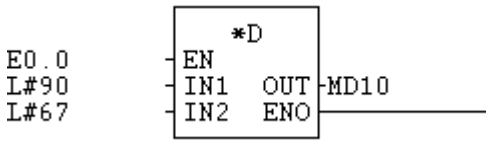


*D

Multipliziere Akku 1 mit Akku 2 als Ganzzahl (32 Bit)

Akku 1 wird mit Akku 2 multipliziert und das Ergebnis im Akku 1 abgespeichert. Solange die Werte in Akku 1 und Akku 2 positiv sind, ist es gleichgültig, ob sie aus einem Word oder aus einem DWord geladen wurden. Könnte jedoch einer der als Word geladenen Operanden negativ sein, wird "*D" zu einem unsinnigen Ergebnis führen. In diesem Fall muss der evtl. negative Operand vor der Multiplikation mit ITD auf das DWord-Format erweitert werden.

```
L  L#300000
L     MD 8
*D
T     MD 20
```

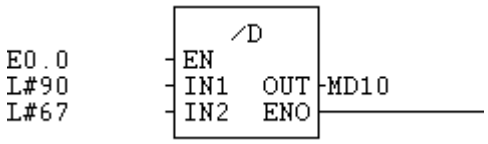


/D

Akku 2 wird durch Akku 1 dividiert und das Ergebnis im Akku 1 abgespeichert. Der Rest geht verloren. Solange die Werte in Akku 1 und Akku 2 positiv sind, ist es gleichgültig, ob sie aus einem Word oder aus einem DWord geladen wurden. Könnte jedoch einer der als Word geladenen Operanden negativ sein, wird "/D" zu einem unsinnigen Ergebnis führen. In diesem Fall muss der evtl. negative Operand vor der Division mit ITD auf das DWord-Format erweitert werden.

Der Rest der Division kann mit der Operation MOD erhalten werden.

```
L  L#300000
L     MD 8
/D
T     MD 20   // in MD 20 steht jetzt 300000 / MD8
```



MOD

Akku 1 wird mit dem Rest der Doppelt-Integer-Divsion von Akku 2 durch Akku 1 geladen. Akku 2 bleibt unverändert.

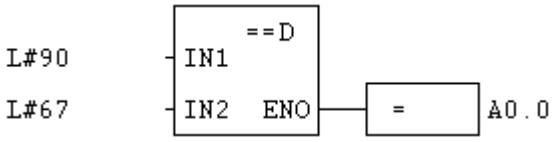
Bei der Integer-Divsion wird diese Operation automatisch ausgeführt und das Ergebnis in den beiden höherwertigen Bytes von Akku 1 gespeichert.

==D

Akku 1 und Akku 2 werden als DInts miteinander verglichen. Wenn sie gleich sind, wird das VKE auf “1” gesetzt, andernfalls auf “0”.

Solange die Werte in Akku 1 und Akku 2 positiv sind, ist es gleichgültig, ob sie aus einem Word oder aus einem DWord geladen wurden. Könnte jedoch einer der als Word geladenen Operanden negativ sein, wird “==D” zu einem unsinnigen Ergebnis führen. In diesem Fall muss der evtl. negative Operand vor der Division mit ITD auf das DWord-Format erweitert werden.

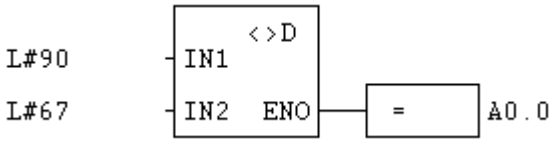
```
L  L#300000
L      MD 8
==D
=      A 0.0 // wenn 300000 gleich MD 8 ist,
                wird A 0.0 auf 1 gesetzt,
                sonst auf 0
```



<>D

Ist Doppel-Integer in Akku 2 <> Akku 1 ? Wenn ja, setze VKE auf 1, sonst auf 0.

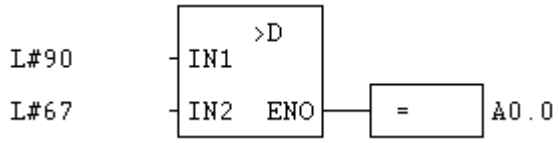
```
L  L#300000
L      MD 8
<>D
=      A 0.0   // wenn 300000 ungleich MD 8 ist,
                  wird A 0.0 auf 1 gesetzt,
                  sonst auf 0
```



>D

Ist Doppel-Integer in Akku 2 > Akku 1? Wenn ja, setze VKE auf 1, sonst auf 0.

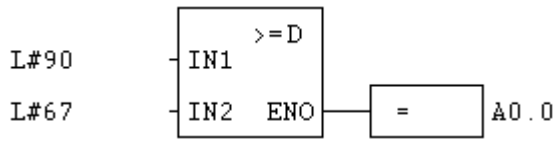
```
L  L#300000
L      MD 8
>D
=      A 0.0 // wenn 300000 größer als MD 8 ist,
                wird A 0.0 auf 1 gesetzt,
                sonst auf 0
```



>=D

Ist Doppel-Integer in Akku 2 >= Akku 1? Wenn ja, setze VKE auf 1, sonst auf 0.

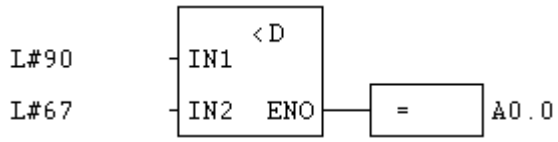
```
L  L#300000
L      MD 8
>=D
=      A 0.0   // wenn 300000 größer oder gleich MD 8 ist,
                  wird A 0.0 auf 1 gesetzt,
                  sonst auf 0
```



<D

Ist Doppel-Integer in Akku 2 < Akku 1 ? Wenn ja, setze VKE auf 1, sonst auf 0.

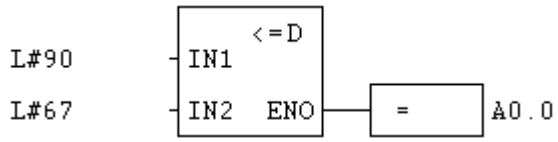
```
L  L#300000
L      MD 8
<D
=      A 0.0 // wenn 300000 kleiner als MD 8 ist,
                  wird A 0.0 auf 1 gesetzt,
                  sonst auf 0
```



<=D

Ist Doppel-Integer in Akku 2 <= Akku 1 ? Wenn ja, setze VKE auf 1, sonst auf 0.

```
L  L#300000
     L      MD 8
     <D
     =     A 0.0 // wenn 300000 kleiner oder gleich MD 8 ist,
                    wird A 0.0 auf 1 gesetzt,
                    sonst auf 0
```



NEGD

Die Doppelt-Integerzahl in Akku1 wird mit -1 multipliziert.

+ (Plus)

Syntax: + 5

Zum Akku 1 wird die als Operand angegebene Konstante addiert. Die anderen Akkus und die Anzeigen werden nicht beeinflusst.

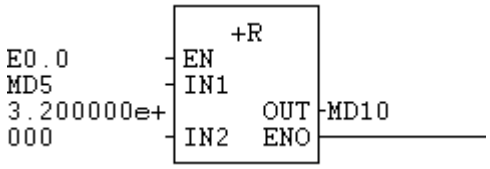
Achtung: TrySim ist beim Zulassen der erlaubten Konstanten großzügiger als STEP®7. “+ P#1.0” wird zum Beispiel akzeptiert, vom Simatic-Manager allerdings nicht. Dies kann zu Problemen beim Exportieren führen.

3.7.7 Operationen mit Real-Zahlen

+R

Die Zahlen in Akku 1 und Akku 2 werden als REAL-Zahlen miteinander multipliziert und das Ergebnis in Akku 1 abgespeichert.
Es ist wichtig, dass beide Operanden bereits im REAL-Format vorliegen. Wenn einer der Operanden als Int oder Dint geladen wurde, muss der vor der Addition mit DTR in eine REAL-Zahl gewandelt werden. Wenn ein als Int geladener Operand negativ sein könnte, muss er außerdem zuvor mit ITD auf das Dword-Format erweitert werden.

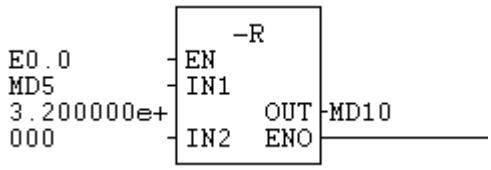
```
L       MD 40
L          2.0      // Der Punkt ist wichtig!
+R
T       MD 44       // in MD 44 steht jetzt MD40 + 2.0
```



-R

Akku 1 wird von Akku 2 abgezogen und das Ergebnis im Akku 1 abgespeichert.
Es ist wichtig, dass beide Operanden bereits im REAL-Format vorliegen. Wenn einer der Operanden als Int oder Dint geladen wurde, muss der vor der Subtraktion mit DTR in eine REAL-Zahl gewandelt werden. Wenn ein als Int geladener Operand negativ sein könnte, muss er außerdem zuvor mit ITD auf das DWord-Format erweitert werden.

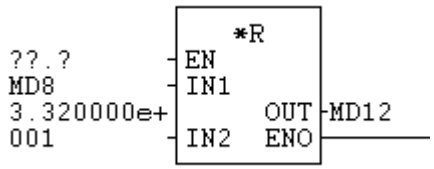
```
L       MD 40
L          2.0      // Der Punkt ist wichtig!
-R
T       MD 44       // in MD 44 steht jetzt MD40 - 2.0
```



****R***

Akku 1 und Akku 2 werden miteinander multipliziert und das Ergebnis in Akku 1 abgespeichert.
Es ist wichtig, dass beide Operanden bereits im REAL-Format vorliegen. Wenn einer der Operanden als Int oder Dint geladen wurde, muss der vor der Multiplikation mit DTR in eine REAL-Zahl gewandelt werden. Wenn ein als Int geladener Operand negativ sein könnte, muss er außerdem zuvor mit ITD auf das DWord-Format erweitert werden.

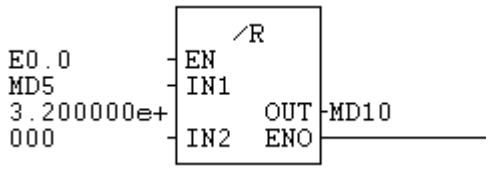
```
L       MD 40
L         2.0     // Der Punkt ist wichtig!
*R
T       MD 44       // in MD 44 steht jetzt MD40 * 2
```



/R

Akku 2 wird durch Akku 1 dividiert und das Ergebnis in Akku 1 abgespeichert. Akku 2 bleibt unverändert.
Es ist wichtig, dass beide Operanden bereits im REAL-Format vorliegen. Wenn einer der Operanden als Int oder Dint geladen wurde, muss der vor der Division mit DTR in eine REAL-Zahl gewandelt werden. Wenn ein als Int geladener Operand negativ sein könnte, muss er außerdem zuvor mit ITD auf das DWord-Format erweitert werden.

```
L      MD 40
L         2.0     // Der Punkt ist wichtig!
/R
T      MD 44      // in MD 44 steht jetzt MD40 / 2
```

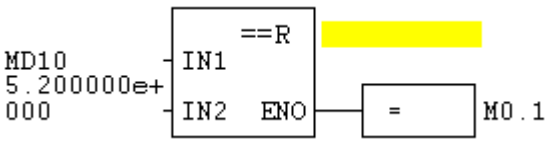


==R

Akku 1 und Akku 2 werden miteinander verglichen und das VKE auf “1” gesetzt, wenn sie gleich sind, andernfalls auf “0”.
Es ist wichtig, dass beide Operanden bereits im REAL-Format vorliegen. Wenn einer der Operanden als Int oder Dint geladen wurde, muss der vor dem Vergleich mit DTR in eine REAL-Zahl gewandelt werden. Wenn ein als Int geladener Operand negativ sein könnte, muss er außerdem zuvor mit ITD auf das DWord-Format erweitert werden.

Der Vergleich von REAL-Zahlen auf “gleich” führt aufgrund von Rundungsfehlern häufig zu Fehlfunktionen des Programms. Statt “==R” sollten Sie lieber die zu vergleichenden Zahlen voneinander subtrahieren und den Absolutwert davon mit “<R” auf eine obere Grenze vergleichen.

```
L      MD 40
L         2.0     // Der Punkt ist wichtig!
==R
A      A 22.4     // Wenn MD40 gleich 2.0 ist,
                     wird A22.4 auf 1 gesetzt,
                     sonst auf 0
```

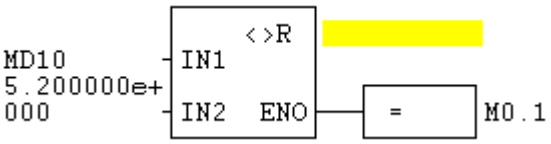


<>R

Akku 1 und Akku 2 werden miteinander verglichen und das VKE auf “1” gesetzt, wenn sie ungleich sind, andernfalls auf “0”.
Es ist wichtig, dass beide Operanden bereits im REAL-Format vorliegen. Wenn einer der Operanden als Int oder Dint geladen wurde, muss der vor dem Vergleich mit DTR in eine REAL-Zahl gewandelt werden. Wenn ein als Int geladener Operand negativ sein könnte, muss er außerdem zuvor mit ITD auf das DWord-Format erweitert werden.

Der Vergleich von REAL-Zahlen auf “ungleich” führt aufgrund von Rundungsfehlern häufig zu unerwartetem Verhalten des Programms. Statt “<>R” sollten Sie lieber die zu vergleichenden Zahlen voneinander subtrahieren und den Absolutwert davon mit “>R” auf eine untere Grenze vergleichen.

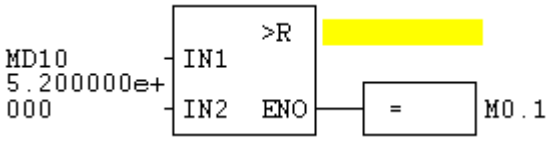
```
L       MD 40
L          2.0     // Der Punkt ist wichtig!
<>R
A       A 22.4     // Wenn MD40 ungleich 2.0 ist,
                      wird A22.4 auf 1 gesetzt,
                      sonst auf 0
```



>R

Akku 1 und Akku2 werden miteinander verglichen und das VKE auf “1” gesetzt, wenn Akku 2 > Akku 1 ist, andernfalls auf “0”.
Es ist wichtig, dass beide Operanden bereits im REAL-Format vorliegen. Wenn einer der Operanden als Int oder Dint geladen wurde, muss der vor dem Vergleich mit DTR in eine REAL-Zahl gewandelt werden. Wenn ein als Int geladener Operand negativ sein könnte, muss er außerdem zuvor mit ITD auf das DWord-Format erweitert werden.

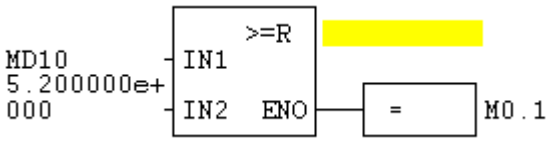
```
L       MD 40
L          2.0     // Der Punkt ist wichtig!
>R
A       A 22.4     // Wenn MD40 größer als 2.0 ist,
                      wird A22.4 auf 1 gesetzt
                      sonst auf 0
```



>=R

Akku 1 und Akku2 werden miteinander verglichen und das VKE auf “1” gesetzt, wenn Akku 2 >= Akku 1 ist, andernfalls auf “0”.
Es ist wichtig, dass beide Operanden bereits im REAL-Format vorliegen. Wenn einer der Operanden als Int oder Dint geladen wurde, muss der vor dem Vergleich mit DTR in eine REAL-Zahl gewandelt werden. Wenn ein als Int geladener Operand negativ sein könnte, muss er außerdem zuvor mit ITD auf das DWord-Format erweitert werden.

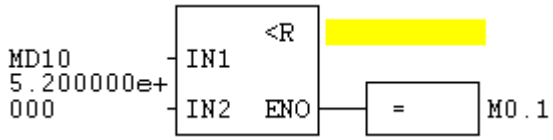
```
L      MD 40
L        2.0   // Der Punkt ist wichtig!
>=R
A     A 22.4  // Wenn MD40 größer oder gleich 2.0 ist,
                 wird A22.4 auf 1 gesetzt,
                 sonst auf 0
```



<R

Akku 1 und Akku2 werden miteinander verglichen und das VKE auf “1” gesetzt, wenn Akku 2 < Akku 1 ist, andernfalls auf “0”.
Es ist wichtig, dass beide Operanden bereits im REAL-Format vorliegen. Wenn einer der Operanden als Int oder Dint geladen wurde, muss der vor dem Vergleich mit DTR in eine REAL-Zahl gewandelt werden. Wenn ein als Int geladener Operand negativ sein könnte, muss er außerdem zuvor mit ITD auf das DWord-Format erweitert werden.

```
L      MD 40
L        2.0   // Der Punkt ist wichtig!
<R
A     A 22.4  // Wenn MD40 kleiner als 2.0 ist,
                 wird A22.4 auf 1 gesetzt,
                 sonst auf 0
```



<=R

Akku 1 und Akku2 werden miteinander verglichen und das VKE auf “1” gesetzt, wenn Akku 2 <= Akku 1 ist, andernfalls auf “0”.
Es ist wichtig, dass beide Operanden bereits im REAL-Format vorliegen. Wenn einer der Operanden als Int oder Dint geladen wurde, muss der vor dem Vergleich mit DTR in eine REAL-Zahl gewandelt werden. Wenn ein als Int geladener Operand negativ sein könnte, muss er außerdem zuvor mit ITD auf das DWord-Format erweitert werden.

```
L      MD 40
L        2.0   // Der Punkt ist wichtig!
<=R
A     A 22.4  // Wenn MD40 kleiner oder gleich 2.0 ist,
                 wird A22.4 auf 1 gesetzt,
                 sonst auf 0
```

ABS

Wenn Akku 1 positiv ist, bewirkt die Operation nichts, andernfalls wird der Wert von Akku 1 negiert und im Akku 1 abgespeichert. Nach ABS ist Akku 1 in jedem Fall positiv (oder “0”).

ABS können Sie nur für im REAL-Format vorliegende Zahlen verwenden. Wenn Sie den Absolutwert von Int- (Dint-) Zahlen benötigen, programmieren Sie

```
        L      0
        L      Zahl
        <I (D)
        SPB Ok
        NEGI (NEGD)
Ok      : .....                .
```

NEGR

Die REAL-Zahl in Akku 1 wird mit -1 multipliziert.

SQRT

Vom Akku 1 wird die Quadratwurzel berechnet und wieder im Akku 1 abgespeichert.

Es ist wichtig, dass der Operand im Akku 1 bereits im REAL-Format vorliegt. Wenn er als Int oder Dint geladen wurde, muss er vor der Berechnung mit DTR in eine REAL-Zahl gewandelt werden.

SQR

Das Quadrat der Zahl im Akku 1 wird berechnet und im Akku 1 gespeichert.

Es ist wichtig, dass der Operand im Akku 1 bereits im REAL-Format vorliegt. Wenn er als Int oder Dint geladen wurde, muss er vor der Berechnung mit DTR in eine REAL-Zahl gewandelt werden. Wenn ein als Int geladener Operand negativ sein könnte, muss er außerdem zuvor mit ITD auf das DWord-Format erweitert werden.

LN

Logarithmus zur Basis e vom Akku 1.

In Akku 1 muss eine Realzahl stehen.

EXP

e hoch Akku 1

Akku 1 muss eine Real-Zahl sein.

3.7.8 Sprung-Operationen

Sprungmarken

Sprungmarken werden in AWL-Netzwerken verwendet, um Zeilen zu markieren, zu denen gesprungen werden soll. Sprungmarken bestehen aus max. 4 Buchstaben und Zahlen und dürfen keine Sonderzeichen außer '_' enthalten. Die Groß- und Kleinschreibung ist signifikant. Sprungmarken werden mit einem ':' abgeschlossen, danach muss ein Befehl kommen, mindestens NOP 0.

Innerhalb eines **Bausteins** darf eine Sprungmarke nur einmal vorkommen.

Absoluter Sprung:
SPA

Bedingte Sprünge:
SPBN SPMZ SPN
SPP SPPZ SPZ SPBNB
SPB SPBB SPBI SPBIN

Schleifen und Sprungverteiler(“case”):
LOOP SPL

Nicht implementierte Sprünge:
SPO SPU SPS

SPA

Springe **a**bsolut

```
SPA   lbl      // lbl ist eine Sprungmarke irgendwo im Baustein
```

Die Programmbearbeitung wird an der angegebene Stelle fortgesetzt.

SPBN

Springe **b**edingt (wenn VKE **n**icht = 1)

```
SPBN lbl      // lbl ist eine Sprungmarke irgendwo im Baustein
```

Wenn das VKE = 0 ist, wird zur angegebenen Stelle gesprungen, sonst wird die Bearbeitung normal fortgesetzt.

SPMZ

```
SPBMZ      lbl      // lbl ist eine Sprungmarke irgendwo im Baustein
```

Wenn das Ergebnis einer vorhergehenden **Int- oder DInt - Rechenoperation** <= 0 ist, wird zu der angegebenen Sprungmarke gesprungen.

Dieser Sprung funktioniert nicht nach Real-Operationen! Siehe aber ABS.

SPM

```
SPBM lbl      // lbl ist eine Sprungmarke irgendwo im Baustein
```

Wenn das Ergebnis einer vorhergehenden **Int- oder DInt - Rechenoperation** < 0 ist, wird zu der angegebenen Sprungmarke gesprungen.

Dieser Sprung funktioniert nicht nach Real-Operationen! Siehe aber ABS.

SPN

```
SPN     lbl     // lbl ist eine Sprungmarke irgendwo im Baustein
```

Wenn das Ergebnis einer vorhergehenden **Int- oder DInt - Rechenoperation** <> 0 ist, wird zu der angegebenen Sprungmarke gesprungen.

Dieser Sprung funktioniert nicht nach Real-Operationen! Siehe aber ABS.

SPP

```
SPP   lbl     // lbl ist eine Sprungmarke irgendwo im Baustein
```

Wenn das Ergebnis einer vorhergehenden **Int- oder DInt - Rechenoperation** > 0 ist, wird zu der angegebenen Sprungmarke gesprungen.

Dieser Sprung funktioniert nicht nach Real-Operationen! Siehe aber ABS.

SPPZ

```
SPPZ lbl     // lbl ist eine Sprungmarke irgendwo im Baustein
```

Wenn das Ergebnis einer vorhergehenden **Int- oder DInt - Rechenoperation** >= 0 ist, wird zu der angegebenen Sprungmarke gesprungen.

Dieser Sprung funktioniert nicht nach Real-Operationen! Siehe aber ABS.

SPZ

```
SPZ   lbl     // lbl ist eine Sprungmarke irgendwo im Baustein
```

Wenn das Ergebnis einer vorhergehenden **Int- oder DInt - Rechenoperation** = 0 ist, wird zu der angegebenen Sprungmarke gesprungen.

Dieser Sprung funktioniert nicht nach Real-Operationen! Siehe aber ABS.

SPBNB

```
SPBNB     lbl     // lbl ist eine Sprungmarke irgendwo im Baustein
```

Wenn das VKE = 0 ist, wird zu der angegebene Stelle gesprungen, sonst wird die Bearbeitung normal fortgesetzt.
Außerdem wird das BIE-Bit auf den Wert des VKE gesetzt.

SPB

Springe **b**edingt

```
SPB   lbl      // lbl ist eine Sprungmarke irgendwo im Baustein
```

Wenn das VKE = 1 ist, wird zu der angegebene Stelle gesprungen, sonst wird die Bearbeitung normal fortgesetzt.

SPBB

```
SPBB lbl      // lbl ist eine Sprungmarke irgendwo im Baustein
```

Wenn das VKE = 1 ist, wird zu der angegebene Stelle gesprungen, sonst wird die Bearbeitung normal fortgesetzt.
Außerdem wird das BIE-Bit auf den Wert des VKE gesetzt.

SPBI

```
SPBI   lbl      // lbl ist eine Sprungmarke irgendwo im Baustein
```

Wenn das BIE-Bit "1" ist, wird zu der angegeben Stelle gesprungen.

SPBIN

```
SPBIN   lbl     // lbl ist eine Sprungmarke irgendwo im Baustein
```

Wenn das BIE-Bit "0" ist, wird zu der angegeben Stelle gesprungen.

Loop

Die Zahl im Akku1 wird um 1 erniedrigt. Wenn Akku1 danach ungleich Null ist, wird zu der angegeben Sprungmarke gesprungen.

Diese Schleife wird 10 mal durchlaufen:

```
      L 10
lbl : T MW 20
      ....
      ....
      L MW 20
      LOOP        lbl          // lbl ist Sprungmarke
```

SPL

Sprung **L**iste

```
Syntax: SPL   err     // err ist Sprungmarke
        SPA   M1      // M1 ist Sprungmarke
        SPA   M2      // M2 ist Sprungmarke
        ..
        SPA   Mx      // Mx ist Sprungmarke
        err:          .....
```

Die im Akku 1 stehende Anzahl von Programmzeilen (die mit SPA beginnen müssen) wird übersprungen. Falls die Liste der SPA-Zeilen nicht lang genug ist, wird stattdessen zur Marke err gesprungen.

3.7.9 Nicht-implementierte Sprünge

SPO

Nicht implementiert.

SPU

Nicht implementiert.

SPS

Nicht implementiert.

3.7.10 Schiebe- und Rotier-Operationen

SRD

Die 32 - Bits des Akku 1 werden um die als Operand (0-32) angegebene Zahl nach rechts verschoben. Nach rechts heraus geschobene Bits gehen verloren, von links wird mit Nullen nachgefüllt.

SSD

Die 32-Bits des Akku 1 werden um die als Operand (0-32) angegeben Zahl nach rechts geschoben. Von links wird mit dem Wert des Bits 31 nachgefüllt. Mithilfe dieser Operation kann man auch negative Zahlen durch Rechtsverschiebung durch Potenzen von 2 teilen.

Ist kein Operand angegeben, wird stattdessen der Akku2-LL verwendet.

RLD

Die 32 Bits im Akku 1 werden um die als Operand angegebene Zahl (0-32) nach links rotiert, d.h., die Bits, die bei einer Verschiebung nach links aus dem Akku geschoben werden, werden auf der anderen Seite wieder eingespeist.

Ist kein Operand angegeben, wird stattdessen der Akku2-LL verwendet.

RLDA

Die 32 Bits im Akku 1 werden um ein Bit nach links rotiert, d.h., das Bit, das bei einer Verschiebung nach links aus dem Akku geschoben wird, wird auf der anderen Seite wieder eingespeist. Allerdings wird bei dieser Operation das links herausgeschobene Bit nicht sofort wieder eingespeist, sondern erst im Übertrags-Bit zwischengespeichert.

SLD

Die 32 - Bits des Akku 1 werden um die als Operand (0-32) angegebene Zahl nach links verschoben. Bits, die nach links hinausgeschoben werden, gehen verloren. Von rechts werden Nullen nachgeschoben.

Ist kein Operand angegeben, wird stattdessen der Akku2-LL verwendet.

Syntax:

```
SLD x     // x ist zwischen 0 und 31
SLD       // der Inhalt von Akku2-LL wird als Parameter verwendet
```

SLW

Die 16 - Bits des Akku 1 - L werden um die als Operand (0-15) angegebene Zahl nach links verschoben. Die nach links über die Akku 1 - L - Grenze geschobenen Bits gehen verloren, sie werden nicht in die höherwertigen Bytes des Akkus übertragen. Von rechts her werden Nullen eingeschoben.
Beispiel:

```
L 1       // 1 im Akku 1
SLW 2  // Schiebe 2 Bits nach links
```

Im Akku 1 steht jetzt W#16#0004 = 2#100

Wenn kein Operand angegeben ist, wird stattdessen die Zahl im Akku 2 LL verwendet.
Beispiel:

```
L 4       // 5 im Akku 1
L 1       // 5 im Akku 2 / 1 im Akku 1
SLW     // Schiebe 1 um 5 Bits nach links
```

Im Akku1 steht jetzt W#16#0010 = 2#1000

RRDA

Die 32 Bits im Akku 1 werden um ein Bit nach rechts rotiert, d.h., das Bit, das bei einer Verschiebung nach rechts aus dem Akku geschoben wird, wird auf der anderen Seite wieder eingespeist. Allerdings wird bei dieser Operation ein rechts herausgeschobenes Bit nicht sofort wieder eingespeist, sondern erst im Übertrags-Bit zwischengespeichert.

RRD

Die 32 Bits im Akku 1 werden um die als Operand angegebene Zahl (0-32) nach rechts rotiert, d.h., die Bits, die bei einer Verschiebung nach rechts aus dem Akku geschoben werden, werden auf der anderen Seite wieder eingespeist.

Ist kein Operand angegeben, wird stattdessen der Akku2-LL verwendet.

SSI

Die 16-Bits des Akku 1 - L werden um die als Operand (0-15) angegebene Zahl nach rechts geschoben. Von links wird mit dem Wert des Bit 15 nachgefüllt. Mithilfe dieser Operation kann man auch negative Zahlen durch Rechtsverschiebung durch Potenzen von 2 teilen.

Ist kein Operand angegeben, wird stattdessen der Akku2-LL verwendet.

SRW

Die 16 - Bits des Akku 1 - L werden um die als Operand (0-15) angegebene Zahl nach rechts verschoben. Nach rechts herausgeschobene Bits gehen verloren, von links wird mit Nullen nachgefüllt.

Ist kein Operand angegeben, wird stattdessen der Akku2-LL verwendet.

3.7.11 Logische Wort- und DWort-Operationen

UD

Die 2 x 32 Bits der Akkus 1 und 2 werden bitweise UND-verknüpft und das Ergebnis im Akku 1 abgespeichert.

UW

Und **W**ort

Akku 1-L und Akku 2-L werden bitweise mit der Operation "Und" verknüpft und das Ergebnis im Akku 1 abgespeichert. Die beiden höherwertigen Bytes des Akkus werden auf Null gesetzt. Akku 2 bleibt unverändert.

OW

Die 16 niederwertigen Bits des Akkus 1 werden einzeln nach folgender Regel gesetzt:
Wenn eines der Bits im Akku 1 oder Akku 2 “1” ist, wird das Bit auf “1” gesetzt, sonst auf “0”.

OD

Die 32 - Bits des Akkus 1 werden einzeln nach folgender Regel gesetzt:
Wenn eines der Bits im Akku 1 oder Akku 2 “1” ist, wird das Bit auf “1” gesetzt, sonst auf “0”.

XOD

EXLUSIV ODER Doppelwort (32 Bit)

Akku1 und Akku 2 werden bitweise mit der Operation “exklusiv Oder” verknüpft und das Ergebnis im Akku 1 abgespeichert. Akku 2 bleibt unverändert.

XOW

EXLUSIV ODER Wort

Akku 1-L und Akku 2-L werden bitweise mit der Operation "exklusiv Oder" verknüpft und das Ergebnis im Akku 1 abgespeichert. Die beiden höherwertigen Bytes des Akkus1 werden auf Null gesetzt. Akku 2 bleibt unverändert.

INVI

Die 16 niederwertigen -Bits des Akku 1 werden umgedreht, d.h. aus 1 wird 0 und aus 0 wird 1.

INVD

Alle 32-Bits des Akku 1 werden umgedreht, d.h. aus 1 wird 0 und aus 0 wird 1.

3.7.12 BCD-Operationen

BCD-Zahl

In den frühen Zeiten der Computertechnik, als das HMI (**h**uman **m**achine **i**nterface) noch aus Lämpchen und Handradschaltern bestand, bemerkte man, dass Menschen die Binärzahlen der Maschinen nur schlecht lesen können und dass es umgekehrt

schwierig ist, mehrstelligen Handradschaltern ein binäres Signal zu entlocken. Daher erfand man die **b**inär **c**odierte **d**ezimale Zahl, eine Chimäre aus beiden Darstellungen. Man nimmt einfach jede einzelne Ziffer einer im Dezimalsystem dargestellten Zahl und übersetzt sie in ein Nibble (Halbbyte) in binärer Darstellung. Etwa so:

```
159     ->     0001   0101   1001
                 1      5      9
```

Mit etwas Übung können Menschen durchaus die Binärdarstellung der Zahlen von 0 bis 9 lernen und auch Handradschalter können mit viel Aufwand so konstruiert werden, dass sie diesen kleinen Zahlenbereich binär codieren. Leider ist dieses unsägliche Datenformat, mit dem es sich schlecht rechnen lässt, bis auf den heutigen Tag am Leben gehalten worden. Der einzige Trost ist, dass die Speicherplatzverschwendung, die damit verbunden ist, inzwischen keine Rolle mehr spielt.

Die einfachste Art, eine BCD - Konstante im Bereich von 0 - 999 zu notieren, besteht in der Verwendung des Counter-Formats C#xyz, z.B. C#642.

Eine BCD-Zahl wird als negativ interpretiert, wenn alle vier Bits des höchstwertigen Nibbles gleich 1 sind. So notiert man z.B. -53(bcd) als W#16#F053.

Siebenstellige BCD-Zahlen können unter Verwendung des DW#16#- Formates notiert werden.

Zur Umwandlung zwischen BCD- und dual-Format gibt es folgende Funktionen:
BTI 3 stellige BCD - Zahl mit Vorzeichen -> Integer
BTD 7 stellige BCD - Zahl mit Vorzeichen -> Double Integer
ITB dualcodierte Zahl von -999 bis +999 -> BCD
DTB dualcodierte Zahl von -9.999.999 bis +9.999.999 -> BCD

ITB

Die Integer-Zahl in Akku 1-L wird in eine 3 - stellige - BCD-Zahl gewandelt.
Bei einer negativen Zahl werden die vier höchstwertigen Bits des Ergebnisses auf “1” gesetzt.

Die Zahl muss zwischen -999 und +999 liegen. Wenn sie außerhalb liegt, findet keine Umwandlung statt. (In einer realen S7 werden außerdem die Flags OV und OS gesetzt.)

DTB

Doppelte Integerzahl in 7-stellige BCD-Zahl mit Vorzeichen wandeln.
Bei einer negativen Zahl werden die 4 höchstwertigen Bits des Ergebnisses alle auf “1” gesetzt.

Die Zahl muss zwischen -9 999 999 und + 9 999 999 liegen. Wenn sie außerhalb

liegt, findet keine Umwandlung statt. (In einer realen S7 werden außerdem die Flags OV und OS gesetzt.)

BTD

BCD im Akku 1 in doppelten Integer wandeln

Der als 7-stellige BCD-Zahl interpretierte Akku 1 wird in eine 4-Bytes-Binärzahl verwandelt und das Ergebnis im Akku 1 abgespeichert.

Das höchstwertigste Bit wird als Vorzeichen interpretiert. Ist es “1”, ist das Ergebnis negativ.

Bei einer ungültigen BCD-Zahl im Akku wird ein Laufzeitfehler ausgelöst. In einer realen S7 können Sie diesen mit dem OB121 abfangen.

Bis zur Version 2.1 hat diese Funktion den Akku1 fälschlicherweise als 8-stellige, immer positive BCD-Zahl interpretiert. Programme, die von dieser Eigenschaft Gebrauch gemacht haben, werden daher jetzt fehlerhaft sein.

BTI

BCD im Akku 1 in Integer wandeln.

Der als 3-stellige BCD-Zahl interpretierte Akku 1 wird in eine 2-Bytes-Binärzahl verwandelt und das Ergebnis im Akku 1-L abgespeichert.

Das höchstwertigste Bit wird als Vorzeichen interpretiert. Ist es “1”, ist das Ergebnis negativ.

Bei einer ungültigen BCD-Zahl im Akku wird ein Laufzeitfehler ausgelöst. In einer realen S7 können Sie diesen mit dem OB121 abfangen.

Bis zur Version 2.1 hat diese Funktion den Akku1 fälschlicherweise als 4-stellige, immer positive BCD-Zahl interpretiert. Programme, die von dieser Eigenschaft Gebrauch gemacht haben, werden daher jetzt fehlerhaft sein. Abhilfe schafft die Verwendung von BTD.

3.7.13 Andere Umwandlungen

DTR

Doppelte Integerzahl in REAL wandeln. Wenn die Zahl als Integer geladen worden ist und negativ sein könnte, muss sie zuvor mit ITD auf das DINT-Format erweitert werden.

Die Umkehrfunktion heißt RND.

ITD

Die Integerzahl in Akku 1 wird zu einer Doppel-Integerzahl erweitert. Für positive Zahlen passiert dabei überhaupt nichts, für negative werden die beiden höherwertigen Bytes des Akkus mit 1'en gefüllt.

TRUNC

Die Realzahl im Akku 1 wird in ein DInt gewandelt. Nachkommastellen werden abgeschnitten.

RND

Die REAL-Zahl im Akku 1 wird durch Runden in eine Ganzzahl verwandelt.

RND+

Die REAL-Zahl im Akku 1 wird durch Runden in eine Ganzzahl verwandelt, dabei wird immer aufgerundet.

RND-

Die REAL-Zahl im Akku 1 wird durch Runden in eine Ganzzahl verwandelt, dabei wird immer abgerundet.

3.7.14 Trigonometrische Operationen

Bogenmaß

Die trigonometrischen Operationen von TrySim erwarten Winkel im Bogenmaß bzw. liefern sie in diesem Format.

```
90°   = Pi / 2    = 1.570796
180°  = Pi        = 3.1415926
```

Wenn Sie einen Winkel in Grad haben und ihn im Bogenmaß benötigen, programmieren Sie:

```
L     ZahlGrd     // in Grad
L     180.0       // nicht 180 ! sondern 180.0
/R
L     3.1415926   // Pi
*R                // jetzt haben Sie die Zahl im Bogenmaß
```

Wenn Sie einen Winkel im Bogenmaß haben und ihn in Grad ausdrücken wollen, programmieren Sie:

```
L     ZahlBog     // im Bogenmaß im REAL-Format
L     3.1415926   // Pi
```

```
/R
L       180.0       // nicht 180 ! sondern 180.0
*R                  // jetzt haben Sie die Zahl in Grad
```

SIN

Der SIN des Akkus 1 wird berechnet und das Ergebnis im Akku 1 abgespeichert. Der Winkel im Akku 1 muss als REAL im Bogenmaß vorliegen.

COS

Der COS des Akkus 1 wird berechnet und das Ergebnis im Akku 1 abgespeichert. Der Winkel im Akku 1 muss als REAL im Bogenmaß vorliegen.

TAN

Der TAN des Akkus 1 wird berechnet und das Ergebnis im Akku 1 abgespeichert. Der Winkel im Akku 1 muss als REAL im Bogenmaß vorliegen.

ASIN

Der ASIN des Akkus 1 wird berechnet und im Akku 1 im Bogenmaß abgespeichert.

Es ist wichtig, dass der Operand im Akku 1 bereits im REAL-Format vorliegen. Wenn er als Int oder DInt geladen wurde, muss der vor der Berechnung mit DTR in eine REAL-Zahl gewandelt werden. Wenn ein als Int geladener Operand negativ sein könnte, muss er außerdem zuvor mit ITD auf das DWord-Format erweitert werden.

```
ASIN(-1)      = - Pi / 2 ( Pi = 3,141592...)
ASIN(0)       = 0
ASIN(1)       = Pi / 2 ( Pi = 3,141592...)
```

ACOS

Der ACOS des Akkus 1 wird berechnet und im Akku 1 im Bogenmaß abgespeichert.

Es ist wichtig, dass der Operand im Akku 1 bereits im REAL-Format vorliegen. Wenn er als Int oder DInt geladen wurde, muss der vor der Berechnung mit DTR in eine REAL-Zahl gewandelt werden. Wenn ein als Int geladener Operand negativ sein könnte, muss er außerdem zuvor mit ITD auf das DWord-Format erweitert werden.

```
ACOS(1)       = 0
ACOS(0)       = Pi / 2 ( Pi = 3,141592...)
ACOS(-1)      = Pi  ( Pi = 3,141592...)
```

ATAN

Der ATAN des Akkus 1 wird berechnet und im Bogenmaß in Akku 1 abgespeichert.

Es ist wichtig, dass der Operand im Akku 1 bereits im REAL-Format vorliegen. Wenn er als Int oder DInt geladen wurde, muss der vor der Berechnung mit DTR in eine REAL-Zahl gewandelt werden. Wenn ein als Int geladener Operand negativ sein könnte, muss er außerdem zuvor mit ITD auf das DWord-Format erweitert werden.

ATAN(-unendlich) = - Pi / 2 (Pi = 3,141592...)
ATAN(0) = 0
ATAN(unendlich) = Pi / 2 (Pi = 3,141592...)

3.7.15 Sonstige Operationen mit Akku 1

TAK

Die Inhalte von Akku 1 und Akku 2 werden miteinander vertauscht.

TAW

Swap!

Tausche die Bytes im Akku 1 - L:
aus LL, LH wird LH, LL

TAD

Die Bytes im Akku 1 werden vertauscht:
aus HH,HL,LH,LL wird LL,LH,HL,HH

INC

Addiere das Argument zum Akku 1

Syntax INC *Zahl*

Die *Zahl* muss zwischen 0 und 255 liegen, sie wird zum Akku 1-LL addiert und das Ergebnis wieder im Akku 1 - LL gespeichert. Die anderen Bytes des Akkus bleiben unberührt.

INC ist eine schöne Abkürzung für:
L *Zahl*
+I

vor allem, weil Akku 2 unberührt bleibt. Man muss nur unbedingt aufpassen, dass das Ergebnis nicht größer als 255 wird, denn es findet keine Übertragung in die höheren Bytes statt.

Ebenso praktisch ist + (Plus).

DEC

Decrementiere Akku 1.

Syntax DEC *Zahl*

Die *Zahl* muss zwischen 0 und 255 liegen, sie wird vom Akku 1-LL abgezogen und das Ergebnis wieder im Akku 1 - LL gespeichert. Die anderen Bytes des Akkus bleiben unberührt.

DEC ist eine schöne Abkürzung für:

```
L    Zahl
-I
```

vor allem, weil Akku 2 unberührt bleibt. Man muss nur unbedingt aufpassen, dass das Ergebnis nicht kleiner als 0 wird, denn sonst gibt es eine unangenehme Überraschung (das Programm rechnet, bei einem Ergebnis von "-5" z.B., "256-5").

Ebenso praktisch ist + (Plus).

+ (Plus)

Syntax: + 5

Zum Akku 1 wird die als Operand angegebene Konstante addiert. Die anderen Akkus und die Anzeigen werden nicht beeinflusst.

Achtung: TrySim ist beim Zulassen der erlaubten Konstanten großzügiger als STEP®7. “+ P#1.0” wird zum Beispiel akzeptiert, vom Simatic-Manager allerdings nicht. Dies kann zu Problemen beim Exportieren führen.

3.7.16 Operationen mit Akku 3 und 4

PUSH

Die Akkus 1, 2 und 3 werden gemeinsam um einen Platz nach oben verschoben. Die Zahl im Akku 1 steht danach im Akku 1 und im Akku 2, die Zahl im Akku 4 geht verloren.

```
      -->
Z4          Z3
Z3          Z2
Z2          Z1
```

Z1 Z1

Dieser Befehl ist nur in SPS der S7-400 Reihe verfügbar.

S7-300 und S7-400 sind eingetragene Warenzeichen der Siemens AG.

POP

Die Akkus 2, 3 und 4 werden gemeinsam um einen Platz nach unten verschoben. Die Zahl in Akku 4 steht danach in Akku 4 und in Akku 3, die Zahl in Akku 1 geht verloren.

-->

Z4 Z4
Z3 Z4
Z2 Z3
Z1 Z2

Dieser Befehl ist nur in SPS der S7-400 Reihe verfügbar.

S7-300 und S7-400 sind eingetragene Warenzeichen der Siemens AG.

LEAVE

Akku 4 und Akku 3 werden gemeinsam um einen Platz nach unten verschoben. Die Zahl im Akku 4 steht danach sowohl in Akku 4 als auch in Akku 3. Die Zahl in Akku 2 geht verloren.

-->

Z4 Z4
Z3 Z4
Z2 Z3
Z1 Z1

Dieser Befehl ist nur in SPS der S7-400 Reihe verfügbar.

S7-300 und S7-400 sind eingetragene Warenzeichen der Siemens AG.

ENT

(Enter) Akku 2 und Akku 3 werden gemeinsam um einen Platz nach oben verschoben. Der Inhalt von Akku 4 geht verloren, Akku 2 bleibt unverändert.

-->

Z4 Z3
Z3 Z2
Z2 Z2
Z1 Z1

Dieser Befehl ist nur in SPS der S7-400 Reihe verfügbar.

S7-300 und S7-400 sind eingetragene Warenzeichen der Siemens AG.

3.7.17 Register-indirekte Adressierung

Warnung bei Verwendung von AR1 oder AR2

Leider müssen wir bei der Verwendung des AR1 und AR2 zu großer Vorsicht aufrufen. Diese Register werden von den Siemens - S7 unter bestimmten Umständen eigenständig modifiziert oder dürfen wegen der Verwendung für S7-interne Zwecke vom Anwender nicht verändert werden. Obwohl sich mit diesen Registern im Prinzip sehr elegant komplexe Operationen (insbesondere in Schleifen) programmieren ließen, werden sie durch die genannten Einschränkungen nahezu nutzlos. Sie müssen bei der Verwendung dieser Register sehr genau wissen, ob und wann das S7-Betriebssystem diese für interne Zwecke verwendet, damit keine unerwarteten Resultate auftreten. Die Suche nach den Ursachen für Fehlfunktionen, die durch eine unzulässige Verwendung entstehen, verschlingt viel mehr Zeit, als benötigt würde, wenn man auf eine Schleife verzichtet und stattdessen 20 bis 50 Zuweisungen oder Transfers explizit ausschreibt.

Die in TrySim simulierte SPS verändert den Inhalt der AR-Register nicht und benötigt sie auch nicht für irgendwelche Operationen, die nicht explizit von Ihnen programmiert worden sind. Dies kann dazu führen, dass ein Programm mit TrySim prima funktioniert, im tatsächlichen Einsatz jedoch nicht. Solche Fehler sind, wie uns berichtet wurde und wie wir aus eigener Erfahrung wissen, sehr schwer zu finden.

Bitte überprüfen Sie die Funktion der Teile eines mit TrySim getesteten Programms, die die Adressregister verwenden, unbedingt vor der Inbetriebnahme auf einer realen S7.

Selbstverständlich werden wir versuchen, potenzielle Problemfälle in TrySim zu erkennen und eine entsprechende Warn-Meldung auszugeben. Dazu müssen wir aber zunächst genau wissen, unter welchen Umständen eine S7 die Adressregister modifiziert oder auf einen nicht vom Anwender veränderten Inhalt angewiesen ist. Für diesbezügliche Hinweise sind wir sehr dankbar.

Eine sichere Alternative ist die Verwendung der speicher-indirekten Adressierung. Leider gibt es dafür nicht solche Operationen wie +AR1 und andere lästige Beschränkungen, wie z.B. dass IN-Parameter nicht als Index verwendet werden dürfen, sondern erst in eine TEMP-Variable übertragen werden müssen.

LAR1 oder 2

Mit dieser Operation können Sie die Adressregister1 oder 2 beschicken. Sie kann sowohl mit, als auch ohne Operand verwendet werden.
Wird sie mit Operand verwendet, muss dieser ein POINTER sein. Wird sie ohne Operand verwendet, wird der Inhalt des Akkus 1 in das entsprechende Adressregister geladen. In diesem Fall muss natürlich vorher ein gültiger Pointer in den Akku 1 geladen werden.

Anmerkung: TrySim erlaubt mehr LAR1/2 Operanden als STEP®7. Dies kann dazu führen, dass ein in TrySim funktionierendes Programm nicht nach STEP®7 exportierbar ist. Versuchen Sie in solchen Fällen, zunächst den Pointer in den Akku

zu laden und dann LAR1/2 ohne Operand zu verwenden.

Beispiel

Schreiben Sie nicht:

```
      LAR1   P##Any1      // #Any1 ist als In - ANY deklariert
```

Sondern:

```
      L      P##Any1      // Lade den Pointer auf #Any1 in Akku1
      LAR1                // Lade diesen Pointer ins AR1
```

!! Warnung bei Verwendung der Adressregister !!

+AR1 oder 2

Diese Operation wird verwendet, um z.B. in Schleifen die Adressregister Adressregister 1 oder 2 zu erhöhen. In den Akku 1 muss vorher die Schrittweite geladen werden, oder im Pointerformat als Operand angegeben werden.

Diese Werte müssen im Akku 1 stehen , um die gewünschte Schrittweite zu erhalten

Schrittweite	Pointerformat	Hex-Format	Beispiel
1 Bit	P#0.1	0001h	A 5.7 -> A 5.8
1 Byte	P#1.0	0008h	E 7.3 -> E 8.3
1 Word	P#2.0	0010h	MW 6 -> MW 8
1 Dword	P#4.0	0020h	DBD10 -> DBD 14

!! Warnung bei Verwendung der Adressregister !!

Beispiel einer Schleife mit AR1

```
      LAR1 P#A6.0         // Lade das Adressregister mit Pointer auf A6.0
      L    4              // 4 Schleifendurchläufe
lbl   T #Idx              // #Idx ist lokale INT-Variable
:
      SET                 // VKE auf "1" setzen
      = [AR1,P#0.1]       // Ziel ist AR1 + P#0.1
      +AR1 P#1.2          // Schrittweite ist 1 Byte, 2 bits
      L Idx
      LOOP lbl
```

Dies hat die gleiche Wirkung wie:

```
      SET
      = A 6.1
      = A 7.3
      = A 8.5
      = A 8.7
```

S7-300 und S7-400 sind eingetragene Warenzeichen der Siemens AG.

TAR1 oder 2

Das Adressregister wird in den angegeben Operanden transferiert, oder, wenn keiner angegeben ist, in den Akku 1.

!! Warnung bei Verwendung der Adressregister !!

TAR

Die Inhalte der beiden Adressregister werden miteinander vertauscht.

!! Warnung bei Verwendung der Adressregister !!

Indirekte Adressierung mit AR1 und AR2

Für einige Anwendungen ist das Verfahren der speicher-indirekten-Adressierung nicht flexibel genug. Daher gibt es in der CPU zwei spezielle Register (AR 1 und AR2, Adressregister 1 u. 2.), mit deren Hilfe der Index erst während des Zugriffs berechnet wird. Vor der ersten Benutzung der Adressregister müssen diese mit einem Zeiger auf einen Operanden geladen werden. Wenn Sie auf den Operanden E 5.6 zugreifen wollen, programmieren Sie:

```
LAR1   P#5.6  // Lade Pointer auf "irgendwas" 5.6 in AR1
```

Jetzt steht die Adresse 5.6 im Adressregister 1. Den Eingang fragen Sie nun mit folgender Operation ab:

```
U E[AR1,P#0.0]
```

Wollen Sie aber die Adresse E 5.7 abfragen, so programmieren Sie:

```
U E[AR1,P#0.1]
```

Der Pointer nach dem Komma wird zu dem Wert im Adressregister 1 addiert und das Resultat als Byte- und Bit-Nummer des Eingangs interpretiert. Dabei wird berücksichtigt, dass Bytes nur 8 Bits haben, Bit 5.9 wird daher zu Bit 6.0.
Wenn Sie jetzt

```
U E[AR1,P#0.2]
```

programmieren, wird der Eingang 6.0 abgefragt.
Das Ganze nennt man registerindirekte-bereichsinterne-Adressierung.
Bereichsintern deswegen, weil (im Beispiel) immer nur Eingänge abgefragt werden.

Natürlich können Sie durch diese nicht nur auf Bits, sondern auch auf Bytes, Words und DWords zugreifen. Beispiel:

```
L MW[AR2,P#2.0]
```

Es gibt nun aber auch die registerindirekte-bereichsübergreifende Adressierung. Da müssen Sie im Vorfeld nicht einmal entscheiden, ob Eingänge, Ausgänge oder Merker abgefragt werden. Das geht folgendermaßen:
Sie laden das AR1 mit der Adresse A 5.3:

```
LAR1   P#A5.3
```

Dann weisen Sie dem Ausgang A 5.3 den Wert des VKE zu, indem Sie programmieren:

```
= [AR1,P#0.0]
```

und entsprechend den Ausgang A 17.2

```
= [AR1,P#11.7]        // (5.3 + 11.7 = 17.2)
```

Auch hier können Sie genauso auf Bytes, Words und DWords zugreifen:

```
L   W[AR1,P#4.0]
```

Eigenartigerweise ist der bereichsübergreifende Zugriff auf Lokaldaten in einer S7-300 nicht gestattet, bei einer S7-400 jedoch erlaubt. Wir haben diesen Zugriff in TrySim gesperrt.

!! Warnung bei Verwendung der Adressregister !!

S7-300 und S7-400 sind eingetragene Warenzeichen der Siemens AG.

Indirekte Adressierung

Sie müssen nicht schon beim Schreiben Ihres Programmes die Operanden endgültig festlegen, sondern können sie erst während der Laufzeit bestimmen lassen. Nützlich ist dies, wenn immer wiederkehrende Operationen mit verschiedenen Operanden durchgeführt werden sollen. Wenn Sie z.B. 14 verschiedene Rezepte in den Datenbausteinen DB 1 – 14 gespeichert haben und die Nummer des aktuellen Rezeptes im MW 20 gespeichert ist, programmieren Sie:

AUF DB[MW 20]

Wenn im MW 20 eine 5 steht, wird durch diese Anweisung der DB 5 aufgeschlagen und das Programm arbeitet im Folgenden mit dem darin gespeicherten Rezept. Dieses Verfahren nennt man “Speicherindirekte Adressierung”, da der Index in den eckigen Klammern eine beliebige Speicherstelle sein kann.

Zum indirekten Zugriff auf einzelne Bits reicht ein Word wegen des großen Adressbereiches von Eingängen, Ausgängen, Merkern und Datenbits von jeweils 65536 Bytes nicht aus, denn schon für die Angabe der Byte-Adresse wird bereits

ein Word benötigt. Für die indirekte Adressierung dieser Datenbereiche wird daher ein Doppelwort benötigt. Der Einheitlichkeit halber gilt dies auch dann, wenn auf Bytes, Words oder DWords zugegriffen wird. Um die Bit-Nummer zu codieren, werden die drei niederwertigsten Bits des DWords verwendet. Da diese Bits dann für die Byte-Nummer nicht mehr zur Verfügung stehen, wird diese einfach mit 8 multipliziert und zur Bit-Nummer addiert. Im Ergebnis werden alle Bits eines Datenbereiches einzeln durchnummeriert.

Bit 0.2	bekommt die Nummer	2	hex 0000 0002
Bit 1.0	bekommt die Nummer	8	hex 0000 0008
Bit 8.3	bekommt die Nummer	67	hex 0000 0043

In der Praxis brauchen Sie sich um diese Details nicht zu kümmern. Wenn Sie das POINTER-Format verwenden, dann programmieren Sie einfach

```
L     P#8.3
T     MD 24
U     A[MD24]
```

um auf den Ausgang A 8.3 zuzugreifen.

3.7.18 Baustein-Operationen

AUF

Datenbaustein aufschlagen.

Wenn Sie innerhalb eines Programmteils häufig Daten aus dem gleichen Datenbaustein benötigen dann "schlagen" Sie diesen "auf". Im Folgenden brauchen Sie bei der Adressierung von Datenwörtern nicht mehr anzugeben, in welchem DB sich diese befinden, die SPS nimmt sie automatisch aus dem aufgeschlagenen Datenbaustein.

Warnung: Wenn Sie während der Arbeit mit einem aufgeschlagenen DB einen Operanden mit dem voll qualifizierten Zugriff ansprechen, wird automatisch der entsprechende DB aufgeschlagen ! Beispiel:

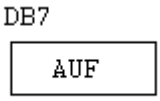
```
AUF     DB 10
L       DBW 20          // Es wird das DW 20 des DB 10 in Akku 1 geladen
L       DB30.DBW 40     // Es wird das DW 40 des DB 30 in Akku 1 geladen
L       DBW 20          // Es wird das DW 20 des DB 30 in Akku 1 geladen
```

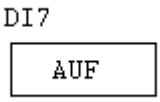
Intern wird durch die Operation AUF das Global-DB-Register der CPU mit dem angegebenen DB geladen.

Sie können die Operation AUF nicht nur mit der Syntax AUF DB xx verwenden, sondern auch mit AUF DI xx.. In diesem Fall wird nicht das Global-DB-Register mit dem angegebenen DB geladen, sondern das Instanz-DB-Register. Alle Zugriffe auf Daten, die ein "I" enthalten (DIX, DIB, DIW oder DID), beziehen sich nachfolgend auf den so aufgeschlagenen DB. Achtung: Auch alle mit "#" bezeichneten Operanden,

die nicht aus dem TEMP-Bereich sind, befinden sich nun in dem als AUF DI.xx aufgeschlagenen DB !

Die Nummer des gerade aufgeschlagenen DB/IDB können Sie mit L DBNO/DINO in den Akku 1 laden.





BEA

Baustein**e**nde **a**bsolut

Wenn bei der Programmbearbeitung dieser Befehl erreicht wird, erfolgt der Rücksprung zum aufrufenden Baustein. Dieser Befehl wird üblicherweise nur in Programmteilen verwendet, die manchmal übersprungen werden, aber auch während der Fehlersuche ist er hilfreich, wenn man ausschließen möchte, dass die gesuchte Fehlfunktion von nachfolgenden Bausteinen verursacht wird.

BEB

Baustein**e**nde **b**edingt

Wenn bei Erreichen dieses Befehls das VKE "1" ist, wird die Bearbeitung des Bausteins abgebrochen und zum aufrufenden Baustein zurückgekehrt. Sonst wird das VKE auf "1" gesetzt und der nächste Befehl bearbeitet. BEB wird meist in Bausteinen verwendet, die bestimmte (wenige) Operationen immer, den Rest des Bausteins aber nur beim Vorliegen einer Bedingung ausführen sollen.

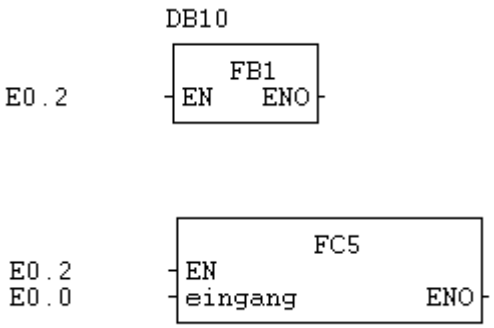
CALL

Bausteinaufruf

Syntax: CALL FB x, DB y
CALL FC x

Mit CALL veranlassen Sie den Sprung zu dem angegebenen Baustein, der eine Funktion(FC) oder ein Funktionsbaustein(FB) sein kann. Beim Aufruf eines FB müssen Sie den DB angeben, der vom FB als Instanz-DB verwendet werden soll. Wenn Sie den Befehl CALL programmieren, erscheinen automatisch die Parameter des aufgerufenen Bausteins. Bei FC müssen Sie für jeden der aufgeführten

Formalparameter einen Operanden angeben. Beim Aufruf eines FB müssen Sie dies nicht: wenn Sie keinen Operanden angeben, wird innerhalb des FB der entsprechende Wert des Instanz-DB verwendet.



Der FB1 bzw. die FC5 werden nur aufgerufen, wenn der Eingang E 0.2 auf “1” ist. Wenn der Aufruf unbedingt erfolgen soll, können Sie den EN-Eingang auch unbeschaltet lassen.

CC

Bedingter Bausteinaufruf (**c**onditional **c**all)

Syntax: CC FBxx
CC FCxx

Der angegebene Baustein wird aufgerufen, wenn das VKE “1” ist, sonst wird der nächste Befehl bearbeitet. Es erscheint keine Parameterliste.

Sie sollten diese Form des Aufrufes nur für Funktionen und Funktionsbausteine verwenden, die keine Parameter und auch keine statischen Variablen haben, denn sonst werden unvorhergesehene Ergebnisse auftreten, in jedem Fall wird Ihr Programm schwer zu verstehen sein.

Wenn Sie beim Aufruf eines FB keinen IDB angeben, wird stattdessen der IDB des aufrufenden Bausteines verwendet ! Das führt nicht oft zu vernünftigen Ergebnissen.

UC

Unbedingter Bausteinaufruf (**u**nconditional **c**all)

Syntax: UC FBxx
UC FCxx

Der angegebene Baustein wird unabhängig vom VKE aufgerufen. Es erscheint keine Parameterliste.

Sie sollten diese Form des Aufrufes nur für Funktionen und Funktionsbausteine

verwenden, die keine Parameter und auch keine statischen Variablen haben, denn sonst werden unvorhergesehene Ergebnisse auftreten, in jeden Fall wird Ihr Programm schwer zu verstehen sein.

Wenn Sie beim Aufruf eines FB keinen IDB angeben, wird stattdessen der IDB des aufrufenden Bausteines verwendet! Das führt nicht oft zu vernünftigen Ergebnissen.

TDB

Die Inhalte der beiden Datenbausteinregister (Global- und Instanzdatenbausteinregister) werden miteinander vertauscht.

Zugriffe mit DBW beziehen sich danach auf den aktuellen Instanz-DB, Zugriffe mit DIW beziehen sich danach auf den bisherigen Global-DB. Das gilt auch für alle mit "#" gekennzeichneten Parametern und Variablen, außer den TEMP-Variablen.

Diese Operation sollten Sie nur verwenden, wenn der Global-DB ebenfalls ein Instanz-DB zum aktuellen Funktionsbaustein ist, denn sonst wird es sehr unübersichtlich.

3.7.19 Null-Operationen

NOP 0 oder 1

Hier macht die SPS gar nichts.

Der Befehl wird vom System als Platzhalter für nicht-beschaltete Eingänge von Flip-Flops u.ä. verwendet, damit die Rückübersetzung von AWL nach FUP oder KOP einfacher ist.

Wenn man in AWL programmiert, benutzt man diesen Befehl meistens deswegen, weil in einer Zeile mit Sprungmarke ein Befehl stehen **muss.**

BLD

Bildbefehl

Dieser Befehl bewirkt nichts. Sie werden ihn nie programmieren. Er erscheint nur im AWL-Code von in FUP- oder KOP – programmierten Netzwerken. Das Programmiersystem fügt ihn ein, um die Rückübersetzung des AWL-Codes in eine FUP- oder KOP-Grafik zu gewährleisten.

3.7.20 Master Control Relais (nicht implementiert)

Master-Control-Relais

Siehe voriges Kapitel.

MCR(

Die Funktionen für das Master-Control-Relais sind in TrySim nicht implementiert.

)MCR

Die Funktionen für das Master-Control-Relais sind in TrySim nicht implementiert.

MCRA

Die Funktionen für das Master-Control-Relais sind in TrySim nicht implementiert.

MCRD

Die Funktionen für das Master-Control-Relais sind in TrySim nicht implementiert.

3.8 Unterschiede der TrySim-SPS zur S7

Einige Eigenschaften der realen S7 sind in TrySim nicht verfügbar. Manche werden jedoch auf andere Weise realisiert, sodass es in Ausnahmefällen zu einem abweichenden Verhalten kommen kann.

S7-300 und S7-400 sind eingetragene Warenzeichen der Siemens AG.

3.8.1 Nicht implementierte Eigenschaften

Folgende Eigenschaften der S7 sind in TrySim nicht oder nicht vollständig vorhanden:
(auch diese Liste ist natürlich nicht vollständig).

Master-Control-Relais

Siehe voriges Kapitel.

Prozessalarme

Alarmbearbeitung ist nicht implementiert. In Grenzen können Sie Alarmbearbeitung dennoch simulieren, da Sie in TrySim auch OB aufrufen können. Programmieren Sie einen bedingten OB-Aufruf nach der Auswertung der Bedingung, die den Alarm auslösen soll.

Zeitalarme

Auch diese Alarme sind nicht vorhanden. Zeitalarme, die in größeren Abständen als die Zykluszeit aufgerufen werden, können Sie durch einen bedingten OB-Aufruf nach einer Zeit nachbilden.

Ab Version 2.2 ist ein Weckalarm (OB 35) vorhanden.

Eingeschränkte Weckalarme

Diese Registerkarte ist unter SPS|CPU-Eigenschaften zu erreichen.

Der OB 35 wird, unabhängig von der eingestellten Simulationsrate, in den hier vorgegebenen Zeitabständen aufgerufen. Weitere Weckalarme sind noch nicht vorhanden.

Eingeschränklte ARRAYs

Arrays von Strukturen, anderen komplexen Typen, sowie Arrays von Arrays und mehrdimensionale Arrays sind noch nicht möglich.

Außerdem können nicht alle Möglichkeiten zur Defaultwert-Vorgabe genutzt werden.

Von folgenden Typen können Arrays gebildet werden:
BOOL, BYTE, CHAR, INT, WORD, DATE, S5TIME, DWORD, DINT, REAL, TIME_OF_DAY, TIME

Eingeschränkt können Arrays auch von UDTs gebildet werden. Dies ist eine Möglichkeit, um das Fehlen von Arrays aus Strukturen zu überbrücken.

Eingeschränkte Funktion von UDT's

UDT (user defined types) können sowohl als Maske zur Erstellung von DB dienen, als auch als Abkürzung zur Deklaration von strukturierten Variablen verwendet werden. Die Bearbeitung ist jedoch gelegentlich noch fehlerhaft. Wenn Sie UDT verwenden wollen, erkundigen Sie sich bitte bei uns, ob eine bessere Version zur Verfügung steht.

Funktion "FR" bei Zeiten und Zählern

Diese Funktionen ist nicht implementiert.

SPU, SPO und SPS

Diese bedingten Sprungbefehle sind nicht implementiert.

STRINGS

Strings sind ab Version 1.5 implementiert. Sie können Strings deklarieren, als Parameter von Funktionen verwenden und im- und exportieren. Auf die einzelnen Buchstaben greifen Sie mit der Notation Stringname[idx] zu.

Folgende Funktionen stehen im Verzeichnis IECfuncs für die Bearbeitung von Strings bislang zur Verfügung:

FC 21 Länge eines Strings ermitteln.

Weitere sind in Vorbereitung, bei Bedarf bitte nachfragen.

3.8.2 Abweichend implementierte Eigenschaften

Und vor Oder

Bei einer S7 wird diese Rechenregel durch das OR-Bit realisiert. Da die exakte Nachbildung dieses Verfahrens zu einer merklichen Reduzierung der Ausführungsgeschwindigkeit geführt hätte, wird die Und-vor-Oder-Regel in TrySim bereits vom Compiler berücksichtigt. Wenn Sie in FUP oder KOP programmieren, werden Sie keine Unterschiede im Verhalten der TrySim-SPS und einer S7 bemerken. Wenn Sie jedoch in AWL Sprünge zwischen verschiedenen Klammerebenen programmieren, kann es zu Abweichungen kommen. Vermeiden Sie solche Sprünge, die nebenbei zu schlecht lesbaren Programmen führen.

S7-300 und S7-400 sind eingetragene Warenzeichen der Siemens AG.

Parameter-Aktualisierung bei Funktionen

Die folgenden Unterschiede betreffen nur Funktionen, nicht Funktionsbausteine.
In einer S7 wird bei einer Zuweisung an einen Out- oder In-Out-Parameter, dessen Aktualparameter nicht ein weiterer Parameter des aufrufenden Bausteins, sondern ein direkter Operand (Ein-/Ausgang, Merker, Datenwort) ist, der Aktualparameter im Moment der Zuweisung verändert. Genauso wird beim Zugriff auf einen In-Parameter, an den ein direkter Operand angeschlossen ist, der tatsächliche Zustand dieses Operanden erkannt, auch wenn sich dieser seit Aufruf des Bausteins geändert hat.

In TrySim erfolgt die Aktualisierung der an Out- und In-Out-Parameter angeschlossenen Aktualparameter erst bei der Rückkehr zum aufrufenden Baustein. Die In- und In-Out-Parameter werden einmal beim Aufruf des Bausteins aktualisiert, danach werden die Aktual-Parameter nicht mehr gelesen.
Die dadurch bedingten Unterschiede treten aber nur zu Tage, wenn Sie auf einen Operanden sowohl direkt als auch über Parameter zugreifen.

Betrachten Sie dieses Programmbeispiel:

S7 TrySim

OB 1			OB 1		
		Status			Status
	SET	1		SET	1
	= M 1.2	1		= M 1.2	1
	CALL FC 2			CALL FC 2	
	Par1: M 1.2			Par1: M 1.2	
	U M 1.2	0		U M 1.2	0
FC 2			FC 2		
OUT	Par1	BOOL	OUT	Par1	BOOL
		Status			Status
	U #Par1	1		U #Par1	1
	U M 1.2	**1**		U M 1.2	1
	CLR	0		CLR	0
	= #Par1	0		= #Par1	0
	U #Par1	0		U #Par1	0
	U M 1.2	**0**		U M 1.2	**1**

Vermeiden Sie daher solche gemischten Zugriffe, die im Übrigen zu schwer zu lokalisierenden Programmfehlern führen können. Im obrigen Beispiel hat sich auf der linken Seite der Zustand des Merkers M 1.2 während der Programmbearbeitung von 1 nach 0 geändert, ohne dass dies durch eine explizite Zuweisung zu erkennen wäre.

S7-300 und S7-400 sind eingetragene Warenzeichen der Siemens AG.

Out-Parameter von Funktionen

Das Folgende betrifft nur Funktionen, nicht Funktionsbausteine.

In einer S7 haben Out-Parameter entgegen den Regeln der IEC 1131 bereits beim Aufruf einen definierten Wert. Sie verhalten sich eigentlich wie In-Out-Parameter. Dies verführt einige Programmierer dazu, Out-Parameter wie In-Out-Parameter zu verwenden, z.B. indem ein Out-Parameter in vielen Durchläufen gar nicht bearbeitet wird und nur gelegentlich zurückgesetzt wird. Wir halten diese Praxis für sehr gefährlich. denn es ist nicht garantiert, dass spätere Ausgabestände einer CPU das

gleiche Verhalten zeigen. Die Folge wäre, dass ein Programm nach Ausfall einer CPU auf einer neuen CPU nicht mehr läuft. Siemens kann man in einem solchen Fall keine Schuld zuweisen: die neue CPU verhält sich ja gemäß IEC 1131. Auch die Portierung des Programms auf ein anderes System, das sich IEC konform verhält, wird durch diesen Mißbrauch der Out-Parameter sehr viel aufwändiger als vorhergesehen.

In TrySim haben daher die Out-Parameter beim Aufruf einer Funktion regelgemäß keinen definierten Wert. Dies kann dazu führen, dass ein Programm, welches auf einer S7 prima läuft, Probleme bereitet, wenn es von der TrySim-internen-SPS bearbeitet wird.

Für die Nutzer der Standard- und Professional-Version bieten wir für EUR 100,- die Option an, das falsche Verhalten der realen S7 in diesem Punkt exakt nachzubilden. Wir tun dies ungern, aber der Kunde ist König.

Was oben über die Out-Parameter gesagt worden ist, gilt sinngemäß auch für die In-Parameter. Hier liegt bei der S7 allerdings ein klarer Verstoß gegen die Bestimmungen der IEC 1131 vor: Ein In-Parameter darf in einer Funktion nicht so verändert werden können, dass der aufrufende Baustein danach mit anderen Daten rechnen muss. Die Versuchung, einem In-Parameter in einer Funktion einen neuen Wert zuzuweisen ist aber relativ gering, daher ist hier nur in Ausnahmefällen mit Problemen zu rechnen. TrySim verhält sich in diesem Punkt IEC 1131 - konform.

S7-300 und S7-400 sind eingetragene Warenzeichen der Siemens AG.

Datentypprüfung

Die Datentypprüfung ist in TrySim nicht so streng wie in STEP®7, solange die Größen der Operanden übereinstimmen. So können Sie z.B. an einen als CHAR deklarierten In-Parameter auch ein BYTE anschließen, ohne dass dies vom Compiler bemängelt wird. In STEP®7 wird dies zu einem Typkonflikt führen.

STEP®7 ist eingetragenes Warenzeichen der Siemens AG.

SPS-Editor

Kapitel

4 SPS-Editor

Mit dem Programmeditor erstellen, modifizieren und testen Sie Ihr Programm. Es sind 3 Darstellungsarten möglich: Anweisungsliste (AWL), Funktionsplan (FUP) und Kontaktplan (KOP).

4.1 Bausteinfunktionen

4.1.1 Neue Bausteine erzeugen

Klicken Sie auf **SPS|Neu** .

- Wählen Sie Bausteinart und geben Sie die Bausteinnummer ein.
- Wählen Sie die gewünschte Darstellungsart durch Klicken eines der Buttons AWL, FUP, KOP in der oberen Symbolleiste.
- Sie können unter **Ansicht|Optionen|SPS-Editor** einstellen, welche Ihre bevorzugte Darstellung ist.
- Das erste Netzwerk wird automatisch eingefügt.

Datenbausteine können Sie mit der SFC22 auch während der Laufzeit erzeugen.

4.1.2 Bausteinarten

- OB Organisationsbausteine
- FB Funktionsbausteine
- FC Funktionen
- DB Datenbausteine
- I-DB Instanz-DB
- UDT-DB UDT
- SFB Systemfunktionsbausteine
- SFC Systemfunktionen
- UDT UDT (**u**ser **d**efined **t**ype)

4.1.3 Bausteine speichern und öffnen

Sie können Programmbausteine auf verschiedene Arten öffnen:

- Klicken Sie **SPS|Öffnen** und wählen Sie einen Baustein aus der Liste aus. Mit den Buttons oberhalb der Liste können Sie die Anzeige nach Bausteinsorten filtern.
- Wählen Sie **SPS|NeuÖffnen**. Jetzt erscheint eine Liste mit allen zuvor geöffneten Bausteinen.
- Sie doppelklicken in der Querverweisliste auf die Zeile, in der der gewünschte Baustein steht. Mit dieser Methode sind Sie auch gleich im richtigen Netzwerk.
- Sie doppelklicken in KOP, FUP oder AWL auf den entsprechenden Bausteinaufruf.
- Sie klicken in KOP, FUP oder AWL mit rechts auf einen Bausteinaufruf und

wählen “Öffnen” aus dem Kontext-Menü.
- Bausteine werden automatisch an der richtigen Stelle geöffnet, wenn beim Übertragen oder während der Laufzeit ein Fehler auftritt.
- Sie gehen im Einzelschrittmodus mit “Trace” in den Baustein hinein.

Sie können Programmbausteine auf verschiedene Arten speichern:
- Wählen Sie **SPS|Speichern**

Wählen Sie **Projekt|Alles Speichern** oder das Symbol 
- Sie schließen den Baustein und beantworten die Frage nach dem Speichern mit “Ja”.
- Sie aktivieren die Checkbox “Automatisch speichern” unter **Ansicht|Optionen|SPS-Editor**. Nun werden die Bausteine beim Schließen ohne lästige Rückfrage gespeichert.
- Vor dem Übertragen in die SPS werden Bausteine grundsätzlich automatisch gespeichert.

4.1.4 Löschen eines Bausteins

Zum Löschen eines Bausteins wählen Sie **SPS|Öffnen**, markieren den Baustein und betätigen die Taste “**Entf**”. Der Baustein ist danach unwiderruflich gelöscht und wird auch aus der SPS entfernt.

Sie können auch mehrere markierte Bausteine gleichzeitig löschen.

Der Punkt “Löschen” ist auch im Kontextmenü des Baustein-Öffnen-Fensters vorhanden.

Datenbausteine können Sie mit der SFC23 auch während der Laufzeit löschen.

4.1.5 Bausteine exportieren und importieren

TrySim ist eine reine Offline-Software. Wenn Sie ein mit TrySim erstelltes Programm auf einer echten SPS laufen lassen wollen, geht dies nur mittels des Original-Entwicklungssystems des Herstellers. Bislang unterstützt TrySim nur die S7-300/400-Reihe von Siemens.

Export nach STEP®7
Sie können das Programm und die Symboltabelle nach STEP®7 exportieren, dort weiter bearbeiten und schließlich in eine echte SPS übertragen.

Import von STEP®7
Sie können ein Programm und die Symboltabelle, die Sie in STEP ®7 geschrieben haben, nach TrySim importieren und dort testen. Wenn Sie Änderungen vornehmen,

können Sie es anschließend nach STEP ®7 zurückübertragen.

Import von STEP®5
Der sowieso sehr eingeschränkte Import von STEP ®5 ist nicht mehr möglich.

4.1.6 AWL

Der AWL-Editor ist im Wesentlichen ein normaler Text-Editor und kann mit den Windows-üblichen Tastenkombinationen bedient werden.
Eine Auswahlliste wie für FUP und KOP existiert für AWL nicht.
Kommentare beginnen mit "//".
Sprungmarken müssen Sie mit einem ":" abschließen.
Durch Rechtsklick erhalten Sie ein Kontextmenü.
Doppelklick auf eine Zeile mit einem Bausteinaufruf öffnet diesen.
Doppelklick auf eine Zeile mit voll qualifiziertem Zugriff öffnet den entsprechenden DB.

Sie können Operanden aus der Anlage direkt übernehmen, indem Sie das entsprechende Element markieren (im Grafik-Fenster, Adressentabelle oder Elementbaum) und dann im AWL-Editor aus dem Kontext-Menü die letzte Zeile wählen. Wenn Sie vorher Text markiert haben, wird dieser durch den Operanden ersetzt, wenn Sie keinen Text markiert haben, wird der Operand an der Cursor-Position eingefügt.

4.1.7 FUP

Die am Häufigsten benötigten Befehle finden Sie auf der Symbolleiste, die erscheint, wenn die Darstellung des aktuellen Bausteins "FUP" ist. Sollte diese Leiste nicht zu sehen sein, klicken Sie mit der rechten Maustaste auf den grauen Balken neben den vorhandenen Symbolen und wählen aus dem erscheinenden Kontextmenü "FUP"-Ansicht.

Weitere Befehle stehen zur Auswahl, wenn Sie auf das Symbol  klicken.
Für einige Befehle sind auch Tastenkürzel vorhanden:

F2	Und-Box
F3	Oder-Box
F8	Neuer Anschluss
F9	Negation an/aus

Wenn das Netzwerk noch leer ist, brauchen Sie auf die Position des Cursors nicht zu achten. Sind jedoch schon Funktionsblöcke vorhanden, müssen Sie den Cursor vor dem Einfügen neuer Elemente auf einen Operanden oder auf die Verbindung zwischen zwei Funktionsblöcken positionieren.

Das Editierfeld der Operanden öffnen Sie durch die Eingabe - Taste oder durch die Eingabe eines Buchstabens, einer Zahl , “#” oder der doppelten Anführungszeichen. Nach Beendigung der Eingabe schließen Sie das Feld mit der Eingabe-Taste oder, wenn Sie Ihre Eingabe nicht übernehmen wollen, mit der **ESC**-Taste.

Sie können Operanden aus der Anlage direkt übernehmen. Markieren Sie dazu das Element im Grafik-Fenster, in der Adressentabelle oder im Elementbaum. Bei Operanden die noch mit einem “?” anfangen (oder ganz leer sind), genügt ein Klick mit der linken Maustaste, um den Operanden aus der Anlage einzufügen. Wollen Sie einen bereits beschrifteten Operanden ersetzen, wählen Sie aus dem Kontextmenü den letzten Punkt.

Sie brauchen die Operanden nicht sofort zu beschriften, solange sie mit einem “?” anfangen. Spätestens wenn Sie das Programm in die SPS übertragen, müssen aber alle obligatorischen Operanden angegeben sein.

Teile des Netzwerkes lassen sich mit der **Entf**-Taste löschen.

Durch Rechtsklick erhalten Sie ein Kontextmenü.

Doppelklick auf einen Baustein öffnet diesen.

4.1.8 KOP

Die am Häufigsten benötigten Befehle finden Sie auf der Symbolleiste, die erscheint, wenn die Darstellung des aktuellen Bausteins KOP ist. Sollte diese Leiste nicht zu sehen sein, klicken Sie mit der rechten Maustaste auf den grauen Balken neben den vorhandenen Symbolen und wählen aus dem erscheinenden Kontextmenü “KOP - Ansicht”.

Weitere Befehle stehen zur Auswahl, wenn Sie auf das Symbol  klicken.
Wenn das Netzwerk noch leer ist, brauchen Sie auf die Position des Cursors nicht zu achten. Sind jedoch schon Strompfade vorhanden, müssen Sie den Cursor vor dem Einfügen neuer Elemente auf einen Kontakt oder auf einen Funktionsblock positionieren.

Das Editierfeld der Operanden öffnen Sie durch die Eingabe-Taste oder durch die Eingabe eines Buchstabens, einer Zahl , “#” oder der doppelten Anführungszeichen. Nach Beendigung der Eingabe schließen Sie das Feld mit der Eingabe-Taste oder, wenn Sie Ihre Eingabe nicht übernehmen wollen mit, der ESC-Taste.

Sie können Operanden aus der Anlage direkt übernehmen. Markieren Sie dazu das Element im Grafik-Fenster, in der Adressentabelle oder im Elementbaum. Bei Operanden die noch mit einem “?” anfangen (oder ganz leer sind), genügt ein Klick mit der linken Maustaste, um den Operanden aus der Anlage einzufügen. Wollen Sie einen bereits beschrifteten Operanden ersetzen, wählen Sie aus dem Kontextmenü den letzten Punkt.

Sie brauchen die Operanden nicht sofort zu beschriften, solange sie mit einem “?” anfangen. Spätestens wenn Sie das Programm in die SPS übertragen, müssen aber alle obligatorischen Operanden angegeben sein.

Teile des Netzwerkes lassen sich mit der “Entf”-Taste löschen.

Durch Rechtsklick erhalten Sie ein Kontextmenü.

Doppelklick auf einen Baustein öffnet diesen.

4.1.9 Operand übernehmen

Einen Operand übernehmen bedeutet, einen in der Anlage bereits verwendeten Operanden im SPS-Programm einzufügen, ohne ihn einzutippen. Wenn Sie z.B. einen Leuchtmelder zur Anzeige eines bestimmten Zustandes in der Anlage erzeugt haben und diesen nun im SPS-Programm ansprechen wollen, müssen Sie nur den Leuchtmelder in der Anlage markieren und danach im SPS-Programm auf die entsprechende Zuweisung klicken. Die Adresse, unter der der Leuchtmelder angesprochen wird, wird dann automatisch im SPS-Programm eingetragen (übernommen), ohne dass Sie ihn erneut eintippen müssen. In FUP/KOP und AWL gelten etwas andere Regeln, wie und wann ein Operand übernommen wird:

4.1.10 Schnelle Operanden-Eingabe über Nummern-Block

Zur Beschleunigung der Eingabe von Operanden auf den Editiermasken, im FUP-/KOP-Editor sowie der Symbol- und Adressentabelle sind einige Tasten auf dem Nummernblock anders belegt:

÷	:	E	bzw. I wenn IEC-Mnemonics
*	:	A	bzw. Q
-	:	M	
,	:	.	

4.2 Netzwerkfunktionen

Neues Netzwerk
Solange Sie die Netzwerke eines nach dem anderen schreiben, werden sie automatisch erzeugt, sowie Sie das letzte Netzwerk mit “Bild runter” oder dem Doppelpfeil auf dem Scrollbar verlassen. Wenn Sie nachträglich Netzwerke einfügen wollen, wählen Sie **SPS |Netzwerk|Neu**. Das neue Netzwerk wird vor dem aktuellen Netzwerk eingefügt. Beachten Sie, dass Sie mit der Funktion **Netzwerk|Einfügen** kein leeres Netzwerk erhalten, sondern das zuletzt Gelöschte bzw. Kopierte.

Netzwerk löschen
Wählen Sie **SPS|Netzwerk|löschen**. Das gelöschte Netzwerk wird im Clipboard

gespeichert und kann mit **Netzwerk|einfügen** an anderer Stelle, auch in einem anderen Baustein, wieder eingefügt werden.

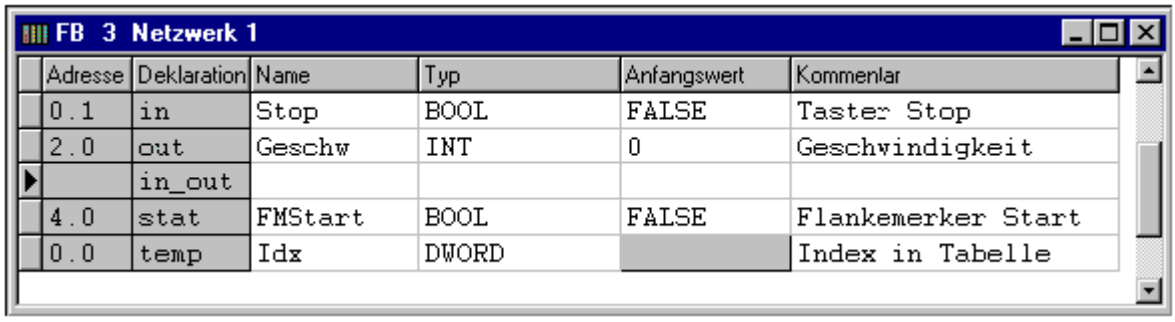
Netzwerk kopieren
Mit **SPS|Netzwerk|kopieren** übertragen Sie das aktuelle Netzwerk in das Clipboard, ohne es zu löschen. Sie können dieses Netzwerk dann an anderer Stelle mit **Netzwerk|einfügen** wieder einfügen.

Netzwerk einfügen
Mit **SPS|Netzwerk|einfügen** fügen Sie ein zuvor gelöschtes oder kopiertes Netzwerk vor das aktuelle Netzwerk wieder ein. Sie können Netzwerke mehrfach wieder einfügen und auch in anderen Bausteinen.

4.3 Editieren des Bausteinkopfes

Den Bausteinkopf brauchen Sie nur zu editieren, wenn Sie den Baustein mit Parametern versehen wollen oder statische / temporäre Variablen benötigen. Nach jeder Änderung des Bausteinkopfes sollten Sie alle Aufrufe dieses Bausteins überprüfen. Dies geht am schnellsten mittels der Querverweisliste, doppelklicken Sie einfach auf jeden Eintrag des Bausteins.

Beschreibung der Spalten



Adresse	Deklaration	Name	Typ	Anfangswert	Kommentar
0.1	in	Stop	BOOL	FALSE	Taster Stop
2.0	out	Geschw	INT	0	Geschwindigkeit
	in_out				
4.0	stat	FMStart	BOOL	FALSE	Flankemerker Start
0.0	temp	Idx	DWORD		Index in Tabelle

In der ersten Spalte wird die automatisch bestimmte Adresse angezeigt. Ein neuer Deklarationstyp beginnt immer auf einer geraden Adresse. Bei Funktionsbausteinen beginnt die Nummerierung der temporären Variablen wieder bei Null, da diese Variablen nicht gemeinsam mit den anderen Werten im Instanz-DB, sondern auf dem Stack gespeichert werden.
In der zweiten Spalte erkennen Sie den Deklarationstyp. Welchen Deklarationstyp ein Parameter/eine Variable erhält, wird durch die Position in der Tabelle festgelegt (siehe etwas weiter unten).
In der dritten Spalte tragen Sie den Namen ein, verwenden Sie dabei keine Sonderzeichen außer “_” und keine Leerzeichen. Auch darf der Name nicht mit einer Ziffer anfangen.
In der Spalte "Typ" tragen Sie den Datentyp ein. Vom Namen des Typs brauchen Sie nur die Buchstaben einzugeben, die ausreichen, den Typ zu bestimmen. Folgende Datentypen stehen zur Verfügung

BOOL, BYTE, INT, WORD, DINT, DWORD, TIMER, COUNTER, S5TIME, TIME, DATE, BLOCK_DB, BLOCK_FB, BLOCK_FC, POINTER, ANY, ARRAY, STRUCT .....

In der Spalte "Anfangswert" tragen Sie am besten zunächst gar nichts ein. Dann setzt das System automatisch einen Nullwert im richtigen Format ein, den Sie nachträglich an Ihre Erfordernisse anpassen können.
In der Kommentarzeile sollten Sie eine möglichst genaue Beschreibung der Variablen eintragen.

Einfügen neuer Zeilen
Gehen Sie mit dem Cursor in eine bereits vorhanden Zeile des gewünschten Deklarationstyps. Wenn Sie eine neue Zeile davor einfügen wollen, betätigen Sie die Taste “Einfg”, wenn Sie die neue Zeile danach einfügen wollen, drücken Sie mehrfach die “Return-” (Eingabe-) Taste.

Löschen von Zeilen
Gehen Sie mit dem Cursor in die 1. Spalte oder klicken Sie darauf. Wenn Sie jetzt die Leertaste betätigen, wird die ganze Zeile markiert. Mit der Taste “Entf” entfernen Sie die Zeile. Betätigen Sie gleichzeitig die “Shift”-(Umschalt-) Taste, so wird die Zeile im Clipboard gespeichert und Sie können sie an anderer Stelle (jedoch nicht in einem anderen Baustein) mit “Shift” - “Einfg” wieder einfügen.

Ändern des Deklarationstyps
Da der Deklarationstyp durch die Position im Bausteinkopf bestimmt wird, können Sie den Typ nur ändern, indem Sie die Zeile markieren (Feld “Adresse” anwählen und dann die Leertaste drücken), mit “Shift” - “Entf” ausschneiden und mit “Shift” - “Einfg” in dem richtigen Abschnitt wieder einfügen.

4.4 Editieren von DB

Beim Editieren eines DB müssen Sie unterscheiden, ob es sich um einen frei erstellten DB handelt oder um einen, der als Instanz-DB zu einem Funktionsbaustein erstellt worden ist. Außerdem kann der DB noch nach der Vorlage eines UDT erstellt worden sein.

1. Editieren eines frei erstellten DB
Wenn Sie einen DB geöffnet haben, können Sie im Menü **Ansicht** zwischen der Deklarationsansicht und der Datenansicht wählen. Nur in der Datenansicht sehen Sie die aktuell in der SPS gespeicherten Werte und können diese auch verändern. In der Deklarationsansicht dagegen können Sie Name, Datentyp, Anfangswert und Kommentar einzelner Zeilen verändern.
Zum Löschen einzelner Zeilen bewegen Sie den Cursor in die erste Spalte und betätigen dann die Leertaste. Die Zeile wird markiert. Sie können sie mit der “Entf”-Taste löschen oder mit “Shift-Entf” in den Zwischenspeicher schieben und an anderer Stelle mit “Shift-Einfg” wieder einfügen. Die Markierung von mehreren Zeilen ist leider noch nicht möglich.

2. Editieren von Instanz-DB und DB, die nach einem UDT erstellt worden sind
In diesen DB ist die Struktur schon bei der Erzeugung unwiderruflich festgelegt worden, sie werden daher automatisch in der Datenansicht geöffnet. Das Einzige, was Sie editieren können, sind die Daten in der SPS.

Wenn Sie einen Datenbaustein während eines Simulationslaufes in der Datenansicht betrachten, können Sie im Menü **Ansicht** den Beobachten-Modus aktivieren, jetzt wird die Ansicht ca. jede Sekunde mit den Werten aus der SPS aktualisiert.
Wenn Sie nur gelegentlich die aktuellen Werte in der SPS benötigen, können Sie **Ansicht|Aktualisieren** wählen, dann werden die Daten nur einmal aus der SPS gelesen.

4.5 Übertragen eines Bausteins in die SPS

Zum Übertragen eines (geöffneten) Bausteines wählen Sie den Menüpunkt **SPS|Übertragen** oder klicken auf das Symbol: 
Andere Bausteine, die von dem gerade übertragenen Baustein benötigt werden, werden automatisch mitübertragen. Sie brauchen also nur den OB 1 zu übertragen und danach nur noch jeden Baustein, den Sie geändert haben.

Benötigte Datenbausteine werden ebenfalls automatisch zur SPS übertragen. Dies gilt sogar für die mittels indirekter Adressierung angesprochenen Datenbausteine. Außerdem werden Datenbausteine in die SPS übertragen, wenn Sie sie öffnen. Beim Öffnen eines DB kann es vorkommen, dass die Daten des Bausteins in der SPS nicht mit den auf der Platte gespeicherten Daten übereinstimmen. In diesem Fall werden Sie beim Öffnen aufgefordert zu entscheiden, ob die Daten von der Datei zur SPS oder von der SPS zur Datei übertragen werden sollen.

Ein Baustein wird ebenfalls automatisch in die SPS übertragen, wenn Sie den Beobachten-Modus anschalten.

TrySim ist (mit Einschränkungen) eine reine Offline-Software. Wenn Sie Bausteine in eine wirkliche SPS übertragen wollen, müssen Sie das Programm exportieren.

In realen SPS-Systemen muss aus Sicherheitsgründen scharf zwischen den Bausteinen in der SPS (online) und denen im Entwicklungssystem (offline) unterschieden werden. In TrySim ist diese Unterscheidung aber nicht wesentlich, sondern eher störend und nach einer Weile werden Sie das Übertragen einfach als “Bestätigen” empfinden.

4.6 Beobachten der Programmbearbeitung

Sie schalten in den Modus ”Beobachten” über den Menüpunkt **SPS|Beobachten** oder
über das Symbol  .

Auf der Statusleiste unten rechts erscheint eine laufenden Fortschrittsanzeige. Wenn diese Anzeige sich nicht bewegt, wird der Baustein vermutlich nicht bearbeitet. Entweder ist dann die Anlage im Zustand ”Stop” (rote Lampe im Anlagenfenster unten links) oder der Baustein wird nicht aufgerufen oder vor dem beobachteten Netzwerk ist ”Baustein Ende” programmiert oder das Programm wird wegen eines fatalen Fehlers nicht mehr bearbeitet (hierüber erhalten Sie aber eine Meldung).
Sie schalten den Beobachten-Modus durch erneute Anwahl des Menüpunktes, durch erneutes Klicken auf das Symbol oder mit der ”ESC”-Taste wieder aus.
Sie können auch mehrere Bausteine gleichzeitig beobachten. Häufig ist es nützlich, beim Beobachten den Einzelschritt-Modus zu verwenden. Während Sie beobachten, sind keine Eingaben möglich.

Beobachten in FUP
Aktive Operanden werden rot dargestellt, inaktive grau. Unterhalb von Byte-, Word- und DWord-Operanden wird der aktuelle Wert dargestellt. Beizeiten steht der aktuelle Wert rechts oberhalb, bei Zählern im unteren Teil des Rechteckes.

Beobachten in KOP
Stromführende Verbindungen werden rot dargestellt, stromlose Verbindungen grau. Die Darstellung der anderen Elemente ist wie bei FUP.

Beobachten in AWL
Rechts neben dem Programmcode werden anstelle des Kommentares folgende Informationen dargestellt:

VKE | Status des Operanden | Akku 1 | Akku 2

Die Darstellung der Akkus wird, so gut es geht, automatisch an den jeweiligen Operanden angepasst. So werden die Akkus nach einer Integer-Addition als Integers dargestellt, nach einer wortweisen Und-Verknüpfung jedoch als Dualzahlen solange 12 Stellen nicht überschritten werden, danach als Hex-Zahlen..
Je genauer Sie die Datentypen in den Deklarations-Köpfen und der Symboltabelle deklarieren, desto besser weiß das System, wie der Akku darzustellen ist. Wenn Sie zum Beispiel im MD 24 immer eine Realzahl speichern, sollten Sie dies in der Symboltabelle auch deklarieren und den Datentyp von dem Defaultwert “DWORD” nach “REAL” ändern.
Sie erkennen die Hex-Darstellung daran, dass sie im Gegensatz zu der Integer-Darstellung mit führenden Nullen geschrieben wird.
Sollte Sie die häufig wechselnde Darstellung der Akkus stören, so können Sie unter Ansicht|Optionen|SPS-Editor auch ein festes Datenformat vorgeben.

4.7 Breakpoints und Einzelschrittmodus

Durch Breakpoints (Haltepunkte) können Sie die Bearbeitung des SPS-Programms an einer beliebigen Stelle unterbrechen. Die Bearbeitung der Simulation wird dann ebenfalls gestoppt.

Nachdem das Programm angehalten wurde, wird das CPU-Fenster eingeblendet

und die LED links unten wird gelb. Auf dem CPU-Fenster werden Ihnen vier Möglichkeiten angeboten, die Bearbeitung des Programms fortzusetzen:

1. **Weiter**: Hier wird die Bearbeitung des Programms normal wieder aufgenommen, bis der nächste Breakpoint erreicht wird. Anstelle dieses Buttons können Sie auch  verwenden.

2. **1 Netz**: Hier wird ein Netz bearbeitet und gestoppt, nachdem die letzte Zeile bearbeitet worden ist. Das ist in AWL etwas irritierend, da man erwartet, dass die Bearbeitung bis zum Anfang des nächsten Netzes fortgesetzt wird. Ein FUP- oder KOP-Netzwerk kann jedoch im Beobachten-Status nur sinnvoll dargestellt werden, nachdem alle Anweisungen bearbeitet wurden. Wenn das Netzwerk einen Bausteinaufruf enthält, wird die Bearbeitung vorzeitig gestoppt, damit Sie die Möglichkeit haben mittels “Trace” (s.u.) die Bearbeitung im aufgerufenen Baustein zu verfolgen.

3. **1 Step**: Hier wird genau eine Programmzeile bearbeitet. Wenn die Programmzeile ein Bausteinaufruf ist, wird dieser Baustein vollständig bearbeitet, aber nicht angezeigt. Dieser Button ist in FUP/KOP nicht sehr nützlich.

4. **Trace**: Auch hier wird das Programm Zeile für Zeile bearbeitet, bei Bausteinaufrufen wird jedoch die Bearbeitung auch im aufgerufenen Baustein Zeile für Zeile vorgenommen.

Wenn Sie auf  klicken, wird das Programm bis zum Ende des aktuellen Zyklus bearbeitet und die Simulation gestoppt. Wird während der Bearbeitung ein weiterer Breakpoint angetroffen, wird dieser ignoriert.

Wenn während der schrittweisen Programmbearbeitung das Ende des Zyklus erreicht wird, wird einmal die Anlagensimulation bearbeitet. Sie erkennen dies an dem Fortschreiten der Simulationszeit im Hauptfenster unten links und an dem kurzen Aufblitzen der grünen LED.

Sie **löschen** einen Breakpoint genauso, wie Sie ihn gesetzt haben: Rechtsklicken und Menüpunkt “Breakpoint” wählen.

Sie können auch Bedingungen für den Breakpoint angeben: wählen Sie dafür nach Rechtsklick: “Edit Breakpoint”.

Wenn Sie während der schrittweisen Bearbeitung einen Baustein in die SPS übertragen wollen (auch zum Ein-/Ausschalten des Beobachtenmodus ist eine Übertragung notwendig), werden Sie darauf hingewiesen, dass dies zurzeit nicht möglich ist, dass der Baustein aber am Ende des Zyklus automatisch übertragen werden wird.

4.7.1 Setzen / Löschen eines Breakpoints in AWL

Sie setzen einen Breakpoint, indem Sie im AWL-Editor auf die gewünschte Zeile mit der rechten Maustaste klicken und aus dem Kontext-Menü “Breakpoint” wählen. Die entsprechende Zeile wird dann blau gefärbt.

Wenn die Bearbeitung des SPS-Programms an diesen Punkt gelangt, wird der Baustein geöffnet und das CPU-Fenster erscheint. Die Bearbeitung des Programms wird unterbrochen, **bevor** die markierte Zeile bearbeitet wird.

Sie editieren einen bereits gesetzten Breakpoint, indem Sie aus dem Kontextmenü “Edit Breakpoint” wählen.

Sie löschen einen Breakpoint, indem Sie aus dem Kontextmenü erneut die nun mit einem Häkchen versehene Zeile “Breakpoint” wählen.

Wenn Sie den Breakpoint in FUP/KOP gesetzt haben, befindet er sich (unsichtbar) unterhalb der letzten Zeile des Netzwerkes.

4.7.2 Setzen / Löschen eines Breakpoints in FUP / KOP

Sie setzen einen Breakpoint, indem Sie im FUP- oder KOP-Editor auf das gewünschte Netzwerk mit der rechten Maustaste klicken und aus dem Kontext-Menü “Breakpoint” wählen. Ein gesetzter Breakpoint wird durch ein blaues Kästchen oben links vom Netzwerk angezeigt.

Wenn die Bearbeitung des SPS-Programms an diesen Punkt gelangt, wird der Baustein geöffnet und das CPU-Fenster erscheint. Die Bearbeitung des Programms wird unterbrochen, **nachdem** das Netzwerk bearbeitet worden ist.

Sie editieren einen bereits gesetzten Breakpoint, indem Sie aus dem Kontextmenü “Edit Breakpoint” wählen.

Sie löschen einen Breakpoint, indem Sie aus dem Kontextmenü erneut die nun mit einem Häkchen versehene Zeile “Breakpoint” wählen.

Wenn Sie das Netzwerk später in AWL darstellen, befindet sich der Breakpoint (unsichtbar) unterhalb der letzten Zeile des Netzwerkes.

4.7.3 Bedingte Breakpoints

Wenn das Programm nur unter bestimmten Bedingungen gestoppt werden soll, können Sie einen bedingten Breakpoint setzen oder einen bereits gesetzten Breakpoint nachträglich mit Bedingungen versehen. Klicken Sie dazu im AWL-, FUP- oder KOP - Editor mit rechts auf den Breakpoint und wählen Sie “Edit Breakpoint”.

Beachten Sie, dass die Bedingung in AWL vor Bearbeitung der aktuellen Zeile

ausgewertet wird, in FUP/KOP jedoch nach Bearbeitung des gesamten Netzwerkes (siehe auch Besonderheiten in FUP/KOP, ein Stück weiter unten).

Das VKE können Sie folgendermaßen auswerten:
1. Es soll nicht berücksichtigt werden, d.h. es wird immer gestoppt, wenn keine Bedingung für den Akku vorgegeben wird.
2. Es wird nur gehalten, wenn das VKE “1” ist.
3. Es wird nur gehalten, wenn das VKE “0” ist.
4. Es wird nur bei einer steigenden Flanke des VKE gehalten.
5. Es wird nur bei einer fallenden Flanke des VKE gehalten.
6. Es wird bei jedem Zustandswechsel des VKE gehalten.
7. Es wird nur gehalten, wenn das VKE keinen Zustandswechsel macht.

Als Referenzwert für 4.) - 7.) gilt immer das VKE, wie es bei der letzten Bearbeitung der Zeile mit dem Breakpoint war, “Steigende Flanke” bedeutet also nicht, dass das VKE jetzt gerade von Null nach Eins geht, sondern dass es im letzten Zyklus Null war und jetzt Eins ist.

Den Akku 1 können Sie mit einem Referenzwert vergleichen. Dazu müssen Sie angeben, wie der Inhalt des Akkus interpretiert werden soll: als INT, DINT oder REAL.

4.7.4 Besonderheiten in FUP / KOP

Für einen gezielten Einsatz von bedingten Breakpoints in FUP / KOP müssen ein paar Punkte beachtet werden:

Es werden das VKE und der Akku 1 ausgewertet, wie sie nach Bearbeitung der letzten Zeile vorgefunden werden. D.h. nur die letzte, das VKE bzw. den Akku verändernde Operation entscheidet, ob die Bedingung erfüllt ist. Bei Unklarheiten ist es unbedingt empfehlenswert, sich das Netzwerk kurz in AWL anzeigen zu lassen.

Wenn Sie ein nicht abgefragtes SR/RS-Flipflop programmiert haben, entscheidet der Zustand des unteren Eingangs über das VKE. Meistens ist man aber mehr an dem Zustand des Flipflops selber interessiert. In diesem Fall sollte eine Zuweisung am Q-Ausgang programmiert werden. Sinngemäß gilt dies für alle Elemente mit Q-Ausgang.

Wenn Sie einen Zustand abfragen wollen, der am Ende des Netzwerkes nicht mehr im VKE ist, z.B. den oberen Eingang eines SR/RS-Flipflops, so müssen Sie sich das Netzwerk in AWL anzeigen lassen und den Breakpoint an der entsprechenden Stelle setzen. Sie können dann wieder nach FUP/KOP umschalten, müssen jedoch beachten, dass im Beobachten-Status nur die Signale richtig angezeigt werden, die vor dem Erreichen des Breakpoints bearbeitet worden sind. Klicken Sie dann auf “1 Netz” im CPU-Fenster, dann wird die Bearbeitung bis zum Ende des Netzwerkes fortgesetzt und alle Signale werden richtig angezeigt.

4.8 Umverdrahten

Auf zwei Arten können Operanden umverdrahtet werden:

1. Über die Adressentabelle
Wenn dort eine Adresse geändert wird, werden nach Rückfrage alle entsprechenden Adressen im Programm automatisch geändert. Durch wählen der Check-Box “Diese Frage nicht mehr stellen” kann die Rückfrage unterdrückt werden. Unter Ansicht|Optionen|SPS-Editor|Umverdrahten kann die Rückfrage wieder aktiviert werden.

2. Über den Menüpunkt **SPS|Umverdrahten**
Dieser Punkt ist nur anwählbar, wenn alle SPS-Fenster geschlossen sind. Wenn hiermit ein Operand umverdrahtet wird, betrifft dies nur die SPS, nicht die Anlage.

Beim Umverdrahten muss unbedingt darauf geachtet werden, dass die neue Adresse nicht bereits im Programm verwendet wird, da sonst zwei verschiedene Adressen zu einer zusammengelegt werden. Ob eine Adresse bereits verwendet wird, kann mittels der Querverweisliste überprüft werden.

Denken Sie auch daran, die Symboltabelle anzupassen. Im Allgemeinen ist es günstiger, vor Beginn des Programmierens die Ein- und Ausgänge dem Schaltplan entsprechend festzulegen.

4.9 Querverweisliste

In der Querverweisliste werden alle Operanden und das zugehörige Symbol mit dem Ort ihres Auftretens im Programm und der mit ihnen durchgeführten Operation aufgeführt. Durch einen Doppelklick auf eine Zeile oder mit “Enter” springt der Editor zu dem entsprechenden Ort. Wenn Sie erkennen, dass dies nicht die von Ihnen gesuchte Programmstelle ist, können Sie den Baustein mit einem einfachen **ESC**-Tastendruck wieder schließen.
Die Querverweisliste ist immer nach Operanden sortiert, Sie haben keinen Einfluss auf die Sortierungsreihenfolge.
Die roten Punkte kennzeichnen die Programmstellen, an denen der Operand **verändert** wird. Bei der Fehlersuche sind diese Stellen häufig am Interessantesten.
Auf fünf Wegen gelangen Sie zur Querverweisliste:

1. Über Menüpunkt **SPS|Querverweise**
2. Indem Sie bei geöffnetem Baustein auf die Statusleiste (unten) des Editorfensters doppelklicken. Wenn der Cursor auf einem Operanden steht, wird automatisch der entsprechende Bereich in der Querverweisliste angezeigt.
3. Klicken Sie in der Adressentabelle rechts und wählen aus dem Kontextmenü “Querverweise”.
4. Wenn Sie die Editiermaske eines Elementes mit SPS-Anschluss geöffnet haben, sehen Sie im Interface-Panel das Symbol 

. Ein Klick und Sie erkennen sofort, an welchen Stellen im Programm das Element bearbeitet wird.
5. Sie klicken in AWL, KOP oder FUP mit rechts auf den Operanden und wählen “Querverweise” aus dem Kontextmenü.

Die Querverweisliste kann mittels **Projekt|Drucken** ausgedruckt werden.

Über das Kontextmenü gelangen Sie auch zur Symboltabelle und können die Elemente, die an den Operanden angeschlossen sind, in der Adressentabelle überprüfen.

4.10 IEC-Funktionen

(noch in Entwicklung begriffen)

Im TrySim - Hauptverzeichnis gibt es ein Unterverzeichnis “IECFuncs”. Darin sind vorgefertigte Funktionen enthalten, die zum Manipulieren, Vergleichen und Konvertieren von STRINGS, Datumsformaten usw. benötigt werden. Daneben sind auch die Köpfe der Funktionen enthalten, die in der Siemens-Bibliothek “TI - S7 Converting Blocks” sind.

Diese Funktionen müssen Sie mit **Projekt|Dateien hinzufügen** in Ihr Projekt importieren und ggfs. umbenennen, wenn die FC-Nummern mit selbst programmierten FC kollidieren.

Bislang sind erst wenige dieser Funktionen tatsächlich ausgeführt, von den Meisten ist lediglich die Deklaration vorhanden. Wenn Sie einige dieser Funktionen benötigen, fragen Sie uns bitte, ob wir sie bereits fertiggestellt haben.

Bereits erstellte IEC-Funktionen:

SFB3 (TP) Zeit als Impuls
SFB4 (TON) Einschaltverzögerung
SFB5 (TOF) Ausschaltverzögerung

Bereits erstelle TI-S7 Funktionen
FC84 (ATT) Eingabe für FIFO / LIFO
FC85 (FIFO) First In -> First Out
FC87 (LIFO) Last In -> First Out

4.10.1 SFB 3 (TP) Zeit als Impuls

Dieser IEC-Baustein entspricht im Wesentlichen der S7-Zeit-als-Impuls. Er muss jedoch mit T#.. beschickt werden, nicht mit S5T#.. . Außerdem läuft die Zeit am Ausgang ET vorwärts, nicht rückwärts.

4.10.2 SFB 4 (TON) Einschaltverzögerung

Dieser IEC-Baustein entspricht im Wesentlichen der S7-Einschaltverzögerung. Er muss jedoch mit T#.. beschickt werden, nicht mit S5T#.. . Außerdem läuft die Zeit am Ausgang ET vorwärts, nicht rückwärts.

4.10.3 SFB 5 (TOF) Ausschaltverzögerung

Dieser IEC-Baustein entspricht im Wesentlichen der S7-Ausschaltverzögerung. Er muss jedoch mit T#.. beschickt werden, nicht mit S5T#.. Außerdem läuft die Zeit am Ausgang ET vorwärts, nicht rückwärts.

4.10.4 FC 87 (LIFO) Auslesen des jüngsten Wertes einer Tabelle

Hiermit können Sie den jüngsten Wert einer Tabelle auslesen, der dieser mittels der Funktion FC 84 (ATT) hinzugefügt worden ist. Der Wert wird dadurch aus der Tabelle entfernt.

Die Werte in diesen Tabellen sind immer 16 Bit breit.

4.10.5 FC 85 (FIFO) Auslesen des ältesten Wertes einer Tabelle

Hiermit können Sie den ältesten Wert einer Tabelle auslesen, der dieser mittels der Funktion FC 84 (ATT) hinzugefügt worden ist. Der Wert wird dadurch aus der Tabelle entfernt.

Die Werte in diesen Tabellen sind immer 16 Bit breit.

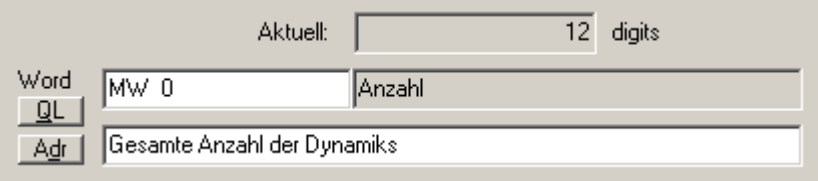
4.10.6 FC 84 (ATT) Hinzufügen eines Wertes zu einer FIFO / LIFO Tabelle

Hiermit können Sie einen Wert zu einer Tabelle hinzufügen, der danach mit den Funktionen FC85 (First In First Out) oder FC 87 (Last In First Out) wieder ausgelesen (und entfernt) werden kann.

Die Werte in diesen Tabellen sind immer 16 Bit breit.

4.11 Interfacepanel

So heißen in TrySim die in den Editiermasken der Elemente vorkommenden Gruppen, mit denen die Anbindung an die SPS hergestellt wird.



In das kleine weiße Feld kann der Operand entweder in absoluter Schreibweise oder als Symbol eingetragen werden. Namen aus dem Deklarationsteil von Bausteinen, die mit "#" anfangen, sind nicht erlaubt (Ausnahme bei Script). E, A und M sowie der Punkt können auch über den Nummern-Block eingegeben werden.

Es gibt Interface-Panels für Bits-, Wordund DWords, jedes Panel akzeptiert nur Operanden, die (zumindest in der Größe) seinem Typ entsprechen.

Wenn ein Symbol für den Operanden vorhanden ist, wird es im Symbol-Feld angezeigt. Falls der Operand aus dem E- A- M-Bereich ist, gelangen Sie über einen Doppelklick auf das Symbol-Feld zur Deklaration in der Symboltabelle und können das Symbol dort ggfs. anpassen.

Im großen weißen Feld wird der Kommentar aus der Symboltabelle angezeigt. Sie können ihn hier auch editieren. Wenn Sie den Operanden bislang noch nicht in die Symboltabelle eingetragen haben, geschieht dies automatisch durch die Eingabe eines Kommentars. Sie können im Interface-Panel keine neuen Symbole zuordnen oder bestehende verändern, dazu müssen Sie die Symboltabelle öffnen.

Mit dem Knopf QL springen Sie zur Querverweisliste, in der alle Verwendungen des Operanden im SPS-Programm angezeigt werden. Aus der QL gelangen Sie mit einem Doppelklick oder über das Kontextmenü zur gewünschten Programmstelle.

Mit dem Knopf Adr springen Sie zur Adressentabelle, in der alle weiteren Verwendungen dieser Adresse in der Anlage aufgeführt sind. Schauen Sie sich auch dort die Punkte des Kontextmenüs an.

Der aktuelle Zustand eines Bits wird durch das kleine Quadrat angezeigt. Durch einen Klick können Sie den Zustand umschalten. Das hat natürlich nur Wirkung, wenn das Bit nicht woanders (in der Anlagen-Simulation oder im SPS-Programm) wieder neu beschrieben wird. Trotzdem kann diese Funktion gerade beim Erstellen einer Simulation sehr nützlich sein. Entsprechendes gilt für das Feld "Aktuell:" Durch einen Klick gelangen Sie in den Editiermodus und können den angegebenen Wert überschreiben.

4.12 Symboltabelle

SPS|Symboltabelle

Abkürzung: 

In der Symboltabelle können Sie den absoluten Operanden, ein Symbol, einen Typ und einen Kommentar zuordnen. Bitte verwechseln Sie die Symboltabelle nicht mit der Adressentabelle, dort sind alle von der Anlage verwendeten SPS-Adressen zusammengefasst.

Sie können die Symboltabelle sowohl nach Absolutoperanden als auch nach Symbolen sortieren. Klicken Sie dazu in der Beschriftungszeile der Tabelle auf die entsprechende Spaltenüberschrift. Wenn Sie mit links klicken, wird automatisch aufsteigend sortiert. Wenn Sie mit rechts klicken, können Sie wählen, ob auf- oder absteigend sortiert wird.

Wenn Sie die Symboltabelle ausdrucken wollen, wählen Sie **Projekt|Drucken** während die Tabelle aktiv ist. Die Tabelle wird so gedruckt, wie sie zuletzt sortiert worden ist.

Sie können die Symboltabelle auch (gleich an der richtigen Stelle) öffnen, wenn Sie im FUP/KOP/AWL-Editor aus dem Kontextmenü “Symboltabelle” wählen.

Mithilfe des Kontextmenüs der Symboltabelle gelangen Sie direkt zu den Querverweisen und zur Adressentabelle.

In der Spalte für den Operanden können “E”, “A” und “M” sowie der Punkt auch über den Nummern-Block eingegeben werden.

4.13 Systemfunktionsbausteine und -funktionen

Diese Bausteine erfüllen vielfältige Spezialaufgaben, wie STRING und Datumsverarbeitung, Kommunikation usw. Sie werden von STEP®7 nicht exportiert.

Die meisten Systemfunktionsbausteine und -funktionen sind in TrySim nicht integriert, d.h. gegebenenfalls müssen Sie sie selbst so programmieren, dass der Rest Ihres Programmes funktioniert. Bevor Sie dies tun, sollten Sie jedoch bei uns nachfragen, ob die von Ihnen benötigten Bausteine nicht bereits von uns programmiert worden sind.

Im TrySim - Hauptverzeichnis gibt es ein Unterverzeichnis “Sonderbausteine”. Hierein sollten Sie alle von Ihnen geschriebenen oder von uns erhaltenen Sonderbausteine kopieren, sie werden dann automatisch bei Verwendung in das aktuelle Projekt übernommen.

Bislang programmiert sind:

SFC 0	Stellen der CPU-Uhr
SFC 1	Auslesen der CPU-Uhr
SFC 20	Kopieren von Speicherbereichen
SFC 21	Füllen von Speicherbereichen mit einem Muster
SFC 22	Erzeugen eines neuen DB

SFC 23 Löschen eines DB
SFC 24 Ermitteln der Länge eines DB
SFC 64 Auslesen der Systemzeit in ms

STEP®7 ist eingetragenes Warenzeichen der Siemens AG.

4.13.1 SFC 0 Stellen der CPU-Uhr

Mit dieser Systemfunktion kann die Uhr der CPU gestellt werden. Dazu müssen Sie zunächst eine lokale DATE_AND_TIME - Variable deklarieren und diese danach mit der IEC-Funktion FC3 auf das gewünschte Datum und die Zeit setzen.

Das Auslesen der Uhr kann mit der SFC 1 geschehen.

4.13.2 SFC 1 Auslesen der CPU-Uhr

Mit dieser Systemfunktion kann die Uhr der CPU ausgelesen werden. Dazu müssen Sie zunächst eine lokale DATE_AND_TIME - Variable deklarieren und diese danach auswerten.

Das Stellen der Uhr kann mit der SFC 0 geschehen.

4.13.3 SFC 20 Block Move

Mit dieser Systemfunktion können Sie den Inhalt eines beliebigen Speicherbereichs in einen anderen kopieren. Sowohl der Quellbereich, als auch der Zielbereich werden als ANY angegeben. Übertragen wird nur die Anzahl von Bytes, die sowohl im Quellbereich als auch im Zielbereich vorhanden ist.
Der Rückgabewert RET_VAL ist, anders als in einer S7, immer “0”.

Falls sie Quell- oder Zielbereich explizit durch eine als ANY deklarierte Variable angeben und Quell- oder Zielbereich in den Lokaldaten liegt, müssen Sie darauf achten, dass die Kennung $87 im Any angegeben wird, nicht $86, denn dies wird von der SFC20 als ihre eigenen Lokaldaten interpretiert und die gibt es nicht.

4.13.4 SFC 21 Fill

Mit dieser Systemfunktion können Sie einen beliebigen Speicherbereich mit einem Muster füllen. Sowohl der Zielbereich als auch das Muster werden als ANY angegeben. Das Muster wird so oft in den Zielbereich kopiert, bis dieser ganz beschrieben ist. Falls die Größe des Zielbereiches kein ganzzahliges Vielfaches der Größe des Musters ist, wird die letzte Kopie des Musters entsprechend abgeschnitten.
Der Rückgabewert RET_VAL ist, anders als in einer S7, immer “0”.

Falls sie Muster- oder Zielbereich explizit durch eine als ANY deklarierte Variable angeben und Muster- oder Zielbereich in den Lokaldaten liegt, müssen Sie darauf

achten, dass die Kennung $87 im Any angegeben wird, nicht $86, denn dies wird von der SFC20 als ihre eigenen Lokaldaten interpretiert und die gibt es nicht.

4.13.5 SFC 22 Create DB

Mit dieser Systemfunktion können Sie während der Laufzeit einen DB erzeugen. Den Bereich der Nummer des neuen DB legen Sie mit den Parametern LOW_LIMIT und UP_LIMIT fest. Die Nummer des tatsächlich erzeugten DB wird über den Parameter DB_NUMBER zurückgegeben. Mittels des Parameters COUNT, der gerade sein muss, bestimmen Sie die Größe des zu erzeugenden DB.
Der Out-Parameter RET_VAL kann folgende Werte annehmen:
W#16#0000: keine Fehler
W#16#80A1: LOW_LIMIT = 0 oder größer als UP_LIMIT
W#16#80A2: COUNT = 0 oder ungerade
W#16#80B1: Kein DB erzeugt, weil keine Nummer frei
Die anderen in einer S7 erkannten Fehlerfälle werden nicht erkannt.

4.13.6 SFC 23 Delete DB

Mit dieser Systemfunktion können Sie während der Laufzeit einen DB löschen. Die Nummer des zu löschenden DB legen Sie mit dem Parameter DB_NUMBER fest. Achten Sie darauf, dass der DB nicht geöffnet ist. Er darf auch nicht in einem der aufrufenden Bausteinen offen gewesen sein, als der Aufruf des aktuell bearbeiteten Bausteins erfolgte.

Der Out-Parameter RET_VAL kann folgende Werte annehmen:
W#16#0000: keine Fehler
W#16#80B1: der DB existiert nicht
Die anderen in einer S7 erkannten Fehlerfälle werden nicht erkannt.

4.13.7 SFC 24 Info über DB

Mit dieser Systemfunktion können Sie während der Laufzeit die Länge eines DBs erkunden. Die Nummer des DB legen Sie mit dem Parameter DB_NUMBER fest. Der DB muss in der SPS vorhanden sein, es reicht nicht, wenn er nur auf der Festplatte ist. Der DB wird automatisch in die SPS übertragen, wenn er entweder im Editor einmal geöffnet worden ist oder wenn er vom SPS-Programm aufgeschlagen worden ist. Die Länge wird (auf die nächste gerade Zahl aufgerundet) über den Parameter DB_LENGTH zurückgegeben. Der Rückgabewert WRITE_PROT ist, anders als in einer S7, immer Null, da in TrySim DB nicht als schreibgeschützt deklariert werden können.

Der Out-Parameter RET_VAL kann folgende Werte annehmen:
W#16#0000: keine Fehler

W#16#80B1: der DB existiert nicht
Die anderen in einer S7 erkannten Fehlerfälle werden nicht erkannt.

Die Länge des aktuell geöffneten DB oder des aktuellen Instanz-DB können Sie auch mit DBLG oder DILG erfahren.

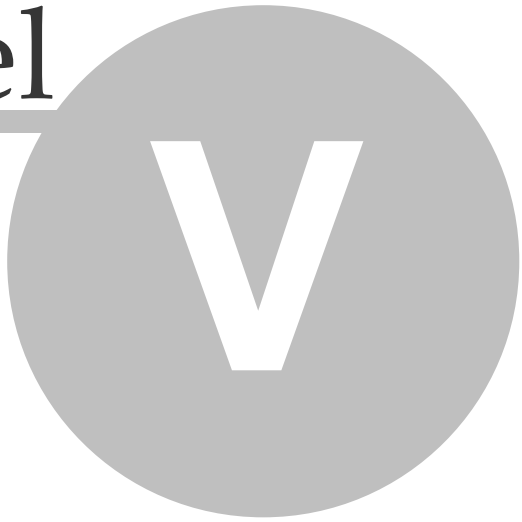
4.13.8 SFC 64 Auslesen der Systemzeit in ms

Diese Systemfunktion hat nur einen out-TIME-Parameter, an dem die Systemzeit in ms ausgegeben wird.
Beachten Sie, dass in einer S7 diese Zeit nach ca. 48 Tagen wieder von Null zu laufen anfängt, sodass es bei Differenzbildung zu unerwarteten Ergebnissen kommen kann.
TrySim kann z.Z. nur 24 Tage simulieren, sodass Sie diesen Effekt mit TrySim nicht testen können.

S7-300 und S7-400 sind eingetragene Warenzeichen der Siemens AG.

Export und Import

Kapitel



5 Export und Import

5.1 Übersicht

TrySim ist eine reine Offline-Software. Wenn Sie ein mit TrySim erstelltes Programm auf einer echten SPS laufen lassen wollen, geht dies nur mittels des Original-Entwicklungssystems des Herstellers. Bislang unterstützt TrySim nur die S7-300/400-Reihe von Siemens.

Export nach STEP®7
Sie können das Programm und die Symboltabelle nach STEP®7 exportieren, dort weiter bearbeiten und schließlich in eine echte SPS übertragen.

Import von STEP®7
Sie können ein Programm und die Symboltabelle, die Sie in STEP ®7 geschrieben haben, nach TrySim importieren und dort testen. Wenn Sie Änderungen vornehmen, können Sie es anschließend nach STEP ®7 zurückübertragen.

Import von STEP®5
Der sowieso sehr eingeschränkte Import von STEP ®5 ist nicht mehr möglich.

STEP®7, STEP®5 S7-300 und S7-400 sind eingetragene Warenzeichen der Siemens AG.

5.2 Export des Programms nach STEP®7

Zum Export des Programms müssen Sie eine Quelle generieren. >Quelle< nennt man eine Text-Datei, die das Programm enthält. Eine Quelle können Sie mit jedem Text-Editor bearbeiten.

- Wählen Sie **Projekt|Exportieren|Bausteine.**
- Bewegen Sie in der Auswahlmaske die zu exportierenden Bausteine mit den Pfeil-Buttons in die rechte Liste. Sie können auch alle Bausteine zugleich bewegen. Über die Reihenfolge der Bausteine brauchen Sie sich keine Gedanken zu machen, sie wird automatisch bestimmt. Sonderfunktionsbausteine und –funktionen sollten Sie nicht exportieren.
  Die Dateien im Exportfilter (Anlage|Export|Edit Export Filter) können zwar nach rechts bewegt werden, von ihnen wird aber keine Quelle erzeugt.
- Wählen Sie “Gesamt-Datei”. Hierdurch erreichen Sie, dass alle Bausteine in eine Quelle geschrieben werden. Diese Datei erhält den Namen Ihres Projektes mit der Erweiterung “AWL”. Wenn Sie “Einzeldateien” anwählen, wird für jeden Baustein eine eigene Quelle angelegt, der den Namen des Bausteins mit der Erweiterung “.AWL” erhält.
  Die “*.AWL”-Dateien werden in dem Verzeichnis Ihres Projektes angelegt. Falls sich dort bereits Dateien mit dem gleichen Namen befinden, werden diese ohne Warnung überschrieben.

- Starten Sie STEP ®7 und legen ein neues Projekt an. Welche Schritte dazu nötig sind, ist in der Dokumentation zu STEP ®7 detailliert beschrieben. Öffnen Sie den Behälter **Projekt|Station|CPU|S7-Programm|Quellen**.
- Wählen Sie den Menüpunkt **Einfügen|Externe Quelle**.
- Jetzt müssen Sie die zuvor erzeugte Quelldatei finden. Der komplette Pfad heißt (wenn Sie die Vorgaben bei der Installation akzeptiert haben) C:\Programme\TrySim\Projekte*IhrProjekt*.
- Nach dem Einfügen finden Sie in dem Behälter die neue Quelle, die wie Ihr Projekt heißt. Öffnen Sie diese Quelle durch einen Doppelklick.
- Wählen Sie den Menüpunkt **Datei|Übersetzen** oder **Einfügen|Übersetzen**. Falls sich bereits Bausteine mit den gleichen Namen wie die Importierten im STEP ®7-Projekt befinden, werden sie ohne Warnung überschrieben.
- Wenn alles gut gegangen ist, steht jetzt im unteren Fenster die Meldung "Compilerergebnis: 0 Fehler, 0 Warnungen" und Ihr Programm ist erfolgreich nach STEP ®7 exportiert. Falls Sie Warnungen erhalten haben, beziehen diese sich zumeist auf die Verwendung von Datenwörtern und sind in aller Regel ungefährlich. Falls Sie Fehlermeldungen erhalten haben, ist die Übersetzung nicht erfolgreich gewesen, bitte lesen Sie dazu den Abschnitt Probleme beim Export.

5.3 Export der Symboltabelle

Für den Export der Symboltabelle sind in TrySim keine besonderen Schritte notwendig, da TrySim die Tabelle in einem Format speichert, das von STEP ®7 unmittelbar gelesen werden kann. Sie sollten jedoch das Projekt speichern, damit auch Ihre letzten Änderungen an der Symboltabelle übernommen werden.

- Starten Sie STEP ®7. Wenn Sie noch kein Projekt angelegt haben, müssen Sie dies jetzt tun.
- Öffnen Sie die Symboltabelle im Behälter **Projekt|Station|CPU|S7-Programm**.

Die Zeilen der TrySim-Symboltabelle werden zu den bereits Vorhandenen in der STEP ®7 – Tabelle hinzugefügt werden. Falls Adressen oder Symbole dadurch doppelt vorkommen, werden diese fett markiert und Sie müssen sie einzeln per Hand löschen oder umbenennen. Falls Ihre Liste in TrySim alle benötigten Symbole enthält, sollte Sie daher vorm Importieren alle Zeilen in der STEP ®7 – Tabelle löschen. Wählen Sie **Bearbeiten|Markieren|Alle** und betätigen anschließend die Taste "Entf".

- Wählen Sie den Menüpunkt **Tabelle|Importieren**.
- Wählen Sie den Dateityp "ASCII – Format (*.ASC)"
- Die Symboltabelle von TrySim heißt "SYMLIST.ASC" und liegt im Projektverzeichnis. Der komplette Pfad heißt (wenn Sie die Vorgaben bei der Installation akzeptiert haben) C:\Programme\TrySim\Projekte*IhrProjekt*.
- Wählen Sie die Datei und klicken auf "Öffnen"
- Jetzt sollten alle Zeilen der TrySim-Symboltabelle in der Tabelle erscheinen.

5.4 Probleme beim Export

Falls Sie während des Übersetzens im STEP ®7 Quelleneditor Fehlermeldungen erhalten haben, werden dies in den meisten Fällen Typkonflikte sein. Fehler treten auch auf, wenn Sie rekursiv programmiert haben, d.h., dass ein Baustein sich direkt oder indirekt selbst aufruft. Dieser Fehler ist nicht zu beheben, Rekursion ist in AWL-Quellen nicht erlaubt. Schließlich kann auch ein Fehler in TrySim selbst vorliegen. Gelegentlich bemängelt STEP ®7 auch Typkonflikte, die gar nicht vorhanden sind.

In jedem Fall sollten Sie beim Übersetzen der Quelle unter **Extras|Einstellung| Quellen** die Checkbox "Bausteine nur bei fehlerfreier Übersetzung erzeugen" abwählen. Häufig lässt sich die Quelle dann beim zweiten oder dritten Versuch übersetzen, da immer mehr von den aufgerufenen Bausteine vorhanden sind. Dieses Mehrfach-Übersetzen ist nur notwendig, wenn Sie ein Programm das erste Mal nach STEP ®7 exportieren. Wenn Sie später in TrySim Änderungen vornehmen und das Programm erneut exportieren, sind alle Bausteine bereits vorhanden und es wird keine Fehlermeldungen geben. Häufig ist es auch hilfreich, zuerst die Symboltabelle in STEP ®7 zu importieren, da einige Anweisungen nur symbolisch dargestellt werden können.

Typkonflikte
Diese resultieren daraus, dass in TrySim die Datentyp-Prüfung nicht so streng ist, wie in STEP ®7. In TrySim können Sie an Funktionsbausteine alle Aktualparameter anschließen, die die gleiche Größe haben wie der Formalparameter. STEP ®7 jedoch akzeptiert nur die Aktualparameter, die exakt den gleichen Typ haben wie der Formalparameter. So ist die Übergabe einer als INT deklarierten Variablen an einen als WORD deklarierten Parameter nicht zulässig, obwohl beide 2 Bytes groß sind. Wenn der STEP ®7 – Compiler daher Typkonflikte meldet, müssen Sie in TrySim die Typen überprüfen und entsprechend anpassen. Dies hat nebenbei den Vorteil, dass die Akku-Darstellung beim Beobachten von AWL-Bausteinen besser gewählt werden kann.
Wenn STEP ®7 Typkonflikte bemängelt, obwohl die Typen exakt identisch sind, empfiehlt es sich, diesen Baustein separat zu übersetzen. Zumindest bei ARRAY-Parametern hilft dies in manchen Fällen.

Rekursive Aufrufe
Wenn Sie einen FB aufrufen, müssen Sie den Instanz-DB angeben. Dieser wird entsprechend des FB-Kopfes erstellt und muss daher nach der FB-Deklaration in der Quelle stehen. Da der Quellen-Compiler von STEP ®7 den Instanz-DB nicht automatisch erzeugt (wie normalerweise im Editor), ist der Instanz-DB noch nicht vorhanden, wenn ein FB sich selbst aufruft. Rekursive Aufrufe sind daher in AWL-Quellen nicht übersetzbar. Vermeiden Sie daher, dass ein Baustein sich selbst aufruft, was nach IEC 61131-3 auch gar nicht zulässig ist.

Fehler in TrySim
Schließlich ist es leider auch möglich, dass uns ein Programmierfehler unterlaufen ist und nicht die richtige Syntax in der Quelle erzeugt worden ist. Wenn Sie die Quellensyntax beherrschen, können Sie den Fehler in der Quelle beheben, wenn nicht, bleibt Ihnen nichts anders übrig, als die fehlerhafte Stelle im Quelleneditor zu

finden (Doppelklick auf die Fehlermeldung), den entsprechenden Baustein zu identifizieren und die Quelle in TrySim erneut zu generieren, jedoch ohne den fehlerhaften Baustein. Diesen müssen Sie dann vor dem Übersetzen in STEP ®7 per Hand eingeben.
Wenn Ihnen ein solcher Fehler unterkommt, benachrichtigen Sie uns bitte, damit wir ihn in der nächsten Ausgabe des Programms beseitigen können.

5.5 Import des Programms von STEP®7

Zum Import des Programmes müssen Sie zunächst in STEP ®7 eine Quelle generieren. Quelle nennt man eine Text-Datei, die das Programm enthält. Eine Quelle können Sie mit jedem Texteditor bearbeiten.

- Starten Sie STEP ®7.
- Öffnen Sie das Projekt.
- Starten Sie den KOP/AWL/FUP-Editor, z.B., indem Sie einen beliebigen Baustein öffnen. Sie müssen den Baustein dann wieder schließen, da von offenen Bausteinen keine Quelle erzeugt werden kann.
- Wählen Sie den Menüpunkt **Datei|Quelle generieren**.
- Geben Sie in die Eingabemaske irgendeinen Namen für die zu erzeugende Quelle an.

Wählen Sie die nach TrySim zu importierenden Bausteine aus; über die Reihenfolge brauchen Sie sich keine Gedanken zu machen. (Siehe auch: Import-Filter)

- Wählen Sie “Operanden: Absolut”.
- Bestätigen Sie mit OK.
- Starten Sie TrySim.
- Öffnen Sie das Projekt.
- Wählen Sie **Projekt|Importieren|Bausteine|S7-AWL-Quelle.**
- Jetzt müssen Sie das STEP ®7 Projekt finden. Vermutlich ist es unter C:\Step7_Vx\S7proj (Doppelklick auf einen Ordner öffnet ihn).
- Öffnen Sie den Ordner mit dem Namen Ihres Projektes.
- TrySim öffnet dann automatisch den Ordner S7ASRCOM. Falls das aus irgendeinem Grund nicht geschieht, müssen Sie es selbst machen.
- In diesem Ordner sind ein oder mehrere Ordner, die 00000001, 00000002 usw. heißen. In einem von diesen Ordnern ist Ihre AWL-Quelle.
- Wenn Sie den 0000000x - Ordner geöffnet haben, werden in dem rechten Fenster die verfügbaren AWL-Quellen angezeigt. Öffnen Sie die Gewünschte mit Doppelklick oder markieren Sie sie und bestätigen Sie mit OK. (Anmerkung.: Die Quellen sind in diesem Verzeichnis nicht unter dem Namen gespeichert, den Sie eingegeben haben, sondern sie sind entsprechend ihres Erzeugungsdatums numeriert: “0000000x.awl”, wobei x eine Hex-Zahl ist, d.h. der zehnte Eintrag heißt “0000000a.awl”, der elfte “0000000b.awl” usw. Falls TrySim die Übersetzung dieser codierten Namen in die richtigen Namen nicht gelingt, wählen Sie die Datei mit der größten Nummer, das ist die, die Sie gerade erzeugt hatten).
- Wenn alles gut gegangen ist, erscheint die Meldung “Import erfolgreich”, andernfalls eine Liste mit den Bausteinen, die nicht importiert werden konnten.

Siehe Abschnitt Probleme beim Import.

5.6 Import-Filter

Beim Testen von größeren Programmen kommt es vor, dass nur Teile der Anlage einer Simulation mit TrySim zugänglich sind. In diesen Fällen gibt es in Ihrem im STEP ®7-System geschriebenen Programm Bausteine, die in TrySim in einer modifizierten Form benötigt werden. Damit diese Bausteine nicht beim Import des gesamten Programms überschrieben werden, können Sie einen Import-Filter definieren, der einen Import dieser Bausteine verhindert.

Wählen Sie **Projekt|Importieren|Bausteine|Edit Import Filter**.

Ein typisches Beispiel ist eine SPS mit Zählerbaugruppe zur Positionserfassung. Hier gibt es von Siemens Standard-Bausteine, die die Kommunikation mit der Baugruppe abwickeln. Letztlich haben diese Bausteine nur die Aufgabe, die Position des bewegten Teils an einer bestimmten Stelle im Datenspeicher abzulegen. Genau dies muss der (modifizierte) Baustein auch in TrySim machen, damit der Rest des Programmes funktioniert. In diesem Fall gehen Sie so vor:

1. Importieren Sie einmal den Standard-Baustein, damit das Interface in TrySim bekannt ist.

2. Modifizieren Sie den Code im Baustein so, dass er die gewünschte Funktionalität hat. In der Regel ist nach dem Import gar kein Code vorhanden, da die Bausteine als Know-How-Protected deklariert sind.

3. Nehmen Sie den Baustein in den Import-Filter auf, damit Ihre Änderungen beim nächsten Import des Programmes nicht versehentlich überschrieben werden.

Wenn Sie Probleme mit dem Modifizieren eines solchen Bausteins haben, wenden Sie sich bitte an uns.

Wenn Sie in folgenden Projekten den geänderten Baustein ebenfalls benötigen, können Sie ihn mit **Projekt|Datei hinzufügen** in das neue Programm einbinden.

5.7 Import der Symboltabelle

Für den Import der Symboltabelle sind in TrySim keine Vorkehrungen notwendig, da STEP ®7 das von TrySim verwendete Dateiformat erzeugen kann. Wenn Sie in TrySim bereits Symbole angelegt haben, werden diese durch den Import überschrieben. Wollen Sie das vermeiden, müssen Sie die TrySim-Tabelle zunächst nach STEP ®7 exportieren, dadurch werden die beiden Listen miteinander verbunden.

- Starten Sie STEP ®7
- Öffnen Sie die Symbolik-Tabelle

- Wählen Sie **Tabelle|Exportieren**
- Wählen Sie für den Dateityp “ASCII Format (*.ASC)”
- Wählen Sie als Dateiname “Pfadname\Symlist.asc”
  Der komplette Pfad heißt (wenn Sie die Vorgaben bei der Installation akzeptiert haben) C:\Programme\TrySim\Projekte*IhrProjekt*.
- Auf die Frage, ob Sie die Datei überschreiben wollen, antworten Sie mit “Ja”
- Die Symbolik-Liste ist jetzt in TrySim verfügbar

Wenn Sie die Symboltabelle aus STEP ®7 exportieren, während TrySim bereits läuft, müssen Sie in TrySim noch **Projekt|Importieren|Symboltabelle** mit der Symboltabelle im TrySim – Projektverzeichnis ausführen, da die Symboltabelle nur beim Start des Programms gelesen wird.

5.8 Probleme beim Import

Wenn Sie beim “Quelle generieren” in STEP ®7 darauf geachtet haben, “Operanden Absolut” anzuklicken, sind Fehler beim Import meistens auf die Verwendung von Sonderfunktionsbausteinen und –funktionen zurückzuführen. Gelegentlich können auch Inkompatibilitäten mit bereits vorhanden Bausteinen die Ursache von Fehlern sein. Rekursive Bausteinaufrufe führen zu nicht behebbaren Fehlern. Einige Konstruktionen sind in dieser Version von TrySim noch nicht implementiert und verursachen beim Importieren Fehler oder Warnungen. Schließlich kann auch ein Fehler in TrySim selbst vorliegen. Der Importer von TrySim ist nur mit Quellen getestet, die vom Siemens SIMATIC ® -Manager generiert worden sind. Wenn Sie ein anderes System verwenden und es zu Fehlern kommt, mailen Sie uns bitte den fehlererzeugenden Quellcode, damit wir TrySim entsprechend korrigieren können.

Symbolische Darstellung der Operanden
Wenn Sie beim “Quelle generieren” die Voreinstellung “Operanden: Symbolisch” beibehalten haben, wird TrySim die Quelle nicht importieren können. Wiederholen Sie den Vorgang mit “Operanden: Absolut”. Dennoch sollten Sie in jedem Fall zuerst die Symboltabelle importieren.

Sonderfunktionsbausteine und –funktionen und Know-How-Protected
Diese Bausteine, die in STEP ®7 zur Ansteuerung von Funktionsbaugruppen und für besondere Aufgaben verwendet werden, werden von STEP ®7 nicht exportiert. Im Verzeichnis Sonderbausteine sind die Deklarationen aller SFB und SFC abgelegt. Sie werden automatisch in Ihr Projekt kopiert, wenn Sie sie aufrufen.
Als Know-How-Protected deklarierte Bausteine sollten Sie nicht nach TrySim exportieren. Um dies zu verhindern, können Sie den Import Filter verwenden.

Inkompatibilitäten mit vorhandenen Bausteinen
Wenn in Ihrem Projekt bereits Bausteine vorhanden sind, die eine abweichende Deklaration von den Importieren haben, kann es gelegentlich zu Importfehlern kommen. Versuchen Sie anhand des Importprotokolls, das automatisch angezeigt

wird, die fehlerverusachenden Bausteine zu identifizieren. Löschen Sie in diesen Fällen die entsprechenden Bausteine und wiederholen den Vorgang.

Rekursive Bausteinaufrufe
Sie können in STEP ®7 FB programmieren, die sich selbst, direkt, oder über einen weiteren Baustein aufrufen. Eine von einem solchen Programm erzeugte Quelle lässt sich allerdings häufig nicht zurückübersetzen. Vermeiden Sie daher Rekursionen.

Nicht implementierte Befehle und Konstruktionen
Siehe: Nicht implementierte Eigenschaften.

Fehler in TrySim
Falls keine der anderen Fehlerquellen vorliegt, ist wahrscheinlich ein Programmierfehler unsererseits die Ursache des Problems. In diesem Fall können Sie nur versuchen, die den Importfehler verursachenden Programmteile so abzuändern, dass TrySim die Quelle versteht. Bitte benachrichtigen Sie uns in solchen Fällen, damit wir Abhilfe schaffen können. Hier ist unsere e-mail-Adresse: info@cephalos.de.

5.9 Import des Programms von STEP®5

Der Import von STEP ®5 ist nicht mehr möglich.

5.10 Import von STEP®5 Zuweisungslisten

Der Import von STEP ®5 Zuweisungslisten ist eingeschränkt möglich.

- Wählen Sie: **Projekt|Importieren|Symboltabelle**
- Wählen Sie für den Dateityp: *.seq

Es werden nur die Eingänge, Ausgänge, Merker, Zeiten und Zähler importiert. Leerzeilen, Kommentarzeilen “;” und Seitenumbrüche “.PA” werden ignoriert. Der Datentyp wird entsprechend dem Operanden als BOOL, BYTE, WORD, DWORD, TIMER oder COUNTER gewählt.

Beispielsitzung

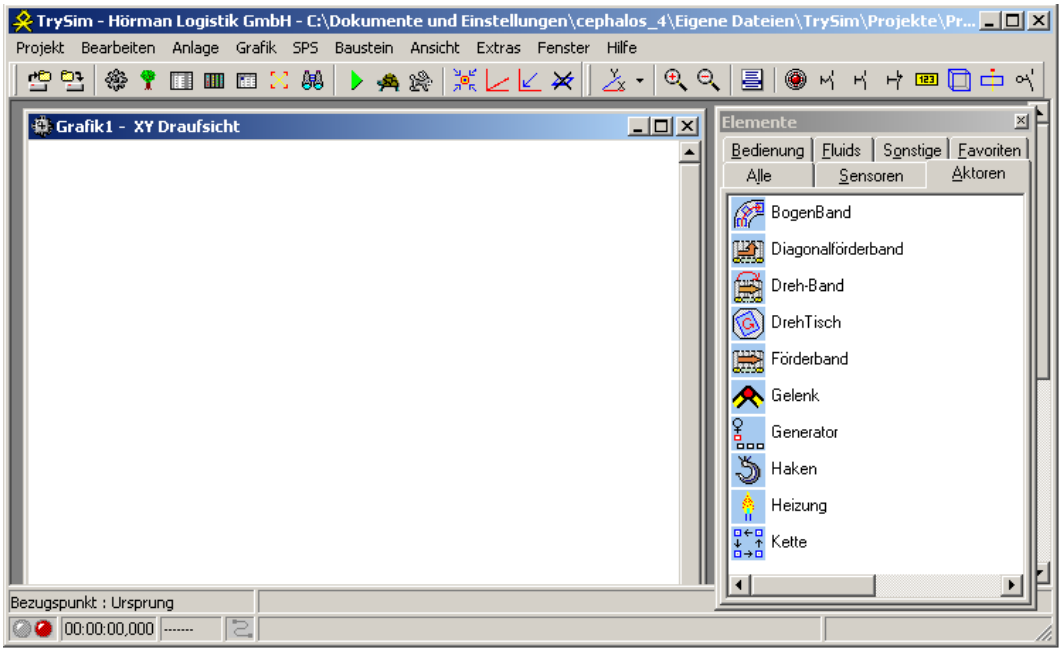
Kapitel

6 Beispielsitzung

6.1 Beispielsitzung

In dieser Beispielsitzung soll eine sehr einfache Anlage erstellt werden, um Ihnen die Grundlagen der Benutzung von TrySim näher zu bringen. Die Aufgabe besteht darin, einen Wagen zwischen zwei Endschaltern hin- und herfahren zu lassen. Wenn Sie keine Vorstellung haben, was eine SPS überhaupt ist, hilft Ihnen vielleicht die kurze Einführung für Anfänger.

Starten Sie TrySim. Wählen Sie **Projekt|Neu.** Es erscheint ein leeres Grafik-Fenster und eine Auswahlliste mit den verfügbaren Elementen.



6.2 Erstellen der Anlage

1. Linearbeweger erzeugen

Wählen Sie die Registerkarte “Aktoren” und erzeugen einen Linearbeweger, indem Sie das entsprechende Symbol mit der Maus packen (linke Maustaste gedrückt lassen). Bewegen Sie die Maus etwa in die Mitte des Grafik-Fensters und lassen Sie die Maustaste los.

2. Kasten erzeugen

Erzeugen Sie genauso einen Kasten (Registerkarte “Sonstiges”). Positionieren Sie ihn etwas oberhalb des Linearbewegers.

Jetzt müssen Sie festlegen, dass der Kasten am Linearbeweger befestigt werden soll.

Öffnen Sie dazu den Elementbaum  . Klicken Sie auf den Kasten und halten die

linke Maustaste gedrückt. Dann ziehen Sie den Kasten im Baum auf den Linearbeweger.

Öffnen Sie die Editiermaske des Linearbewegers, indem Sie mit der rechten Maustaste auf den Strich klicken. Achten Sie darauf, dass Sie erst klicken, wenn sich der Cursor in eine Hand verwandelt hat. Schieben Sie die Editiermaske so zur Seite, dass Sie Linearbeweger und Kasten sehen können.
Schieben Sie mit der Maus den Schieberegler unter dem Feld “HotSpot” etwas nach rechts. Dadurch wird der HotSpot des Linearbewegers samt Kasten nach rechts verschoben.
Schließen Sie das Editierfenster mit OK.

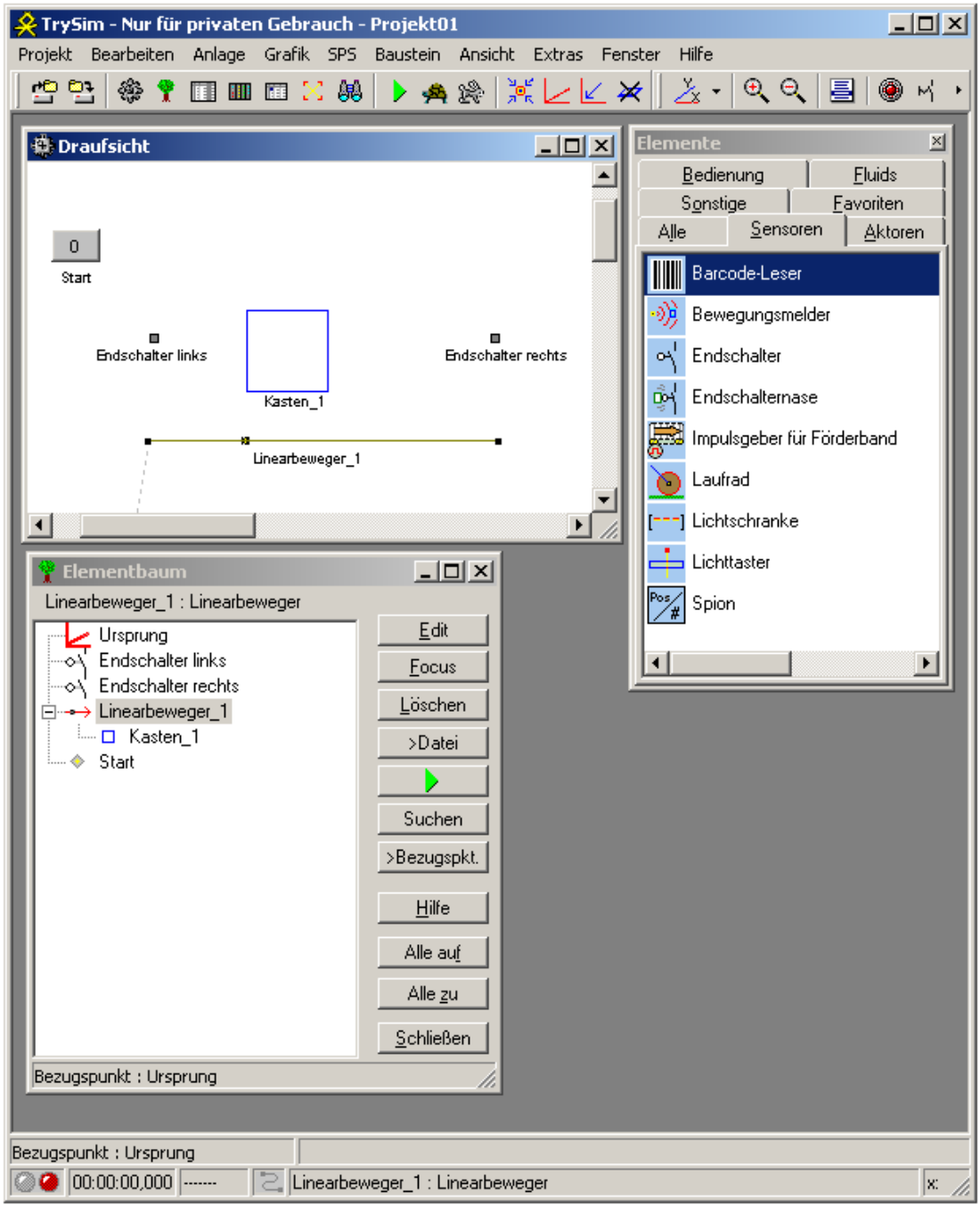
3. Endschalter erzeugen
Erzeugen Sie einen Endschalter (Registerkarte “Sensoren”).
Positionieren Sie ihn etwas links neben dem Kasten. Achten Sie darauf, dass er so nahe am Kasten ist, dass ihn der Kasten bei der Bewegung des Linearbewegers nach links noch erreicht.
Öffnen Sie die Editiermaske des Endschalters durch klicken mit der rechten Maustaste. Klicken Sie auf den Pfeil neben dem Eingabefeld Master (Registerkarte "VAR") und wählen den Kasten aus. Dadurch weiß der Endschalter, dass er vom Kasten betätigt werden soll.
Tragen Sie in das Feld Name “Endschalter links” ein.

Erzeugen Sie genauso einen weiteren Endschalter, den Sie rechts neben dem Kasten positionieren. Nennen Sie ihn “Endschalter rechts”.

Wenn die Beschriftungen Sie stören, stellen Sie sie durch “packen” mit der linken Maustaste an einen anderen Ort, oder öffnen Sie die Editiermasken und wählen den Haken neben dem Feld "Beschriftung" ab.

4. Start-Taster erzeugen
Erzeugen Sie einen Schließer - Taster (Registerkarte “Bedienung”) und plazieren Sie ihn irgendwo im Fenster. Öffnen Sie die Editiermaske und nennen Sie ihn “Start”.

5. Speichern
Die Anlage ist jetzt fertig, Sie sollten sie speichern: **Projekt|Alles Speichern.** Sie können dem Projekt einen eigenen Namen geben, oder den Vorschlag “Projekt01” akzeptieren.
Die Auswahlliste können Sie jetzt durch Klicken auf das Kreuz oben rechts schließen.

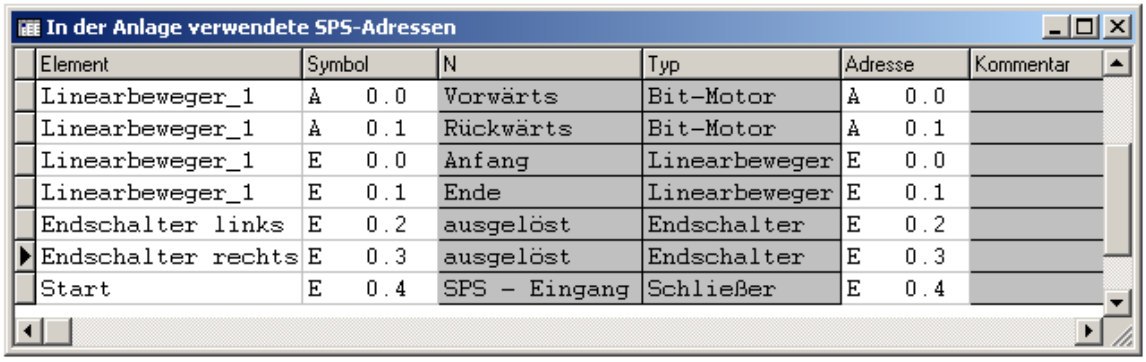


6.3 Erstellen des SPS-Programms

TrySim vergibt automatisch die SPS-Adressen, wenn Sie neue Elemente erzeugen. Wenn Ihr Programm später auf einer wirklichen SPS laufen soll, müssen Sie diese Adressen natürlich entsprechend der Ein- und Ausgangsbelegung anpassen.

Sie können sich übersichtlich alle vergebenen Adressen anschauen und ggf. anpassen, wenn Sie **Anlage|Adressentabelle** wählen.
Wenn Sie die Reihenfolge der Erzeugung der Elemente eingehalten haben, sieht

diese Tabelle so aus:



In der Anlage verwendete SPS-Adressen

Element	Symbol	N	Typ	Adresse	Kommentar
Linearbeweger_1	A 0.0	Vorwärts	Bit-Motor	A 0.0	
Linearbeweger_1	A 0.1	Rückwärts	Bit-Motor	A 0.1	
Linearbeweger_1	E 0.0	Anfang	Linearbeweger	E 0.0	
Linearbeweger_1	E 0.1	Ende	Linearbeweger	E 0.1	
Endschalter links	E 0.2	ausgelöst	Endschalter	E 0.2	
Endschalter rechts	E 0.3	ausgelöst	Endschalter	E 0.3	
Start	E 0.4	SPS - Eingang	Schließer	E 0.4	

1. Organisationsbaustein 1 öffnen

Öffnen Sie den OB 1 (Organisationsbaustein 1) durch **Baustein|Öffnen** oder indem Sie auf das Symbol  klicken.

2. Netzwerk für “Kasten vorwärts” erzeugen

Erzeugen Sie ein SR-Flipflop durch Klicken auf das Symbol 

(oben rechts) in der Symbolleiste.

Geben Sie in das gelbe Feld überhalb des Flip-Flops seine Adresse “A 0.0” ein. Das Editierfeld dazu wird automatisch aktiv, wenn Sie einen Buchstaben eingeben. Sie können es auch mit “ENTER” explizit öffnen. A 0.0 ist “Motor vor”. Nach der Eingabe müssen Sie das Editierfeld mit “ENTER” wieder schließen.

Setzen Sie vor den “S” - Eingang eine "Oder-Box". Klicken Sie dazu den “??..”-Namen vor dem Eingang an, oder bewegen Sie den Cursor mit den Pfeil-Tasten. Wählen Sie das Symbol 

aus der Symbolleiste.

Beschalten Sie die Eingänge der "Oder-Box" mit “E 0.2” (das ist der Endschalter links) und “E 0.4” (das ist der Taster Start).

Beschalten Sie den “R”-Eingang mit “E 0.3” (das ist der Endschalter rechts). Statt den Text “E 0.3” einzutippen, können Sie auch den Endschalter anklicken und danach auf den Operanden “??” klicken.

Beschalten Sie die Zuweisung am “Q”-Ausgang des Flip-Flops ebenfalls mit “A 0.0” oder löschen Sie sie mit der “Entf”-Taste.

3. Netzwerk für “Kasten rückwärts” erzeugen

Wechseln Sie in das nächste Netzwerk mit der “Bild runter”-Taste oder klicken Sie auf den nach unten weisenden Doppelpfeil an der rechten Seite des Netzwerkfensters.

Erzeugen Sie ein weiteres SR-Flip-Flop “A 0.1” (Motor zurück).

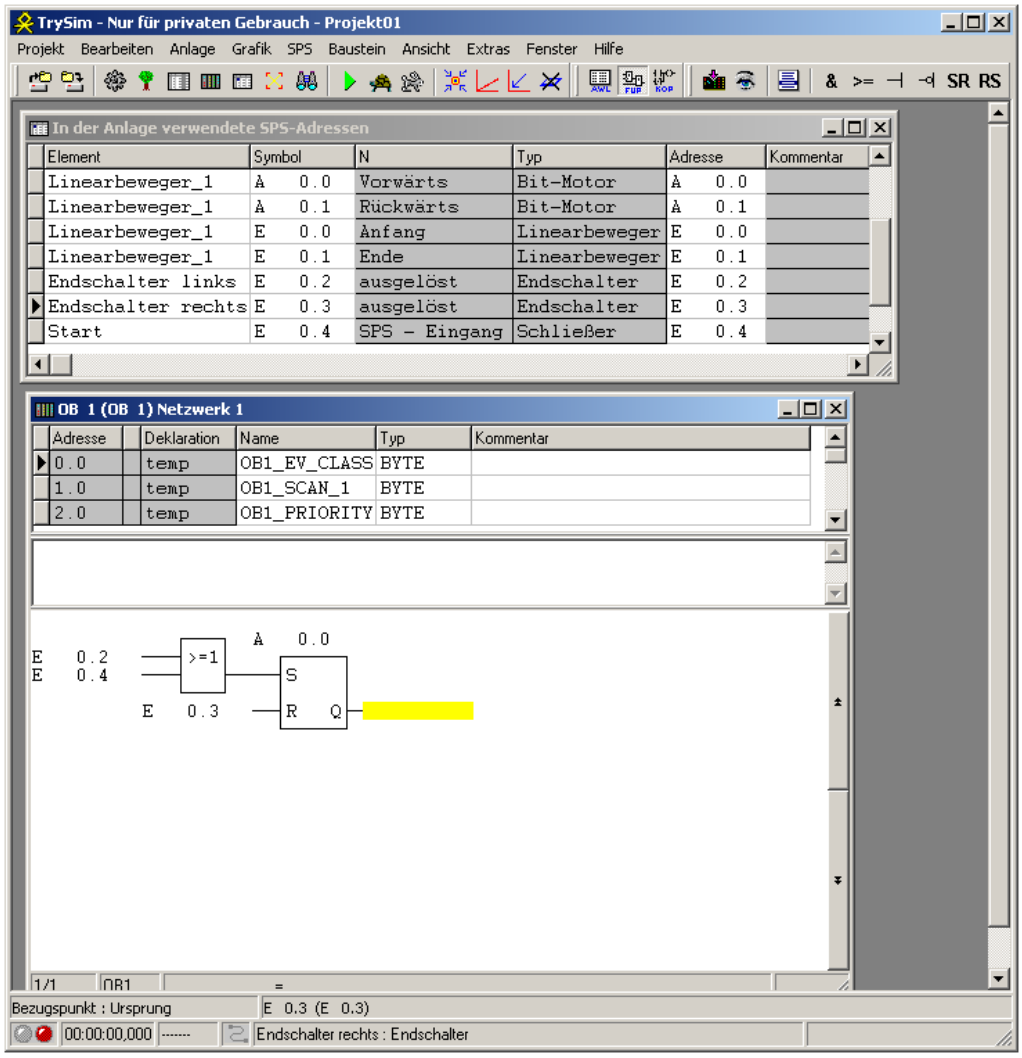
Beschalten Sie den “S”-Eingang mit “E 0.3” (Endschalter rechts) und den “R”-Eingang mit “E 0.2” (Endschalter links) und löschen Sie die Zuweisung am “Q”-Ausgang.

4. Programm in die SPS übertragen

Übertragen Sie das Programm in die SPS durch Klicken auf das Symbol  (Übertragungssymbol) neben dem Auge  in der Symbolleiste.

5. Speichern

Das Programm ist jetzt fertig und Sie sollten es speichern: **Projekt|Alles speichern** .



Hinweis: Das ist natürlich kein schönes Programm, so sind z.B. die Ausgänge “Motor vor” und “Motor zurück” nicht gegeneinander verriegelt, aber es ist einfach zu schreiben und darauf kam es hier an.

6.4 Starten der Simulation

Wechseln Sie wieder zur Anlage, indem Sie auf das Anlagen-Fenster klicken oder das Zahnrad  aus der Symbolleiste wählen.

1. Starten der Simulation

Starten Sie die Simulation, indem Sie auf das  - Symbol klicken. Jetzt werden abwechselnd die Anlage und das SPS-Programm bearbeitet.

2. Starten der simulierten Anlage

Klicken Sie auf den von Ihnen erzeugten “Start”-Taster. Jetzt sollte, wenn alles gut gegangen ist, der Kasten nach rechts fahren, bis er den Endschalter erreicht, dann umkehren bis zum linken Endschalter und immer hin- und herfahren.
Falls der Wagen nicht hin und her fährt, siehe hier: Mögliche Fehler beim Beispiel.

3. Justieren der Start- und Stop-Rampen

Die Endschalter werden bei Betätigung rot, aber das ist schwer zu sehen, da der Wagen ja sofort wieder in die andere Richtung fährt. Durch Verlängern der Bremszeit des Motors lässt sich hier Abhilfe schaffen:

> Stoppen Sie die Simulation durch Klicken auf  . Sie können sie auch dadurch stoppen, indem Sie mit der rechten Maustaste irgendwo auf das Weiße des Fensters klicken.
>
> Öffnen Sie die Editiermaske des Linearbewegers mit Rechtsklick und klicken Sie auf den Button "Antrieb". Jetzt öffnet sich die Editiermaske des Motors vom Linearbeweger. Geben Sie im Feld "Hochlaufzeit" “2 s” ein. Bestätigen Sie 2 x mit “OK” und starten Sie die Anlage erneut. Jetzt ist deutlich zu erkennen, wie die Endschalter ihre Farbe wechseln.

6.5 Beobachten der Programmbearbeitung

Wechseln Sie wieder zum Programm durch Klicken auf einen sichtbaren Teil des OB1-Fensters oder durch Klicken auf das SPS-Symbol  neben dem Zahnrad  .

Klicken Sie auf das Auge  , um den Beobachten-Modus zu aktivieren.

Jetzt können Sie den Zustand der Ein- und Ausgänge beobachten.

Wechseln Sie mit der Bild-Rauf-Taste in das erste Netzwerk und schieben Sie die Fenster so, dass gleichzeitig Ihr Taster “Start” und der entsprechende Eingang an der Oder-Box zu sehen ist. Wenn Sie mit der linken Maustaste den Taster betätigen, sehen Sie, wie der Eingang und die Oder-Box rot werden.

6.6 Mögliche Fehler beim Beispiel

Gar nichts passiert:

Haben Sie den Änderungshinweis bei der Adresstabelle berücksichtigt?
Haben Sie die Simulation gestartet? (grüne LED links unten).
Haben Sie das Programm in die SPS übertragen?
Haben Sie das Programm wie beschrieben erstellt?
Falls Sie einen Fehler im SPS-Programm gefunden und korrigiert haben, müssen Sie das Programm danach neu übertragen!

- Der HotSpot des Linearbewegers fährt zwar, aber der Kasten bewegt sich nicht:
  Sie haben den Kasten nicht am Linearbeweger befestigt.
  Öffnen Sie die den Elementbaum  und schieben Sie den Kasten mit der Maus auf den Linearbeweger

- Ein Endschalter wird vom Kasten nicht erreicht:
  Stoppen Sie die Simulation mittels des Symbols  oder
  klicken Sie mit rechts irgendwo auf das Weiße des Anlagefensters.
  Packen Sie den Endschalter mit der linken Maustaste
  und schieben Sie ihn an die gewünschte Position.

- Ein Endschalter fährt mit dem Kasten mit und wird daher nie erreicht:
  Sie haben den Endschalter versehentlich am Kasten oder am Linearbeweger befestigt.
  Öffnen Sie die Editiermaske des Endschalters
  und wählen für den Vater den Ursprung,
  dadurch wird der Endschalter am Ursprung befestigt,
  der sich per Definition nicht bewegt.

- Der Kasten bedeckt einen Endschalter, aber dieser wird nicht rot:
  Der Endschalter weiß nicht, dass er vom Kasten bedeckt werden soll;
  öffnen Sie die Editiermaske des Endschalters und wählen Sie in der Auswahlliste,
  die erscheint, wenn Sie auf den Pfeil neben dem Feld Master klicken,
  den Kasten.

- Der Kasten bedeckt einen Endschalter, dieser wird auch rot, aber der Kasten kehrt nicht um:
  Vermutlich haben Sie einen Fehler im Programm gemacht. Überprüfen Sie:
  A 0.0 wird von E 0.2 oder E 0.4 gesetzt
  A 0.0 wird von E 0.3 zurückgesetzt
  A 0.1 wird von E 0.3 gesetzt
  A 0.1 wird von E 0.2 zurückgesetzt
  Falls Sie einen Fehler im SPS-Programm gefunden und korrigiert haben, müssen Sie das Programm danach neu übertragen!

Ende der Beispielsitzung

6.7 Kapitel für Anfänger

6.7.1 Was ist eine SPS ?

Dieser Teil der TrySim-Hilfe ist nur für diejenigen gedacht, die noch nie mit einer SPS gearbeitet haben. Erfahrene Anwender werden nichts Neues finden und stattdessen vieles vermissen - Vollständigkeit ist hier aber nicht unser Anliegen.

Normale Computer sind dafür gemacht, dass sie von Menschen benutzt werden. Dementsprechend haben sie als Eingabemöglichkeiten eine Tastatur, eine Maus und manchmal noch einen Scanner. Die Ergebnisse, die der Computer liefert, werden über einen Monitor, einen Drucker und ggfs. über Lautsprecher ausgegeben.

Eine SPS (**s**peicher**p**rogrammierbare **S**teuerung) ist im Prinzip ein Computer wie Ihr PC, allerdings sind die Ein- und Ausgabemöglichkeiten der SPS dafür ausgelegt, Maschinen zu steuern. Maschinen haben keine Finger, eine Tastatur wäre also unnütz und die meisten SPS haben auch keine. Stattdessen haben Maschinen z.B. Lichtschranken, Endschalter und Knöpfe (die hier Taster genannt werden) mit denen die Bediener der Maschine vorgeben, was geschehen soll. Allen diesen Elementen ist gemeinsam, dass sie über eine Leitung ein binäres Signal liefern, also entweder eine “1” oder eine “0”. Daher befinden sich an einer SPS Digital-Eingänge, die je aus einer Klemme bestehen, an die man die Leitung anschließen kann.

Genauso wie die Eingabemöglichkeiten einer SPS an die Erfordernisse einer Maschine angepasst sind, sind es die Ausgabemöglichkeiten: Die Digital-Ausgänge sind als Klemmen ausgeführt, die entweder Spannung führen oder nicht. Daran kann man dann z.B. Ventile und Relais anschließen mit deren Hilfe die Bewegungen der Maschine kontrolliert werden.

Wann die Ausgänge eingeschaltet werden, wird durch das Programm innerhalb der SPS vorgeschrieben. Dieses Programm muss von dem Programmierer erstellt werden. Dazu gibt es verschiedene, speziell an die Bedürfnisse von SPS angepasste Programmiersprachen, u.a. FUP (Funktionsplan), KOP (Kontaktplan) und AWL (Anweisungsliste). Um ein Gefühl dafür zu bekommen, wie eine solche Programmierung im Prinzip aussieht, sollten Sie sich das Beispiel “Für_Anfänger_1” anschauen, dort mit den Tastern spielen und beobachten, wann die Leuchtmelder leuchten.

Beim Schreiben von SPS-Programmen ist es häufig nützlich, Zwischenergebnisse speichern zu können. Dazu gibt es u.a. die Merker.

Will man ein größeres SPS-Programm schreiben, sollte man dieses strukturieren.

Was ist eine Lichtschranke ?

Lichtschranken dienen dazu, das Vorhandensein eines Objektes zu detektieren. Sie sind in fast jedem Supermarkt vorhanden um zu erkennen, dass Sie einen Einkaufswagen durch die Eingangsschranke schieben wollen. Lichtschranken bestehen aus einem Sender, der Licht aussendet und einem Empfänger, der überprüft, ob dieses Licht bei ihm ankommt. Kommt kein Licht an, befindet sich offensichtlich irgendetwas zwischen dem Sender und dem Empfänger. Eine Lichtschranke liefert also nur die Information: “Lichtweg frei” oder “Lichtweg

unterbrochen". Diese Information wird vom Empfänger über ein Spannungssignal der SPS mitgeteilt. Dabei gibt es zwei Möglichkeiten:

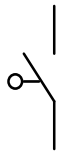
1. Spannung da bedeutet : Lichtweg unterbrochen
2. Spannung da bedeutet : Lichtweg frei

Welche von diesen Möglichkeiten gewählt wird, lässt sich meistens am Empfänger einstellen. Im Prinzip sind beide Möglichkeiten gleichberechtigt, doch manchmal führen Sicherheitsüberlegungen dazu, eine zu bevorzugen.

Häufig sind Sender und Empfänger auch im gleichen Gehäuse untergebracht. Der Lichtweg führt dann vom Sender zu einem Reflektor und von dort zurück zum Empfänger.

Was ist ein Endschalter ?

Ein Endschalter (auch Grenztaster, Initiator, usw.) ist eigentlich wie ein normaler Lichtschalter, allerdings ist er dafür ausgelegt, von Maschinenteilen betätigt zu werden, nicht von Menschenhänden. Er liefert nur die Information: "Maschinenteil da" und "Maschinenteil nicht da". Ein Beispiel für einen Endschalter, das jeder kennt, ist der Lichtschalter im Kühlschrank: Ist die Tür offen, ist das Licht an, ist sie geschlossen, ist das Licht aus.
Endschalter gibt es in tausend Formen, je nach dem Maschinenteil, von dem sie betätigt werden sollen, es ist daher schwer zu beschreiben, wie sie aussehen. Mechanische Endschalter haben häufig eine Rolle, damit bei vorbei gleitenden Maschinenteilen kein zu großer Verschleiß auftritt. Hier das Symbol für Endschalter in Schaltplänen:



Häufig arbeiten Endschalter aber auch berührungslos, meistens wird hierbei die Induktivitätsänderung einer Spule durch das Vorhandensein von Eisen ausgenutzt, es gibt aber auch kapazitive Endschalter, die nicht nur auf Eisen ansprechen. All diese berührungslosen Endschalter haben eine Auswerteelektronik, die letztlich nur das Signal erzeugt: "Teil da" oder "Teil nicht da", sodass in der SPS-Praxis nicht unterschieden werden muss, um welche Art es sich handelt. Die SPS erkennt sowieso nur, ob an dem Digitaleingang, an dem der Endschalter angeschlossen ist, Spannung anliegt oder nicht. Dabei gibt es wie bei der Lichtschranke zwei Möglichkeiten:

1.) Spannung da bedeutet : Teil ist da
2.) Spannung da bedeutet : Teil ist nicht da

Anders als bei den Lichtschranken gibt es bei den Endschaltern (ES) eine klare Bezeichnung für die beiden Fälle: Wenn ein ES das Vorhandensein eines Teiles durch "Spannung da" signalisiert, wird er "Schliesser" genannt. Signalisiert er

umgekehrt das Vorhandensein eines Teiles durch “Spannung nicht da”, wird er “Öffner” genannt (zur klaren Kennzeichnung der Funktion eines ES reicht dies dennoch nicht aus, s.u.). Im Prinzip sind beide Möglichkeiten gleichberechtigt, da in einer SPS sehr leicht durch eine Negation die Bedeutung eines Signales umgekehrt werden kann, doch manchmal führen Sicherheitsüberlegungen dazu, eine zu bevorzugen.

Bei der Erstellung der Dokumentation zu einer Steuerung ist es sehr wichtig, eindeutig zu vermerken, was die Bedeutung des Signals “Spannung da” ist, im Falle des Kühlschranks wäre dies z.B. “1 = Tür ist offen”.

Sicherheitsüberlegungen

Bei dem Entwurf der Steuerung einer Maschine muss man immer daran denken, dass auch etwas schief gehen kann. Der weitaus häufigste Fehlerfall ist dabei, dass eine Spannung nicht dort ansteht, wo sie erwartet wird. Der umgekehrte Fall (Spannung fälschlicherweise da) kommt (insbesondere bei berührungslosen Endschaltern und Lichtschranken) natürlich auch vor, aber mit sehr viel geringerer Wahrscheinlichkeit. Wenn man sich überlegt, wieviel Aufwand notwendig ist, ein Spannungssignal von A nach B zu bringen, ist das auch ganz einleuchtend. Es ist einfach viel wahrscheinlicher, dass eine Klemme sich löst oder eine Leitung bricht, als dass eine andere spannungsführende Leitung versehentlich Kontakt mit der Leitung bekommt, über die das uns interessierende Signal übertragen wird.

Wenn man mögliche Fehler berücksichtigen will, sollte man vom wahrscheinlicheren Fall ausgehen und überlegen, wie sich ein möglicher Schaden minimieren lässt. Wenn ein Wagen z.B. wie in unserer Beispielsitzung zwischen zwei Endschaltern hin- und herfahren soll, dann ist der größere Schaden zu erwarten, wenn der Wagen nicht anhält. In diesem Fall sollten die Endschalter also als Öffner, d.h. mit der Bedeutung “1 = Wagen nicht da” ausgelegt werden. Denn wenn dann die Leitung des ES bricht oder eine Klemme sich löst, wird keine Spannung vom ES zu SPS mehr übertragen und das bedeutet ja “0 = Wagen da”. Der Wagen bleibt also stehen oder kehrt um, auf jeden Fall aber fährt er nicht über den Endschalter hinaus gegen die mechanische Begrenzung. In diesem Fall nämlich müsste der zur Störungsbeseitigung gerufene Elektriker zwei Probleme lösen: 1.) Warum hat der Motorschutzschalter angesprochen? und 2.) Warum funktioniert der Endschalter nicht?

Die Grundregel heißt also: Der sichere Fall wird durch “Spannung da” signalisiert.

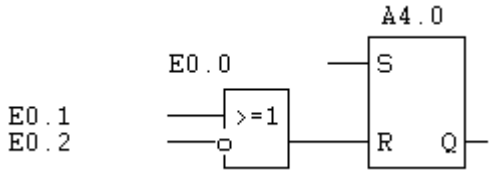
Sehr schön sehen kann man die Einhaltung dieser Regel an der Not-Aus-Schaltung jeder Maschine. Alle Not-Aus-Taster werden als Öffner in Reihe geschaltet. Nur dann, wenn dieser Kreis geschlossen ist, können die gefährlichen Teile der Anlage aktiv werden. Tritt eine Unterbrechung des Kreises auf, bleibt alles stehen. Natürlich könnte man auch alle Not-Aus-Taster als Schliesser auslegen und parallel schalten. Bei der Betätigung eines dieser Schließer würde dann ein Schütz anziehen, das alle gefährlichen Bewegungen stoppt. Dann hätte man aber keine ständige Kontrolle, ob alle Not-Aus-Taster in Ordnung sind, denn einen Bruch des Kabels zu einem der

Taster würde man erst dann bemerken, wenn man ihn benutzen will - zu spät.

Negation

Vom Inneren einer SPS aus werden die Eingänge als Bedingungen gesehen, die entweder “wahr” oder “falsch” (meist mit “1” und “0” abgekürzt) sein können. “1” bedeutet dabei immer: “Am Eingang liegt Spannung an”. Die Bedeutung eines Signals, dass über einen Eingang in die SPS gelangt, ist damit aber noch nicht festgelegt. Bei einem Eingang, der mit “Not-Aus” beschriftet ist, z.B. bedeutet eine “1”: “alles in Ordnung, Maschine kann laufen”. (Siehe: Sicherheitsüberlegungen). Verwendet man diesen Eingang im SPS-Programm, muss man seine Bedeutung berücksichtigen. Denken Sie sich einen Motor (angeschlossen an den Ausgang `A 4.0`), der durch einen EIN-Taster (`E0.0`) gestartet und durch einen AUS-Taster (`E 0.1`) gestoppt werden soll. Außerdem gibt es noch einen Not-Aus-Kreis, der an den Eingang `E0.2` angeschlossen ist. Natürlich soll der Motor auch gestoppt werden, wenn Not-Aus ausgelöst worden ist (`E0.2`=0).

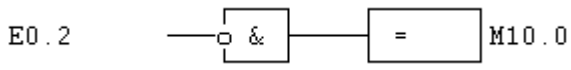
Das Programm zur Steuerung dieses Motors in FUP könnte dann so aussehen:




```
E  0.0    EIN      Taster EIN
E  0.1    AUS      Taster AUS
E  0.2    ENotAus  Not-Aus ist ausgelöst (=0)
A  4.0    Motor    Motor einschalten
```

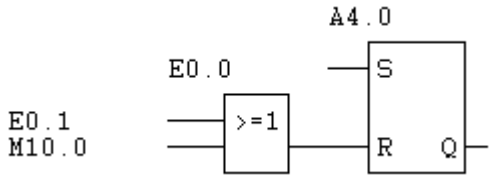
Die eigentliche Bedeutung des Netzwerkes ist: Der Motor soll gestartet werden, wenn der Taster EIN (`E0.0`) betätigt wird. Der Motor soll gestoppt werden, wenn entweder der Taster AUS (`E0.1`) betätigt wird, oder wenn Not-Aus (`E0.2`) ausgelöst worden ist. Dass Not-Aus ausgelöst worden ist, wird aber ja durch das Ausbleiben der Spannung am Eingang `E0.2`, also “0” erkannt! Der Not-Aus-Eingang muss also **negiert** verwendet werden. Im Netzwerk oben ist dies durch den kleinen Kreis geschehen. Aus der “Abfrage auf 1” wird dadurch eine “Abfrage auf 0”.

Dies ist natürlich nur ein Beispiel für den Gebrauch einer Negation. In SPS-Programmen kommen Negationen sehr häufig vor und fehlende oder falsch gesetzte Negationen sind eine der häufigsten Fehlerquellen. Gerade bei Eingängen, deren Bedeutung der Intuition zuwider läuft, wie dem Not-Aus-Eingang, ist es häufig ratsam, die Negation einmal am Anfang des Programmes vorzunehmen, das Ergebnis in einem Merker zu speichern und im folgenden nur noch diesen Merker zu verwenden. Zum Beispiel so:




```
E  0.2    ENotAus  Not-Aus ist ausgelöst (=0)
M 10.0    NotAus   Not-Aus ist ausgelöst
```

Dieses Netzwerk schreibt man ganz am Anfang des Programmes. Der Merker `M 10.0` hat nun die Bedeutung “Not-Aus ist ausgelöst” und wird im Weiteren so verwendet:




```
E  0.0     EIN     Taster EIN
E  0.1     AUS     Taster AUS
M 10.0     NotAus  Not-Aus ist ausgelöst
A  4.0     Motor   Motor einschalten
```

Jetzt sieht man ganz klar und ohne weiter nachzudenken: Der Motor wird gestoppt, wenn auf den AUS-Taster gedrückt wird, oder wenn Not-Aus ausgelöst worden ist.

SPS-Programm

Wenn man eine typische Maschine betrachtet, fällt einem zunächst auf, dass sich etwas bewegt. Bei Anlagen der Verfahrenstechnik fällt dies nicht so ins Auge, aber wenn man Röntgenblicke hätte, die in die Ventile und Pumpen schauen könnten, wäre der Eindruck der gleiche. All die Bewegungen haben den Zweck, ein Produkt zu erzeugen oder zu bearbeiten. Der Konstrukteur einer Anlage oder Maschine konzentriert sich daher zunächst auf die Aktoren, denn nur durch diese geschieht ja etwas. Wenn er dann beschreiben muss, wann welcher Aktor aktiv werden soll, merkt er, dass er dazu Informationen über den Prozess braucht und diese Informationen liefern ihm die Sensoren, die er also auch noch vorsieht. Am Ende gibt es dann einen Plan, welche Aktoren und Sensoren in die Anlage/Maschine eingebaut werden sollen und eine Beschreibung, wie all diese zusammenarbeiten sollen. Damit eine SPS entsprechend dieser Beschreibung funktionieren kann, muss aus der Beschreibung ein Programm gemacht werden, d.h., sie muss in eine Sprache übersetzt werden, die die SPS “versteht”. Dieses Programm besteht aus allem, was die SPS “wissen muss” und zusätzlich noch aus Kommentaren und weiteren Informationen, die für Menschen notwendig sind, das Programm auch im Nachhinein zu verstehen und ggfs. modifizieren zu können.
Zum Erstellen des Programms für eine SPS gibt es spezielle Software, die i.A. auf normalen PCs läuft und die es dem Programmierer ermöglicht, auf einfache Weise die menschen-sprachliche Beschreibung des Konstrukteurs in die Maschinensprache zu übersetzen, die die SPS versteht.
Wenn das Programm erstellt ist, wird es in die SPS übertragen, dazu muss natürlich der PC mit der SPS über ein Kabel verbunden werden. Wenn das Programm in die SPS übertragen ist, kann die Verbindung wieder getrennt werden und die SPS wird dann genau das tun, was ihr durch das Programm aufgetragen wird.

Digital-Eingang

Einen Digital-Eingang einer SPS kann man von zwei Seiten aus betrachten: Von außen ist er einfach eine Klemme, an die man einen Draht anschließen kann, der entweder Spannung führt oder nicht. Vom Inneren der SPS (also vom Programm aus) gesehen, ist ein Digital-Eingang eine Speicherstelle, die entweder den Wert “1” oder “0” hat und beliebig oft ausgelesen werden kann.

Es gibt auch Analog-Eingänge, mit denen variable Spannungen oder Ströme eingelesen werden, aber von denen soll hier zunächst nicht die Rede sein.

Digital-Ausgang

Einen Digital-Ausgang einer SPS kann man von zwei Seiten aus betrachten: Von außen ist er einfach eine Klemme, die entweder Spannung führt oder nicht und an die man einen Draht anschließen kann. Hiermit kann man dann Ventile, Relais, Schütze oder Leuchtmelder ansteuern. Vom Inneren der SPS (also vom Programm aus) gesehen, ist ein Digital-Ausgang eine Speicherstelle, der man entweder den Wert “1” oder “0” zuweisen kann.

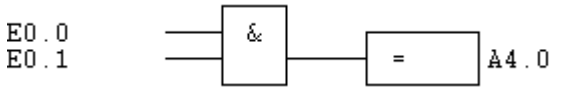
Es gibt auch Analog-Ausgänge, mit denen variable Spannungen oder Ströme ausgegeben werden können, aber von denen soll hier zunächst nicht die Rede sein.

Anweisungslite (AWL)

Dies ist eine der möglichen Darstellungsarten eines SPS-Programms. Die Operationen werden hier (wie der Name schon sagt) als eine Liste von Anweisungen untereinander geschrieben. Für logische Operationen (UND, ODER usw.) halten wir AWL für eine denkbar schlechte Darstellungsart. Wenn man jedoch beginnt, in einer SPS zu Rechnen oder ganz allgemein kompliziertere Operationen durchführen will, ist AWL allerdings deutlich übersichtlicher. Da dieser Teil der TrySim-Hilfe nur für absolute Anfänger gedacht ist, verzichten wir auf eine Beschreibung.

Funktions-Plan (FUP)

Dies ist eine der möglichen Darstellungsarten eines SPS-Programms. Die logischen Operationen werden hierbei durch Blöcke dargestellt, die ihrer Funktion entsprechend bezeichnet werden. Einfachstes Beispiel für einen Funktionsplan ist eine UND-Verknüpfung:



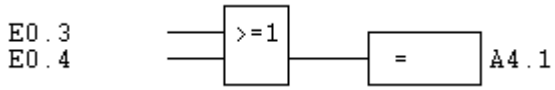

```
E  0.0     Taster1     Taster 1 Presse ab
E  0.1     Taster2     Taster 2 Presse ab
A  4.0     PressAb     Ventil Presse ab
```

Hier bedeutet das Symbol “&” in dem Funktionsblock, dass der Ausgang eine “1”

führen soll, wenn Eingang 1 (E 0.0) **und** Eingang 2 (E 0.1) eine “1” führen. Nur dann wird der Ausgang A4.0 eingeschaltet.

Es können auch mehr Eingänge am &-Block vorhanden sein, dann wird der Ausgang nur dann eine “1” führen, wenn alle Eingänge eine “1” führen.

Genauso einfach ist die ODER - Verknüpfung:




```
E  0.3     THupe1    Taster Hupe drinnen
E  0.4     THupe2    Taster Hupe draußen
A  4.1     Hupe      Warnungshupe
```

Hier wird der Ausgang geschaltet, wenn E0.3 **oder** E0.4 eine “1” führen. Das Symbol “>=1” kommt von der Sprechweise “Der Ausgang wird geschaltet, wenn die Anzahl der Eingänge, die eine “1” führen, größer oder gleich 1 ist.”

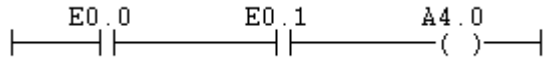
Auch an den “>=1”-Block können mehr als zwei Eingänge angeschlossen werden. Der Ausgang wird dann geschaltet, wenn mindestens einer der Eingänge eine “1” führt.

Kontakt-Plan (KOP)

Dies ist eine der möglichen Darstellungsarten eines SPS-Programms. Die logischen Operationen werden hier so dargestellt, wie man sie in herkömmlicher Schaltungstechnik verwirklichen würden.

Im Kontaktplan fließt ein gedachter Strom von links nach rechts. Die Digital-Eingänge der SPS (an die ja Sensoren, Taster oder Schalter der Anlage angeschlossen sind) werden durch Unterbrechungen ---| |--- dargestellt, die den Strom daran hindern weiter zu fließen. Falls der entsprechende Eingang eine “1” führt, wird dieser gedachte Kontakt geschlossen und der Strom kann weiter fließen. Wenn er bis zum Ausgang kommt, der durch zwei Klammern dargestellt wird ----( )----- , wird der entsprechende Digital-Ausgang aktiviert.

Eine UND-Verknüpfung in KOP wird so dargestellt:

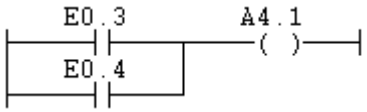



```
E  0.0     Taster1     Taster 1 Presse ab
E  0.1     Taster2     Taster 2 Presse ab
A  4.0     PressAb     Ventil Presse ab
```

Nur dann, wenn der Eingang E0.0 **und** der Eingang E0.1 auf “1” sind, wird der Ausgang A4.0 geschaltet.

Möchte man eine UND-Verknüpfung von mehr als zwei Eingängen, muss man die entsprechende Anzahl in Reihe schalten.

Eine ODER-Verknüpfung wird in KOP so dargestellt:




```
E  0.3     THupe1   Taster Hupe drinnen
E  0.4     THupe2   Taster Hupe draußen
A  4.1     Hupe     Warnungshupe
```

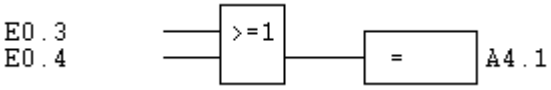
Hier hat der gedachte Strom zwei Möglichkeiten von links zum Ausgang A4.1 zu fließen:
über den Kontakt E0.3 **oder** über den Kontakt E0.4. Wenn also mindestens einer dieser beiden Kontakte geschlossen ist (d.h., der zugehörige Digital-Eingang führt eine “1”) wird der Ausgang geschaltet werden.

Möchte man eine ODER-Verknüpfung von mehr als zwei Eingängen, muss man die entsprechende Anzahl von Kontakten parallel schalten.

Zwei weitere Darstellungsarten von SPS-Programmen sind der Funktionsplan FUP und die Anweisungsliste AWL.

Merker

Die eigentliche Aufgabe einer SPS ist es, die Eingänge zu lesen, sie logisch zu verknüpfen und daraus den Zustand der Ausgänge zu ermitteln. Einfachstes Beispiel: Eine Warnungshupe (`A4.1`) soll ertönen,



wenn entweder auf einen der Taster Hupe1 (`E0.3`) oder Hupe2 (`E0.4`) gedrückt wird. Das Programm in FUP würde dann so aussehen:

```
E  0.3     THupe1   Taster Hupe drinnen
E  0.4     THupe2   Taster Hupe draußen
A  4.1     Hupe     Warnungshupe
```

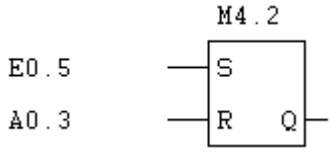
Nicht immer aber sind SPS-Programme so einfach. Häufig gibt es die Notwendigkeit, Zustände zu speichern, zumindest ist es nützlich. Dazu gibt es die **Merker**, das sind Zellen im Speicher der SPS, in denen Informationen abgelegt und später wieder ausgelesen werden können. Ein einzelner Merker kann nur “0” oder “1” speichern. Merker werden benannt wie die Ein- und Ausgänge: Zunächst kommt die Bezeichnung des Speicherbereiches, in diesem Fall ein M, dann die Byte-Nummer und schließlich, getrennt durch einen Punkt, die Bit-Nummer. Beispiele für Merkerbezeichnungen sind:

```
M  0.0
M 22.6
M 29.3
```

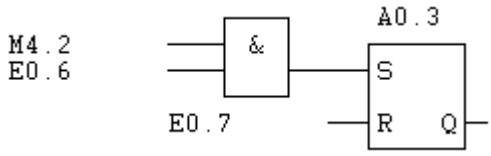
Die Speicherung von Zuständen kann, wie oben bereits angemerkt, notwendig oder

nützlich sein.

Notwendig ist eine Speicherung, wenn z.B. ein Tor sich nur dann öffnen soll, wenn zunächst auf einen Taster und danach auf einen Weiteren gedrückt wird. In diesem Fall muss gespeichert werden, dass bereits auf den ersten Taster gedrückt wurde. Das Programm für diesen Fall könnte so aussehen:



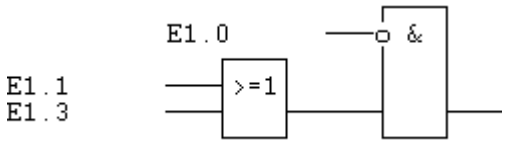

```
E  0.5     Taster1     Taster 1
M  4.2     Taster1OK   Merker Es ist auf Taster 1 gedrückt worden
A  0.3     TorAuf      Motor Tor öffnen
```




```
M  4.2     Taster1OK   Merker Es ist auf Taster 1 gedrückt worden
E  0.6     Taster2     Taster 2
A  0.3     TorAuf      Motor Tor öffnen
E  0.7     ESTorAuf    Endschalter Tor offen
```

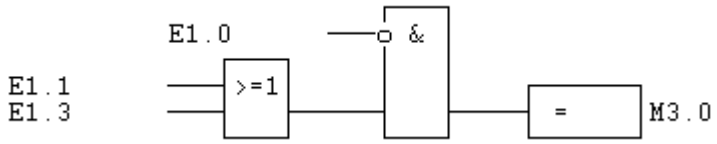
Im ersten Netzwerk wird gespeichert, dass auf den Taster1 gedrückt worden ist. Wenn das Tor geöffnet wird, ist diese Information nicht mehr wichtig und wird daher gelöscht. Im zweiten Netzwerk wird das Öffnen des Tores durch den Taster2 gestartet, aber nur dann, wenn vorher Taster1 gedrückt wurde.

Nützlich ist die Verwendung von Merkern dann, wenn es dadurch gelingt, das SPS-Programm übersichtlicher darzustellen. Denken Sie sich eine Maschine, bei der sehr viele Aktionen nur stattfinden sollen, wenn 1.) Keine Störung vorliegt und 2.) Die Automatik aktiv ist ODER getasteter Handbetrieb vorliegt. Als FUP sieht diese Bedingung so aus:




```
E  1.0     Stoerung    Sammelstörung
E  1.1     Auto        Schalter Automatik
E  1.3     HandTast    Taster Getasteter Handbetriebe
```

Ohne Merker müssten Sie diese Bedingung zig-mal im Programm schreiben. Das ist erstens nervig und zweitens unübersichtlich. Unter Verwendung von Merkern schreiben Sie einmal:




```
E   1.0     Stoerung    Sammelstörung
E   1.1     Auto        Schalter Automatik
E   1.3     HandTast    Taster Getasteter Handbetriebe
M   3.0     Betrieb     Keine Störung UND Auto oder Hand getastet
```

Im ganzen weiteren Programm können Sie dann den Merker `M3.0` anstelle dieser Bedingung verwenden.

Anmerkung: In STEP®7 sind temporäre und statische Variablen für diese Aufgaben häufig viel besser geeignet als Merker, die Idee ist aber die Gleiche und mit Merkern einfacher zu erläutern, da diese immer vorhanden sindund nicht extra deklariert werden müssen.

Materialaustausch

In diesem Beispiel gehen wir von zwei Anlagenteilen aus. Um das konkreter zu machen: Anlagenteil 1 (AT1) wird mit dem Rohstoff beschickt und formt daraus Stücke der gewünschten Form. Die notwendigen Befestigungslöcher können aber nur vom Anlagenteil 2 (AT2) gebohrt werden, daher muss ein vom AT1 vorgefertigtes Stück an AT2 übergeben werden. Denken wir uns im AT2 ein dafür vorgesehenes Fach. Zunächst muss der AT1 sicherstellen, dass das Aufnahmefach leer ist. Dies muss ihm irgendwie vom AT2 signalisiert werden. Wenn dann das vorgefertigte Stück im Aufnahmefach abgelegt ist, muss der AT1 dem AT2 signalisieren “OK, Stück ist zur Abholung bereit”.

Wenn die beiden Anlagenteile von der gleichen SPS gesteuert werden, muss dieser, den Materialaustausch begleitende Signalaustausch, innerhalb des Programmes nachgebildet werden.

Fehlersuche

Wenn eine SPS-gesteuerte Maschine, die schon lange in Betrieb ist, nicht wie erwartet funktioniert, dann liegt es nur in den seltensten Fällen an dem SPS-Programm. Meistens ist irgendein Sensor defekt, oder ein Aktor arbeitet nicht wie vorgesehen. Dennoch ist es bei der Fehlersuche sehr hilfreich, zu sehen, wie die SPS die Eingänge auswertet und daraus die Ausgänge bestimmt.

Von außen ist es z.B. häufig nicht feststellbar, ob ein binärer Sensor richtig funktioniert. In einer guten Dokumentation würde stehen: “dieser Sensor liefert eine “0”, wenn er betätigt wird”. Das ist aber in den seltensten Fällen so vermerkt, daher kann man von außen kaum entscheiden, ob der Sensor intakt ist oder nicht.

Bei den Aktoren ist es häufig einfacher, aber spätestens, wenn es sich um pneumatische oder hydraulische Ventile handelt, ist unklar: sollte für eine korrekte Funktion dieses Ventil jetzt mit Spannung beaufschlagt werden oder nicht?

Ein Blick in das Programm der SPS verschafft da häufig schnellen Aufschluss, besonders wenn man von der Möglichkeit, die in allen modernen SPS vorhanden ist, Gebrauch macht, sich den Signalfluss online anzeigen zu lassen (beobachten). Wenn diejenigen, die mit der Funktion der Maschine vertraut sind, sagen: Dieser Zylinder müsste jetzt vorfahren und Sie mit Ihrem Blick in die Innereien der SPS sagen: das tut er aber darum nicht, weil diese Lichtschranke nicht anspricht, dann wird man das Problem erst einmal bei der Lichtschranke suchen. Vielleicht ist sie verdreckt vom Staub, vielleicht ist sie auch einfach defekt.

Abschließend: Fehlerhafte Sensoren sind die häufigste Fehlerquelle bei vorher funktionierenden Maschinen. Mit weitem Abstand kommen die fehlerhaften Aktoren, z.B. klemmende Ventile, festgebrannte Schütze oder defekte Motoren. Recht selten nur ist ein Fehler im Programm für eine Fehlfunktion verantwortlich. Kurz nach einer Inbetriebnahme sind diese Fehler wohl für einen großen Teil der Fehlfunktionen einer Maschine/Anlage verantwortlich. Wenn die Maschine/Anlage aber einige Zeit störungsfrei gelaufen ist, sagen wir mal ein halbes Jahr, dann sind Störungen fast immer auf defekte Sensoren, selten auf defekte Aktoren zurückzuführen. Fehler im Programm können natürlich auch noch nach langer Zeit zutage treten und ganz selten ist tatsächlich die SPS defekt und muss ausgetauscht werden. Das kommt vor, sollte aber immer erst dann in Betracht gezogen werden, wenn wirklich jede andere Fehlerquelle ausgeschlossen wurde.

Beobachten der Programmbearbeitung

Zur Erleichterung der Fehlersuche gestatten (fast) alle Entwicklungssysteme für SPS-Programme, der CPU quasi bei der Arbeit zuzuschauen. Dazu muss der PC, auf dem Sie programmieren natürlich mit einem Kabel mit der SPS verbunden sein. Auf dem Bildschirm wird dann der gewünschte Programmausschnitt angezeigt und die Werte, die Eingänge, Ausgänge oder Merker haben, werden farbig dargestellt oder als Zahlen angezeigt. Durch den Vergleich dessen, was man sieht, mit dem, was man erwartet, lassen sich Programmfehler oder Fehler durch defekte Sensoren dann leichter finden. Dies bedeutet natürlich, dass man eine Erwartung haben muss. Obwohl das Beobachten eine ungemein nützliche Funktion ist, kann es einem nicht das Denken abnehmen.

Da die Zykluszeit einer SPS sehr viel schneller ist, als Sie gucken können und da die aktuellen Werte aus der SPS über die vergleichsweise langsame Verbindung zum PC übertragen werden müssen, können Sie nicht die Bearbeitung jedes einzelnen Zyklus' verfolgen, sondern Sie sehen immer nur Momentbilder, nicht aber was zwischen diesen Bildern geschieht. Bei einem sehr schnellen Wechsel der beobachteten Werte wird dadurch die Interpretation des Gesehenen recht schwierig.

Ausgabebaugruppe

Wie in der Einleitung erwähnt, muss eine SPS Maschinenelemente betätigen können. Üblicherweise werden unter dem Stichwort “Ausgabebaugruppe” eine Menge technischer Daten, die auch unbestritten wichtig sind und deren Sinn man

verstehen muss, angegeben. Für den Programmierer aber ist eine Ausgabebaugruppe einfach nur das Tor zur Welt. Wenn Sie in Ihrem SPS-Programm berechnet haben, dass auf dieses Formteil jetzt eine Kraft von 7.45 kN angewendet werden muss, dann transferieren Sie die entsprechende Zahl auf die Ausgabebaugruppe, die dann dem Rest der Steuerung vorgibt, das die gewünschte Kraft angewendet werden soll.

Prozessabbild der Eingänge

Wenn Sie in einem Programm einen Eingang verwenden, müssen Sie sich eigentlich nicht darum kümmern, wie die SPS diese Abfrage tatsächlich durchführt. Sie schreiben z.B. in FUP den Namen des Eingangs an einen UND-Block und meinen damit: wenn an dem Eingang Spannung anliegt, dann soll diese Bedingung als wahr gelten. Analoges gilt, wenn Sie in KOP den Namen eines Einganges über einen Kontakt schreiben.

Manchmal ist es aber doch nützlich zu wissen, was im Detail in der SPS geschieht. Hier kommt das Prozessabbild der Eingänge (PAE) ins Spiel. Dies ist ein Bereich im Speicher der SPS. Vor dem Aufruf des OB1 fragt das Betriebssystem alle vorhanden Eingangsbaugruppen einmal ab und schreibt die Ergebnisse in eben diesen Speicherbereich. Wenn das Programm dann bearbeitet wirdund darin Abfragen der Eingänge von Ihnen programmiert sind, werden die Eingangsbaugruppen nicht erneut abgefragt, sondern es wird der im PAE abgelegte Wert verwendet.

Dieses Verfahren hat zwei Vorteile:

1.) Es steigert die Bearbeitungsgeschwindigkeit der SPS enorm. Der Zugriff auf einen Speicher innerhalb der CPU geht sehr schnell, die Abfrage eines Eingangs einer Eingabebaugruppe dauert jedoch um ein Vielfaches länger. Der Geschwindigkeitsvorteil durch die Zwischenschaltung eines PAE ergibt sich nun zum einen dadurch, dass Eingänge in aller Regel innerhalb eines Programmes nicht nur einmal, sondern mehrfach verwendet werden. Zum anderen werden die Eingänge einer Digital-Eingabebauppe nicht einzeln, sondern in Gruppen (von z.B. acht) abgefragt.

2.) Sie können sich beim Programmieren darauf verlassen, dass sich der Wert eines Eingangs während eines Zyklus' nicht ändert. In kleinen Programmen kommt dieser Vorteil nicht so zum Tragen, in großen Programmen wird es dadurch aber sehr erleichtert, das genaue Verhalten der SPS vorherzusagen und so die Korrektheit eines Programmes zu überprüfen.

Prozessabbild der Ausgänge

Für das Prozessabbild der Ausgänge (PAA) gilt im Wesentlichen das Gleiche wie bei dem Abbild der Eingänge, bitte lesen Sie daher zunächst dort.

Wenn Sie einen Ausgang innerhalb des Programmes auf “1” setzen, wird die CPU nicht sofort eine Meldung die Ausgabebaugruppe senden, sondern zunächst nur das

entsprechende Bit im PAA setzen. Erst am Ende des Zyklus' wird der neue Wert zur Ausgabebaugruppe übertragen. Wenn Sie also einen Ausgang innerhalb des Zyklus' mehrfach auf “1” und dann wieder auf “0” setzen (was man i.A. vermeiden sollte), wird der Ausgang nicht flackern, sondern er wird auf den Wert gesetzt, den das Bit im PAA am Ende des Zyklus' hatte.

Strukturieren eines SPS-Programms

Es gibt viele Gründe, ein Programm (nicht nur ein SPS-Programm) zu strukturieren. In diesem Abschnitt wird ein Aspekt, nämlich die Vorteile bei der Fehlersuche in fremden Programmen und auch die Probleme, die trotzdem noch auftreten, besonders herausgehoben. Die Bücher zu diesem Thema füllen ganze Regalwände, wir können also wirklich nur die Idee grob umreißen.

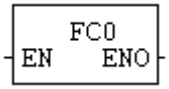
Wenn man ein SPS-Programm geschrieben hat, muss irgendwie festgelegt werden, wie dieses Programm von der SPS bearbeitet wird. Um dies zu erläutern, wählen wir hier stellvertretend für SPS im Allgemeinen die Siemens S7. Das Prinzip ist überall das Gleiche und auch, wenn die Namen jeweils verschieden sind, werden Sie sich schnell auf andere SPS umstellen können.

Wenn man das Programm in die SPS überträgt, kann man es in viele Bausteine unterteilen. Unter all diesen Bausteinen ist der Organisationsbaustein 1 (OB1) besonders ausgezeichnet. Nach dem Start der SPS und nachdem einige Vorbereitungen erledigt sind, liest sie die Digital-Eingänge und überträgt sie ins Prozessabbild der Eingänge. Anschließend ruft die SPS den OB1 auf, d.h., sie führt die Befehle, die dort stehen, einen nach dem anderen aus. Wenn sie damit fertig ist, überträgt sie alle Ausgänge des Prozessabbildes auf die entsprechenden Digital-Ausgabe-Baugruppen. Dann ruft sie wieder den OB1 auf. Dieser Vorgang wiederholt sich, bis die SPS gestoppt wird.

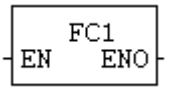
Der Programmierer muss also alles, was die SPS tun soll, im OB1 festlegen. Bei komplizierteren Programmen würde dies jedoch zu einem riesigen OB1 führen, der kaum noch zu verstehen und zu warten wäre. Darum gibt es die Möglichkeit, logisch zusammengehörige Teile eines Programmes in einer Funktion oder einem Funktionsbaustein zu schreiben. Im OB1 muss dann nur noch stehen: “Und jetzt führe bitte alle Befehle aus, die in der FunktionX aufgeführt sind”. Sind alle diese Befehle dann ausgeführt, wird die Bearbeitung im OB1 nach diesem Aufruf fortgesetzt. Im Idealfall besteht der OB1 dann nur noch aus einigen Bausteinaufrufen.

Für die Fehlersuche ist dies ungemein hilfreich. Denken Sie sich, sie sollten einen Fehler in der Steuerung einer Ihnen unbekannten Anlage suchen, der im Anlagenteil 2 aufgetreten ist. Sie schließen ihr Programmiergerät an und finden folgenden OB1:

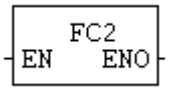
Aufruf Schnittstelle zum Bediengerät



Aufruf Steuerung Anlagenteil 1



Aufruf Steuerung Anlagenteil 2



Wo würden Sie den Fehler suchen? Natürlich im FC 2 und damit ist der Fehler schon einmal auf ein Drittel des Programmes eingegrenzt.
Wenn die Steuerung des Anlagenteils 2 auch wieder sehr umfangreich ist, kann es sein, dass der Programmierer diese nochmals in einzelne Funktionen und Funktionsbausteine untergliedert hat.

Dies ist die grundlegende Idee der strukturierten Programmierung. Jetzt folgt, warum die Fehlersuche in strukturieren Programm nicht ganz so einfach ist, wie es zunächst aussieht.

In der Theorie sollte es also möglich sein, mittels der strukturierten Programmierung übersichtliche und leicht zu verstehende Programme zu schreiben. In der Praxis sieht es leider häufig ganz anders aus. Und dafür gibt es mehrere Gründe:

Zum einen werden die beiden Anlagenteile ja nicht vollkommen unabhängig voneinander funktionieren. Und genauso, wie die wirklichen Anlagenteile z.B. Material austauschen, müssen die Programmbausteine, die für die Steuerung des jeweiligen Anlagenteils zuständig sind, Informationen austauschen. Eine Möglichkeit, dies zu bewerkstelligen sind die Parameter von Funktionen und Funktionsbausteinen, die die benötigten Kommunikationsmöglichkeiten bereitstellen, die aber die Fehlersuche auch dementsprechend schwieriger machen, weil die strenge Trennung der Bausteine und damit des möglichen Fehlerortes, aufgehoben wird.
Zum anderen werden oft die gleichen Bausteine von verschiedenen Stellen im Programm aufgerufen, auch wenn die von der entsprechenden Programmstelle kontrollierten Anlagenteile weit voneinander entfernt sind. Die Schwierigkeiten, die damit verbunden sind, sind im Kapitel Schwierigkeiten bei der Fehlersuche in mehrfach verwendeten Bausteinen beschrieben.

Außerdem muss man ganz klar sagen, dass es bei komplexeren Anlagen immer mehrere Möglichkeiten gibt, das Programm zu strukturieren. Bei der Fehlersuche muss man also zunächst einmal offen sein, man muss versuchen, die Prinzipien, die der Programmierer bei der Strukturierung seines Programmes zugrunde gelegt hat, zu verstehen. Diese Prinzipien können durchaus ganz andere sein, als die, die man selbst erlernt hat.

Vorbereitungen nach dem Start einer SPS

Wenn eine SPS eingeschaltet wird, sei es durch Anlegen der Versorgungsspannung oder durch Betätigung eines “Run”-Schalters, wird zunächst ein Selbsttest vorgenommen, von dem Sie als Anwender/Programmierer aber nichts

mitbekommen, solange er erfolgreich verläuft. Dann wird ein Anlaufbaustein (in einer S7-300 ist dies der OB100) aufgerufen. Hier können Sie programmieren, was vor dem ersten Start des zyklisch aufgerufenen OB1 einmalig geschehen soll, z.B. Initialisierungen aller Art. Für einfache Programme ist es häufig nicht notwendig, den Anlaufbaustein zu programmieren.

Aktoren, oder auch Aktuoren oder Stellglieder

Unter diesem Sammelbegriff fasst man alle Teile einer Anlage zusammen, die irgendeine Bewegung unmittelbar veranlassen (z.B. ein Motor) oder anders aktiv auf den Prozess einwirken (z.B. eine Heizung).

Aktoren werden sehr häufig nicht direkt an eine Ausgabebaugruppe einer SPS angeschlossen, sondern benötigen ein vermittelndes Schaltelement (Relais, Schütz), da die Ausgänge der SPS die notwendige Leistung zur Betätigung des Aktors nicht liefern können.

Im Gegensatz dazu stehen die Sensoren, die Informationen über den Prozess erfassen, aufbereiten und an die SPS weiterleiten.

Sensoren

Unter diesem Sammelbegriff werden alle Elemente einer Anlage zusammengefasst, die irgendwelche Informationen über den Prozess erfassen, entsprechend aufbereiten und über eine Eingabebaugruppe der SPS mitteilen.

Sensoren sind meistens so ausgelegt, dass sie direkt an eine Eingabebaugruppe angeschlossen werden können.

Im Gegensatz dazu stehen die Aktoren.

Parameter

Mit Parameter bezeichnet man in diesem Zusammenhang einen Platzhalter, der während des Programmierens einer Funktion oder eines Funktionsbausteines verwendet wird. Beim Schreiben des Programms muss man sich dann noch nicht festlegen, mit welchen Eingangswerten der Baustein arbeiten muss. Dies wird erst während der Laufzeit des Programmes, also dann, wenn die SPS arbeitet, ermittelt.

Schwierigkeiten bei der Fehlersuche in mehrfach verwendeten Bausteinen

Einer der großen Vorteile der strukturierten Programmierung ist es, dass sehr ähnliche Aufgaben der SPS von dem gleichen Baustein erfüllt werden können. Man braucht also ein Problem nur einmal zu lösen. Wenn ein ähnliches Problem auftaucht, verwendet man einfach den gleichen Baustein noch einmal. In diesen Fällen muss man dem Baustein dann über Parameter mitteilen, welches Problem aktuell gelöst werden soll.

Wenn man jedoch während einer Fehlersuche die Bearbeitung des Bausteins beobachtet, ergeben sich hieraus häufig Schwierigkeiten, weil nicht klar ist, welches Problem der Baustein gerade bearbeitet. Da SPS-Programme ja in aller Regel zyklisch bearbeitet werden, wird ein Baustein, der z.B. zweimal im Programm aufgerufen wird, jede Sekunde zig-mal für die Lösung des einen Problems und ebenso oft für die Lösung des anderen Problems bearbeitet. Da man beim Beobachten immer nur sehr viel weniger Momentbilder sehen kann, sieht man dem Baustein also in unregelmäßiger Folge mal bei der Lösung des einen und dann wieder des anderen Problems zu. Dies führt zu schnell springenden Anzeigen auf dem Bildschirm, die sich kaum interpretieren lassen. Es gibt natürlich im Programmiergerät Mechanismen, diese Schwierigkeit zu umgehen, aber das führt hier zu weit.

Menübefehle

7 Menübefehle

7.1 Projekt

Hier sind alle Funktionen aufgeführt, die in vielen Anwendungen auch unter “Datei” laufen.

Einige der Punkte haben in Abhängigkeit vom aktiven Fenster wechselnde Bedeutung: “Drucken” z.B. druckt immer das Objekt im aktuellen Fenster.

7.1.1 Projekt | Neu

Hiermit erzeugen Sie eine neue Anlage, die außer dem Ursprung keine Elemente und keine SPS-Programmbausteine enthält.

7.1.2 Projekt | Öffnen

Hiermit öffnen Sie ein bereits bestehendes Projekt. Sie brauchen das aktuelle Projekt nicht zu schließen, es wird automatisch geschlossen. Wenn Sie Änderungen vorgenommen haben, werden Sie gefragt, ob Sie diese speichern wollen. Beachten Sie bitte, dass bereits das Laufenlassen der Simulation eine Änderung bedeutet, da sich ja die Positionen der beweglichen Elemente geändert haben könnten.

Projekte werden von TrySim immer in einem Verzeichnis gespeichert. Sie müssen also keine besondere Projektdatei finden, sondern nur das Verzeichnis, in dem Ihr Projekt steht.

7.1.3 Projekt | Neu Öffnen

Hier werden alle zuletzt bearbeiteten Projekte angezeigt.

7.1.4 Projekt | Alles Speichern

Es werden die Elemente der Anlage mit ihren aktuellen Positionen gespeichert.

- Es werden alle offenen Programmbausteine gespeichert
- Es werden alle Datenbausteine mit den aktuellen Werten gespeichert

Abkürzung:  Die Funktionstaste F2 können Sie nur verwenden, wenn nicht eines der Programmbaustein-Fenster aktiv ist, denn hier hat F2 die Bedeutung “Und-Box” oder “Neuer Schliesser”.

7.1.5 Projekt | Speichern unter

Hier müssen Sie ein Verzeichnis angeben, in dem Ihr Projekt gespeichert werden soll. Wenn das Verzeichnis nicht existiert, wird es automatisch erzeugt. TrySim legt für jedes Projekt einen eigenen Order mit dem Namen des Projektes an. Sie können auch weitere Informationen in diesem Verzeichnis speichern, sollten aber keine von TrySim erzeugten Dateien löschen.

7.1.6 Projekt | Dateien hinzufugen

Dieser Menüpunkt ist ähnlich wie Importieren. Nützlich ist er vor allem dann, wenn Sie Programmbausteine aus bereits existierenden Projekten auch im aktuellen Projekt benötigen. Sie könnten genauso gut auf der Windows-Ebene die entsprechenden Dateien kopieren.

7.1.7 Projekt | Kommentar

Sie öffnen den Kommentar-Editor mit **Projekt|Kommentar**. Sie können den Kommentar auch mit jedem anderen Editor bearbeiten, der das rtf -Format beherrscht (z.B. MS-Word) oder gleich eine vorhandene Projekt-Definition als **PrjKom.rtf** im Verzeichnis Ihres Projektes abspeichern. Achten Sie dabei darauf, als Datei-Typ explizit “Rich Text Format” anzugeben, nur die Erweiterung .rtf führt bei einigen Textverarbeitungsprogrammen noch nicht zum richtigen Format.

7.1.8 Projekt | Exportieren

Sie können
- SPS-Programmbausteine
- Symboltabellen

exportieren. Der Anlagen-Export ist aber kein echter Export, sondern es wird nur die aktuelle TrySim-Anlage kopiert.

7.1.9 Projekt | Importieren

Sie können
- SPS-Programmbausteine
- Symboltabellen
- RsLogix-DataTypes

importieren. Der Anlagen-Import ist aber kein echter Import, sondern es wird nur eine bestehende TrySim-Anlage kopiert.

7.1.10 Projekt | Drucken

Hierdurch wird das Objekt im gerade aktiven Fenster gedruckt. Dies kann sein:

Die Anlage
Ein Programmbaustein
Ein Datenbaustein
Die Symboltabelle
Die Adressentabelle
Die 3D-Ansicht

Gedruckt wird immer mit dem jeweils aktuellen Zoom, Ansichtrichtung,

Darstellungsart, Sortierung usw.

Nicht gedruckt werden können:
Querverweisliste
Elementbaum

Wenn mehrere Programm- und Datenbausteine gedruckt werden sollen, sollten Sie SPS | Drucken wählen.

7.1.11 Projekt | Beenden

Hiermit beenden Sie TrySim.

7.2 Bearbeiten

Dieses Menü enthält verschieden Punkte, je nachdem, welche Art von Fenster gerade aktiv ist. Erwähnenswert sind besonders diejenigen, die sich auf die Anlage beziehen.

Alle Punkte dieses Menüs funktionieren auch, wenn die 3D-Ansicht das oberste Grafik-Fenster ist. Da in der 3D-Ansicht keine Elemente markiert werden können, muss die Markierung über den Elementbaum erfolgen. Bei einer gut strukturierten Anlage stellen die Punkte dieses Menüs ein mächtiges Werkzeug dar, um die 3D-Ansicht den Erfordernissen entsprechend zu konfigurieren.

7.2.1 Bearbeiten | Edit

Hierdurch wird die Editiermaske des oder der markierten Elemente aufgerufen. Die Wirkung ist die gleiche, wie ein kurzer Rechts-Klick auf ein markiertes Element oder ein etwas längerer Rechts-Klick mit anschließender Auswahl “Eigenschaften”.

7.2.2 Bearbeiten | Alle markieren

Hierdurch werden alle Elemente der Anlage markiert, d.h., auch diejenigen, die in dem aktuellen Fenster nicht sichtbar sind.

Falls Sie diese Funktion aus dem Elementbaum oder der Adressentabelle aufrufen, achten Sie bitte darauf, das markierte Elemente im Baum oder der Tabelle vor Ausführung der von Ihnen geplanten Aktion nicht mehr zu ändern. Sowohl im Baum als auch in der Tabelle kann nämlich nur ein Element markiert sein.

7.2.3 Bearbeiten | Alle Kinder markieren

Hierdurch werden die Kinder und Kindeskinder aller markierten Elemente ebenfalls markiert.

Nützlich ist dies, wenn danach die gemeinsame Editiermaske dieser Elemente aufgerufen wird (z.B. über Bearbeiten|Edit). Die gemeinsam editierbaren Eigenschaften von Elementen, die außer dem (Groß-) Vater nichts gemein haben, sind i.A. auf Fixierung und auf das Anzeigen der Beschriftung beschränkt. Die Sichtbarkeit kann zwar auch geändert werden, aber dies ist meistens schneller über Einblenden/Ausblenden-Punkte dieses Menüs zu bewerkstelligen.

Falls Sie diese Funktion aus dem Elementbaum oder der Adressentabelle aufrufen, achten Sie bitte darauf, das markierte Elemente im Baum oder der Tabelle vor Ausführung der von Ihnen geplanten Aktion nicht mehr zu ändern. Sowohl im Baum als auch in der Tabelle kann nämlich nur ein Element markiert sein.

7.2.4 Bearbeiten | Markierung umkehren

Hierdurch wird die aktuelle Markierung umgekehrt, d.h., markierte Elemente werden unmarkiert und unmarkierte Elemente werden markiert. Bitte beachten Sie, dass sich diese Operation auf alle Elemente der Anlage bezieht, d.h., auch auf diejenigen, die in dem aktuellen Fenster nicht sichtbar sind.

Falls Sie diese Funktion aus dem Elementbaum oder der Adressentabelle aufrufen, achten Sie bitte darauf, markierte Elemente im Baum oder der Tabelle vor Ausführung der von Ihnen geplanten Aktion nicht mehr zu ändern. Sowohl im Baum als auch in der Tabelle kann nämlich nur ein Element markiert sein.

7.2.5 Bearbeiten | Ausblenden

Hierdurch werden alle markierten Elemente in dem obersten Grafikfenster ausgeblendet. Die Wirkung ist die gleiche, als würden Sie die Editiermasken der Elemente öffnen und im Grafikfilter  das entsprechende Fenster abwählen.

Diese Funktion kann auch ausgeführt werden, wenn ein Element im Elementbaum oder (weniger wichtig) in der Adressentabelle markiert worden ist. Dies ist sehr nützlich, wenn Sie beim Editieren oder Testen großer Projekte das aktuelle Fenster von störenden Elementen bereinigen wollen. Wenn Sie die Anlage gut strukturiert haben (ein Blick in den Elementbaum zeigt Ihnen, ob das so ist), ist die Funktion Kinder ausblenden besonders effektiv.

7.2.6 Bearbeiten | Kinder ausblenden

Hierdurch werden alle Kinder und Kindeskinder des markierten Elementes in dem obersten Grafikfenster ausgeblendet. Die Wirkung ist die gleiche, als würden Sie die Editiermasken der Kinder des Elementes öffnen und im Grafikfilter  das entsprechende Fenster abwählen.

Diese Funktion kann auch ausgeführt werden, wenn ein Element im Elementbaum

oder (weniger wichtig) in der Adressentabelle markiert worden ist. Dies ist sehr nützlich, wenn Sie beim Editieren oder Testen großer Projekte das aktuelle Fenster von störenden Elementen bereinigen wollen. Wenn Sie die Anlage gut strukturiert haben (ein Blick in den Elementbaum zeigt Ihnen, ob das so ist), können Sie so mit einem Klick ganze Funktionsgruppen ausblenden.

Bitte lesen Sie auch die Hinweise zum Strukturieren großer Anlagen unter **Anlage | Arbeiten mit dem Anlageneditor | Gruppen | Strukturieren großer Anlagen**.

7.2.7 Bearbeiten | Einblenden

Hierdurch wird das markierte Element in dem obersten Grafikfenster eingeblendet. Die Wirkung ist die gleiche, als würden Sie die Editiermasken der Elemente öffnen und im Grafikfilter  das entsprechende Fenster anwählen.

Da unsichtbare Elemente in den Grafikfenstern nicht markiert werden können, ist diese Funktion nur auszuführen, wenn ein Element im Elementbaum oder (weniger wichtig) in der Adressentabelle markiert worden ist. Dies ist sehr nützlich, wenn Sie beim Editieren oder Testen großer Projekte im aktuellen Fenster ein bestimmtes Element angezeigt haben möchten. Wenn Sie die Anlage gut strukturiert haben (ein Blick in den Elementbaum zeigt Ihnen, ob das so ist), kann mit der Funktion Mit Kindern einblenden eine ganze Funktionsgruppe mit einem Klick eingeblendet werden.

7.2.8 Bearbeiten | Mit Kindern einblenden

Hierdurch werden alle markierten Elemente samt ihrer Kindern und Kindeskinder in dem obersten Grafikfenster eingeblendet. Die Wirkung ist die gleiche, als würden Sie die Editiermasken aller betroffenen Elemente öffnen und im Grafikfilter  das entsprechende Fenster anwählen.

Diese Funktion kann auch ausgeführt werden, wenn das Element im Elementbaum oder (weniger wichtig) in der Adressentabelle markiert worden ist. Dies ist sehr nützlich, wenn Sie beim Editieren oder Testen großer Projekte im aktuellen Fenster eine bestimmte Funktionsgruppe angezeigt haben möchten. Für einen effektiven Einsatz dieser Funktion ist es notwendig, dass Sie die Anlage gut strukturiert haben.

Bitte lesen Sie auch die Hinweise zum Strukturieren großer Anlagen unter **Anlage | Arbeiten mit dem Anlageneditor | Gruppen | Strukturieren großer Anlagen**.

7.2.9 Bearbeiten | In Bildmitte anzeigen

Hierdurch wird das markierte Element in der Mitte des obersten Grafikfensters angezeigt.

Diese Funktion ist eigentlich nur dann sinnvoll, wenn das Element im Elementbaum oder (weniger wichtig) in der Adressentabelle markiert worden ist. Auf dem Elementbaumfenster gibt es dafür auch einen eigenen Button “Focus”, der die gleiche Funktion hat, da es häufig nötig ist, ein Element, dass man im Elementbaum gefunden hat, auch räumlich zu lokalisieren.

Damit ein Element im Elementbaum schnell gefunden werden kann, muss die Anlage gut strukturiert sein. Lesen Sie daher bitte auch die Hinweise zum Strukturieren großer Anlagen unter **Anlage | Arbeiten mit dem Anlageneditor | Gruppen | Strukturieren großer Anlagen**.

7.3 Anlage

In diesem Menü finden Sie alles, was Sie brauchen, um die Maschine zu bauen und zu modifizieren. Außerdem können Sie hier die Simulation starten und stoppen, sowie die Simulationsgeschwindigkeit verändern und einen anderen Blickwinkel auf die Anlage einstellen.

7.3.1 Anlage | Start

Abkürzung:  -Symbol in der Symbolleiste.

Hierdurch wird der Editier-Modus beendet und die Simulation gestartet. Nun wird abwechselnd die Anlage simuliert und dann das SPS-Programm ausgeführt. Beim ersten Start der Simulation in der TrySim-Sitzung wird vor dem OB 1 einmal der Anlaufbaustein OB 100 ausgeführt.

Viele Erst-Benutzer von TrySim müssen sich erst daran gewöhnen, dass es meistens zwei Start - und zwei Stop- Knöpfe gibt. Mit den Buttons in der Symbolleiste oder über das Menü starten und stoppen Sie die Simulation. Mit den Tastern, die Sie selbst erstellen, wird dann die simulierte Maschine gestartet und gestoppt.

Im Einzelschrittmodus wird durch diesen Menüpunkt die Bearbeitung des SPS-Programms bis zum nächsten Breakpoint fortgesetzt.

7.3.2 Anlage | Stop

Abkürzung:  -Symbol aus der Symbolleiste.

Hierdurch wird die Simulation angehalten und die Anlage in den Editier-Modus geschaltet. Das SPS-Programm können Sie auch während laufender Anlage

editieren und übertragen.

Viele Erst-Benutzer von TrySim müssen sich erst daran gewöhnen, dass es zwei Start - und zwei Stop- Knöpfe gibt. Mit den Buttons in der Symbolleiste oder über das Menü starten und stoppen Sie die Simulation. Mit den Tastern, die Sie selbst erstellen, wird dann die simulierte Maschine gestartet und gestoppt.

Die Anlage wird auch gestoppt und in den Editiermodus geschaltet, wenn Sie eine neues Element hinzufügen, oder irgendwo im Weißen des Grafik-Fensters rechtsklicken.

Im Einzelschrittmodus wird durch diesen Menüpunkt das SPS-Programm bis zum Ende des aktuellen Zyklus' bearbeitet und die Simulation dann gestoppt. Wenn während der Programmbearbeitung ein weiterer Breakpoint angetroffen wird, wird dieser ignoriert.

7.3.3 Anlage | Langsamer

Abkürzung:  Schildkröte in der Symbolleiste.

Sie können die virtuelle Zeit der simulierten Maschine langsamer oder schneller laufen lassen. Dadurch können Sie kritische Momente in Ruhe analysieren, während Sie Prozesse, die bereits fehlerfrei laufen, schnell vorbeiziehen lassen. Die Genauigkeit der Simulation wird durch die Zeitlupe oder den Zeitraffer nicht beeinflusst, die Simulationswiederholrate bleibt immer auf dem unter **Ansicht|Optionen|Simulation** eingestellten Wert.

Sie können die Simulationsgeschwindigkeit auch automatisch mit dem Speed-Trigger verändern lassen.

7.3.4 Anlage | Schneller

Abkürzung: Kaninchen in der Symbolleiste.

Sie können die virtuelle Zeit der simulierten Maschine langsamer oder schneller laufen lassen. Dadurch können Sie kritische Momente in Ruhe analysieren, während Sie Prozesse, die bereits fehlerfrei laufen, schnell vorbeiziehen lassen. Die Genauigkeit der Simulation wird durch die Zeitlupe oder den Zeitraffer nicht beeinflusst, die Simulationswiederholrate bleibt immer auf dem unter **Ansicht|Optionen|Simulation** eingestellten Wert.

Sie können die Simulationsgeschwindigkeit auch automatisch mit dem Speed-Trigger verändern lassen.

7.3.5 Anlage | Echte Zeit

Hierdurch wird eine eventuell aktivierte Zeitlupe oder Zeitraffer deaktiviert, die virtuelle Zeit in der Statusleiste läuft dann so, wie Sie es gewohnt sind.

Der Short-Key F8 funktioniert nicht, wenn ein FUP- oder KOP-Baustein aktiv ist, dort bedeutet er “Neuer Anschluss” bzw. “Verzweigung öffnen”.

Sie können die Simulationsgeschwindigkeit auch automatisch mit dem Speed-Trigger verändern lassen.

7.3.6 Anlage | Erweitert

Hier können Sie Geschwindigkeit der Simulation in den Stufen 100: 1, 10: 1, 1: 1, 1: 10 und 1: 100 einstellen.

Zeitlupe

Sie können die virtuelle Zeit der simulierten Maschine langsamer oder schneller laufen lassen. Dadurch können Sie kritische Momente in Ruhe analysieren, während Sie Prozesse, die bereits fehlerfrei laufen, schnell vorbeiziehen lassen. Die Genauigkeit der Simulation wird durch die Zeitlupe oder den Zeitraffer nicht beeinflusst, die Simulationswiederholrate bleibt immer auf dem unter Ansicht| Optionen|Simulation eingestellten Wert.

Zeitlupe erhalten Sie auch durch (mehrfaches) Klicken auf die Schildkröte  .

Sie können die Simulationsgeschwindigkeit auch automatisch mit dem Speed-Trigger verändern lassen.

Zeitraffer

Sie können die virtuelle Zeit der simulierten Maschine langsamer oder schneller laufen lassen. Dadurch können Sie kritische Momente in Ruhe analysieren, während Sie Prozesse, die bereits fehlerfrei laufen, schnell vorbeiziehen lassen. Die Genauigkeit der Simulation wird durch die Zeitlupe oder den Zeitraffer nicht beeinflusst, die Simulationswiederholrate bleibt immer auf dem unter Ansicht | Optionen | Simulation eingestellten Wert.

Zeitraffer erhalten Sie auch durch (mehrfaches) Klicken auf das Kaninchen  .

Die Bezeichnung “1: 100” verstehen Sie bitte als “So schnell wie möglich”. Auch einfache Anlagen kann TrySim selbst auf schnellen Rechnern kaum mehr als mit 50-facher Geschwindigkeit laufen lassen (bei einer Simulationsrate von 100 ms).

Sie können die Simulationsgeschwindigkeit auch automatisch mit dem Speed-Trigger verändern lassen.

7.3.7 Anlage | Bibliothek

Häufig benötigte Anlagenteile können Sie in einer Bibliothek speichern. Markieren Sie dazu die zusammengehörigen Teile und wählen Sie **Anlage|Bibliothek| Speichern Als.** Beim Laden eines Anlagenteils aus der Bibliothek werden die SPS-Adressen nach Rückfrage entweder provisorisch belegt, Sie müssen dann

noch Ihren Anforderungen angepasst werden, oder die Adressen werden unverändert übernommen. Die Namen der eingefügten Elemente werden mit einem _Nr versehen, um Eindeutigkeit zu erreichen.

Um die Namen der neu eingefügten Elemente schnell anzupassen, können Sie vor dem Abspeichern eines Moduls Platzhalter in die Namen einfügen. Ein Platzhalter beginnt mit dem Zeichen “$” und endet beim nächsten Leerzeichen. Wenn Sie das Modul später einfügen, erscheint eine Liste mit allen verwendeten Platzhaltern. Jetzt können Sie für jeden Platzhalter einen neuen Wert eingeben, der dann stattdessen bei allen eingefügten Elementen verwendet wird. Nützlich ist dies vor allem dann, wenn die Elemente Ihres Moduls eine Gruppen-Nr. haben, die bei jedem Einfügen des Moduls einen anderen Wert erhält.

Bislang ist nur ein Platzhalter pro Name möglich.

Bitte beachten Sie, dass die Bauteile z.Z. an genau der Position eingefügt werden, an der sie sich beim Abspeichern befunden haben. Das ist nicht schön, aber wir sind bislang noch nicht dazu gekommen, den Einfügeort z.B. per Maus festlegen zu lassen. Bis es soweit ist, sollten Sie nur Elemente in der Bibliothek speichern, die sich nahe des Ursprungs (unten links) befinden, sonst kann es Ihnen bei einer zu kleinen Anlage, in die eingefügt wird, passieren, dass die neuen Elemente außerhalb des sichtbaren Bereiches landen.

Dieser Menüpunkt ist nur verfügbar, wenn eines der Grafik-Fenster aktiv ist.

7.3.9 Anlage | Medien

**

Innerhalb eines TrySim - Projektes kann es bis zu 16 verschiedene Medien geben, die jeweils einen Namen haben und durch ihre Dichte und Viskosität beschrieben werden. Sie erzeugen ein neues Medium, indem Sie entweder den Medien-Manager aufrufen, oder indem Sie auf der Editiermaske eines Behälters bei einem der Medien “Neues Medium” wählen.

Die Flüssigkeiten in TrySim werden als Mischung aus diesen 16 Medien betrachtet, die Dichte und die Viskosität werden durch eine nach Anteil gewichtete Mittelwertbildung über alle Medien berechnet. Falls das für Ihre Anwendung eine zu starke Vereinfachung ist, können Sie den Reaktor verwenden, um ein neues Medium mit den gewünschten Eigenschaften zu erzeugen.

Medien-Manager

**

Sie erreichen den Medien-Manager entweder über **Anlage|Medien** oder über den Button “Medien-Manager” auf den Editiermasken der Behälter.

Der Medien-Manager dient dazu, Medien neu zu definieren, zu editieren und zu löschen.

Wenn Sie Medien löschen wollen, müssen Sie den Medien-Manager über **Anlage| Medien** aufrufen.

7.3.10 Anlage | Dynamiks | ... löschen

Durch Generatoren werden Dynamiks erzeugt und durch Vernichter sollten Sie sie nach dem Durchlauf durch die Maschine wieder vernichten lassen. Am Anfang schafft es aber kaum ein Dynamik, störungsfrei durch die Maschine zu laufen. Wäre dies anders, hätten wir keinen Grund gehabt, TrySim herzustellen. Irgendwann liegt alles voll mit verirrten Dynamiks und mit diesem Menüpunkt beseitigen Sie alle auf einmal.

Sie können bei stehender Simulation auch einzelne oder mehrere Dynamiks löschen: Markieren und “Entf” drücken, oder sie mit der Maus an ihren richtigen Platz zurückstellen.

7.3.11 Anlage | Reset Zeit

Hierdurch wird die unten in der Statuszeile laufende virtuelle Zeit zurückgesetzt.

7.4 Grafik

7.4.1 Grafik | Eigenschaften

Sie können bis zu 16 Fenster verwenden, die jeweils einen anderen Ausschnitt, Blickwinkel und Zoom haben. Auf den Editiermasken der Elemente können Sie über den Grafik-Filter festlegen, welche Elemente in den einzelnen Fenstern angezeigt werden sollen. Die Fenster aktivieren Sie über das Menü **Grafik**.

Der erste Punkt dieses Menüs ist **Grafik|Eigenschaften**, mit dem Sie den Eigenschaften-Editor der gerade aktiven Grafik aufrufen. Mit diesem Editor können Sie den Namen des Fensters eingeben und festlegen, aus welcher Blickrichtung das Fenster die Anlage darstellen soll. Die Blickrichtung kann auch eingefroren werden, um eine versehentliche Änderung zu verhindern.

Weiterhin können Sie festlegen, welche Element-Typen dieses Fenster aufnehmen soll.
Dabei gibt es 3 Möglichkeiten: Bei “Immer” wird der entsprechende Element-Typ in jedem Fall in die Liste der anzuzeigenden Elemente aufgenommen. Bei “Wenn offen” wird dieser Element-Typ nur aufgenommen, wenn das Fenster während der Erzeugung des Elementes gerade geöffnet ist. Bei “Nie” wird der Element-Typ nicht in die Liste aufgenommen. Beachten Sie, dass Änderungen dieser Einstellungen nur für zukünftige Elemente gelten. Sie können aber mittels der Buttons “Jetzt” die Einstellungen auch für bereits erzeugte Elemente wirksam machen.

Mit den Checkboxen “Fix vert. Scrollbar” und “Fix horz. Scrollbar” können Sie den aktuellen Ausschnitt einfrieren, sodass er nicht versehentlich mit der Maus verstellt werden kann. Wenn auch nur eine dieser Checkboxen angeklickt ist, können Sie

weder den Zoom, noch die Blickrichtung des Fensters ändern.

Die Grafik-Fenster 1 bis 9 können Sie auch über die Shortkeys **Alt+1** bis **Alt+9** aufrufen.

7.5 SPS

In diesem Menü finden Sie alles, was Sie brauchen, um das SPS-Programm zu editieren.

7.5.1 SPS | Reset SPS

Hierdurch werden alle Eingänge, Ausgänge, Merker, Zeiten und Zähler, mit Ausnahme der unter CPU-Eigenschaften ausgenommenen, auf “0” zurückgesetzt. Es werden keine Bausteine aus dem Speicher der SPS gelöscht, die Daten in den Datenbausteinen bleiben erhalten.

Siehe auch:
SPS Urlöschen

7.5.2 SPS | Urlöschen

Hiermit löschen Sie alle Bausteine aus dem SPS-Speicher und setzen alle Eingänge, Ausgänge, Merker, Zeiten und Zähler zurück. Die SPS befindet sich danach in dem gleichen Zustand, wie nach dem Starten von TrySim.

7.5.3 SPS | Bausteine im AS

Hier werden alle Bausteine in der SPS mit ihrer Größe angezeigt. Die angegebene Größe (in Bytes) entspricht nicht derjenigen, die der Baustein in einer S7 haben würde, sondern ist nur ein grober Anhaltspunkt.

7.5.4 SPS | CPU

Wenn Sie auf CPU klicken, wird der aktuelle SPS-Zyklus (OB2, OB1, OB3) noch beendet, danach werden die Akkus und das VKE sowie weitere Informationen aus der CPU ausgelesen und in einem kleinen Fenster angezeigt. Die Anlage und das SPS-Programm laufen weiter, die Daten im Fenster sind nur eine Momentaufnahme.

Wenn Sie die CPU-Daten an einem bestimmten Punkt im SPS-Programm wissen möchten, setzen Sie einen breakpoint. Dann stoppen die SPS und die Anlage an dieser Stelle und das CPU-Fenster erscheint mit den aktuellen Werten. Jedesmal, wenn Sie auf “Weiter” klicken, wird ein Zyklus der SPS und eine Anlagensimulation durchgeführt. Wenn Sie die Simulation normal fortsetzen möchten, müssen Sie den breakpoint entfernen und den Baustein erneut übertragen (oder auf das Auge klicken, dadurch wird vor dem Aktivieren des Beobachten-Modus' automatisch

übertragen).

7.5.5 SPS | CPU Eigenschaften

Sie erreichen diese Einstellungen über **SPS|CPU-Eigenschaften**.

Registerkarten:
Taktmerker
Weckalarme
Remanenz
Zykluszeitüberwachung
Zeitsystem, CPU-Uhr

Taktmerker

Diese Registerkarte ist unter SPS|CPU-Eigenschaften zu erreichen.

Wenn Sie die Check-Box “aktiviert” anklicken, können Sie ein Merker-Byte angeben, dessen Bits vom Betriebssystem der virtuellen SPS periodisch an- und ausgeschaltet werden.
Die Frequenzen / Periodendauern (in der virtuellen Zeit) der einzelnen Bits sind:

Bit 0	10	Hz	0,1 s
Bit 1	5	Hz	0,2 s
Bit 2	2,5	Hz	0,4 s
Bit 3	2	Hz	0,5 s
Bit 4	1,25	Hz	0,8 s
Bit 5	1	Hz	1,0 s
Bit 6	0,625	Hz	1,6 s
Bit 7	0,5	Hz	2,0 s

Wenn Sie mit den Taktmerkern einen blinkenden Leuchtmelder ansteuern wollen, sollten Sie die Sonderform des Leuchtmelders Blinker verwenden, da das Blinken wegen der variablen Simulations-Geschwindigkeit sonst schlecht zu sehen ist.

Bei Verwendung des 10Hz - Bits beachten Sie bitte, dass die Simulationsrate standardmäßig auf 100 ms eingestellt ist und dass bei Beibehaltung dieser Einstellung das Bit immer “1” oder “0” sein wird. Unter Ansicht|Optionen|Simulation können Sie die Simulationsrate ändern.

Weckalarme

Diese Registerkarte ist unter SPS|CPU-Eigenschaften zu erreichen.

Der OB 35 wird, unabhängig von der eingestellten Simulationsrate, in den hier vorgegebenen Zeitabständen aufgerufen. Weitere Weckalarme sind noch nicht vorhanden.

Remanenz

Diese Registerkarte ist unter SPS|CPU-Eigenschaften zu erreichen.

Wenn Sie den Menüpunkt Reset SPS anwählen, werden alle Merker auf “0” gesetzt, ebenfalls alle Zeiten und Zähler. Ausgenommen davon sind die Bereiche, die Sie hier angegeben.

Zykluszeitüberwachung

Diese Registerkarte ist unter SPS|CPU-Eigenschaften zu erreichen.

Die Zykluszeitüberwachung spricht an, wenn das SPS-Programm mehr als 1 sec in echter Zeit benötigt. In den allermeisten Fällen haben Sie dann eine Endlosschleife programmiert. Sie ist standardmäßig nicht aktiviert, da es nicht ausgeschlossen ist, dass sie gelegentlich zu ungeklärten Abstürzen führt. Eine Änderung dieser Option wird erst nach einem Neustart von TrySim wirksam.

Wenn Sie eine Endlosschleife programmiert haben, ohne dass die Zykluszeitüberwachung aktiviert war und Sie zudem noch die Option “Automatisch Starten” (Ansicht|Optionen|Anlage) angewählt haben, dann tritt folgendes Problem auf: Sofort nach dem Starten von TrySim wird auch das SPS-Programm gestartet und hängt in der Endlosschleife.
Starten Sie TrySim dann mit der Option -n, dann wird kein Projekt mehr automatisch gestartet. Wenn Sie nicht wissen, wie man ein Programm unter Windows mit einer Option startet, dann geben Sie im DOS-Fenster ein: \programme\trysim\trysim -n. (vorausgesetzt, Sie haben TrySim in dem vorgeschlagenen Verzeichnis installiert, sonst müssen Sie den Pfad entsprechend anpassen).

Zeitsystem

Diese Registerkarte ist unter SPS|CPU-Eigenschaften zu erreichen.

In der CPU läuft eine Uhr, die beim Start von TrySim auf das aktuelle Datum und die aktuelle Zeit gesetzt wird. Diese Uhr läuft aber nur, wenn auch die Simulation läuft. Sie können die Uhrzeit sowohl unter SPS|CPU_Eigenschaften stellen als auch mit der SFC 0. (Vorher mit der IEC-Funktion FC3 eine lokale DATE_AND_TIME richtig belegen). Mit der SFC 1 können Sie die Uhrzeit auslesen.

7.6 Baustein

7.6.1 Baustein | Öffnen

Hierdurch öffnen Sie einen der Programmbausteine. Im Dialogfenster können Sie filtern, welche Bausteinarten angezeigt werden sollen.

Dieser Punkt wird auch zum Löschen von Programmbausteinen verwendet. Markieren Sie im Dialogfenster den zu löschenden Baustein und drücken Sie auf “Entf”, danach verlassen Sie das Dialogfenster über “Abbrechen”.

7.6.2 Baustein | Neu Öffnen

Neben diesem Menüpunkt werden die zuletzt geöffneten Bausteine angezeigt, damit Sie häufig benötigte Bausteine nicht im Baustein-öffnen-Dialog auswählen müssen.

7.6.3 Baustein | Speichern unter

Der aktuelle Baustein wird unter einem anderen Namen gespeichert. Nützlich, um eine Kopie des aktuellen Bausteines zu erstellen, die dann nur geringfügig modifiziert werden soll, oder um eine Sicherungskopie anzulegen vor Änderungen, von denen man nicht weiß, ob man sie später nicht bereuen wird.

Zur Zeit ist eine Änderung des Bausteintyps (OB,FB,FC) bei "Speichern unter" noch nicht möglich. Wenn Sie dies wünschen, müssen Sie jedes Netzwerk einzeln aus dem alten Baustein kopieren (**SPS|Netzwerk|kopieren**) und in dem neuen Baustein wieder einfügen (**SPS|Netzwerk|einfügen**).

7.6.4 Baustein | Drucken

Wählen Sie **SPS|Drucken.** Es erscheint ein Auswahlfenster mit zwei Feldern. Markieren Sie die zu druckenden Bausteine auf der linken Seite und bewegen Sie sie mit den Pfeil-Buttons auf die rechte Seite. Sie können auch mehrere Bausteine markieren, indem Sie sie nacheinander anklicken. Die Bausteine im rechten Feld werden in der Reihenfolge gedruckt, wie sie ausgewählt wurden.

Symbolisch/Absolut Sie können wählen, ob die Bausteine in der Darstellungsform ausgedruckt werden, in der Sie sie das letzte Mal gespeichert haben, oder Sie können erzwingen, dass symbolisch bzw. absolut ausgedruckt wird.

AWL/FUP/KOP Sie können wählen, ob die Netzwerke in der Sprache ausgedruckt werden, in der Sie sie das letzte Mal gespeichert haben, oder Sie können eine der Sprachen erzwingen. Im ersten Fall kann die Sprache dann von Netzwerk zu Netzwerk verschieden sein. Wenn Sie den Ausdruck in FUP oder KOP erzwingen wollen, sich ein Netzwerk aber nicht so darstellen lässt, wird es in AWL ausgedruckt.

Symbol-Kommentar Diese Option hat nur in AWL eine Funktion. Wenn sie angewählt ist, wird für alle Zeilen, die keinen Zeilenkommentar haben, stattdessen der Kommentar des Operanden aus der Symboltabelle ausgedruckt.

Wenn Sie nur den gerade bearbeiteten Baustein ausdrucken wollen, wählen Sie **Projekt|Drucken**. Der Baustein wird so ausgedruckt, wie Sie ihn sehen, d.h., mit der aktuellen Einstellung von “mit/ohne symbolische Darstellung” und der von Ihnen für jedes Netzwerk vorgegebenen Darstellungsart AWL/FUP/KOP.

7.6.5 Baustein | Bausteineigenschaften

Das wichtigste Feld der Bausteineigenschaften ist der Baustein-Kommentar. Der Menüpunkt ist nur vorhanden, wenn ein SPS-Fenster aktiv ist.

7.7 Ansicht

Die meisten Funktionen, die unter diesem wechselnden Menü angeboten werden, sind selbsterklärend.

7.7.1 Ansicht | Symbolleiste

Hiermit können Sie die Sichtbarkeit der Symbolleiste an- und abschalten. Wir empfehlen aber, die Symbolleiste immer angeschaltet zu lassen, denn sie beschleunigt nach der Eingewöhnungsphase die Arbeit mit TrySim erheblich. Lediglich Nutzer, die keine höhere Auflösung als 640 x 480 wählen können und die daher jeden verfügbaren Quadradzentimeter benötigen, sollten die Symbolleiste ausschalten und sich stattdessen die Short-Keys merken.

7.7.2 Ansicht | Statusleiste

Hier können Sie wählen, ob die Statusleiste des TrySim-Hauptfensters angezeigt werden soll. In der Statusleiste sind Anzeigen für den Anlagenstatus (grün: Simulation läuft, rot: Simulation steht, gelb: Einzelschrittmodus) sowie die virtuelle Zeit und etwaige Meldungen.

7.7.3 Ansicht | Dimension

Mit diesem Menüpunkt wählen Sie aus, aus welcher Richtung die Anlage im aktuellen Fenster angezeigt werden soll.

XY :	Aufsicht von oben	
YZ :	Ansicht von rechts	(ausgehend von der Aufsicht)
XZ :	Ansicht von “unten”	(ausgehend von der Aufsicht)

Wenn Sie eine 3D-Ansicht der Anlage wünschen, wählen Sie: **Anlage|3D-Ansicht**.

7.7.4 Ansicht | Symbolische Darstellung

Wenn Sie den Operanden des SPS-Programms in der Symboltabelle Namen zugeordnet haben, werden diese anstelle der absoluten Adressen angezeigt. Durch Abwahl der “symbolischen Darstellung” werden wieder die absoluten Adressen angezeigt.

Wenn Sie nur von einem einzelnen Operanden die absolute Adresse wissen wollen, ist es günstiger, ihn anzuklicken oder den Cursor auf ihn zu stellen: Dann wird die absolute Adresse in der Statusleiste angezeigt.

7.7.5 Ansicht | Zoom anpassen

Hierdurch wird das Bild der Anlage soweit vergrößert oder verkleinert, dass es gut in das aktuelle Fenster passt.

Danach wird automatisch Anlage | Fenster anpassen ausgeführt, dadurch wird das Höhen-/ Breiten-Verhältnis des Fensters an das der Anlage angepasst und es bleibt mehr Platz für andere Ansichten oder SPS-Bausteine.

Die Größe der Anlage können Sie unter Ansicht | Optionen | Anlage einstellen.

7.7.6 Ansicht | Fenster anpassen

Hierdurch wird das aktuelle Fenster genau so groß gemacht, wie die darin dargestellte Anlage. Der graue Rand, der noch bleibt, ist notwendig, damit Windows nicht automatisch die Scrollbars einblendet.

Die Größe der Anlage können Sie unter Ansicht | Optionen | Anlage einstellen.

7.7.7 Ansicht | Beobachten (bei akt. DB-Fenster)

Wenn Sie einen DB öffnen und sich die aktuellen Daten in der **Datenansicht** anzeigen lassen, werden die Daten einmal aus der SPS gelesen und auf den Bildschirm geschrieben. Die Daten können sich aber ändern, während die Simulation läuft. Wenn Sie diesen Menüpunkt wählen, werden die angezeigten Daten ca. jede Sekunde neu aus der SPS gelesen.

7.8 Extras

7.8.1 Extras | Quick Kom/Script

Dieses “Stay on top”-Fenster (Extras|Quick Kom/Script) ist hilfreich, um einen schnellen Überblick über die zu Elementen verfassten Kommentare und Scripte zu bekommen.

Normalerweise wird immer der Kommentar/das Script des aktuell markierten Elements gezeigt, das Sie wechseln können, ohne das Fenster zu schließen. Mit dem Button “Freeze” kann die Anzeige eingefroren werden, z.B. um ähnliche Scripte mit Kopieren und Einfügen von einem Element zum anderen zu übertragen.

Editieren ist in diesem Fenster nicht möglich.

7.8.2 Extras | Quick Selected

Dieses “Stay on top”-Fenster ist hilfreich, um einen schnellen Überblick über die ausgewählten Elemente zu erhalten.

So können Sie Elemente auswählen:

- Klicken Sie im Modus "Stop" auf ein Element in einem 2D-Grafik-Fenster
- Klicken Sie auf ein Element im Elementbaum
- Zum Hinzufügen weiterer Elemente halten Sie die Großschreib- oder Steuerungstaste gedrückt
- Mehrere Elemente können Sie auch durch Einkreisen mit der Maus markieren
- Im Menü “Bearbeiten” gibt es den Punkt “Alle Kinder markieren”

So heben Sie die Auswahl wieder auf:
- Klicken Sie zweimal (kein Doppelklick!) irgendwo in das Weiße eines Grafik-Fensters
- Klicken Sie bei gedrückter Steuerungstaste auf ein bereits markiertes Element
- Klicken Sie auf ein Element in dieser Schnellanzeige

So öffnen Sie die Editiermaske:
- Bei mehreren markierten Elementen ist es günstiger, auf die Enter-Taste zu drücken
- Kurzer Rechts-Klick im Grafikfenster
- Etwas längerer Rechts-Klick im Grafikfenster und dann “Eigenschaften” auswählen
- Bei vielen Elementen und im Elementbaum geht auch ein Doppelklick. In den Grafikfenstern hat ein Doppelklick für einige Elemente aber eine andere Bedeutung (z.B. Digitalanzeige und Streifen)

Wenn Sie häufig gleiche Gruppen von Elementen markieren wollen, lesen Sie bitte unter Anlage | Arbeiten mit dem Anlageneditor | Gruppen.

Editieren ist in diesem Fenster nicht möglich.

Unter Extras|Optionen können Sie einstellen, ob das Fenster beim Start automatisch geöffnet werden soll.

7.9 Fenster

Dieses Menü ist genau wie in allen Windows-Anwendungen.

7.10 Hilfe

Wie man die Hilfe benutzt, kann man nur durch Ausprobieren erlernen.

Optionen

8 Optionen

8.1 Übersicht

Zu den Optionen gelangen Sie über **Extras|Optionen**.

Simulation	Simulationsrate, Grafikfrequenz
Verzeichnisse	Projektverzeichnis
Symbolleisten	Aktive Symbolleisten
Anlage	Anlagengröße, Fang, Autostart
SPS-Editor	Darstellung, Beobachten, Speichern

8.2 Optionen | Simulation

Simulationswiederholrate

In TrySim läuft eine virtuelle Zeit mit der Auflösung 1 ms. Hier können Sie einstellen, wie viele der virtuellen ms vergehen sollen, bevor die Anlage erneut simuliert und der OB 1 des SPS-Programms erneut bearbeitet wird. In der Wirklichkeit entspricht dieser Punkt etwa der Zykluszeit der SPS. Je kleiner Sie diese Zahl stellen, desto genauer ist die Simulation, bei umfangreichen Anlagen sinkt dadurch aber der maximale Zeitraffer-Faktor.

Durch die Menüpunkte **Anlage|Schneller/Langsamer** wird die Simulationsrate nicht geändert, hierdurch wird nur die Geschwindigkeit der virtuellen Uhr beeinflusst.

!WICHTIG! Die Simulationswiederholrate beeinflusst ganz unmittelbar die Geschwindigkeit, mit der TrySim arbeiten kann. Wenn Sie die Simulationsrate auf 1 ms stellen, dann muss TrySim 100 mal mehr rechnen als wenn sie auf 100 ms stehen würde. Wenn TrySim auf Ihrem Rechner zu langsam läuft, dann prüfen Sie bitte, ob Sie nicht eine viel zu kleine Simulationswiederholrate eingestellt haben. Manchmal ist eine schnelle Simulation nur für ein einziges Element notwendig. Bitte fragen Sie uns in solchen Fällen nach Lösungsmöglichkeiten. Sie erreichen uns unter der Telefonnummer +49 4961 916393 • Faxnummer +49 4961 67469 oder per E-mail unter info@cephalos.de.

CPU-Overspeed

Hiermit können Sie einstellen, wie oft der OB 1 zwischen zwei Anlagensimulationsschritten bearbeitet werden soll. Im Allgemeinen ist es nicht nützlich, den OB 1 mehrfach zwischen zwei Simulationsschritten laufen zu lassen, denn die Eingänge haben dann ja immer den gleichen Wert und die Ausgänge werden nicht ausgewertet, in Einzelfällen macht es vielleicht dennoch Sinn.

Grafikwiederholrate

Hier können Sie einstellen, wie oft die Grafik neu gezeichnet wird. Das Zeichnen kostet trotz der einfachen Grafik von TrySim eine Menge Zeit. Wenn Sie große Projekte programmiert haben und Wert auf einen großen Zeitrafferbereich legen, sollten Sie eine relativ langsame Grafikwiederholrate einstellen. Wenn kleine Projekte mit flüssigen Bewegungen dargestellt werden sollen, muss die Grafikwiederholrate möglichst klein eingestellt werden.

Welche Einstellung optimal ist, hängt von der Taktrate Ihres Rechners und der Anlagengröße ab, aber eine zu klein eingestellte Grafikwiederholrate schadet i.A. nicht, da TrySim automatisch seltener zeichnet, wenn es wegen zu großer Simulationsgeschwindigkeit überlastet ist.

8.3 Optionen | Verzeichnisse

Diese Registerkarte hat nur einen Eintrag auf dem Sie voreinstellen können, unter welchem Pfad Ihre Projekte gespeichert werden.

8.4 Optionen | Anlage

Maßeinheiten
Die Einheit mm ist jetzt voreingestellt. Die anderen in früheren Versionen verfügbaren Einheiten werden nicht mehr unterstützt.

Größe der Anlage
Hiermit geben Sie den weiß dargestellten Teil der Anlage vor. Sie können diese Werte jederzeit ohne Schaden für die Anlage ändern. Es empfiehlt sich, diese Werte ungefähr entsprechend der tatsächlichen Anlagengröße einzustellen, da dann mit dem Befehl **Ansicht|Fenster anpassen** leichter Ordnung in ein Wirrwarr von Fenstern gebracht werden kann.

Automatisch starten
Wenn diese Option aktiviert ist, wird die geladene Anlage sofort nach dem Starten von TrySim in Bewegung versetzt. Sie ist besonders nützlich für die Weitergabe der kostenlosen Laufzeitversion von TrySim an Endkunden, damit in deren Hause die Funktion zukünftiger Anlagen geprüft und diskutiert werden kann.

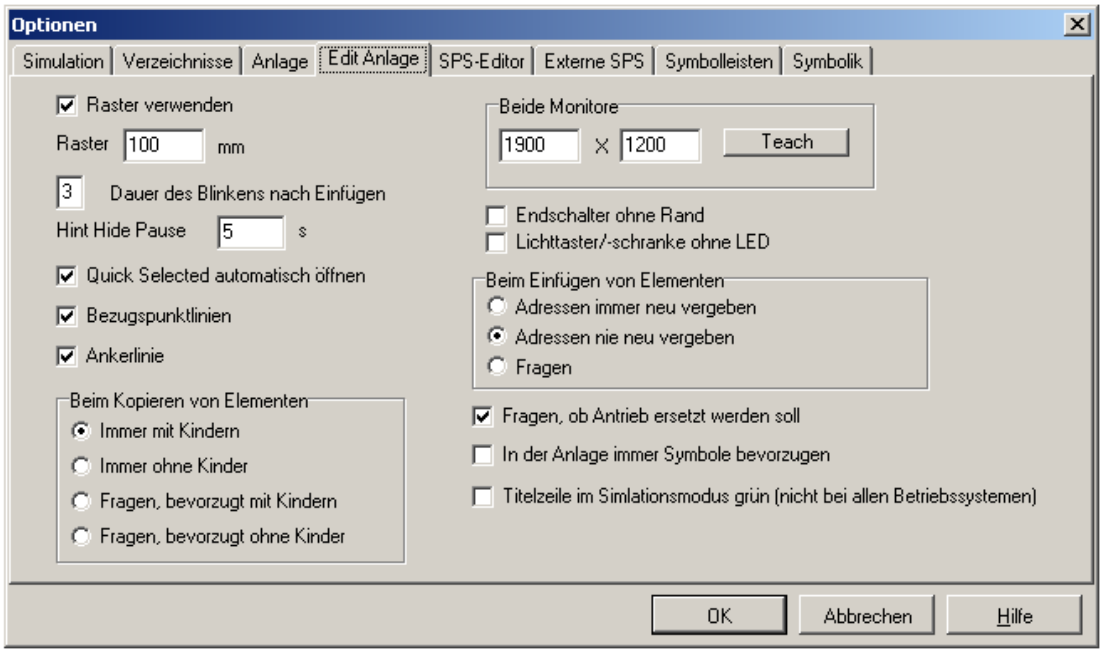
Extrem störend ist diese Option, wenn Sie eine Endlosscheife programmiert haben und die Zykluszeitüberwachung nicht aktiviert ist: sofort nach dem Start hängt sich TrySim auf. Starten Sie TrySim in diesem Fall mit dem Parameter “N” (für Nicht automatisch starten).

Default-Nennweite (Rohre u.ä)
Dieser Wert wird für neue Rohre, Pumpen und Ventile verwendet.

Plattformdicke
Wenn Dynamiks auf irgendeiner Plattform stehen (Förderband, anderes Dynamik, Platte, usw.) muss festgelegt werden, wie dick die aktive Schicht dieser Plattform ist. Die aktive Schicht liegt (unsichtbar) unter der oberen Fläche der Plattform. Befinden sich die unteren Ecken des Dynamiks tiefer als diese Schicht, z.B., wenn es von der Seite gegen die Plattform geschoben wird, wird das Dynamik nicht mehr angehoben. Ohne diese Einrichtung sind die Dynamiks häufig auf andere hinaufgesprungen, was sehr unrealistisch war. Bis zur Version V2.9 war dieser Wert fest auf 40 mm eingestellt. Wenn man ihn zu klein macht, fallen die Dynamiks gelegentlich durch die Plattform hindurch. Macht man ihn zu groß, tritt das o.g. Springen auf.

Ecken nach innen
Beim Aufeinanderstapeln von Dynamiks muss mindestens eine Ecke auf dem darunterliegenden Dynamik stehen, sonst fangen größere Stapel an zu “zappeln”. Wenn die Dynamiks etwas unterschiedliche Größe haben, ist diese Bedingung häufig nicht mehr erfüllt. Durch diesen Wert können die unteren Ecken für die interne Berechnung etwas zur Mitte hin geschoben werden. Dadurch können auch hohe Stapel unterschiedlich großer Dynamiks gebildet werden. I.A. braucht dieser Wert nicht angepasst zu werden. Wenn Sie aber sehr kleine Dynamiks haben, sollten Sie ihn kleiner machen, wollen Sie sehr unterschiedlich große Dynamiks aufeinander stapeln, sollten Sie ihn größer machen.

8.5 Optionen | Edit Anlage



- Raster verwenden: Bei Verschiebung mit der Maus werden die Elemente nur im angegebenen Raster positioniert.

- Dauer des Blinkens nach Einfügen: Damit man leichter sieht, was man eingefügt hat, blinkt das Hauptelement der Einfügung. Für erfahrene Nutzer kann die Dauer des Blinkens lästig sein, daher kann man sie hier einstellen.

- Hint Hide Pause : Hiermit kann man einstellen, wie lange die kleinen Hinweise beim Zeigen auf ein Element angezeigt werden.

- Quick Selected automatisch öffnen: Wen diese Info stört, kann sie hier abwählen.

- Beim Kopieren von Elementen: Im Allgemeinen ist “Immer mit Kindern” die beste Einstellung. Je nach Arbeitsweise und Anlagenart kann aber auch eine andere

Einstellung vorteilhafter sein.

- Beide Monitore : Wenn zwei Monitore verwendet werden, ist es lästig, die Größe des TrySim-Hauptfensters einzustellen, wenn dieses z.B. durch einen Bildschirmschoner verändert wurde. Die hier eingestellte Größe lässt sich dann unter Ansicht|Beide Bildschirme schnell wiederherstellen.

- Endschalter ohne Rand: Wenn die Endschalter sehr klein sind, besteht das Bild oft nur noch aus dem Rand und man kann den Zustand nicht mehr gut erkennen. Diese Option unterdrückt das Zeichnen des Randes. Bei sehr großen Anlagen trägt sie auch ein kleines bisschen zur Beschleunigung der Grafikerstellung bei.

- Lichttaster/-schranke ohne LED: Hier gilt sinngemäß das Gleiche wie bei den Endschaltern.

- Beim Einfügen von Elementen: TrySim vergibt neue Adressen, damit keine Überschneidungen auftreten (funktioniert aber nicht ordnungsgemäß). Normalerweise liegen die Adressen ja schon aus dem Elektroplan fest, dann ist es nützlicher, die alten Adressen beim Kopieren beizubehalten, weil man aus dem Symbol und dem Kommentar dazu schnell erkennen kann, wie die neue Adresse lauten soll.

- Fragen, ob Antrieb ersetzt werden soll: Wenn man eine Anlage mit vielen Förderbändern o.ä. bereits erstellt hat und erst nachträglich komplexe Antriebe, z.B. Frequenzumformer mit einem Script einfügen will, ist die Frage :„Wollen Sie den bestehenden Antrieb wirklich ersetzen?" sehr lästig. Im Normalfall ist diese Frage jedoch ein Schutz gegen versehentliches Überschreiben (Löschen) eines funktionierenden Antriebs.

8.6 Optionen | SPS-Editor

Akku-Darstellung

Im Beobachten-Modus versucht das System anhand der Operationen und Operanden zu ermitteln, welches die lesbarste Darstellung der jeweils 32 Bit breiten Akkus ist. Dies führt zu einer häufig wechselnden Darstellung der Akkus. Wenn Sie dies als störend empfinden, können Sie hier ein festes Datenformat vorgegeben.

Darstellung

In diesem Punkt können Sie festlegen, welche Darstellung bei neu erzeugten und importierten Bausteinen voreingestellt werden soll (FUP, KOP, AWL). Sie können nachträglich aber die Darstellung für jedes Netzwerk einzeln ändern.

Allgemein

Automatisch Speichern

Beim Schließen eines Bausteins wird dieser ohne Rückfrage gespeichert.

Beim Schließen übertragen

Beim Schließen eines Bausteines wird dieser automatisch in die SPS übertragen. Hierdurch haben Sie die Gewißheit, dass alle (außer den noch editierten) Bausteine auf der Festplatte und im SPS-Speicher übereinstimmen.

Lästig ist diese Einstellung, wenn Sie einen fehlerhaften Baustein schließen wollen. Sie können sie aber auch dann noch abwählen. Wenn diese Checkbox angewählt ist, wird außerdem automatisch gespeichert, da nur gespeicherte Bausteine in die SPS übertragen werden können.

IEC-Mnemonics

Hiermit schalten Sie auf die international übliche Bezeichnung für Operationen und Operanden. Aus E wird I, aus A wird Q usw.

Abstand Code -> Kommentar

Diese Option wird nur in AWL benötigt. Sie legt fest, wie viele Zeichen der Operation und
dem Operanden eingeräumt werden, bis der Kommentar, die Symbolinformation (falls unter Ansicht angewählt) oder (beim Beobachten) die Statusinformationen angezeigt werden. Im Interesse einer kompakten Darstellung sollte sie so klein wie der längste symbolische Operand sein, aber nicht kleiner. Wer Bezeichnungen wie “DatenbausteinMotorDaten”.”StatusWort”.”Störung” verwendet, sollte etwas um 60 einstellen, weil die Kommentare (u.ä) sonst in einem kaum zu lesenden Flattersatz dargestellt werden.

Beobachten

Beim AWL-Beobachten nicht löschen

Das VKE, der Operanden-Status und die Akkus entsprechen bei übersprungenen Zeilen nicht den aktuellen Werten, sondern sie behalten ihren Wert von der letzten Bearbeitung bei. Die optimale Darstellung dieses Sachverhaltes ist es, (wie z.B. Siemens es tut) diese Zeilen andersfarbig darzustellen. Obwohl es sich einfach anhört, ist es uns aber in AWL bislang nicht gelungen, daher löschen wir die Beobachten-Werte nicht-aktueller Zeilen einfach. Manchmal will man aber unbedingt wissen, welche Werte in den Akkus gestanden haben, als ein Programmteil das einzige Mal in fünf Minuten bearbeitet wurde. Daher haben wir als Notlösung diese Check-Box eingebaut, mit der man das Löschen nicht-aktueller Zeilen verhindern kann (Siehe aber auch: Breakpoints).

Laufbalken beim Beobachten

Der blaue Laufbalken unten rechts ändert sich, wenn die erste Zeile im Netzwerk zumindest gelegentlich bearbeitet wird. Diese Information bewahrt Sie davor, angestrengt auf ein bestimmtes Ereignis zu warten, das nie eintreten wird, weil die Anlage steht, die SPS abgestürzt ist, oder vorher ein Baustein-Ende programmiert worden ist. Manchmal kann dieser Balken aber auch stören, deshalb können Sie hier selbst entscheiden, ob Sie ihn sehen wollen oder nicht.

Umverdrahten

Wenn Sie in der Adressentabelle die Adresse eines Elementes ändern, ist es häufig wünschenswert, dass die entsprechenden Adressen im SPS-Programm automatisch angepasst werden. Hier können Sie einstellen, ob dies immer, nie oder

nur nach Rückfrage geschehen soll.

8.7 Optionen | Symbolleisten

Hier können Sie einstellen, welche Symbolleisten angezeigt werden sollen. Wir empfehlen Ihnen, alle Symbolleisten anzeigen zu lassen, da die Symbole die Arbeit mit TrySim beschleunigen.

Sie können die Symbolleisten auch an- und abwählen, indem Sie mit rechts auf das leere Grau des Symbolbereichs klicken.

Index

- B -

- C -

- D -

- E -

- F -

- G -

- H -

- I -

- K -

- L -

- M -

- N -

- O -

- P -

- Q -

- R -

- S -

- T -

- U -

- V -

- W -

- X -

- Z -