Anwender-Handbuch

Vorteile des Programmierwerkzeugs Saia PG5®

- **Programm-Portabilität:** Saia PG5-Programme sind auf allen Saia PCD®-Plattformen lauffähig.
- **Programm-Organisation nach Dateien** (mit mehrerenProgrammblöcken) er-leichtert den gleichzeitigen Einsatz von Programm-Dateien in mehreren Saia PCD®-Steuerungen.
- **Programmier- und Debug-Umgebung** im selben Werkzeug vereint.
- **Einfaches Programmieren von Bedienpanel** mit integriertem Web Editor.
- **Starker Befehlssatz,** unterstützt mit einer Vielzahl von Assembler-Direktiven.

Die Eigenschaften des Saia PG5®

- **Symbol-Manager** verwaltet alle lokalen, globalen und Netzwerk-Symbole sowie Symbol-Gruppen. Dank Auto-Allokation weitgehender Verzicht auf feste Adressierung.
- **Projekt-Manager** verwaltet komplexe Anlagen mit vernetzten PCDs inklusive Displays und Dokumentation.
- **Online-Funktionen** für Inbetriebnahme und Fehlersuche über SAIA S-Bus®, Ethernet-TCP/IP, Modem usw.
- **Integrierte Programmierumgebungen:**
  - FUPLA (Funktionsplan)
  - S-Edit (Instruktions-Liste IL oder Anweisungsliste AWL)
  - GRAFTEC (Ablaufplan)
- **Integrierte Netzwerk-Editoren** für SBC S-Bus, PROFIBUS DP, LONWORKS®.
- **Umfangreiche Zusatzbibliotheken** erweitern den PG5-Funktionsumfang.

Ausgabe 26/732 D11

Inhaltsverzeichnis

Vorwort

Dieses Dokument dient als Einführung in die programmierbaren SAIA PCD-Steuerungen und nicht als detailliertes, ausführliches Handbuch. Es konzentriert sich daher auf die wichtigsten Punkte und wendet sich an Anwender, die schnell praktische Erfahrungen sammeln wollen. Tiefergehende und umfassende Informationen bieten die Hilfetexte des Programmier-Werkzeugs PG5 oder die detaillierten Handbücher auf der Dokumentations-CD.

Optimale Ausbildungsbedingungen erreichen Sie mit folgenden Programmen, Dokumenten sowie diesem Material:

- CD PG5 Version 2.0
- Dokumentations CD 26/803
- 1 PCD2.M5540 [1] Steuerung
- 1 PCD2.E110 Modul mit 8 digitalen Eingängen
- 1 PCD2.A400 Modul mit 8 digitalen Ausgängen
- 1 PCD8.K111 Programmier-Kabel

Die notwendigen Installationsanweisungen für PG5 Version 2.0 finden Sie auf der PG5 CD unter: PG5_InstallationGuide_D.pdf.

Beibehaltene englische Bezeichnungen aus den PG5-Menüs, -Befehlen, -Optionen und -Tasten sind in diesem Handbuch *kursiv* dargestellt.

Wir wünschen Ihnen viel Erfolg bei Ihrer Ausbildung und mit Ihren zukünftigen Projekten mit SAIA PCD-Produkten.

Saia-Burgess Controls AG, Ihr Partner

[1] un autre PCD peut aussi convenir

Anpassungen

Chronologie

Datum	Kapitel	Seite	Beschreibung

Inhaltsverzeichnis

1 PCD - Inbetriebnahme, Quick Start

Als Ihr Ansprechpartner für PCD-Ausstattung, empfehlen wir eine direkte Herangehensweise: bewältigen Sie die Erzeugung einer kleinen, lebensechten Anwendung. Das ist einfach, selbst wenn Sie bisher keine Erfahrungen mit den Produkten von Saia PCD gemacht haben. In diesem Einstiegskapitel wird alles detailliert beschrieben.

In diesem Beispiel wird beschrieben, wie eine PCD2.M5540 angeschlossen wird. Programmierung und Tests erfolgen mit den Programmierungswerkzeugen Saia PG5®.

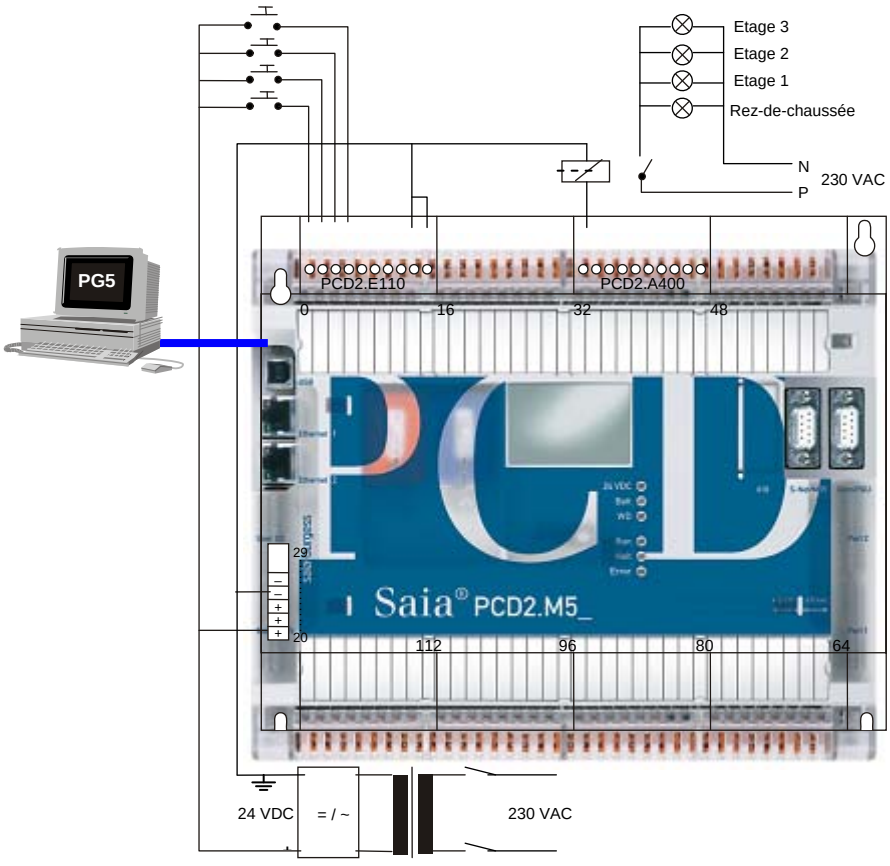
Die nachfolgenden Kapitel dieses Dokuments geben den Inhalt dieses Einstiegskapitels detaillierter wieder und enthalten viele weitere Informationen, wie beispielsweise Beschreibungen der verfügbaren Symbole, Programmstrukturen und die Programmierung von Befehlslisten.

1.1 Installation der Hardware

1.1.1 Beispiel: Treppenhausbeleuchtung

Die Inbetriebnahme einer Saia PCD® wird am Beispiel einer Treppenhausbeleuchtung beschrieben. Das Gebäude hat ein Erdgeschoss und drei Stockwerke. Auf jeder Etage gibt es eine Taste zum Einschalten der Beleuchtung. Nach kurzem Drücken einer der Tasten, werden alle 4 Lampen im Treppenhaus 5 Minuten lang eingeschaltet.
Die Tasten sind mit den 4 Eingängen E0, E1, E2 und E3 der PCD verbunden.
Die 4 Lampen werden über ein Relais ein-/ ausgeschaltet. Das Relais wird über einen einzigen Ausgang (A32) der PCD gesteuert.

1.1.2 Anschlussschema für PCD2.M5540



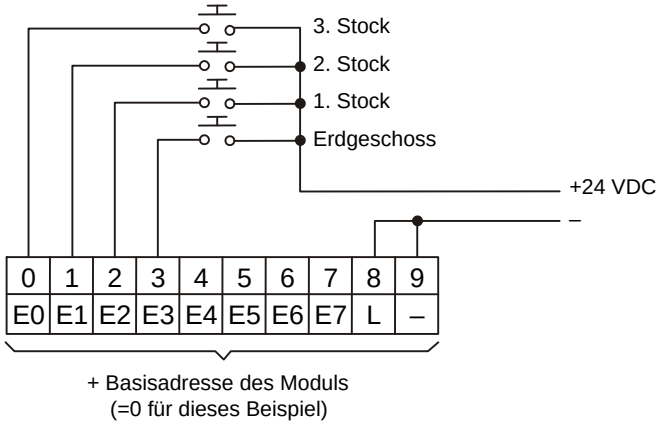
1.1.3 Bestücken der PCD2.M5540

1. Die mitgelieferte Lithium-Batterie von 3.0 V einsetzen.
2. Ein Modul PCD2.E110 an Steckplatz 1 (Adressen 0 bis 15) einsetzen.
3. Modul gegen die Gerätemitte bis zur Endposition schieben und die Halteklinke einrasten. Damit stehen 8 digitale Eingänge für 24 VDC mit den Adressen E0 bis E7 zur Verfügung. Verwendet werden nur die Eingänge E0 bis E3.
4. Ein Modul PCD2.A400 an Steckplatz 3 (Adressen 32 bis 47), wie zuvor beschrieben, einsetzen. Damit stehen 8 digitale Ausgänge (A32 bis A39) für 24VDC / 0,5A zur Verfügung.
   Verwendet wird nur der Ausgang A32.

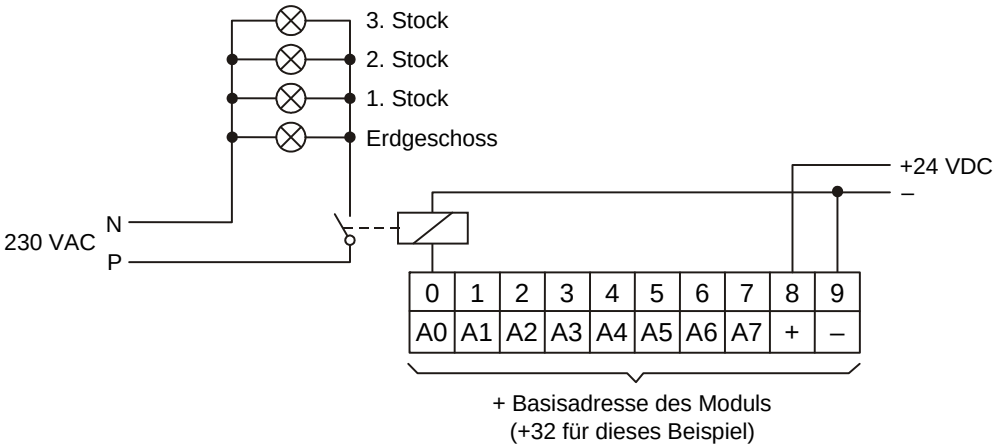
1.1.4 Verdrahtung

1. Die Speisung 24 VDC an den Schraubklemmen 20 (+) und 23 (–) anschliessen. Zulässige Speisespannungen sind:24 VDC ±20% geglättet oder 19 VAC ±15% zweiweggleichgerichtet

2. Die Taster für die Treppenhausbeleuchtung gemäss folgendem Schema anschliessen.



3. Treppenhausbeleuchtung und Relais gemäss folgendem Schema anschliessen.



4. USB-Schnittstelle des PC mit der PCD verbinden

1.2 Programmerstellung

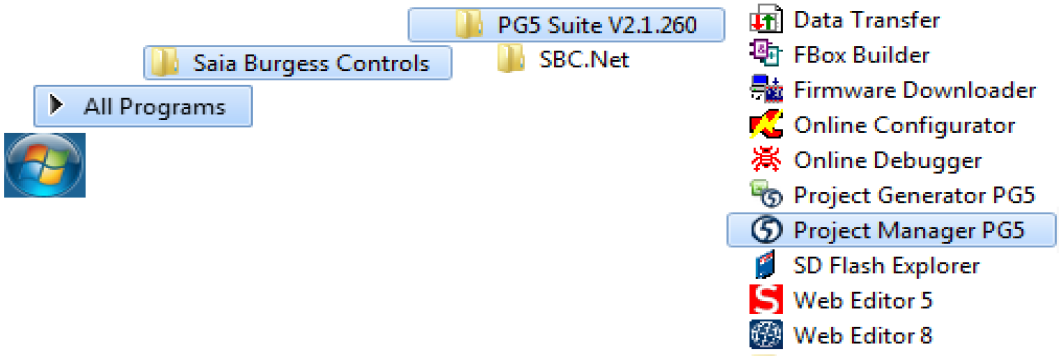
1.2.1 Software Installation

Das Saia PG5®-Programmierwerkzeug für Saia PCD® auf dem PC installieren (falls noch nicht geschehen), gemäss der mit der CD gelieferten Anleitung (CD:\PG5\InstallationGuide_D.pdf).

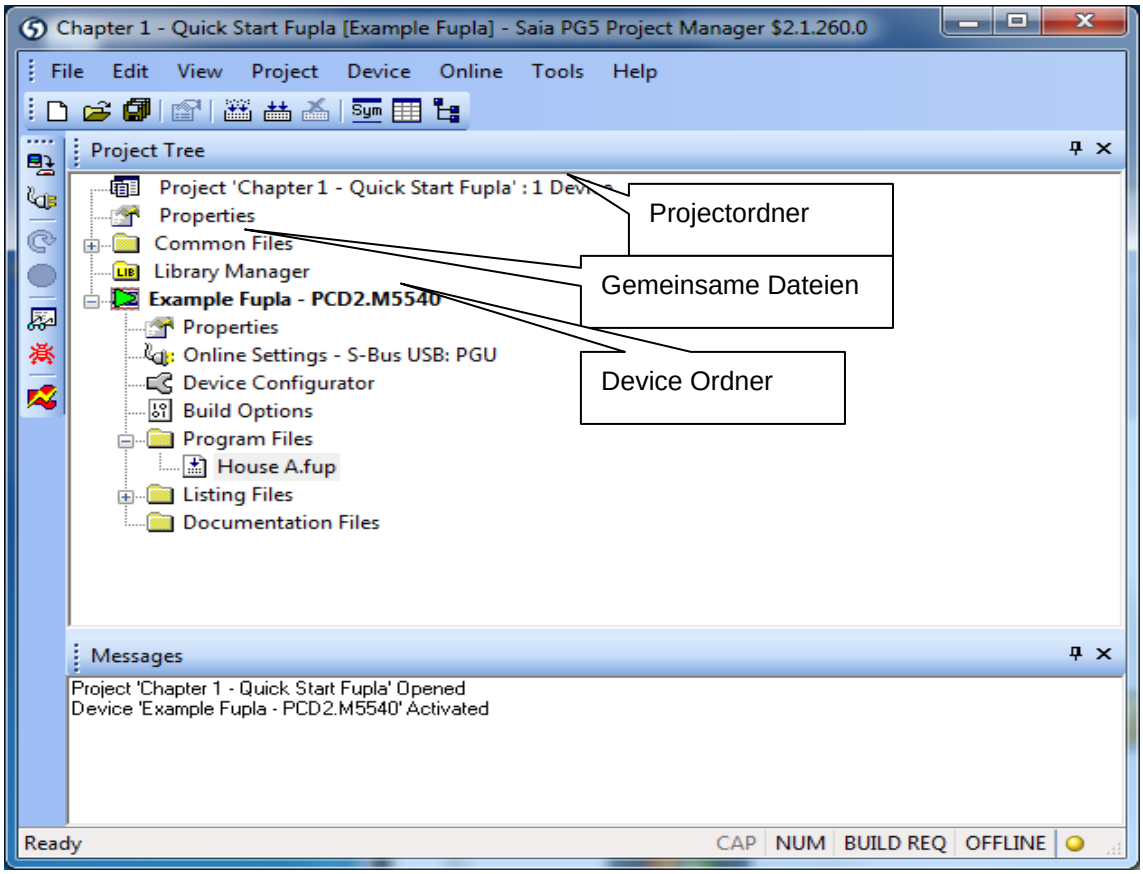
1.2.2 PG5 starten

Den PG5 Projekt Manager öffnen mit:

Start → Programs → Saia Burgess Controls → PG5 Suite 2.1 → Project Manager PG5



Der *Saia PG5 Project Manager* wird angezeigt. Die Struktur des neuen Projektes wird im *Project Tree* dargestellt. (Sollte dieses Fenster nicht erscheinen, verwenden Sie den Befehl *View, Project Tree*.)



Die Ordner im Fenster *Project Tree* enthalten nach bestimmten Kriterien angeordnete Informationen:

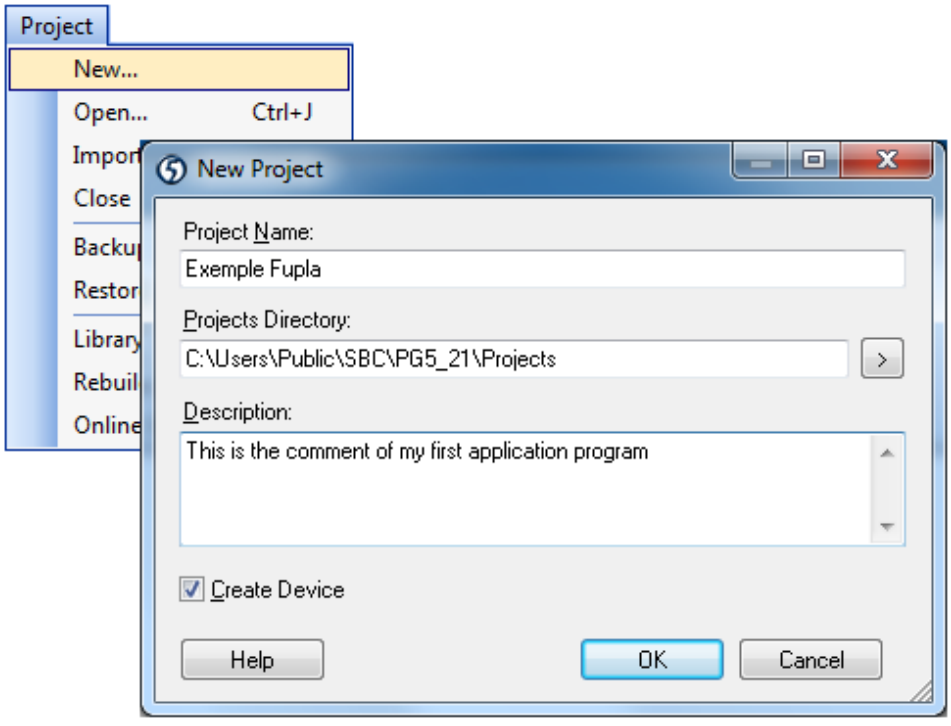
Der Name des Hauptordners gibt den Projektnamen sowie die Anzahl der im Projekt verwendeten Geräte an.
Dateien, die von mehreren Geräten verwendet werden, können im Ordner *Common Files* gespeichert werden.
Darunter befinden sich die Geräteordner (jeder steht für eine PCD).

Alle Geräteordner enthalten folgende Unterordner:

Online Settings, *Device Configurator* und *Build Options*.
Program Files enthält die Dateien zu den Programmmodulen.
Files enthält Dateien, die während des Programmbuilds *(Build)* entstehen. Für den unerfahrenen Nutzer sind diese Ordner uninteressant.

1.2.3 Neues Projekt öffnen

Bevor ein neues Programm geschrieben wird, muss ein neues oder bestehendes Projekt mit den notwendigen Definitionen, einigen Konfigurationsparametern und den für das Anwenderprogramm erforderlichen Dateien geöffnet werden.
noch kein Projekt existiert, wählen Sie *Project, New*.



1.2.4 Bestehendes Projekt öffnen

Ein bereits bestehendes Projekt kann über den Menübefehl *Project, Open…* geöffnet werden. Das Projektverzeichnis wird nun nach allen Projektdateien (.saia5pj) durchsucht und eine Liste der Projektdateien wird angezeigt. Doppelklicken Sie auf ein Projekt in der Liste oder wählen Sie ein Projekt aus der Liste aus und klicken Sie

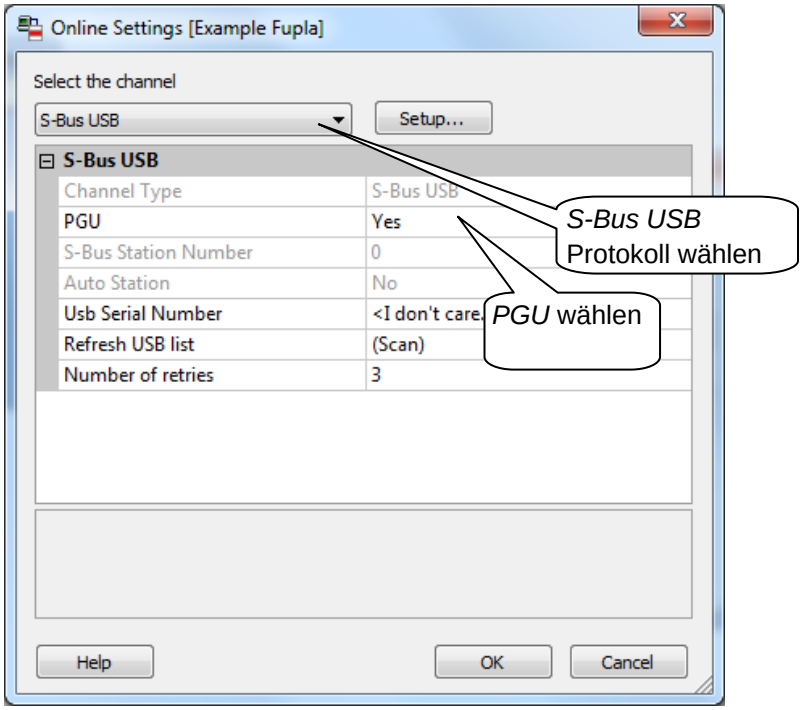
auf die Schaltfläche *Open*. Mit der Schaltfläche *Browse…* können Sie auch direkt nach Projekten oder Geräteordnern suchen.

1.2.5 Konfiguration

Bevor mit einer am Projekt beteiligten device gearbeitet werden kann, müssen die Konfigurationsparameter definiert werden, um zu gewährleisten, dass das Programmierwerkzeug und die PCD das zu erstellende Anwenderprogramm unterstützen.

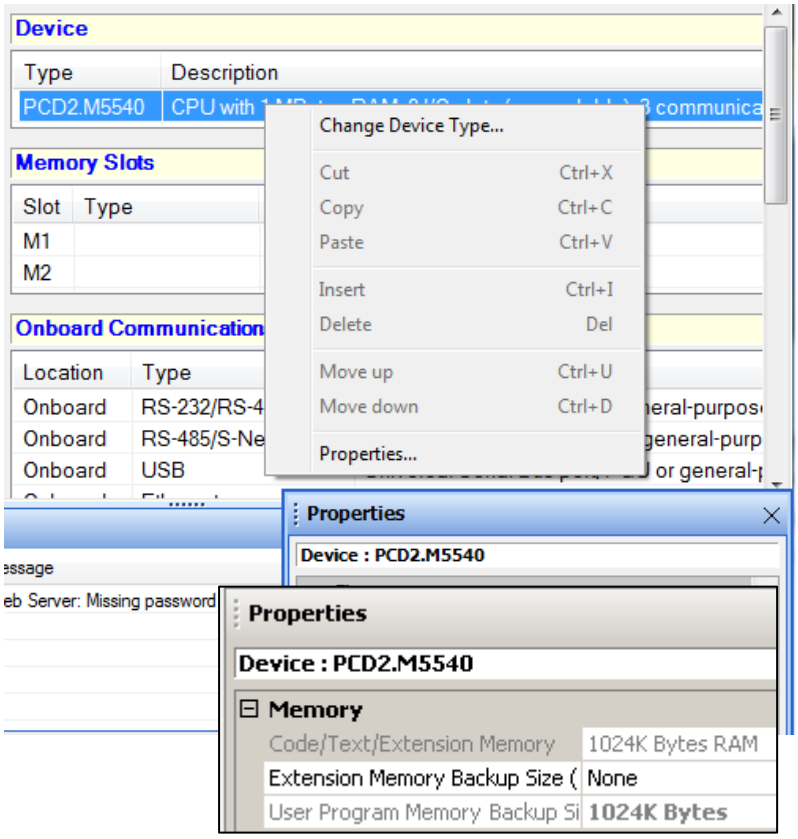
Unter *Online settings (Online Einstellungen)* können die Parameter für die Kommunikation zwischen PCD und PC eingestellt werden. Es gibt dafür mehrere Möglichkeiten. Für diese Übung wird das Default-Protokoll (S-Bus USB) gewählt.

***Channel S-Bus* USB**



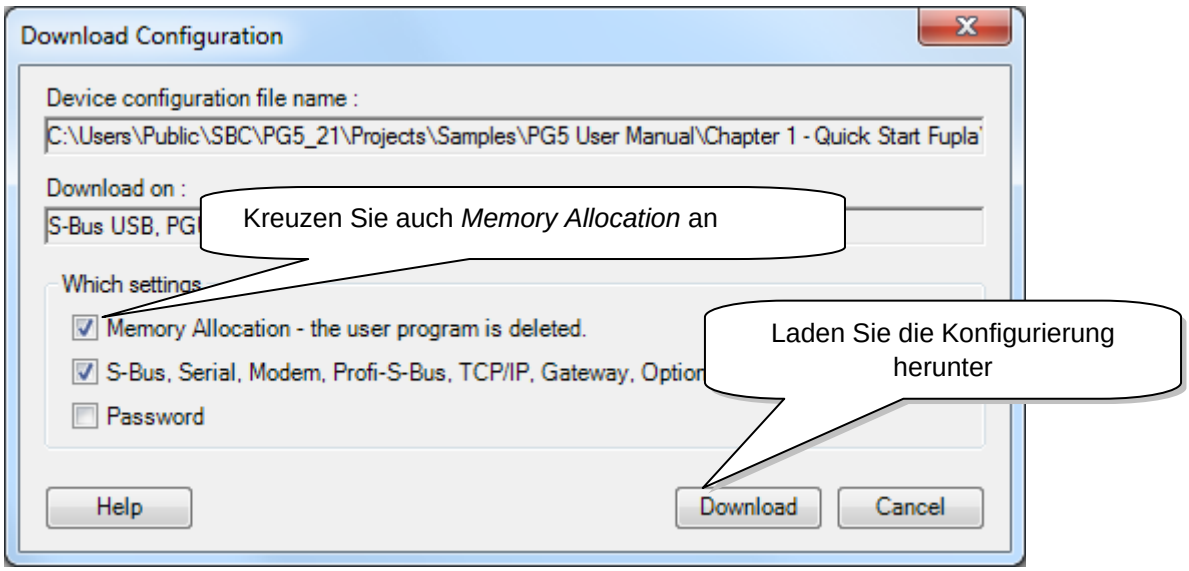
Der *Device Configurator* definiert die Geräteart, die Speichergröße, die Anzahl der S-Bus-Stationen, die Kommunikationsschnittstellen usw. An dieser Stelle sollen jedoch nicht alle Optionen beschrieben werden. Es ist allerdings wichtig, dass die korrekte Geräteart und Speichergröße ausgewählt werden. Die PCD2.M5540 wird immer mit den standardmässigen 1024 Kbyte RAM ausgeliefert.

Wählen Sie den richtigen PCD-Typ aus dem Kontextmenü *Change Device Type*, um ein neues *device* zu definieren.



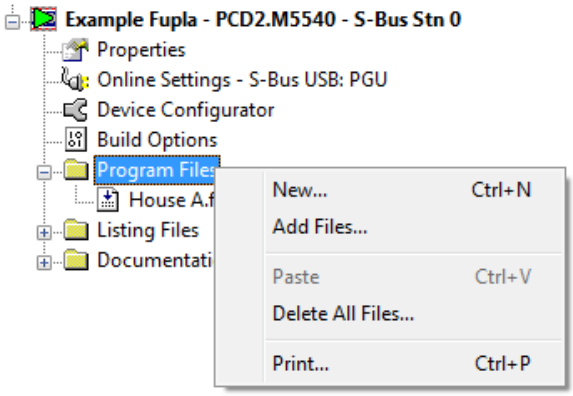
Der Kontextmenübefehl *Properties* zeigt das Eigenschaftenfenster an. Hier kann die Speichergröße definiert werden.

Die Konfiguration muss immer mit dem Menübefehl *Online, Download Configuration…* in die PCD geladen werden.



1.2.6 Eine Programmdatei hinzufügen

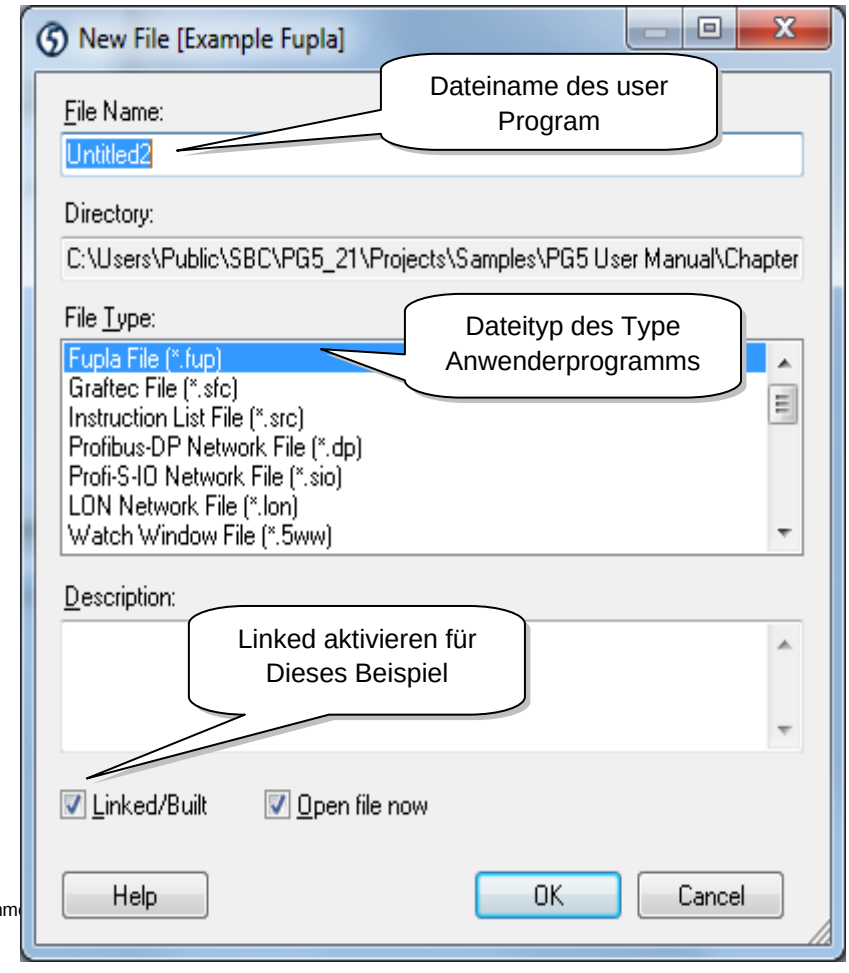
Die PCD-Anwenderprogramme befinden sich in einer Datei. Es gibt mehrere Möglichkeiten, um eine Programmdatei einem Projekt hinzuzufügen:
Im Fenster *Project Tree d*en Ordner *Program File* markieren, rechte Maustaste drücken um das Kontext-Menü aufzurufen und *New (neue Datei)* wählen...



Weitere Möglichkeiten sind:
In der Symbolleiste die Taste *New File* drücken oder mit Menü-Befehl *File, New...*

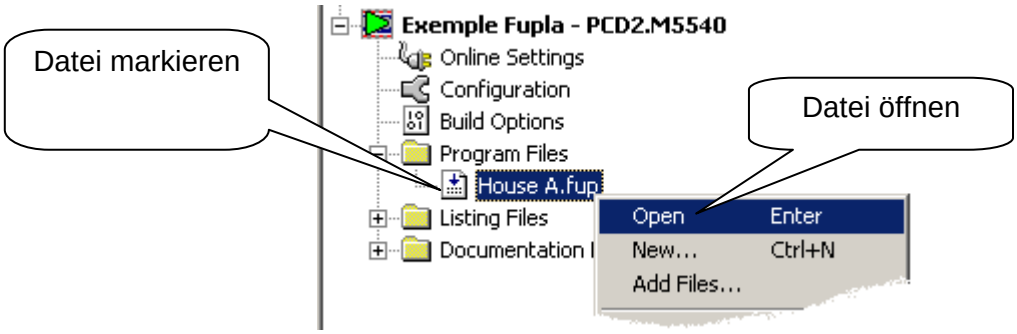
Im Fenster *New File* werden Name und Typ des Anwenderprogramms festgelegt, zwei sehr wichtige Informationen.

Für das Schreiben der PCD-Anwenderprogramme stehen mehrere Editoren zur Verfügung.Der Benutzer kann den für das Anwenderprogramm am besten geeigneten Editor frei wählen.Für dieses Beispiel ist es *Fupla File (*.fup).*
Dies ist ein universell verwendbarer Editor.

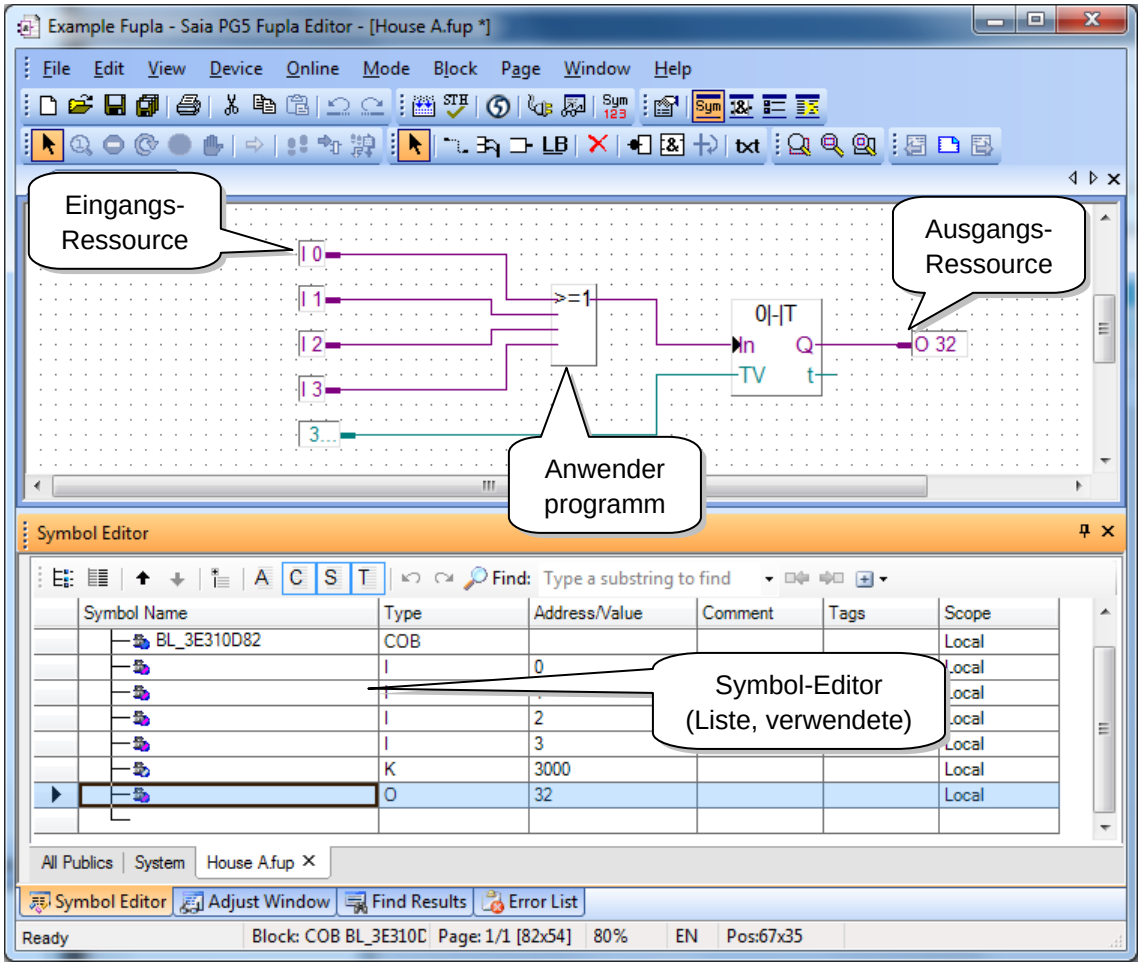


1.2.7 Datei öffnen

Wenn es im Ordner schon eine Programmdatei gibt, kann diese wie folgt geöffnet werden:
Im Fenster *Project* den Ordner *Program Files* öffnen, die betreffende Datei markieren, mit Doppelklick öffnen oder mit der rechten Maustaste das Kontext-Menü öffnen und die Datei mit *Open* öffnen.



1.2.8 Programm editieren



Ressourcen editieren

Ressourcen sind Informationen, die für die Erstellung des PCD-Anwenderprogrammes benötigt werden, z.B. die Schalter für die Treppenhausbeleuchtung.
Wir editieren Symbole in den Connectors auf der Fupla-Seite. „Read“-Symbole befinden sich auf der linken, „Write“-Symbole auf der rechten Seite.

Für das Beispiel Treppenhausbeleuchtung gibt es die 4 Lichtschalter als Eingangs-Ressourcen I 0, I 1, I 2 und I 3 (Inputs) und die Ausgangs-Ressource O 32 (Output) für die Ansteuerung des Treppenhausautomaten. Die gewünschte Einschaltdauer von 5 Minuten für die Treppenhausbeleuchtung wird als Konstante im Eingangsconnector eingetragen, als Vielfaches von Zehntelsekunden.

Der Wert der Konstante ist 3000 (5 Min. x 60 Sek. x 10 = 3000).

Add Connectors

Um einen Connector und sein Symbol zu einer Fupla-Seite hinzuzufügen, drücken Sie die Taste *Add Connectors* auf der Werkzeugleiste und platzieren die Maus auf der Fupla-Seite. Ein „Read“-Eingangsconnector kann durch Drücken der linken Maustaste hinzugefügt werden. Ein „Write“-Ausgangsconnector kann durch Gedrückthalten der Umschalttaste und Drücken der linken Maustaste hinzugefügt werden. Der Connector, den Sie gerade hinzugefügt haben, ist einsatzbereit. Ihm kann jetzt ein Symbol zugewiesen werden. Im Connector wird ein Mauszeiger angezeigt. Wenn Sie das Symbol im Connector nicht sofort editieren möchten, drücken Sie die Escape-Taste und setzen den nächsten Connector.

Show Hide Symbols Editor

Um ein bereits auf der Fupla-Seite vorhandenes Connectorsymbol zu editieren oder zu ändern, wählen Sie den Connector mit einem Doppelklick aus. Im Connector wird nun ein Mauszeiger angezeigt. Sie können jetzt die Adressen I 0 bis I 3 oder Ausgang O 32 oder die Konstante eintragen. Vergewissern Sie sich, dass sich zwischen dem Buchstaben I und der Eingangsadresse stets ein Leerzeichen befindet. Dasselbe gilt für den Ausgang.

Zum Editieren der Eingangs-Ressourcen werden mit der Maus nacheinander
4 Zellen in der linken Spalte der Programmseite markiert und die Adressen
I 0 bis I 3 eingetragen. Ebenso werden die Zeitkonstante 3000 (links) und der Ausgang O 32 (rechts) eingetragen.

Bitte beachten, dass der Adressentyp (I oder O) und die Adressenwerte (0 bis 3 und 32) durch einen Leerschlag getrennt sein müssen.

Die Ressourcen erscheinen sofort im Symbol-Editor *Symbols.* Wenn der Symbol-Editor nicht sichtbar sein sollte, kann er jederzeit via *View, Symbols Editor* angezeigt werden. Oder Taste *Show/Hide Symbol Editor* drücken:

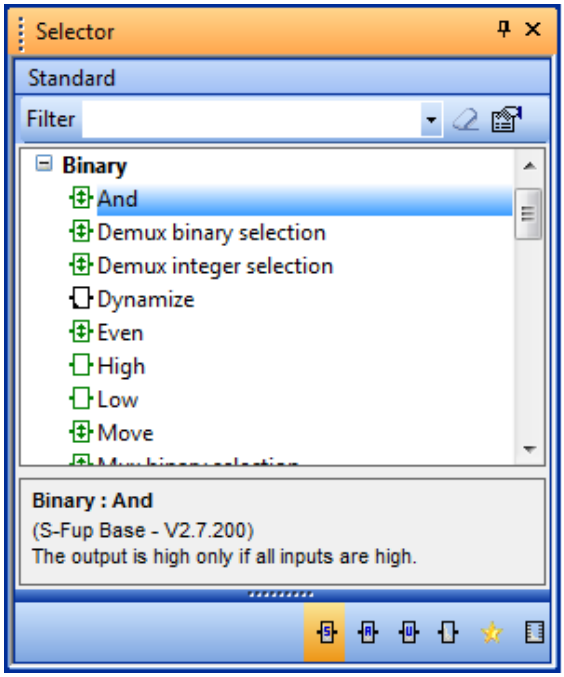
Anmerkung:
Standardmässig kann jede neue Seite bereits über Ränder mit Connectors links und rechts verfügen. Wenn Sie neue Seiten lieber ohne diese Connectors anzeigen lassen wollen, so dass Sie die Connectors nach eigenem Belieben setzen können, deaktivieren Sie bitte die entsprechende Option im Menü: *View, Options…, Workspace, New pages with side connectors*.

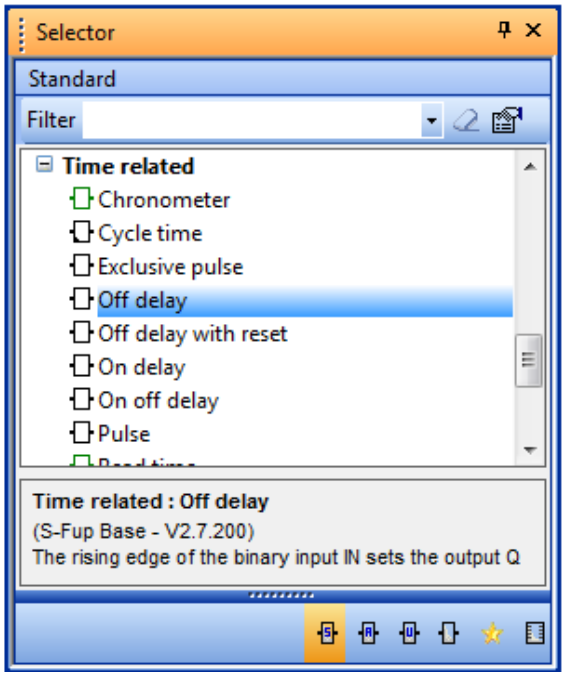
Um leere Connectors zu entfernen, die sich am linken oder rechten Rand der Seite befinden, wählen Sie folgendes Menü: *Page, Remove Unused connectors.*
Um Connectors erneut auf einer leeren Seite zu platzieren, wählen Sie folgendes Menü: *Page,Add Side Connectors.*

Programmfunktionen editieren

Die Programmfunktionen werden auf der Fläche zwischen den „Read"- und „Write"-Connectors editiert. Dazu werden graphische Symbole (Funktionsblöcke) platziert, aus denen die Anwenderprogramme aufgebaut werden.
Die verschiedenen Funktionsblöcke werden im Fenster *Fbox Selector* ausgewählt werden.

Show/Hide





Die erste im Beispiel benötigte Funktion dient dazu die Beleuchtung mit einem kurzen Impuls eines Treppenhausschalters einzuschalten. Es handelt sich dabei um eine Oder (Or)-Funktion. Sie gehört zur Familie der Binär-Funktionen *(Binary)* in der Standard-Bibliothek.

Mit der zweiten Funktion (Off delay) wird die Einschaltdauer von 5 Minuten festgelegt. Sie gehört zur Familie der Zeit-Funktionen *(Time related)* in der Standard-Bibliothek.

Weitere Angaben über eine im Fenster *FboxSelector* markierte FBox erhält man unter *Fbox Info* nach dem Öffnen des Kontextmenüs mit der rechten Maustaste.

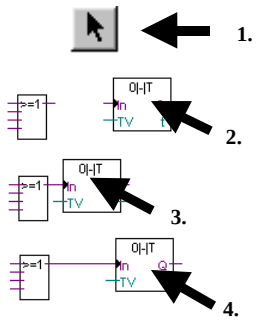
Nachdem im Fenster *Fbox Selectior* ein Funktionsblock gewählt wurde, wird er mit der linken Maustaste auf der Fläche zwischen den Ressourcen-Spalten platziert.

Bei verschiedenen Funktionsblöcken, wie z.B. der *Oder-Logik*, kann die Anzahl der Eingänge gewählt werden. Dies geschieht durch vertikales Ziehen der Maus bei gedrückter linker Maustaste.

Funktionsblöcke anschliessen

1. Möglichkeit, für gerade, horizontale Verbindungen**)**

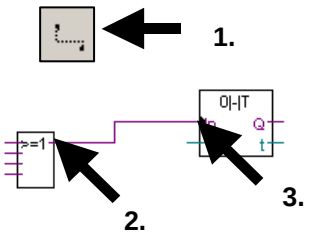
1. Taste *Line Mode* drücken
2. Graphisches Funktionssymbol mit Mauszeiger markieren und linke Maustaste drücken
3. Taste gedrückt halten und Symbol horizontal zum betreffenden Anschluss schieben. Maustaste nicht loslassen.
4. Symbol wieder an seinen ursprünglichen Platz ziehen und Maustaste loslassen.



2. Möglichkeit, mit Richtungswechsel

1. Taste *Auto Lines Mode* drücken
2. Ausgangspunkt mit Mauszeiger markieren, waagrechte Linie so weit wie gewünscht ziehen und dort linke Maustaste drücken.
3. Endpunkt der gezogenen Linie mit Mauszeiger markieren, senkrechte Linie so weit wie gewünscht ziehen und dort linke Maustaste drücken
4. Endpunkt der gezogenen Linie mit Mauszeiger markieren, waagrechte Linie bis zum Anschluss ziehen und dort linke Maustaste drücken

Wenn nötig, kann das Ziehen einer Linie mit der rechten Maustaste abgebrochen werden.



Linie, Funktionsblock, Symbol oder Connector löschen

Taste *Delete Object* drücken und die zu löschende Linie, Funktionsblock, Symbol oder Connector mit der Maus markieren und Taste loslassen.

1.3 Programm aufbauen und in die PCD übertragen

1.3.1 Programm aufbauen (Build)

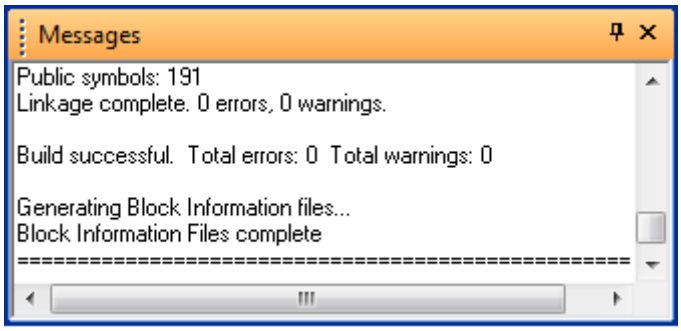
Rebuild All Files

Damit das fertig editierte Programm von der PCD gelesen und ausgeführt werden kann, muss es im Project Manager via Menü *Device Rebuild All Files* Program oder mit der Taste *Rebuild All Files*

im Fupla-Editor oder im Project Manager aufgebaut (kompilieren, assemblieren und linken) werden.

Im Fenster *Messages* werden die Ergebnisse der verschiedenen Programm-aufbereitungs- Schritte *(Compiler, Assembler, Linker* etc.) angezeigt
Wenn das Programm korrekt editiert wurde, wird die Build-Funktion mit der Meldung abgeschlossen: *Build sucessful. Total errors 0 Total warnings: 0*

Eventuell aufgetretene Fehler werden mit roten Meldungen angezeigt. Mit einem Doppelklick der Maustaste auf eine rote Meldung, kann der betreffende Fehler im Anwenderprogramm einfach lokalisiert werden.



1.3.2 Programm in die PCD übertragen (Download)

Download Program

Das Anwenderprogramm ist jetzt bereit. Es muss nur noch vom PC in die PCD übertragen werden. Dies geschieht mit der Taste *Download Program*

oder dem Menü-Befehl *Online, Download Program* im Project Manager.

Bei eventuell auftretenden Kommunikationsproblemen sind die Konfigurationseinstellungen (*Online Settings)* sowie die
PC < - > PCD-Verbindung mit dem Kabel PCD7.K111, USB zu überprüfen.

1.4 Programmfehlersuche und Behebung (Debugging)

Die erste Programmversion ist nicht immer perfekt. Ein seriöser Test ist immer angebracht. Der Programmtest wird von demselben Programm-Editor unterstützt, mit dem das Programm erstellt wurde.

1. Taste *Go On /Offline* drücken

2. Programm mit der Taste *Run* starten

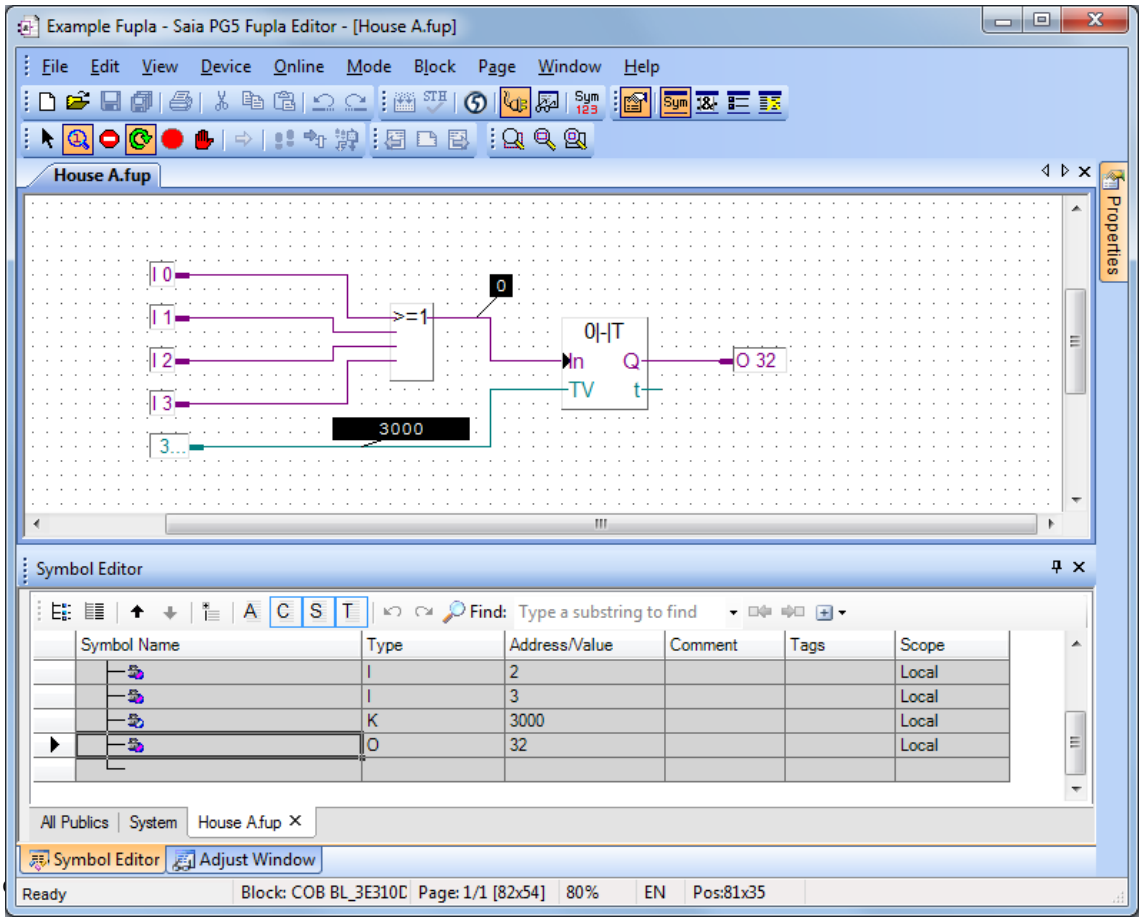
Dabei die LED *RUN* auf der PCD beobachten.
Nach dem Drücken der Taste *Run* muss die LED *RUN* leuchten. Dies bedeutet, dass die PCD das Anwenderprogramm ausführt.

Nach dem Drücken der Taste *Stop* darf die LED *RUN* nicht mehr leuchten.
Die PCD hat die Programmausführung angehalten.

Wenn der Editor *Online* ist und die PCD im *RUN*-Betrieb, kann jede einzelne vom Programm verwendete Ressource angezeigt werden:

- Der logische Zustand der binären Informationen wird mit einer dicken oder dünnen Linie angezeigt (dick = 1, dünn = 0)

Andere Datenwerte können angezeigt werden, indem Sie die linke Maustaste auf dem Connector drücken und dadurch ein *Probe*-Fenster aufrufen: Markieren Sie die Taste *Add Probe* und den Link mit der Maus.



1.4.1 Programm korrigieren

Bei Programmänderungen ist wie folgt vorzugehen:

1. OFFline gehen (mit Taste *Go On /Offline*)
2. Programm ändern

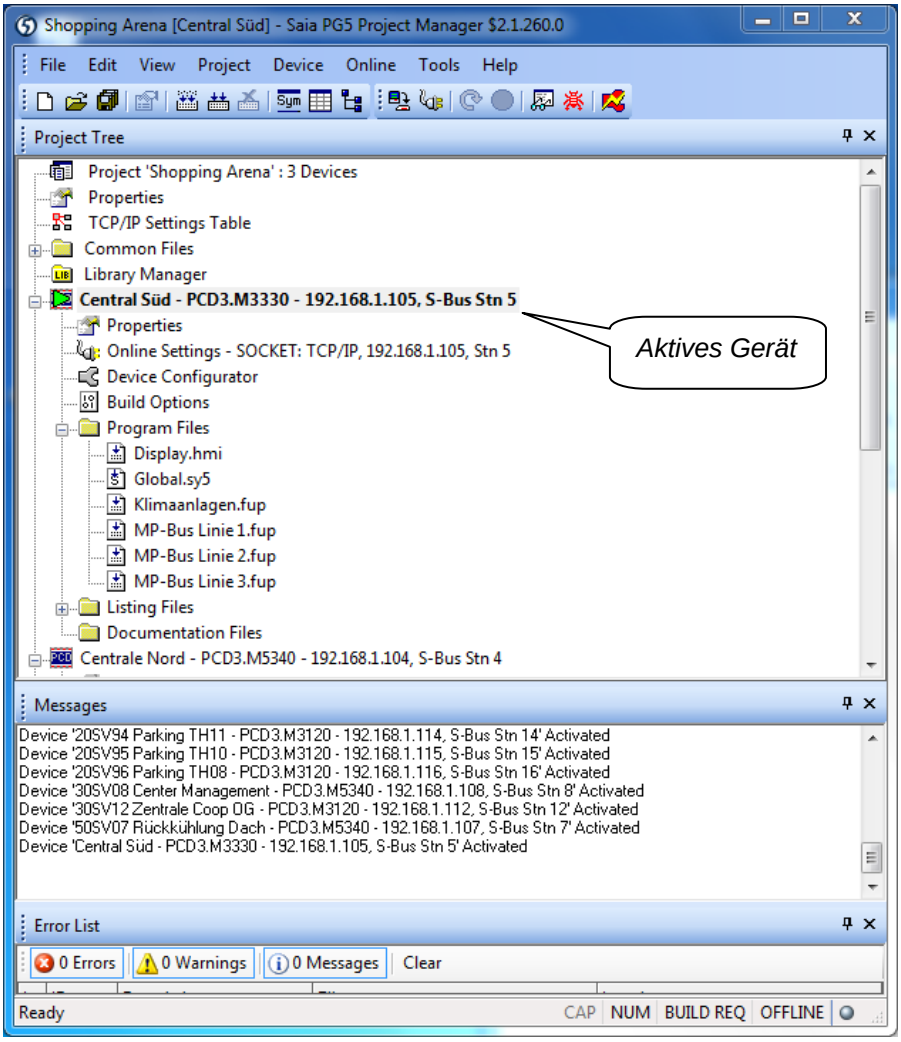
3. Programm neu aufbauen (mit Taste *Build*)
4. Programm in die PCD übertragen (mit Taste *Download Program)*

Inhaltsverzeichnis

2 Projektmanagement

2.1 Projektorganisation

Zu modernen Automatisierungsprozessen gehören meist mehrere in einem Netzwerk zusammengeschlossene Geräte, in dem jedes Gerät eine spezielle Funktion übernimmt. Beispielsweise verfügt ein Gebäudesteuerungssystem über unterschiedliche Geräte für Beleuchtung, Heizung, Lüftung, Automatiktüren in der Tiefgarage usw.



Der *Saia PG5 Project Manager* liefert eine umfassende Übersicht über alle Geräte und Dateien in einem bestimmten Projekt. Alle Funktionen werden hier gestartet. Zum Beispiel: Hinzufügen neuer Programmdateien zum Projekt, Öffnen von Dateien zum Schreiben von Programmen, Hardwarekonfiguration, Erstellung und Downloaden der Programme in die *Devices*, Sicherungs- und Wiederherstellungsprozesse, Ausgabe von Fehler- und Warnmeldungen bei der Programmerstellung usw.

Das Fenster mit dem *Project Tree* enthält eine hierarchische Darstellung der einzelnen Geräte und deren Konfiguration und Programmdateien. Zum Öffnen des Fensters wählen Sie den Befehl *View, Project Tree*.

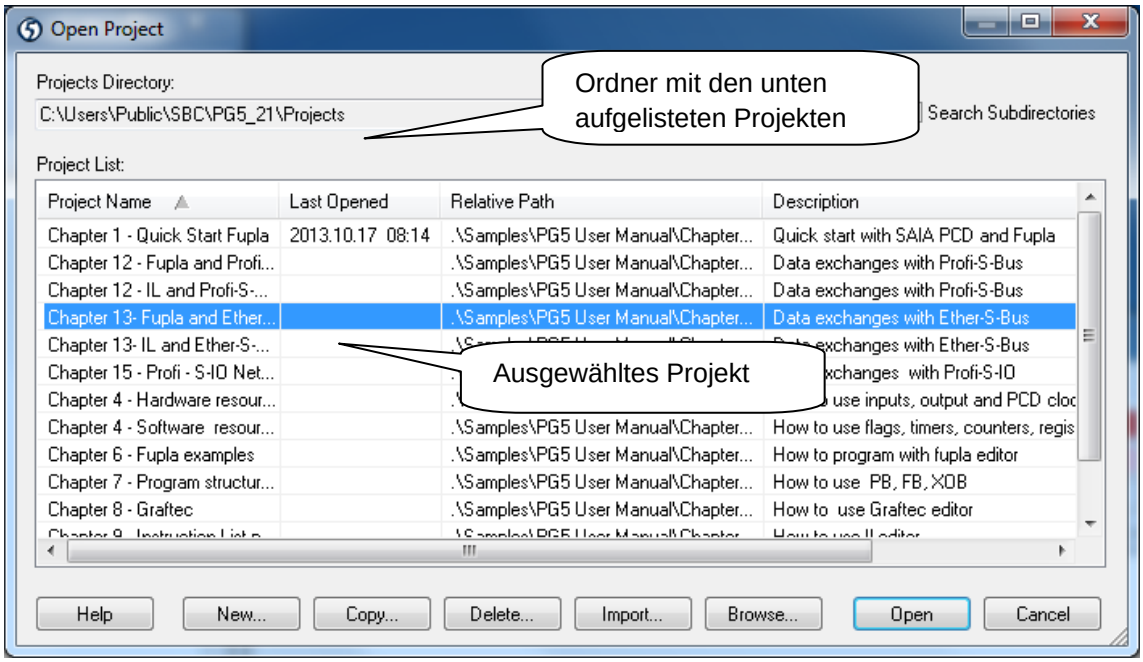
Das Fenster *Messages* enthält allgemeine Informationen und zeigt bei der Programmerstellung Fehler- und Warnmeldungen an. Zum Öffnen des Fensters wählen Sie den Befehl *View, Messages* auswählen.

Die übrigen *View*-Fenster enthalten Listen mit den verwendeten Symbolen und Medien und die Programmstruktur. Hier gibt es ebenfalls eine Funktion für den Querverweis von Symbolen.

Das *active device* ist mit einem grünen Dreieck gekennzeichnet. Viele Menüs und die Schaltflächen der *Tool Bar* können für das *active device* genutzt werden. Um zu einem anderen *active device* zu wechseln, wählen Sie ein anderes Gerät im *Project Tree* oder führen Sie den Befehl *Device, Set Active* aus.

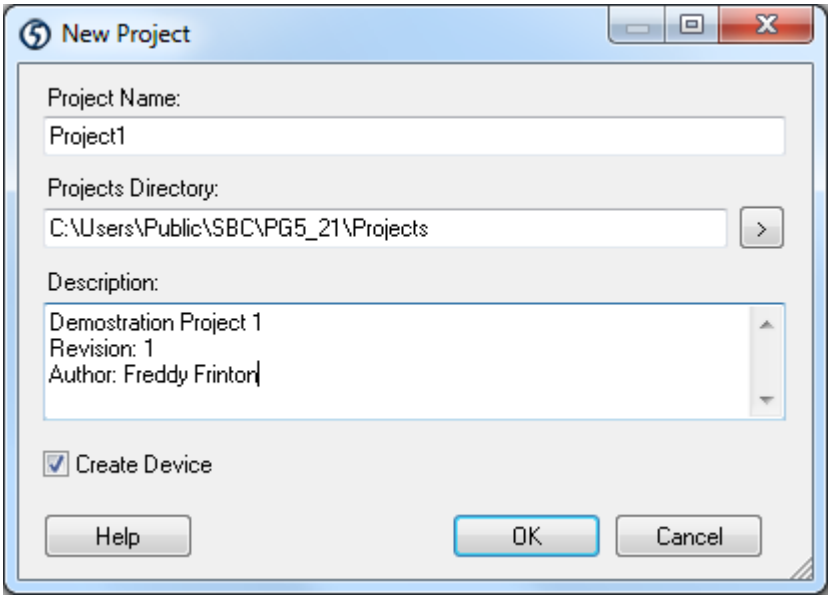
2.1.1 Öffnen eines Projekts

Die PG5 wird mit allen Beispielen in diesem Handbuch installiert. Mit dem Befehl *Project, Open…* wird über alle Projektdateien (.saia5pj) im *Projects Directory* eine Liste mit allen Projekten angezeigt. Zum Öffnen des Projekts führen Sie einen Doppelklick aus oder wählen das Projekt und klicken dann auf *Open*. Alternativ klicken Sie auf die Schaltfläche *Browse…* und wählen dann ein bestimmtes Projekt oder eine Gerätedatei (.saia5pc) in einem anderen Verzeichnis.



2.1.2 Erstellen eines neuen Projekts

Um ein neues Projekt zu beginnen, führen Sie den Befehl *Project, New…* aus, geben den Projektnamen im Feld *Project Name* ein (Standard = *ProjectX*) und klicken dann auf OK.



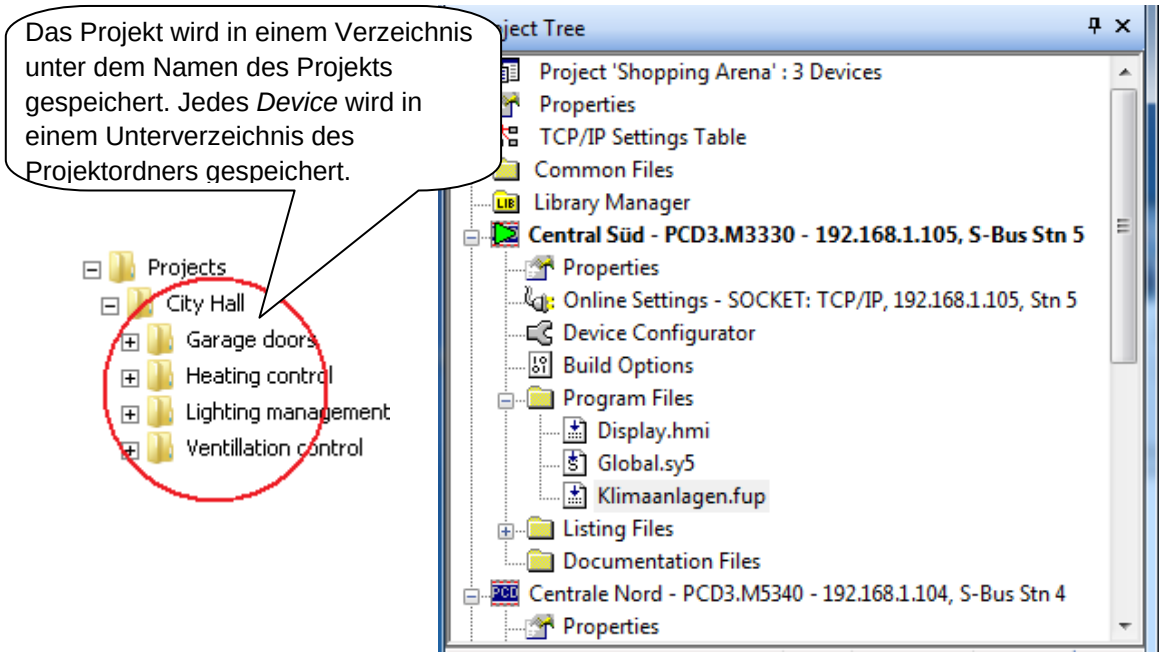
Project Name: Name des neuen Projekts.
Projects Directory: Name des Ordners mit dem Projekt.
Description: Freitext, Beschreibung des Projekts.
Create Device: Es wird automatisch ein einzelnes Gerät mit dem Namen des Projekts erstellt.

2.1.3 Speichern von Projekten auf dem PC

Standardmässig werden alle Projekte abhängig von der Windows-Version in den folgenden Projektverzeichnissen gespeichert:

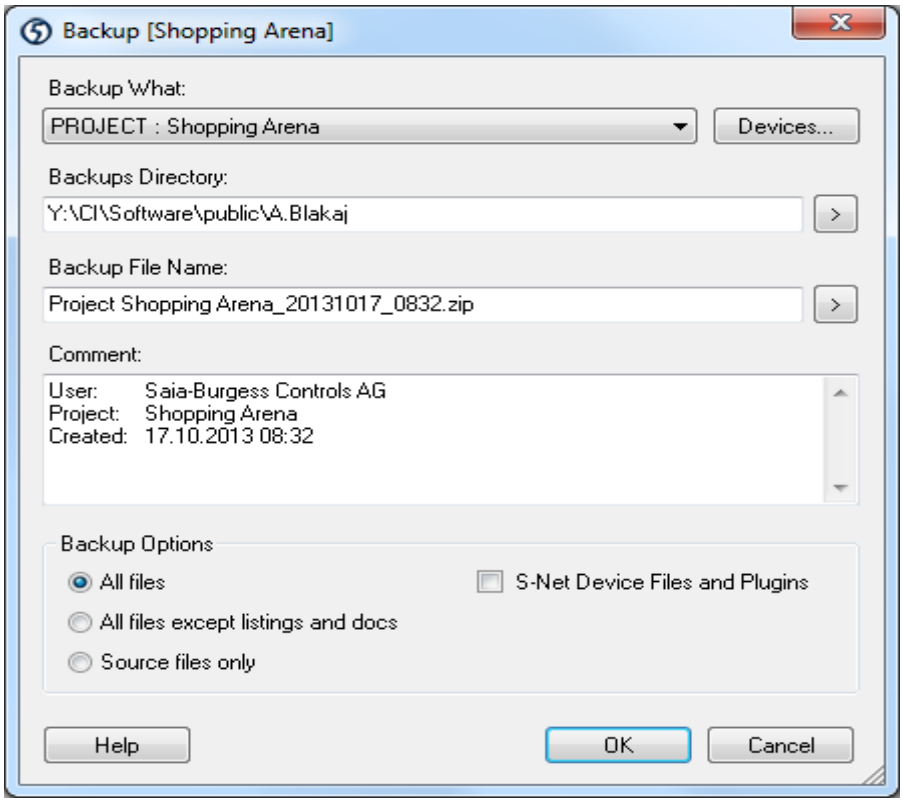
XP: **C:\Dokumente und Einstellungen\All Users\SBC\ PG5_21 \Projects**
Vista: **C:\Users\Public \SBC \PG5_2_1 \Projects**

Falls nötig, können Projekt- (und Bibliotheks-)verzeichnisse mit dem Befehl *Tools, Options…* geändert werden:



2.1.4 Sichern und Wiederherstellen eines Projekts oder Geräts

Um ein Projekt zu sichern, müssen dieselbe Ordnerstruktur und alle Quelldateien des Projekts beibehalten werden. Der Befehl *Project, Backup…* komprimiert entweder das gesamte Projekt oder nur ein einzelnes Gerät in eine standardmässige .zip-Datei, die über den Befehl *Project, Restore…* wiederhergestellt werden kann.



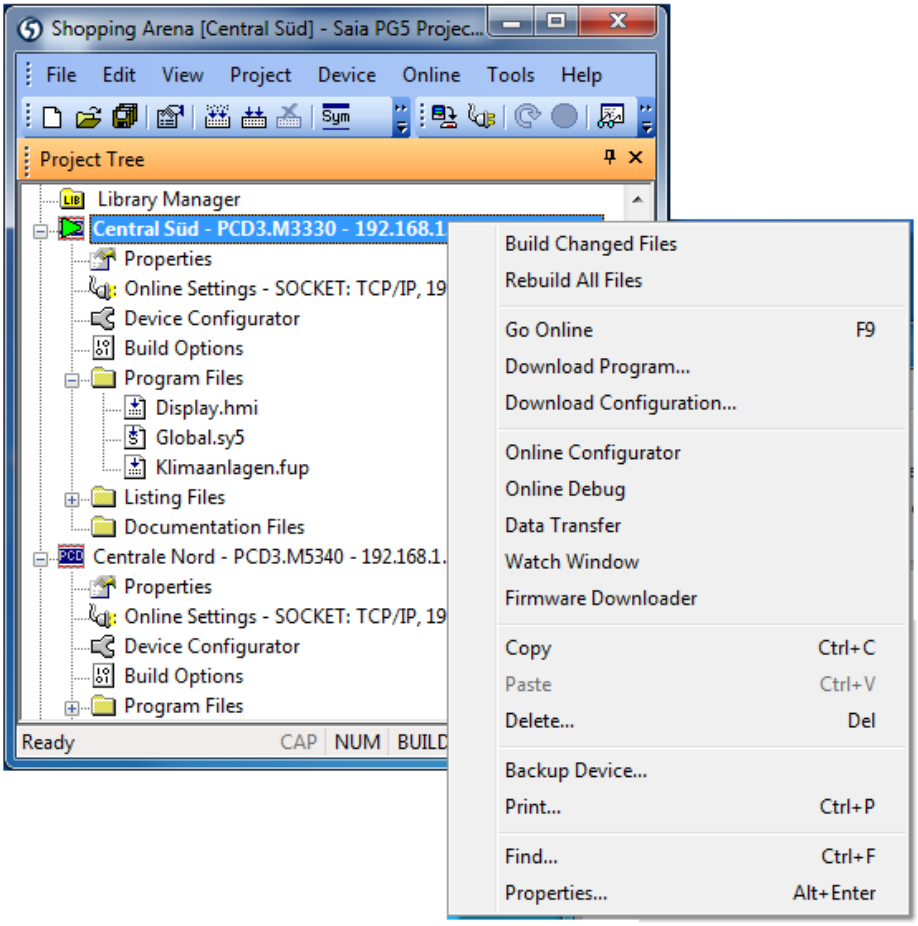
Backup Project or Single Device
Hier wird ausgewählt, ob das gesamte Projekt oder nur ein Einzelgerät des Projekts gesichert werden soll. Standardmässig wird das gesamte Projekt gesichert.

To Compressed File
Pfad der komprimierten .zip-Datei, die erstellt wird. Der standardmässige Name enthält den Projekt- oder Gerätenamen und das Datum/die Uhrzeit in folgendem Format:
d:\<Pfad>\<Name>_JJJJMMTT_SSMM.zip

Comment
Freitext mit Beschreibung der Sicherung. Standardmässig wird hier der Benutzername, Projektname und Datum/Uhrzeit der Sicherung angegeben.

Backup What
Hier wird festgelegt, welche Dateien gespeichert werden. Es müssen nicht alle Dateien gespeichert werden, ausschliesslich Quell- und Konfigurationsdateien sind wichtig.

2.2 Das Fenster *Project Tree*



Kontextmenü des Geräts

Über den Befehl *View, Project Tree* gelangen Sie zum *Project Tree* mit einer strukturierten Übersicht der Projektinformationen.

2.2.1 Ordner *Project*

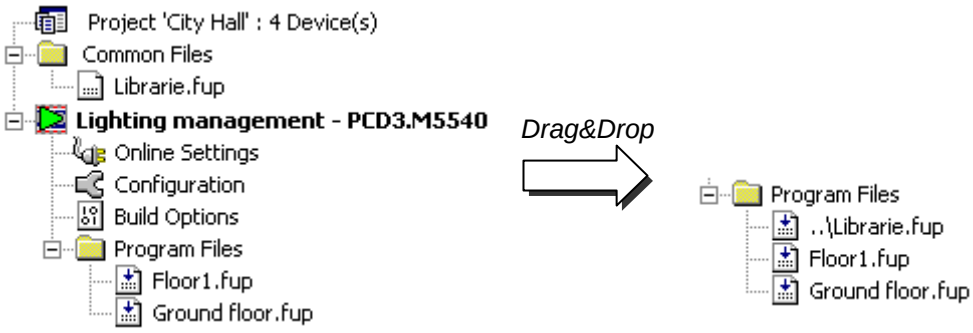
Der oberste Ordner steht für das Projekt. Er trägt den Projektnamen und zeigt die Anzahl der *Devices* in diesem Projekt an. Mit den folgenden Befehlen können Sie Geräte im Projekt verwalten:

Hauptmenü *Device, New…* oder Kontextmenü *New Device…*
Mit diesem Befehl wird ein neues *Device* zum Projekt hinzugefügt.

Hauptmenü *Device, Import…* oder Kontextmenü *Import Device…*
Mit diesem Befehl kann ein bestehendes *Device* aus einem anderen Projekt importiert werden. Wenn ein Gerät aus einer älteren PG4- oder PG5-Version importiert wird, wird es in das neue V2-Format konvertiert.

Hauptmenü *File, Properties...* oder Kontextmenü *Properties…*
Mit diesem Befehl können die Projekteinstellungen angezeigt oder geändert werden: Name, Beschreibung und Attribute mit Lesezugriff.

2.2.2 Ordner für gemeinsam genutzte Dateien



Dieser Ordner enthält Dateien, die von mehreren Geräten in einem Projekt benutzt werden können. Diese Dateien können in den Ordner *Program Files* der einzelnen Geräte kopiert, eingefügt oder hinübergezogen werden. Der Name einer gemeinsam genutzten Datei im Ordner *Program Files* beginnt mit zwei Punkten („..\“). So wird angezeigt, dass sich die Datei im übergeordneten Verzeichnis eine Ebene weiter oben befindet.

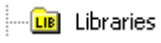
Die Datei kann entweder im Ordner *Common Files* oder aus dem Ordner *Program Files* geöffnet werden. In beiden Fällen kann dieselbe Datei geändert werden, die Änderungen wirken sich dann auf alle Geräte aus, die in der Datei angegeben sind.

Sie benötigen die folgenden Hauptbefehle, um gemeinsam genutzte Dateien hinzuzufügen:

File, New…* oder Kontextmenü *New File…
Mit diesem Befehl erstellen Sie eine neue Datei im Ordner *Common Files*.

Kontextmenü *Add Files…*
Mit diesem Befehl kopieren Sie eine oder mehrere bestehende Dateien in das Verzeichnis *Common Files*. Es können nicht nur PG5-Programmdateien, sondern auch Betriebs- und Wartungsdokumente (Word- und Excel-Dateien usw.) hinzugefügt werden. Diese Dateien werden im PG5-Projekt gespeichert und können aus dem *Project Tree* über einen Doppelklick geöffnet werden.

2.2.3 Bibliothekenordner



Über diesen Ordner öffnen Sie das Fenster *Library Manager*, in dem alle verfügbaren Bibliotheken angezeigt werden.

Die Liste *Installed Libraries* zeigt alle Bibliotheken des Bibliothekenverzeichnisses, die für alle Projekte genutzt werden können.

Die Liste *Libraries Copied To Project* zeigt alle Bibliotheken, die in das lokale Bibliotheken-Unterverzeichnis des offenen Projekts kopiert wurden. Nur für das offene Projekt können diese Bibliotheken genutzt werden. Installierte Bibliotheken können in das Projekt hinübergezogen werden oder über die Auswahl der Bibliothek und einen Klick auf die Schaltfläche *Copy To Project* in das Projekt kopiert werden.

Benutzerbibliotheken oder Versionen, die nicht standardmässig mit der PG5 geliefert werden, sollten immer zusammen mit dem Projekt gespeichert werden, damit das Projekt vollständig ist und immer erstellt werden kann.

Liegen verschiedene Versionen derselben Bibliothek vor, kann die für den Build-Prozess zu verwendende Bibliothek über die Kästchen in der Spalte *Used* ausgewählt werden.

2.2.4 Geräteordner

Jeder *Device*-Ordner enthält die Konfiguration und die Dateien für eine einzelne Steuerung.

Sie benötigen die folgenden Hauptbefehle für die Geräteverwaltung:

Kontextmenü *Device, Set Active*

Mit diesem Befehl wird das Gerät im *Project Tree* aktiviert. Das *active device* ist mit einem grünen Dreieck gekennzeichnet. Es können ebenfalls viele Schaltflächen des Hauptmenüs und der *Tool Bar* für das *active device* verwendet werden.
Anmerkung: Dieser Befehl wird nur angezeigt, wenn die Option *Tools, Options…, Project Manager Activate Device according to Project Tree location* auf *No* gesetzt wurde. Ist die Option auf *Yes* gesetzt, aktiviert der *Project Manager* abhängig von der Auswahl im *Project Tree* das Gerät automatisch.

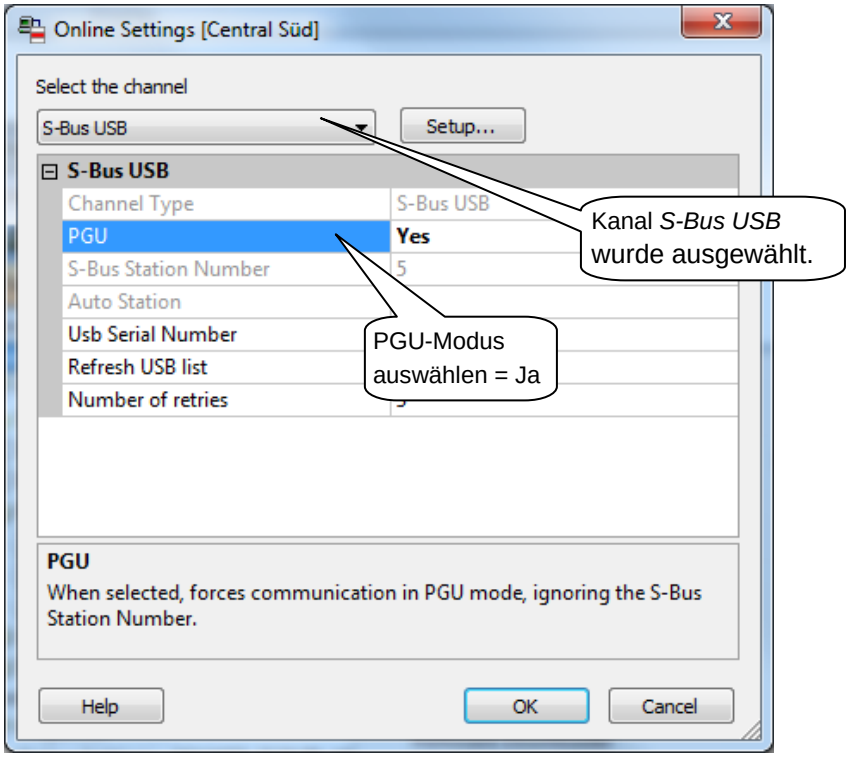
File, Properties…* oder Kontextmenü *Properties…
Mit diesem Befehl können die Eigenschaften eines Geräts angezeigt oder geändert werden: Name, Beschreibung und Option für den Lesezugriff.

[Edit,] Copy, Paste, Delete
Mit dem Befehl *Copy/Paste* kann ein komplettes Gerät im Projekt mit all seinen Dateien und Konfigurationen kopiert werden. Mit *Delete* werden ein Gerät und all seine Dateien in den Papierkorb verschoben.

2.2.5 *Online Settings*

Über diese Verzweigung gelangen Sie zum Fenster *Online Settings*, in dem die Kommunikationsoptionen für den PCD-Anschluss festgelegt werden. Es stehen mehrere Protokolle zur Verfügung: *PGU*, *S-Bus*, *S-Bus USB*, usw. Allerdings bieten nur die Optionen *PGU* und *S-Bus USB* die vollständige Protokollkommunikation, die der *Device Configurator* benötigt.

***Channel* S-Bus USB**



Über die Schaltfläche *Add* können neue Kanäle mit ihren eigenen Typen und Parametern erstellt werden. Diese werden dann in der Liste *Online Settings* der *Channels* angezeigt.

2.2.6 Anschluss des PCs an die PCD

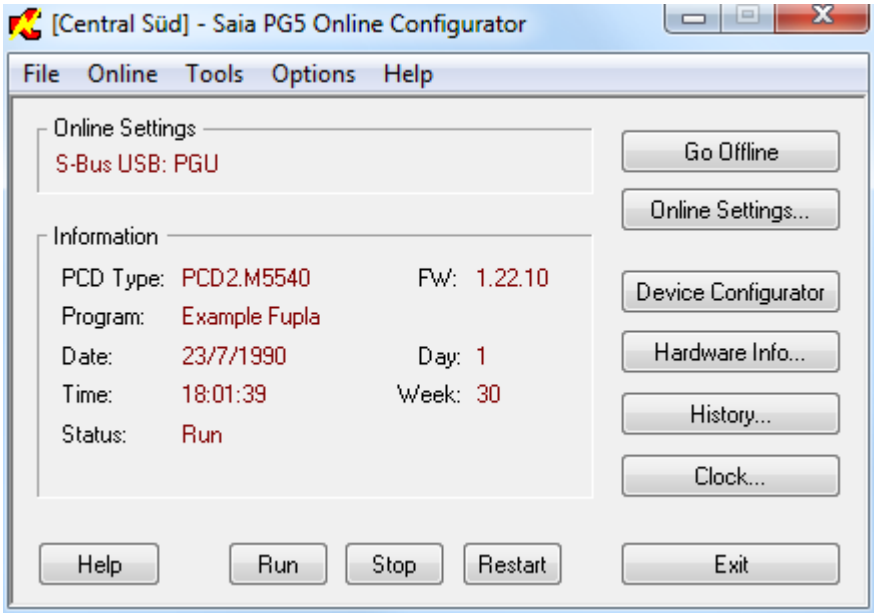
***Channel* S-Bus USB**

Die USB-Verbindung steht nur auf den neuen PCD2- und PCD3-Geräten zur Verfügung. Verwenden Sie ein beliebiges standardmässiges USB-Kabel.

Online Configurator

Überprüfen der Verbindung

Mit der Schaltfläche *Online Configurator* in der Symbolleiste oder über den entsprechenden Menübefehl können Sie die Verbindung zur PCD herstellen und Einzelheiten einsehen. Werden die Informationen in roter Schrift angezeigt, konnte die Verbindung erfolgreich hergestellt werden (die Einzelheiten hängen von der angeschlossenen PCD ab).



Konnte die Verbindung nicht hergestellt werden, wird diese Fehlermeldung angezeigt. Prüfen Sie, ob die PCD mit Strom versorgt wird und prüfen Sie die *Online Settings* und den Kabelanschluss.

2.2.7 *Device Configurator*

Configuration

Im *Device Configurator* werden die Hardware und physischen Funktionen der Steuerung festgelegt, z.B. Gerätetyp, Speicher, Kommunikationskanäle, zugehörige Module und E/As. Hier wird ebenfalls geprüft, ob die Stromversorgung ausreicht. Über den *Device Configurator* können ebenfalls Etiketten für die E/A-Module gedruckt werden.

Um eine PCD in Betrieb zu nehmen, müssen mindestens der PCD-Typ und die Speichergrösse konfiguriert werden. Andere Einstellungen, wie die Kommunikation und E/As können zu einem späteren Zeitpunkt konfiguriert werden.

Am einfachsten beginnen Sie mit der Konfiguration, indem Sie die PCD mit dem USB-Kabel verbinden und die aktuellen Konfigurationseinstellungen aus der PCD über den Befehl *Online, Upload Configuration…* oder entsprechende Schaltflächen in der Symbolleiste auslesen.

Wurde der Speicher der PCD noch nicht konfiguriert, weist er möglicherweise Standardeinstellungen auf. Prüfen Sie immer, ob diese Einstellungen Ihrer Hardware und Anwendung entsprechen.

2.2.8 Build-Optionen

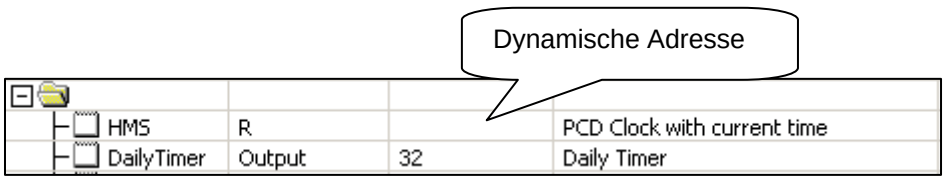
Build Options

Diese Optionen kommen zum Einsatz, wenn das Benutzerprogramm erstellt wird.

Media Allocation	
Last Timer	31
Timer Timebase in milliseconds (10..10000)	100
Has Volatile Flags	**Yes**
Last Volatile Flag	2999
⊞ Dynamic Registers	2000; 4095
⊞ Dynamic Texts	3000; 3499
⊞ Dynamic Data Blocks	3500; 3999
⊞ Dynamic RAM Texts	2000; 2499
⊞ Dynamic RAM Data Blocks	2500; 2999
⊞ Dynamic Timers	5; 31
⊞ Dynamic Counters	1400; 1599
⊞ Dynamic Volatile Flags	2500; 2999
⊞ Dynamic Nonvolatile Flags	7500; 8191

Medienzuweisung
Hier werden Adressbereiche für dynamische Register, Zähler, Timer und Flags reserviert. Wenn das Programm erstellt wird, werden den dynamischen Symbolen, die im Benutzerprogramm und in Fupla-FBoxen festgelegt wurden, automatisch Adressen zugewiesen.

Ein dynamisches Symbol ist ein Symbol, für das keine absolute Adresse festgelegt wurde:



Es ist nicht immer erforderlich, die dynamischen Adressbereiche zu ändern. Die Standardeinstellungen können in der Regel für die meisten Anwendungen verwendet werden.

Wenn jedoch folgende Fehlermeldung während des Build-Prozesses eines grossen Programms erscheint:
Fatal Error 2368: Dynamic space overflow for type: R,
muss der dynamische Adressbereich für den in der Fehlermeldung angezeigten Medientyp erweitert werden.

Wenn die Steuerung über einen EPROM- oder Flash-Speicher als Hauptspeicher verfügt, müssen die dynamischen Bereiche der RAM-Texte und RAM-DB ab Adresse 4000 aufwärts konfiguriert werden, sodass sich diese Texte und DBs im beschreibbaren RAM-Speicher befinden.

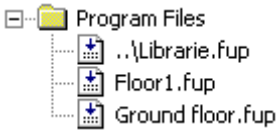
Last Timer
Timer und Zähler verwenden beide dieselben Adressbereiche. Der Wert „Letzter Timer“ legt die Aufteilung zwischen den Timern und Zählern fest (über diesen Wert wird die DEFTC-Anweisung erstellt). Der Adressbereich der dynamischen Timer darf darunter liegen oder bis zu diesem Wert reichen, der Adressbereich für die dynamischen Zähler muss darüber liegen. Wenn beispielsweise der letzte Timer bei 31 liegt, dann gilt für die Timer T 0 bis 31 und für die Zähler C 32 bis 1599.

Timer Timebase in milliseconds
Die standardmässige Zeitbasis, mit der die Timer herunterzählen, ist einmal pro 0,1 Sekunde (100 ms). Dieser Wert kann falls nötig geändert werden. Bitte beachten Sie, dass die Zeitbasis keinerlei Einfluss auf Fupla-Programme hat. Ausschliesslich IL-Programme sind von diesem Parameter betroffen.

Anmerkung: Legen Sie keine unnötig hohe Anzahl an Timern oder eine unnötig kleine Zeitbasis fest, denn dies wird die Zykluszeiten Ihres Programms bremsen.

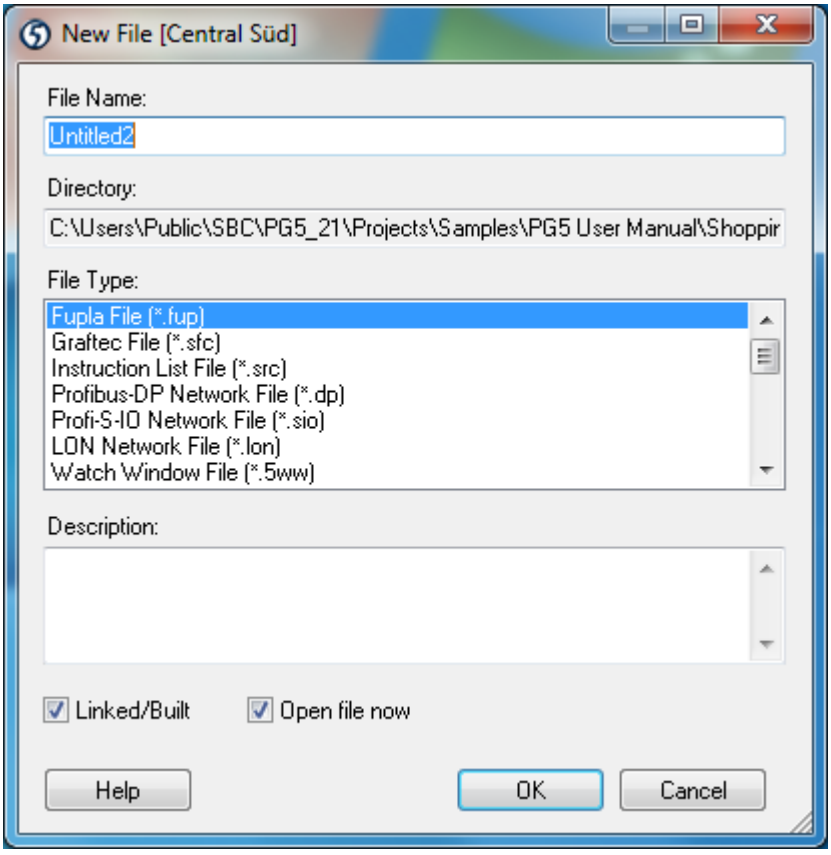
Dynamic Nonvolatile Flags
Standardmässig sind alle Flags nicht flüchtig. Flüchtige Flags werden beim Aufstarten immer auf 0 gesetzt, nicht flüchtige Flags behalten ihre Werte bei. Falls nötig kann über den Parameter *Last Volatile Flag* ein flüchtiger Bereich festgelegt werden. (Die oben stehende Abbildung zeigt flüchtige Flags für die Adressen F 0 bis F 2999).

2.2.9 Ordner für Programmdateien



In diesem Ordner befinden sich die Dateien, aus denen sich das Programm des Geräts zusammensetzt. Sie benötigen die folgenden Hauptbefehle für die Programmdateien:

File, New…* oder Kontextmenü *New File…
Mit diesem Befehl erstellen Sie eine neue Datei und fügen diese im Ordner hinzu.

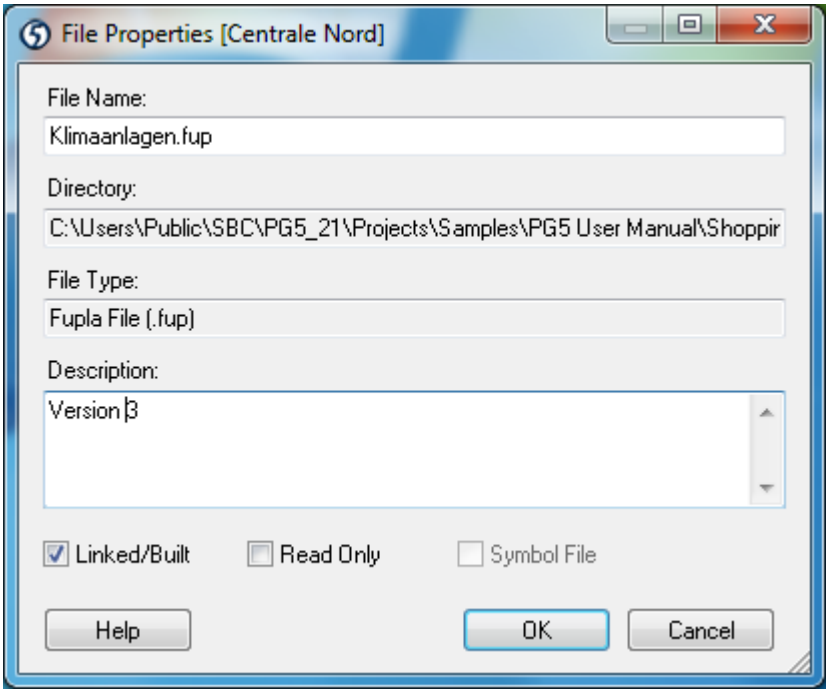


File Name: Name der zu erstellenden Datei.
Directory: Verzeichnis des Geräts, kann nicht geändert werden.
File Type: Der zu erstellende Dateityp.
Description: Freitext für die Beschreibung einer Datei, Verlauf, Versionsinformationen usw.
Linked/Built: Wird diese Option nicht aktiviert, wird die Datei während des Build-Prozesses ignoriert. Sie ist dann kein Bestandteil des Benutzerprogramms.
Open File Now: Ist standardmässig aktiviert, die Datei wird sofort im entsprechenden Editor geöffnet.

Device, Add Files…* oder Kontextmenü *Add Files…
Mit diesem Befehl kann eine oder mehrere Dateien zur Liste *Program Files* hinzugefügt werden. Die Dateien können in das Geräteverzeichnis kopiert oder mit

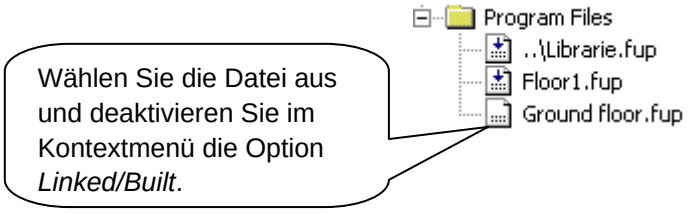
einem Pfad in der Option *Copy files into device directory* im Dialog *Add Files* verlinkt werden.

File, Properties…* oder Kontextmenü *Properties…
Mit diesem Befehl können die Eigenschaften der ausgewählten Datei angezeigt oder geändert werden: Name, Beschreibung und die Optionen *Linked/Built*, *Read Only* und *Symbol File*. Aktivieren Sie die Option *Symbol File*, wenn die Datei nur die Definitionen der globalen Symbole enthält.



[*Edit,*] *Copy, Paste, Delete*
Mit *Copy/Paste* kann eine Kopie der Datei in der aktuellen Liste *Program Files* oder in einem beliebigen Gerät des Projekts erstellt werden. Mit *Delete* werden ein Gerät und alle seine Dateien in den Papierkorb verschoben.

Gerätedateien



Dateien mit einem Pfeil im Symbol werden während des Build-Prozesses verarbeitet. Diese Dateien sind Bestandteil des PDC-Programms, ihr Code und ihre Daten werden in den Speicher der PCD geladen.

Dateien ohne Pfeil im Symbol werden im Build-Prozess **nicht** bearbeitet. Diese Dateien werden ignoriert und nicht in den PCD-Speicher geladen. Dies ist hilfreich im Fall von Test- und Betriebscode, der dann im endgültigen Programm nicht erscheint.

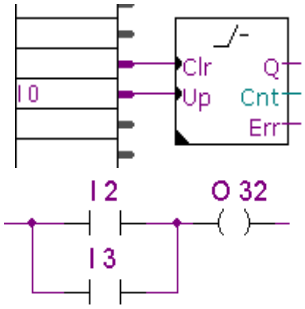
2.2.10 Dateiarten

Für ein Gerät gibt es mehrere Programmdateien unterschiedlicher Art. Jeder Dateityp verfügt über einen zugehörigen Editor für ein spezielles Anwendungsgebiet.

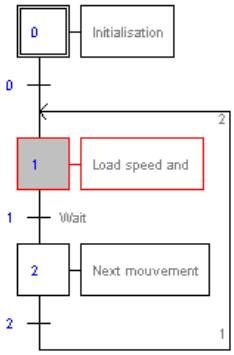
Instruktionsliste-Editor (*.src)
Hier kann die Programmierung in Textform mit IL-Befehlen vorgenommen werden. Geeignet für alle Anwendungen. Der Code ist schnell und effizient, es ist jedoch sehr viel Programmiererfahrung erforderlich.

```
COB  0
     0
STH  I 0
DYN  F 9
INC  C 53
ECOB
```

Fupla-Editor (*.fup)
Hier können Programme in der Form von Funktionsplänen und Kontaktdiagramme erstellt werden. Programmier-erfahrung ist nicht erforderlich. Für die schnelle Umsetzung von HLK-Anwendungen und Kommunikationsnetzwerken (Modem, LON, Belimo, EIB usw.) stehen zahlreiche Bibliotheken zur Verfügung.

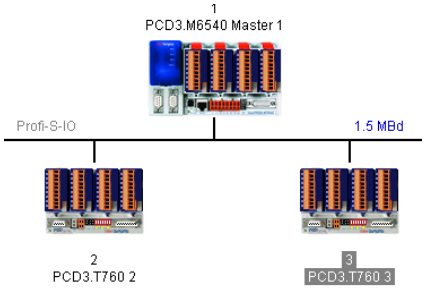


Graftec-Editor (*.sfc)
Hierbei handelt es sich um ein Tool für die Programmstrukturierung in IL (Instruktionsliste) und Fupla. Besonders geeignet für sequenzielle Anwendungen mit Wartezeiten bei internen oder externen Ereignissen.
Der Graftec-Editor ist das optimale Tool für die Programmierung von Befehlen für Motoren, Aktoren usw.



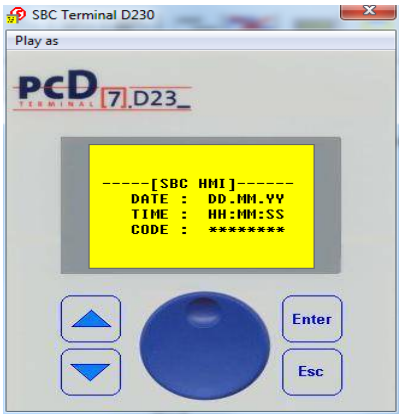
Web-Editor (*.prj)
Editor für Webseiten zur Prozessüberwachung und Kontrolle. Die Seiten können auf der PCD, einem Web-Panel (PCD7.Dxxx) oder auf der Festplatte des PCs gespeichert werden. Web-Panels und PCs zeigen die Seiten in einem Standardbrowser wie Internet Explorer an und können mit einem beliebigen Kommunikationsnetzwerk verwendet werden.

S-Net Netzwerk-Konfigurator (*.dp, *.lon, *.srio)



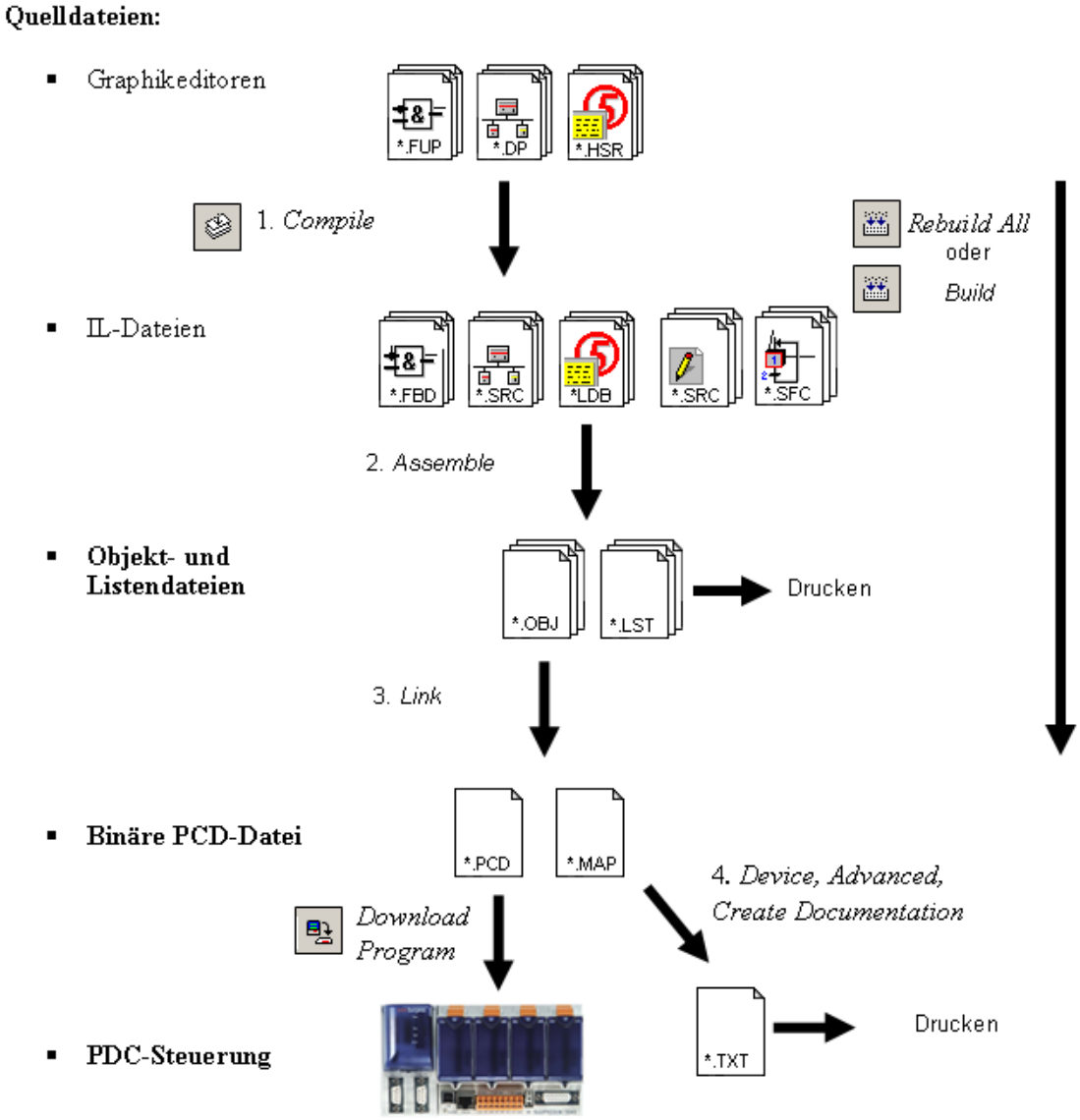
Wird für die Konfiguration von Geräte- und Kommunikationsnetzwerken verwendet: Profibus DP, LON und SRIO.

HMI-Editor (*.hmi)
Mit diesem Editor können einfache Menüs mit PCD7.D1xx- und PCD7.D2xx-Terminals (zusätzlich zu PG5 installiert) konfiguriert werden.



2.3 Programmerstellung

Die PCD kann die in den Editoren erstellten Textdateien nicht direkt verarbeiten. Diese Dateien müssen zunächst zu einer binären ausführbaren Datei im .pcd-Format kompiliert und/oder zusammengestellt werden, wie folgende Abbildung zeigt:



1. Während des Kompiliervorgangs werden die Graphikdateien in Dateien in Instruktionslisten (*.fbd, *.src, *.hsr) umgewandelt.
2. Bei der Zusammenstellung entstehen binäre Objektdateien (*.obj) und ein Bericht (*.lst), der ausgedruckt oder für die Fehlersuche spezieller Assemblerfehler verwendet werden kann.
3. Dann werden die Objektdateien (*.obj) in einer einzigen binären ausführbaren Datei (*.pcd) zusammengeführt, die dann in die PCD geladen werden kann.
4. IL-Dokumentationsdateien werden im Project Manager über den Befehl *Device, Advanced, Create Documentation* erstellt. Die entstandenen Dateien werden im Ordner *Documentation Files* angezeigt.

2.3.1 *Build Changed Files, Rebuild All Files, Rebuild All Devices*

Rebuild All

Mit dem Befehl *Device, Rebuild All Files* wird der Kompiliervorgang gestartet, alle *Linked/Built*-Dateien des *active device* werden zusammengestellt und in einer Datei zusammengefügt.

Der Befehl *Device, Build Changed Files* führt zum selben Ergebnis, jedoch werden nur Dateien berücksichtigt, die seit dem letzten Build verändert wurden. Der Prozess ist sehr viel schneller, insbesondere bei grossen Programmen.

Mit dem Befehl *Project, Rebuild All Programs…* wird die Option *Rebuild All Files* für alle Geräte im aktuellen Projekt ausgeführt. Nach der Ausführung dieses Befehls wird im Fenster *Messages* die Anzahl der Geräte angezeigt, die mit und ohne Fehler erstellt wurden. Über einen Doppelklick auf die roten Build-Fehlermeldungen gelangen Sie zu den Fehlermeldungen der einzelnen Geräte.

2.3.2 Allgemeine *Build Options*

Die Build-Optionen, die von allen Projekten und Geräten gemeinsam genutzt werden, werden über den Befehl *Tools, Options* im Bereich *Build* konfiguriert.

Build	
Ask before saving changed files	Yes
Stop build on first error	No
Download after successful build	No
Download without confirmation	No
Clear message window on build	No
Create Listing files (.lst)	Yes
Page titles and page breaks	No
Disable $NOLIST	No
Hide Graftec parameters	No
Expand Macros	Yes
Cross-reference list	No
Create Map file (.map)	Yes
Create Documentation files (.txt)	No
Lines per page for listing and documentation files	60

Ask before saving changed files
Wurde diese Option auf *Yes* gesetzt, holt die PG5 eine Bestätigung zum Speichern der Quelldateien ein, die geändert, jedoch vor der Programmerstellung nicht gespeichert wurden. Steht die Option auf *No*, werden die Dateien automatisch gespeichert.

Stop build on first error
Wählen Sie *Yes*, wenn Sie die Programmerstellung anhalten möchten, wenn der erste Fehler im Fenster *Messages* angezeigt wird.

Download program after successful build
Steht die Option auf *Yes*, wird das Programm automatisch nach jedem erfolgreichen Build in die PCD geladen.

Download without confirmation
In der Regel beginnt der Ladevorgang mit einem Dialogfenster, in dem der Benutzer den Download über einen Klick auf die Schaltfläche *Download* starten muss. Wurde diese Option auf *Yes* gesetzt, beginnt der Download sofort, es wird kein Dialogfenster angezeigt. Diese Option ist deaktiviert, ausser der Befehl *Download program after successful build* wurde ausgewählt.

Clear message window on build

Der Inhalt des Fensters *Messages* wird zu Beginn eines jeden Build-Prozesses gelöscht.

Create Listing files (.lst)

Mit dieser Option werden Berichtsdateien (.lst) erstellt. Diese können im Ordner *Listing Files* eingesehen werden.

Create Map file (.map)

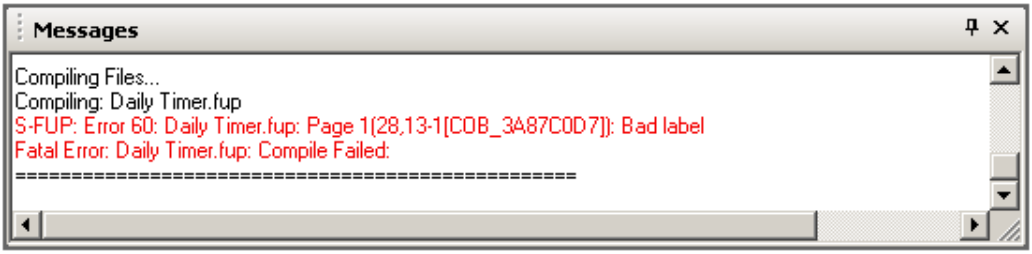
Mit dieser Option wird die Verknüpfung zur Berichtsdatei (.map) erstellt. Diese kann im Ordner *Listing Files* eingesehen werden.

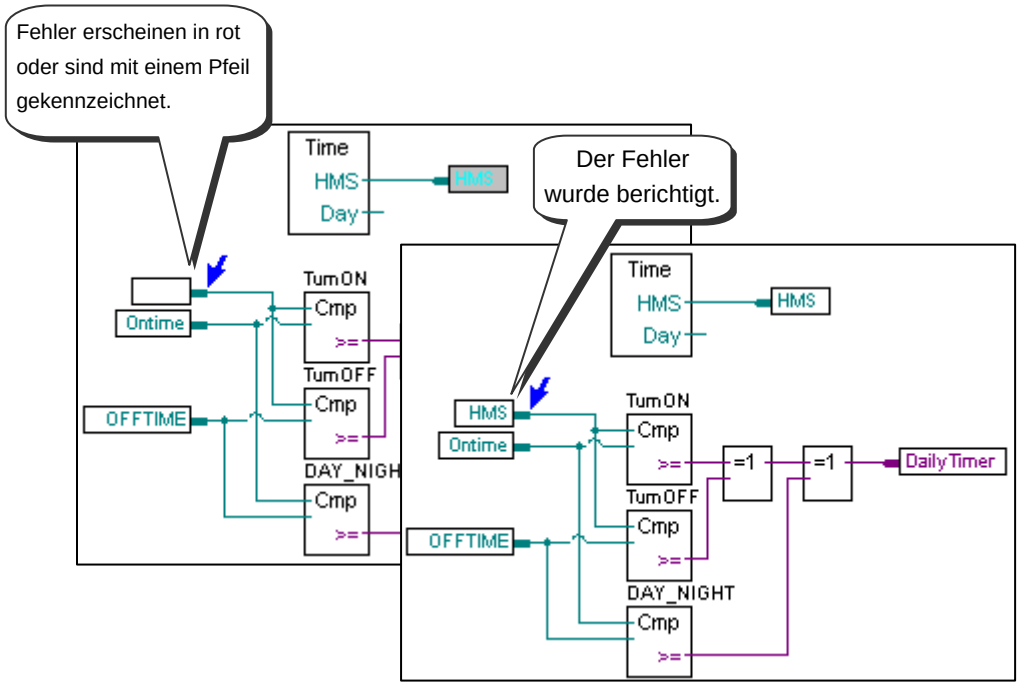
2.4 Fenster *Messages*

Das Fenster *Messages* enthält Informationen zum Fortschritt einer Programmerstellung. Es werden die verschiedenen Phasen des Builds angezeigt: Kompilierung, Zusammenstellung und Verknüpfung. Wenn das Programm korrekt ausgeführt wurde, erscheint am Ende des Build-Prozesses die Nachricht:

Build successful. Total errors 0 Total warnings: 0

Fehler werden in einer entsprechenden Meldung in roter Schrift angezeigt. Über einen Doppelklick auf eine Fehlermeldung öffnet sich der Editor, wo nach Möglichkeit die Fehlerstelle angezeigt wird.
Wenn Sie die Fehlermeldung auswählen und F1 drücken, wird Hilfe zum Fehler angezeigt, falls vorhanden.



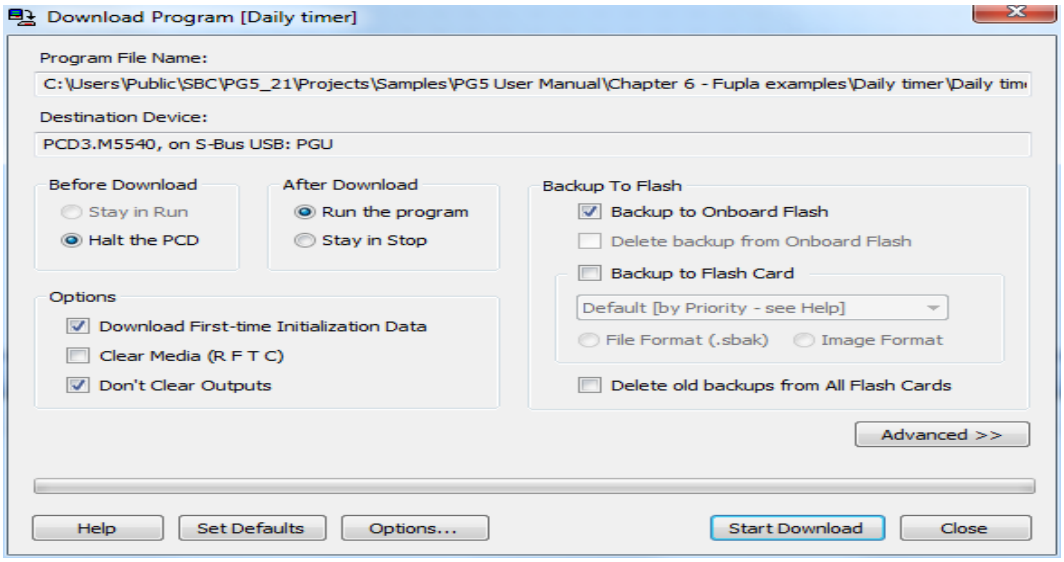


2.5 Laden des Programms in die PCD

2.5.1 *Download Program*

Download Program

War der Build-Prozess erfolgreich, wird über den Befehl *Online, Download Program* oder eine Schaltfläche in der Symbolleiste das ausführbare Programm in den PCD-Speicher geladen.



Program File Name

Dies ist der Standardpfad der PDC-Datei des *active device*.

All

Mit diesem Befehl wird das gesamte Programm geladen (*Code, Text/DB, Extension Memory, Downloadable Files*).

Changed Blocks

Mit dieser Option werden nur die Blöcke (COB, PB, FB, SB, ST, TR, XOB) geladen, die seit dem letzten Download geändert wurden. Diese Option sollte nur aus Zeitersparnisgründen verwendet werden, wenn kleinere Korrekturen während der Programmentwicklung geladen werden sollen. Über die Schaltfläche *Changed Blocks* kann eine Liste mit den geänderten Blöcken angezeigt werden. Weitere Einzelheiten entnehmen Sie bitte der Hilfe.

Download in Run

Mit der Funktion *changed blocks* können geänderte Blöcke geladen werden, ohne die Programmausführung anhalten zu müssen. Der fehlerfreie Betrieb des Programms hängt von den vorgenommenen Änderungen ab. Weitere Einzelheiten entnehmen Sie bitte der Hilfe.

Verwenden Sie diese Option NICHT, wenn Sie sich nicht sicher sind, ob die vorgenommenen Änderungen korrekt sind.

Selected Segments

Mit dieser Funktion werden nur die Bereiche geladen, die in den Optionen *Selected Segments* festgelegt wurden:

Code Segment = Programm, *Text/DB Segment* = Texte und DBs 0…3999, *Extension Memory* = Texte und DBs 4000+, *Downloadable files* = Konfigurationsdateien, wie z.B. BacNet-Konfiguration.

First-time Initialisation Data

Während des Programmdownloads können bestimmte Medien (R T C F) initialisiert werden. *First-time initialisation data* ist folgendermassen festgelegt **:=**

```
symbol EQU type [address] := initialisation_value
```

oder

Untitled1.fup	ROOT		
symbole0	R	10 := 314	First time initialisation value = 324
symbole1	R	11	

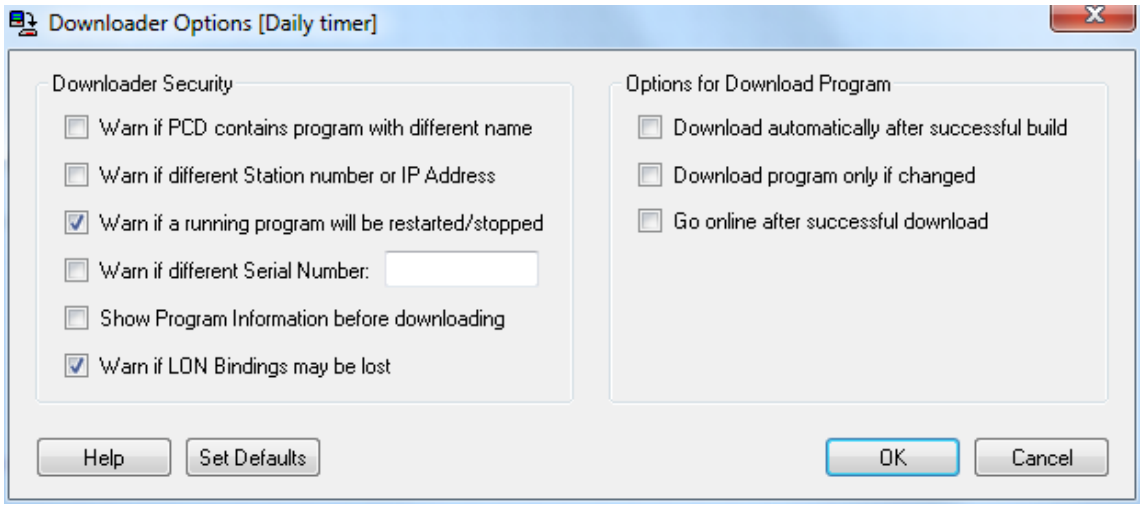
Die Option *First-time initialisation data* wird nicht bei jedem Programmstart initialisiert, sondern nur dann, wenn ein **neues** Programm gestartet wird. Die anderen Daten können über den Code im aufstartenden XOB 16 initialisiert werden.

Copy User Program to Flash

Wenn das Programm erfolgreich in den RAM-Speicher geladen wurde, wird es im Backup-Flash-Speicher gespeichert, falls vorhanden.

2.5.2 *Download Options*

Zusätzliche *download options* können über den Menübefehl *Tools, Options* oder über die Schaltfläche *Options* im Dialogfenster *Download Program* festgelegt werden. Dadurch kann der Downloadprozess individuell gestaltet werden.



Download program only if changed

In diesem Fall werden unveränderte Programme nicht heruntergeladen. Es besteht kein Bedarf, ein Programm herunterzuladen, das nicht verändert wurde.

Download only the changed blocks

Siehe vorhergehende Seite.

Verify all PCD memory writes

Alle Daten, die auf die PCD geschrieben wurden, werden abgerufen und verglichen. Diese Option sollte in der Regel nicht aktiviert werden, da sich sonst die Programmdownload-Zeit verdoppelt. Verwenden Sie diese Option nur, wenn Sie ein seltenes Problem im Speicher-Chip der PCD vermuten.

Run the program after successful download

Mit dieser Funktion wird das Gerät automatisch in Betrieb genommen, nachdem das Programm heruntergeladen wurde.

ACHTUNG: Diese Option sollten Sie nur dann aktivieren, wenn Sie sich sicher sind, dass das Programm korrekt läuft oder wenn kein Risiko besteht, dass bei einem Ausfall Menschen oder Objekte zu Schaden kommen könnten.

Go online after successful download

Mit dieser Funktion wird das Gerät nach einem erfolgreichen Download automatisch online geschaltet.

Backup user program to Flash after download

Kopiert das Programm automatisch in den Flash[1]-Backup-Speicher.

1) PCD2.M170, PCD2.M480, PCD4.M170 et PCD3

Auch wenn diese Option nicht ausgewählt wird, kann nach dem Download eine Kopie erstellt werden. Verwenden Sie hierzu den Befehl *Online, Flash Memory, Copy Program To Flash*.

Warn if PCD contains program with different name

Mit dieser Funktion wird der Name des Programms in der PCD mit dem Namen des herunterzuladenden Programms verglichen. Wenn sich diese Programmnamen unterscheiden, wird eine Meldung angezeigt, um das Herunterladen auf die falsche PCD zu verhindern.

Warn if a running program will be stopped

Das Herunterladen eines Programms hält die PCD an. Abhängig von der Anwendung oder eines Prozesses könnte es gefährlich sein, ein laufendes Programm anzuhalten, z.B. wenn es einen Teilchenbeschleuniger oder ein Kernkraftwerk steuert. Wurde diese Option aktiviert, wird eine Warnmeldung ausgegeben, bevor das Programm angehalten wird.

Do not clear Outputs on download or restart

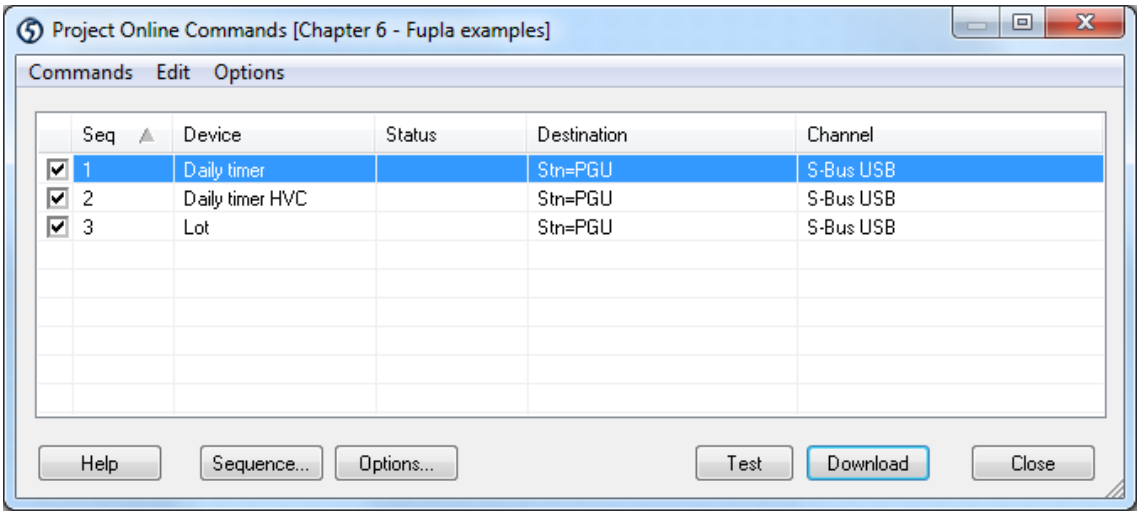
Diese Option kann sich in HLK-Anwendungen als nützlich erweisen. Mit ihr wird verhindert, dass die Belüftung oder Beleuchtung während eines Programm-Downloads abgeschaltet wird. Sie sollte nicht zusammen mit anderen Anwendungen aktiviert werden.

Auto close Up/Download dialog boxes on success

Wurde diese Option aktiviert, schliessen sich die Dialogfenster *Up/Download* automatisch nach einem erfolgreichen Download und bleiben nur dann geöffnet, wenn ein Fehler vorlag.

2.6 Befehle für alle Geräte

Sind der PC und alle Geräte des Projekts über ein Netzwerk miteinander verbunden, liefert das Dialogfenster *Project, Online Commands* einige nützliche Befehle für die Steuerung oder das Downloaden auf alle Geräte im Netzwerk.



Das Dialogfenster enthält eine Liste mit allen Geräten des Projekts. Die Geräte, für die die Befehle gelten sollen, können über Kästchen aktiviert werden. Standardmässig sind alle Geräte aktiviert. Im Kontextmenü der Geräteliste finden Sie eine Reihe von Befehlen, mit denen Sie diese Kästchen bearbeiten können.

Options, Device Sequence

Standardmässig werden die Befehle in der Reihenfolge an die Geräte gesendet, in der sie im *Project Tree* festgelegt sind (alphabetische Reihenfolge). Über diese Option wird ein Dialogfenster angezeigt, in dem Sie die Reihenfolge ändern können.

Options, Options For 'Download Programs'
Mit dieser Funktion können die Optionen für den Befehl *Download Programs* konfiguriert werden.
Anmerkung: Einige dieser Optionen könnten sehr kritisch sein, da hier festgelegt wird, wann für die Geräte die Befehle *Stop* oder *Run* gelten.
Commands, Test
Mit dieser Option wird geprüft, ob alle ausgewählten Geräte online sind und Befehle empfangen. Fällt diese Prüfung negativ aus, überprüfen Sie die *Online Settings* des Geräts und stellen Sie sicher, dass es mit der Stromquelle verbunden und angeschaltet ist.
Commands, Download Programs
Mit dieser Option werden die Programme in alle ausgewählten Geräte geladen.
Commands, Set Clock
Mit dieser Funktion wird die Systemzeit des PCs in alle ausgewählten Geräte kopiert.
Commands, Run/Stop
Mit dieser Option erhalten alle ausgewählten Geräte den Befehl *Run* oder *Stop*.

2.7 Selbstladende Dateien

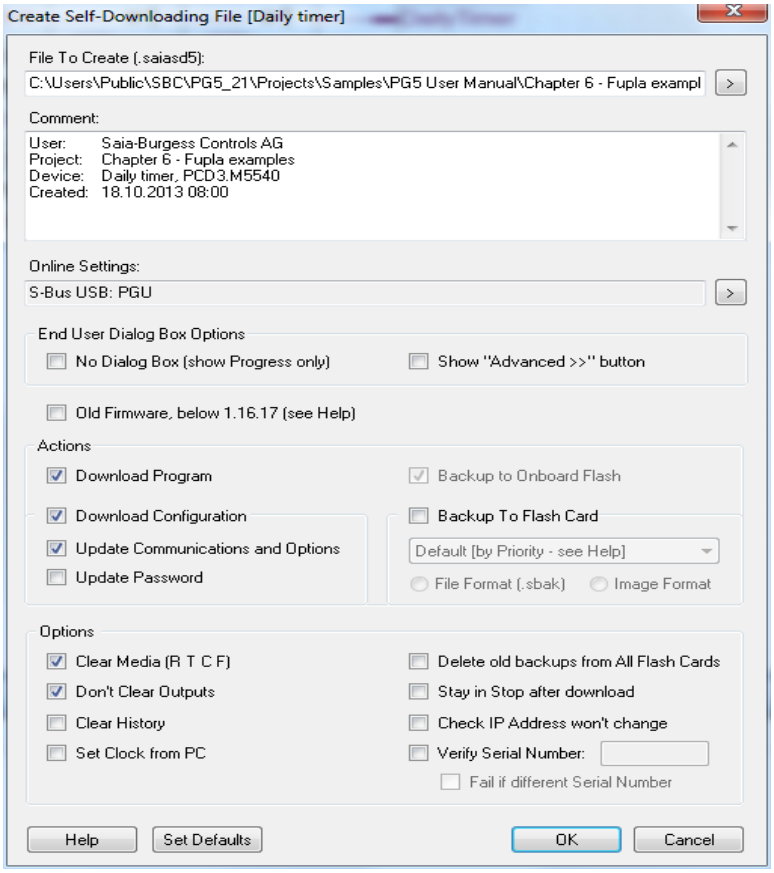
Die selbstladenden Dateien vereinfachen das Herunterladen von Programmen und Konfigurationen auf PCDs vor Ort.

Es wird eine selbstladende *.saiasd5*-Datei erstellt, die alle Informationen für die Aktualisierung des Programms und der Konfiguration auf der PCD enthält. Der PG5-Programmierer schickt lediglich diese Datei per E-Mail an die zuständige Person der PCD vor Ort.

Wenn Sie eine *.saiasd5*-Datei öffnen, wird das Dialogfenster *Download Self Downloading File* geöffnet. Verschiedene Parameter und Optionen der Datei stimmen mit vordefinierten Einstellungen im PG5-Projekt überein. Die Person vor Ort kann diese Optionen unverändert lassen oder sie vor dem Herunterladen auf die PCD ändern.

Somit wird kein Fachwissen zu PG5 benötigt, um Programme oder Konfigurationen in die PCDs zu laden. Diese Funktion kann ohne eine Installation der PG5 oder Benutzerlizenzen ausgeführt werden. Das Paket *Stand Alone Online Tools* muss jedoch auf dem PC installiert sein.

2.7.1 Erstellen einer selbstladenden Datei



Mit dem Befehl *Device, Advanced, Create Self-Downloading File* kann eine selbstladende Datei für das *active device* erstellt werden.

In diesem Dialogfenster können Sie Parameter und Optionen für vor Ort selbstladende Dateien erstellen. Diese Optionen sind identisch mit denjenigen, die bereits aus anderen Menüs bekannt sind: *Download Configuration and Download Program*, jetzt können jedoch auch einige neue Optionen genutzt werden.

Es wird empfohlen, die *Online Settings* und *Device Configuration* auf ihre Richtigkeit zu prüfen und einen *Build*-Prozess erfolgreich auszuführen, bevor die *.saiasd5*-Datei erstellt wird.

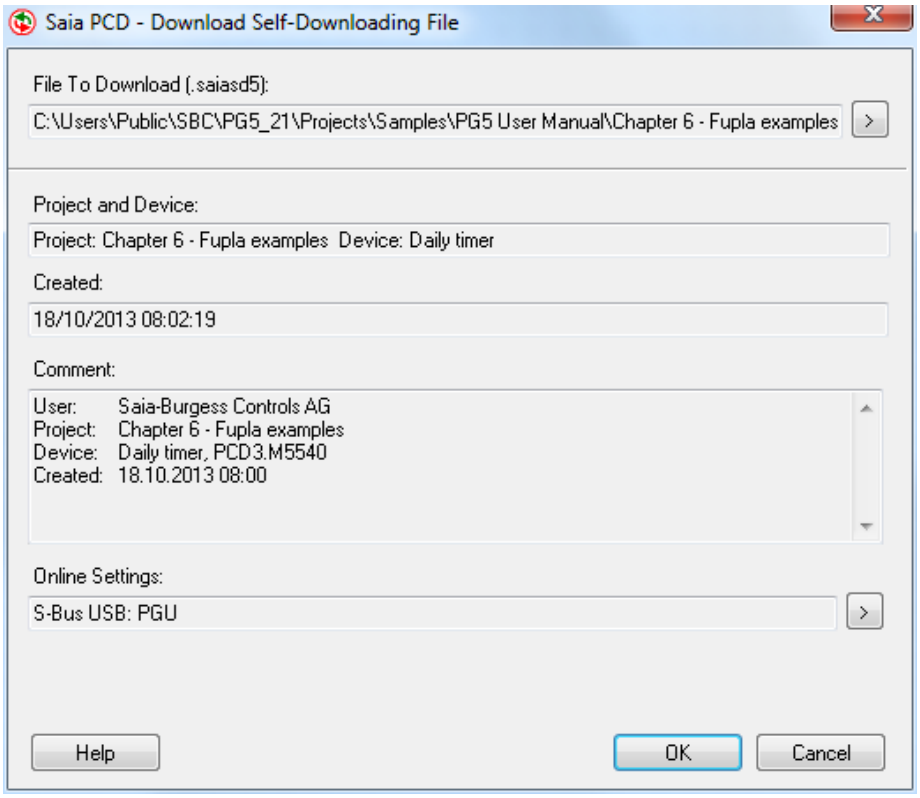
File To Create (*.saiasd5)
Geben Sie den Pfad der zu erstellenden Datei ein. Um einen Pfad auszuwählen, verwenden Sie die Schaltfläche **>**.
No Dialog box (Progress only)
Mit dieser Funktion wird die *.saiasd5*-Datei heruntergeladen, ohne das Dialogfenster *Download Self-Downloading File* anzuzeigen. Der Download beginnt sofort, es wird lediglich ein Fenster mit einem Fortschrittsbalken angezeigt.
Show "Advanced >>" button
Mit dieser Option sind die erweiterten Einstellungen für den Endanwender nicht sichtbar, sodass keine der Einstellungen vor dem Herunterladen der Datei geändert werden kann.
Verify Serial Number

Während des Downloads wird geprüft, ob die Seriennummer der PCD mit der Nummer im Feld *Serial Number* übereinstimmt. Diese Seriennummer ist für jede PCD einmalig und kann daher verwendet werden, um sicherzustellen, dass die Datei auf die richtige PCD heruntergeladen wird.
Anmerkung: Die Seriennummer wird nur von PCD3-Systemen der aktuellen Version unterstützt. Mit dem *Online Configurator* kann die Nummer online abgerufen werden. Verwenden Sie dazu das Menü *Online, Information command*.

2.7.2 Herunterladen einer selbstladenden Datei

Das Paket *Stand-alone Online Tools* muss auf dem PC installiert sein. Weitere Einzelheiten entnehmen Sie bitte der PG5-Installationsanleitung.

Öffnen Sie die Datei *.saiasd5* aus dem *Windows Explorer*, indem Sie einen Doppelklick auf die Datei ausführen. Es wird das folgende Dialogfenster geöffnet, damit Sie Zielort und Einzelheiten überprüfen können, bevor Sie mit dem Download beginnen. Wird die Schaltfläche "*Advanced* >>" angezeigt, können zusätzliche Optionen vor dem Download konfiguriert werden, dies ist in der Regel jedoch nicht notwendig. Starten Sie den Download-Prozess, indem Sie auf die Schaltfläche *OK* klicken.



2.8 Flash-Backup-Speicher

Bestellnummer:	PCD7.R5xx	PCD3.R5xx	PCD.R600
Steckplätze:	M1/M2	E/A-Steckplatz 0..3	E/A-Steckplatz 0..3
Systeme:	PCD1.Mxxx0 PCD2.M5, PCD3	PCD3	PCD3

Alle PCD-Modelle verfügen über einen internen Flash-Speicher. Sie arbeiten darüber hinaus mit Wechselspeichern mit weitaus grösserer Kapazität, die an dedizierten oder E/A-Steckplätzen angeschlossen werden können. Bei Flash-Speichern besteht der Vorteil darin, dass beim Abstellen der Stromquelle im Gegensatz zu RAM keine Daten verloren gehen. Auf dem Flash-Speicher kann eine Sicherungskopie des Benutzerprogramms, eine Kopie des PG5-Quellcodes für das Programm und/oder Daten in einer Datei gespeichert werden, zu der das Benutzerprogramm der PCD über Lese- und Schreibzugriff verfügt.

2.8.1 Speichern des ausführbaren Programms

Das PCD-Programm wird im RAM-Speicher abgespeichert. Bei einem Stromausfall oder im Fall einer leeren Ersatzbatterie kann der Inhalt des Speichers verloren gehen, und das Programm wird nicht starten, wenn die Stromversorgung wiederhergestellt wird. Um eine Sicherungskopie des Programms zu erstellen (*Code/Text/Extension*), kann es auf den Flash-Speicher gespeichert werden, dessen Inhalt sich bei einem Stromausfall nicht ändert.

Über den Menübefehl *Online, Flash Memory, Copy Program To Flash…* wird das Programm auf die Flash-Karte kopiert, über den Befehl *Copy Program From Flash…* wird es wiederhergestellt.
Dies kann automatisch ausgeführt werden, wenn die Download-Option *Backup user program to Flash after download* aktiviert wird.

Geht das Benutzerprogramm im RAM verloren, stellt die PCD beim Aufstarten automatisch das Programm aus dem Flash-Speicher auf dem RAM wieder her.

Wir empfehlen einen Flash-Speicher in Kombination mit Ihrer PCD, damit Sie sich vor unerwünschtem Datenverlust schützen können.

Anmerkung: Die Quelldateien des Programms müssen getrennt gesichert werden, da nur die ausführbare Datei in die PCD geladen wird. Siehe Angaben im nächsten Abschnitt.

2.8.2 Speichern des Programmquellcodes

Die Programmquelldateien werden auf der Festplatte des PCs gespeichert. Nur der ausführbare Code, der im *Build*-Prozess erstellt wird, wird in den Speicher der PCD geladen. Ohne die Quelldateien können die Programme im Rahmen von Aktualisierungen oder Wartungsprozessen nicht geändert werden. Daher ist es

wichtig, dass Sie Sicherungskopien anlegen und immer die aktuelle Version für die Wartungstechniker parat halten.

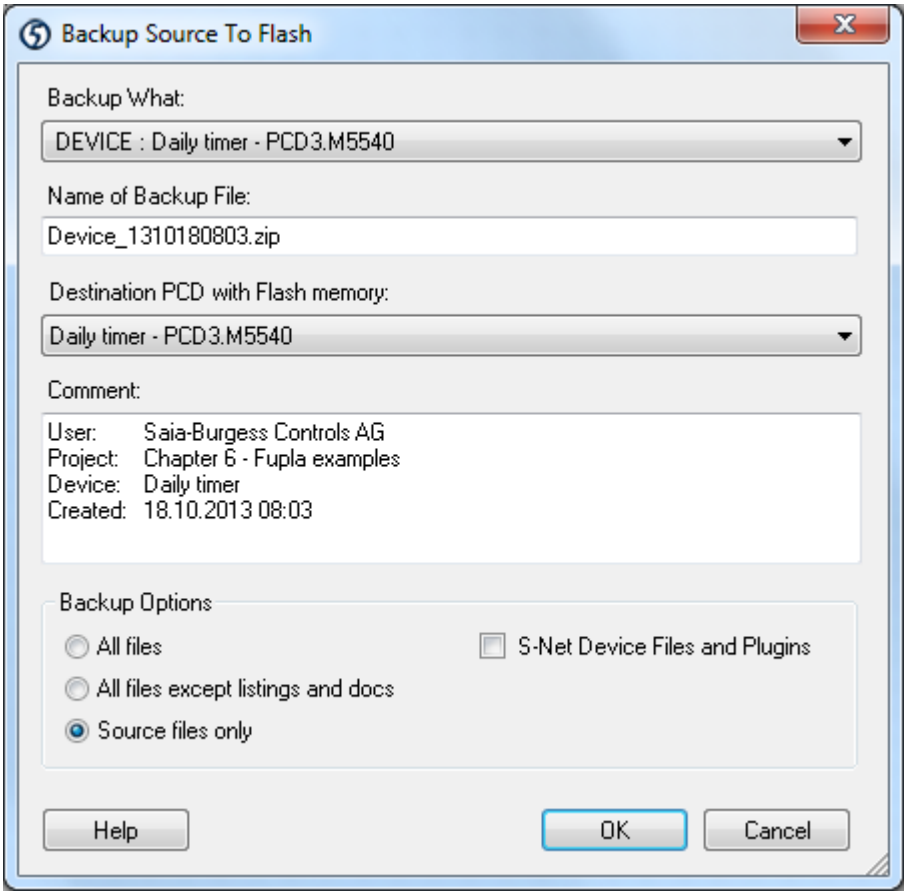
Quelldateien können auf zwei Arten abgespeichert werden:

- Mit dem Befehl *Project, Backup…* werden alle Verzeichnisse und Dateien in einer .zip-Datei gespeichert. Mit dieser Einzeldatei können die Verzeichnisstruktur und das Dateilayout ganz einfach beibehalten werden. Das Projekt wird ausgehend von der .zip-Datei über den Befehl *Project, Restore…* wiederhergestellt. Weitere Einzelheiten entnehmen Sie bitte Abschnitt 2.1.4.

- Mit dem Befehl *Online, Flash Memory, Backup Source To Flash…* wird die Sicherungsdatei im .zip-Format erstellt und auf den Flash-Speicher der PCD über FTP und eine Ethernet-Verbindung heruntergeladen. Die Quelldatei wird über das Hochladen und Wiederherstellen der Datei mit dem Befehl *Online, Flash Memory, Restore Source From Flash…* wiederhergestellt. Dieser Vorgang wird im Folgenden beschrieben.

Sicherung der Quelldateien auf den Flash-Speicher

Der Befehl *Online, Flash Memory, Backup Source To Flash…* beginnt mit der Komprimierung des gesamten Projekts oder eines einzelnen Geräts in eine standardmässige .zip-Datei:



Backup Project or single Device
Standardmässig ist das *active device* ausgewählt. Dies kann jedoch geändert werden, um eine Sicherungskopie des Gesamtprojekts zu erstellen.
Name of Backup File

Name der zu erstellenden Sicherungsdatei. Wenn sie auf dem Flash-Dateisystem der PCD gespeichert werden soll, darf der Name nicht länger als 23 Zeichen einschliesslich der Dateiendung ".zip" sein.

Destination PCD with Flash Memory

Hier wird die PCD angezeigt, in die die .zip-Datei geladen wird.

Comment

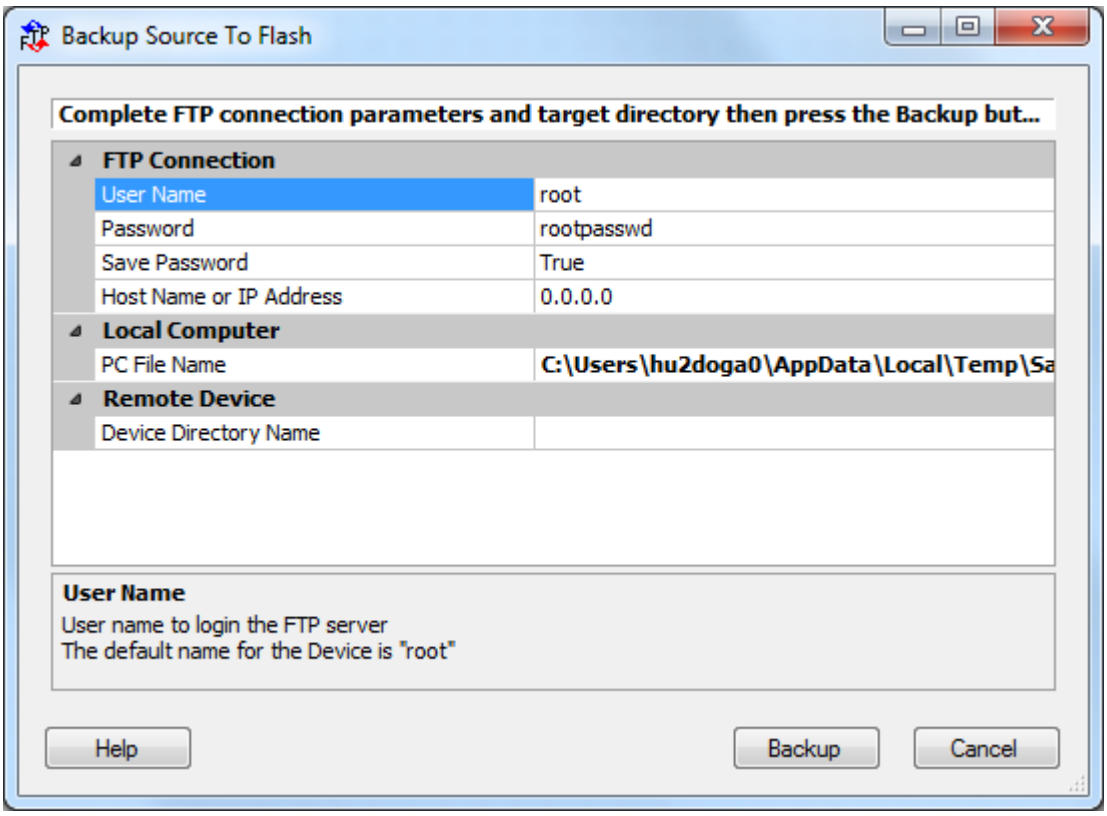
Freitext, standardmässig werden hier Benutzername, Projekt, die Namen der Geräte und Datum/Uhrzeit angegeben. Sie können hier ebenfalls eine Revisionsnummer und sonstige Einzelheiten angeben.

Backup What

Hier werden die zu sichernden Dateien ausgewählt. Es ist nicht notwendig, alle Dateien zu sichern. Ausschliesslich die Quelldateien sind wichtig.

Wenn Sie auf OK klicken, wird die .zip-Datei erstellt. Danach wird der FTP-Downloader gestartet.

FTP-Downloader



User Name
Name, mit dem sich der Benutzer am FTP-Server identifiziert. Wurde kein Benutzer festgelegt, verwenden Sie die Standardeinstellung: *root*

Password
Der Zugriff auf den FTP-Server ist passwortgeschützt. Wurde kein Passwort festgelegt, verwenden Sie die Standardeinstellung: *rootpasswd*

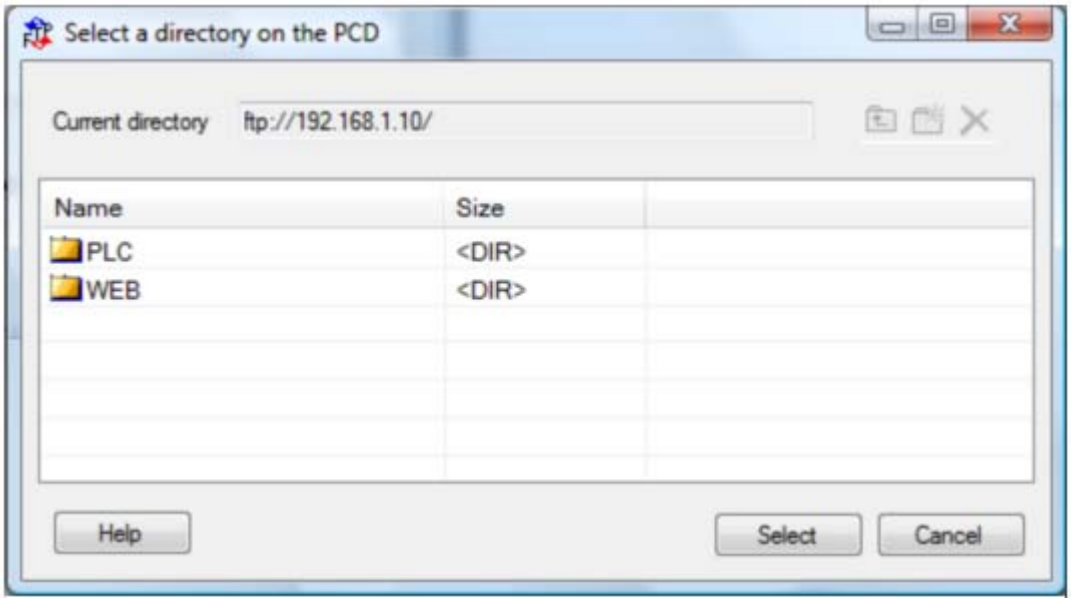
Anmerkung: Möglicherweise sind das FTP-Passwort und das PCD-Passwort nicht identisch. Das FTP-Server-Passwort ist in einer Konfigurationsdatei im Flash-Speicher mit dem Namen *FTPConfig.txt* festgelegt. Das Passwort für den Zugriff auf die PCD ist im *Device Configurator* festgelegt.

Save Password
Wurde diese Option auf *True* gesetzt, wird das Passwort für den nächsten Aufruf dieser Funktion gespeichert.

IP Address
IP-Adresse der Ziel-PCD.

PC File Name
Name der herunterzuladenden Datei. Standardmässig ist hier der Pfad der .zip-Datei angegeben, die im vorhergehenden Schritt erstellt wurde.

PCD Directory Name
Wählen Sie diesen Eintrag aus und klicken Sie auf die Schaltfläche rechts. So erhalten Sie eine Liste mit Flash-Speicherkarten vom FTP-Server. Wählen Sie den gewünschten Zielort für die Datei aus.



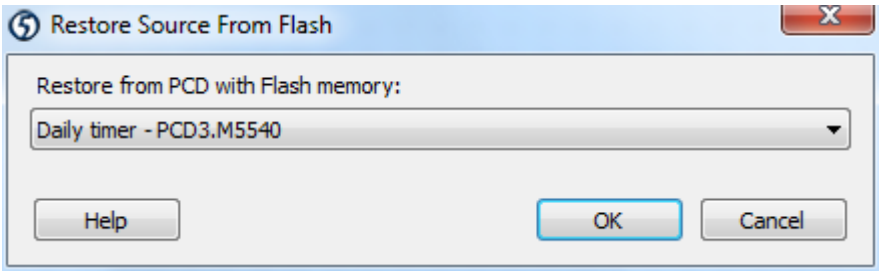
Es dauert womöglich eine kurze Zeit, bis die oben angegebenen Informationen heruntergeladen und angezeigt werden. Stehen mehrere Flash-Speicher zur Verfügung, werden die einzelnen Speicher mit der Verzeichnisstruktur angezeigt. Wählen Sie den Flash-Speicher und das Verzeichnis. Falls nötig können die Verzeichnisse mit den zugehörigen Schaltflächen erstellt und gelöscht werden.

M1_FLASH und M2_FLASH
Flash-Speicherkarten an den Steckplätzen M1 und M2 der PCD.
SL0FLASH, SL1FLASH, usw.
Flash-Speichermodule an den Steckplätzen 0 bis 3 der PCD.

Anmerkung: Diese Funktion wird nur von den neuen PCD-Modellen mit Flash-Speicher mit dem neuen *Flash File System* unterstützt.

Wiederherstellen eines Projekts oder eines Geräts aus dem Flash-Speicher

Dies wird über den Befehl *Online, Flash Memory, Restore Source From Flash...* ausgeführt. Damit wird die .zip-Datei aus dem Flash-Speicher geladen und entzippt.



Nach einem Klick auf OK wird das Dialogfenster mit dem FTP-Uploader mit denselben Parametern wie für den Downloadprozess angezeigt. Mit der Schaltfläche *PCD Directory Name* stellen Sie die Kommunikation mit dem FTP-Server her und können den Flash-Speicher und das Verzeichnis mit der wiederherzustellenden .zip-Datei auswählen. Bestätigen Sie den Upload oder das Wiederherstellen des Projekts oder Geräts.

2.8.3 Sichern von Daten in einer Datei

Die FBox-Bibliothek mit dem Namen *File System* unterstützt den Zugriff auf Datendateien auf dem Flash-Speicher. Für diese Dateien besteht Lese- und Schreibzugriff. Sie können über FTP hoch- und heruntergeladen werden.

Weitere Informationen zur verfügbaren Anzahl und den Typen des Flash-Speichers, zur Konfiguration des FTP-Servers und zu den Möglichkeiten bei der Erstellung von Datendateien mit Fupla oder IL-Programmen entnehmen Sie bitte den folgenden Handbüchern:

SBC FTP Server und SBC Flash-Dateisystem
Hardware-Handbuch für die PCD3-Serie
Hardware-Handbuch für PCD2.M5
Hardware-Handbuch für PCD1.Mxxx0

2.9 Die *View*-Fenster

Die Informationen in diesen Fenstern sind womöglich nicht ganz korrekt, wenn Build-Fehler auftraten. In PG5 V2 werden jedoch alle verfügbaren Informationen angezeigt. In V1.x sind diese Fenster immer leer, wenn es zu einem Build-Fehler kam.

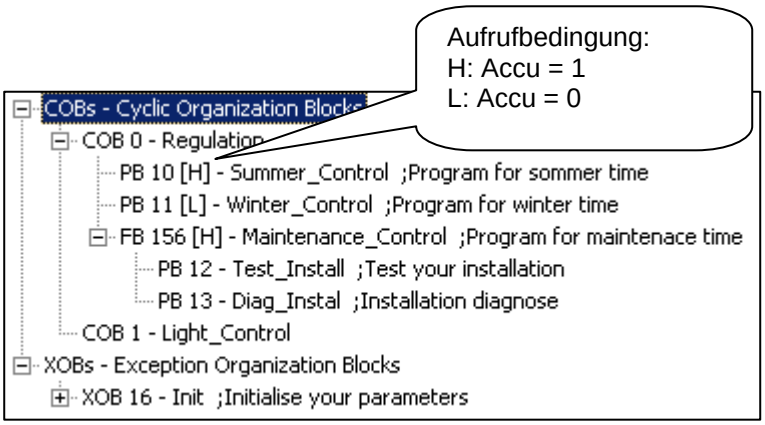
2.9.1 *Block Call Structure*

Block Call Structure

Ein Saia-PCD-Programm setzt sich aus einer Baumstruktur mit Organisationsblöcken zusammen, die den Code der Anwendung enthalten. Jeder Block umfasst einen eigenen Dienst: zyklische Programmierung (COB), sequenzielle Programmierung (SB), Unterprogramme (PB), Funktionen mit Parametern (FB) und Ausnahmeroutinen (XOB).

Die Gesamtstruktur der Blöcke, aus denen sich das Programm zusammensetzt, wird über einen Klick auf die Schaltfläche *Block Call Structure* in der Symbolleiste angezeigt. Es kann ebenfalls der Menübefehl *View, Block Call Structure* ausgeführt werden.

Das folgende Beispiel zeigt ein Programm mit den Blöcken COB 0, COB 1, XOB 16, PB 10, PB11 und FB 156. Bitte beachten Sie, dass COB 0 die drei Unterblöcke (PB 10, PB 11 und FB 156) nur bedingt aufruft. Die Aufrufbedingung wird in Klammern angegeben.



2.9.2 Ansichten *Global Symbols* und *Data List*

Über die Befehle *View, Global Symbols* und *View Data List* werden die vom Programm verwendeten Symbole und Daten angezeigt.

All Publics Symbols

- Über die Option *All Publics Symbols* werden alle globalen Symbole, die in allen Dateien des *active device* verwendet werden, im Symbol Editor angezeigt. Diese Liste kann nicht bearbeitet werden. Zur Bearbeitung der globalen Symbole müssen Sie die Datei öffnen, in der sie festgelegt sind.

- Die *Data List* zeigt alle Symbole und Daten, die vom *active device* verwendet werden. Globale und lokale Symbole und absolute Adressen werden ebenfalls angezeigt. Diese Liste kann nicht bearbeitet werden.

enliste

Symbol	Type	Address/...	Scope	Module	Comment
DailyTimer	O	32		Daily Timer.fbd	Daily Timer
HMS	R	2003	AUTO	Daily Timer.fbd	PCD Clock with current time
OFFTIME	R	2004	AUTO	Daily Timer.fbd	Switch off time
ONTIME	R	2005	AUTO	Daily Timer.fbd	Switch on time

Wurde ein lokales Symbol festgelegt, jedoch nicht im Programm verwendet, wird es in diesen Fenstern nicht angezeigt.

2.9.3 Verweisliste

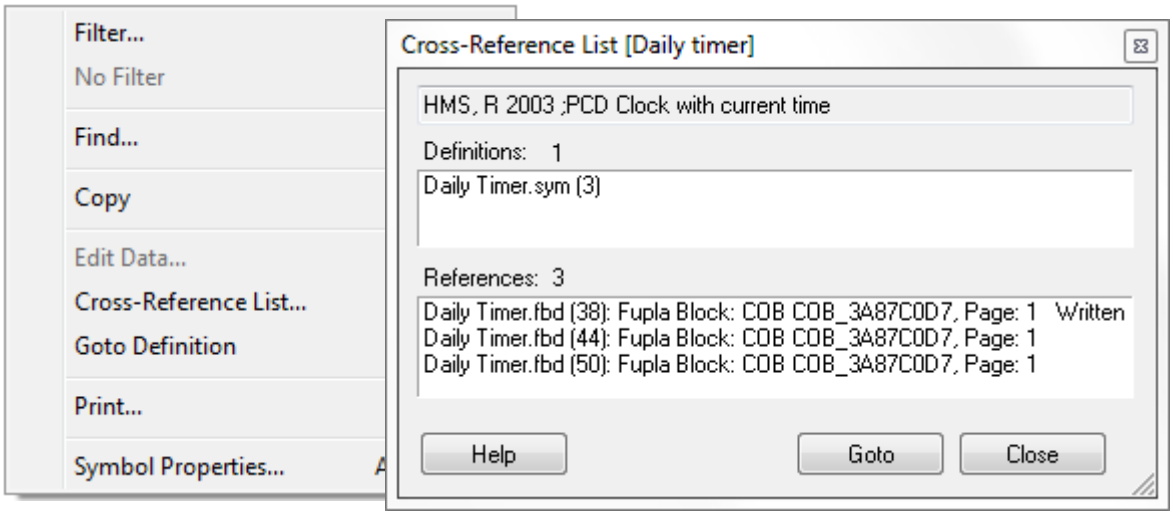
In den Ansichten *All Publics Symbols* und *Data List* können Symbole ausgewählt und die zugehörige *cross-reference list* angezeigt werden. Hierbei handelt es sich um eine Liste mit allen Programmstellen, an denen das Symbol verwendet wird.

Jeder Eintrag zeigt den Dateinamen und Block, in dem das ausgewählte Symbol verwendet wird. Auch Zeilen- oder Seitenzahl werden angezeigt. Es wird ebenfalls angegeben, ob das Symbol an dieser Stelle geändert werden kann.

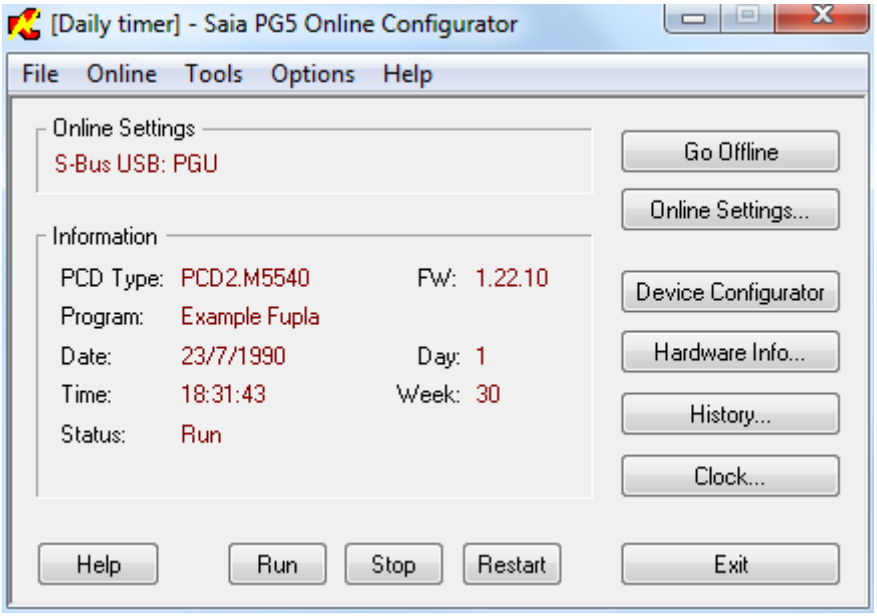
Die Liste *Definitions* gibt an, wo das Symbol festgelegt wurde, z.B. wo die IL-EQU-Angabe gefunden werden kann. In der Liste *References* wird angegeben, wo das Symbol im Programm genutzt wird.

Bei Blöcken gibt die Kennzeichnung „>>“ an, wo der entsprechende Block zu finden ist.

Um das Programm mit dem verwendeten Symbol anzuzeigen, wählen Sie die Definition oder Referenz und klicken Sie auf die Schaltfläche *Goto*.



2.10 Der Online-Konfigurator



Online Settings	Kommunikationsparameter für die PCD-Verbindung
PCD Type	Der Typ der angeschlossenen PCD
Version	Version der Firmware in der PCD
Program Name	Name des Programms in der PCD (Gerätename)
Date	Datum der PCD-Uhr, falls vorhanden
Time	Uhrzeit der PCD-Uhr, falls vorhanden
Day	Wochentag: 1 = Montag, ... 7 = Sonntag
Week	Woche 1 bis 52
Status	Betriebsart: Run, Stop, Halt, Conditional Run

Werden die roten Informationen nicht angezeigt oder erscheint eine Fehlermeldung, bedeutet dies, dass der *Online Configurator* nicht mit der PCD kommunizieren konnte.
Bitte prüfen Sie Folgendes:

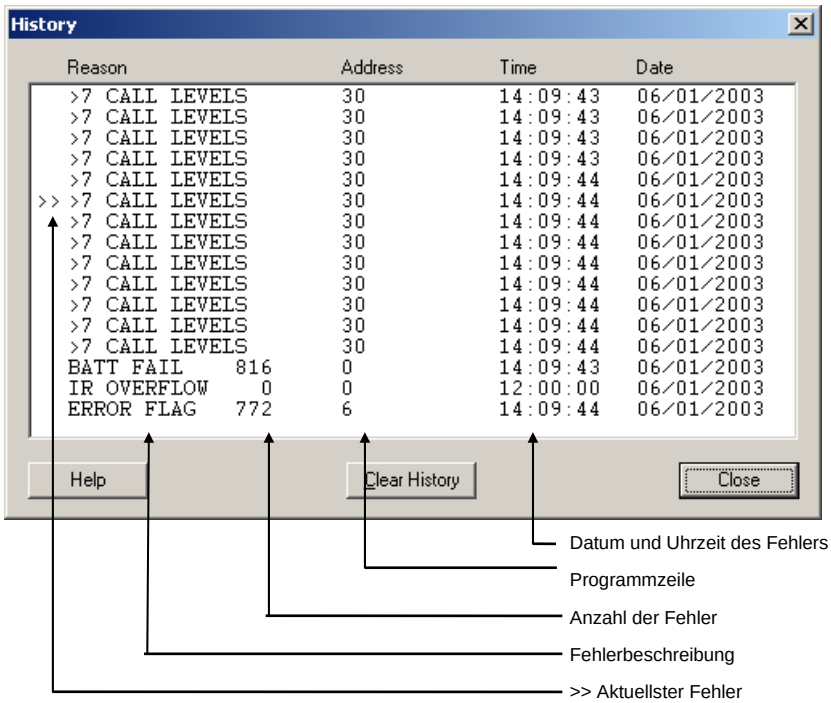
- Die PCD ist korrekt mit dem PCD8.K111- oder USB-Kabel verbunden, an das Netzwerk angeschlossen und eingeschaltet.
- Die Kommunikationsparameter sind korrekt. Klicken Sie auf die Schaltfläche Online Settings.

2.10.1 Gerätekonfigurator

Die Konfiguration wird hochgeladen und der Device Configurator wird geöffnet.
Weitere Einzelheiten entnehmen Sie bitte der Dokumentation des Konfigurators.

2.10.2 PCD-History

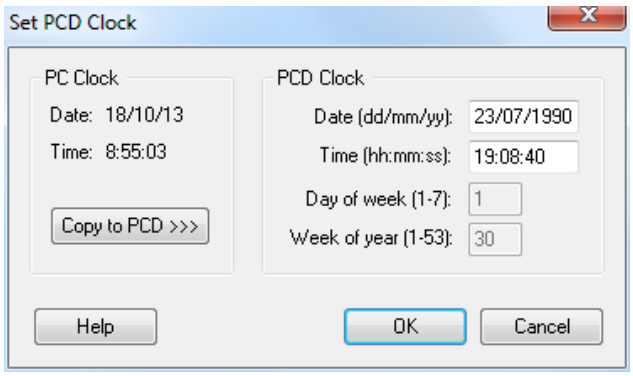
Im Fenster *History* werden alle Hardware- oder Software-Fehler angezeigt, die während des PCD-Betriebs auftraten. History-Meldungen werden immer in diese Liste aufgenommen, auch wenn der zugehörige XOB-Handler programmiert wurde. Prüfen Sie diese Liste, wenn die *Error*-Leuchte der PCD brennt.



Anmerkungen:

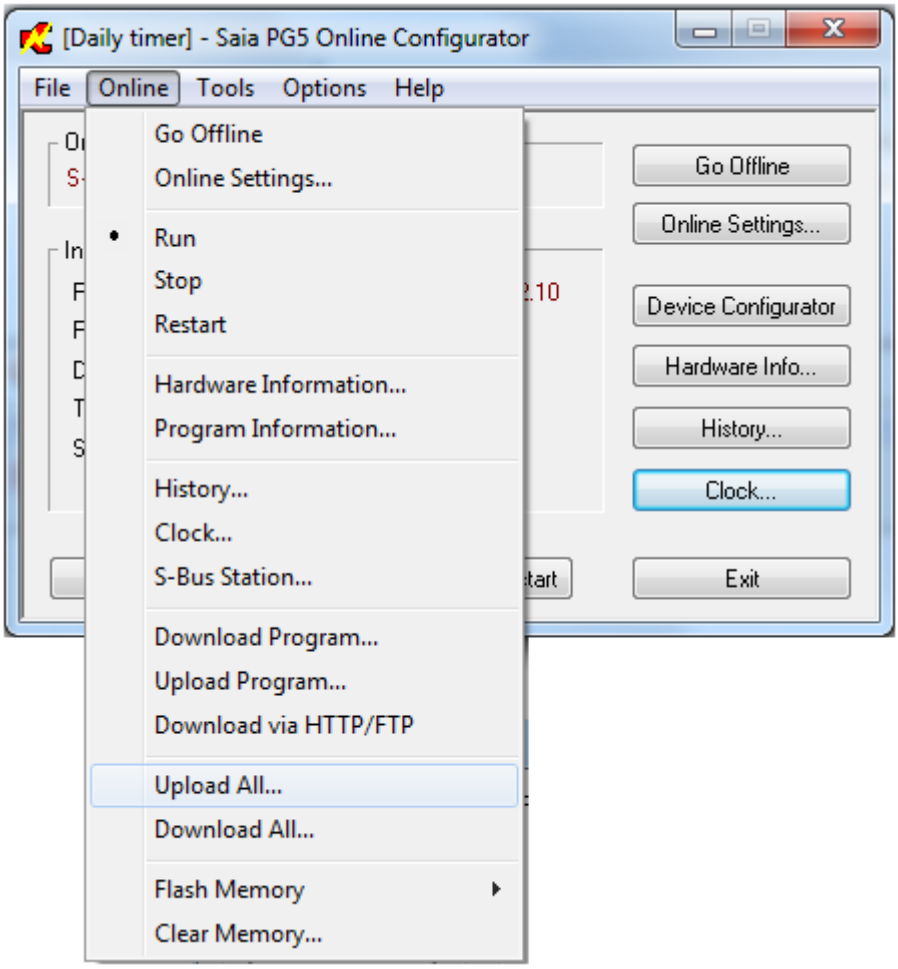
- Trat der Fehler in einer Programmzeile auf, wird unter *Address* die Zeilenzahl angegeben. War dies nicht der Fall, wird ein Hexadezimal-Wert angezeigt.
- XOB 0-Meldungen (Strom aus) werden nur dann angezeigt, wenn XOB 0 programmiert wurde.

2.10.3 Einstellen der PCD-Uhr



Die meisten PCDs verfügen über eine Echtzeit-Uhr, die Datum und Uhrzeit angibt. Über die Schaltfläche *Clock…* wird das o.a. Dialogfenster geöffnet, in dem Datum und Uhrzeit eingesehen und angepasst werden können. Mit der Schaltfläche *Copy to PCD* >> können Datum und Uhrzeit der PC-Systemzeit sofort für die PCD-Uhr übernommen werden. Optional können Datum und Uhrzeit manuell eingegeben und übertragen werden, wenn Sie auf OK klicken.

2.10.4 Speichern von Programm und Daten aus RAM



Hierbei handelt es sich um einen interessanten Befehl, mit dem Sie das Benutzerprogramm und die Konfiguration und darüber hinaus alle Register, Flags, Timer, DB, Texte usw. aus dem RAM-Speicher der PCD speichern und wiederherstellen können. Dieser Prozess ist äusserst hilfreich, wenn Sie Programme in andere PCDs kopieren, eine Installationskopie erstellen, die PCD wechseln oder einfach nur die PCD in einem früheren gespeicherten Status wiederherstellen.

Tools, Upload All…
Mit diesem Befehl wird der RAM-Inhalt in eine Datei im Format .im5 (PG5-Bildspeicherdatei) geladen und gespeichert.

Tools, Download All…
Mit diesem Befehl wird eine .im5-Datei in den RAM der PCD geladen.

2.10.5 *Create Diagnostic File*

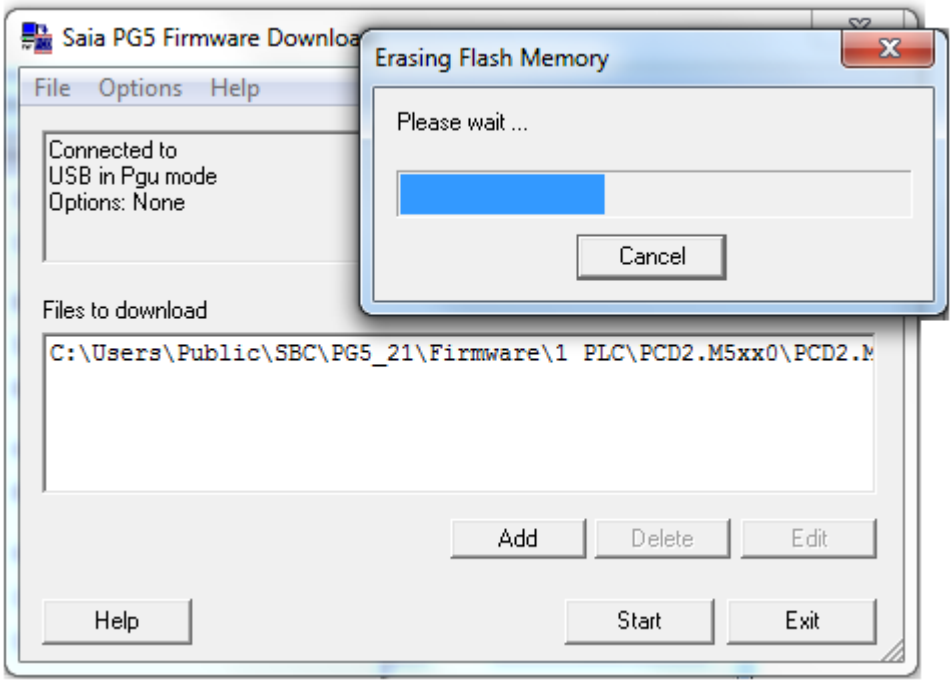
Mit dieser nützlichen Funktion können Sie eine Datei mit allen Informationen erstellen, die benötigt werden, wenn Sie Hilfe vom technischen Support-Team von Saia in Anspruch nehmen. Diese Datei enthält Einzelheiten zum PCD-Typ, zur Firmware-Version usw.

Verwenden Sie den Befehl ***Tools, Create Diagnostic File…*** und notieren Sie sich den Pfad der erstellten Datei.

2.10.6 Firmware-Downloader

Damit eine PCD mit den aktuellen Funktionen arbeiten kann, muss möglicherweise die Option *firmware* in der PCD aktualisiert werden. Dies ist ganz einfach, da alle neuen PCDs die Firmware im Flash-Speicher ablegen. Die aktuelle Firmware-Version kann dem Hauptfenster des *Online Configurator* entnommen werden.

Die Firmware kann über den Befehl *Tools, Firmware Downloader* im Project Manager oder dem Online Configurator heruntergeladen werden.



Über die Schaltfläche *Add* wird eine neue Firmware-Datei (.blk) in die Liste *Files to download* aufgenommen. In der Liste wird angegeben, welche Datei zuletzt heruntergeladen wurde. Bei Sonderanwendungen kann die Liste mehrere Dateien enthalten, bitte stellen Sie jedoch für den normalen Gebrauch sicher, dass nur die Firmware-Datei für die angeschlossene PCD verfügbar ist. Die aktuellsten Firmware-Dateien stehen in einem Verzeichnis auf der PG5-Vertriebs-CD zur Verfügung.

Über den Befehl *Options, Online Settings…* werden die Kommunikationsparameter in der Regel im *S-Bus USB* oder *PGU*-Modus festgelegt.

Klicken Sie auf die Schaltfläche *Start*, um den Download der Firmware zu starten. Nach wenigen Sekunden wird ein Dialogfenster mit einem Fortschrittsbalken angezeigt.

Nach Abschluss blinken die *Run*-, *Halt*- und *Error*-LEDs. Die PCD speichert die Daten. Nachdem die LEDs aufgehört haben zu blinken, warten Sie bitte ca. 1 Minute, bevor Sie die PCD ausschalten oder Ihre Arbeit fortsetzen.

Inhaltsverzeichnis

3 Device configurator

Der *Saia PG5 Device Configurator* ermöglicht die Konfiguration der Hardwareparameter des Steuerung: den Typ des *Device*, den Speicher, die Kommunikationskanäle, die Ein-/Ausgangsmodule, er kontrolliert jedoch auch den Verbrauch der Ein-/Ausgangsmodule bei der internen Stromversorgung des PCD und druckt die Etiketten, die auf die E/A-Module geklebt werden.

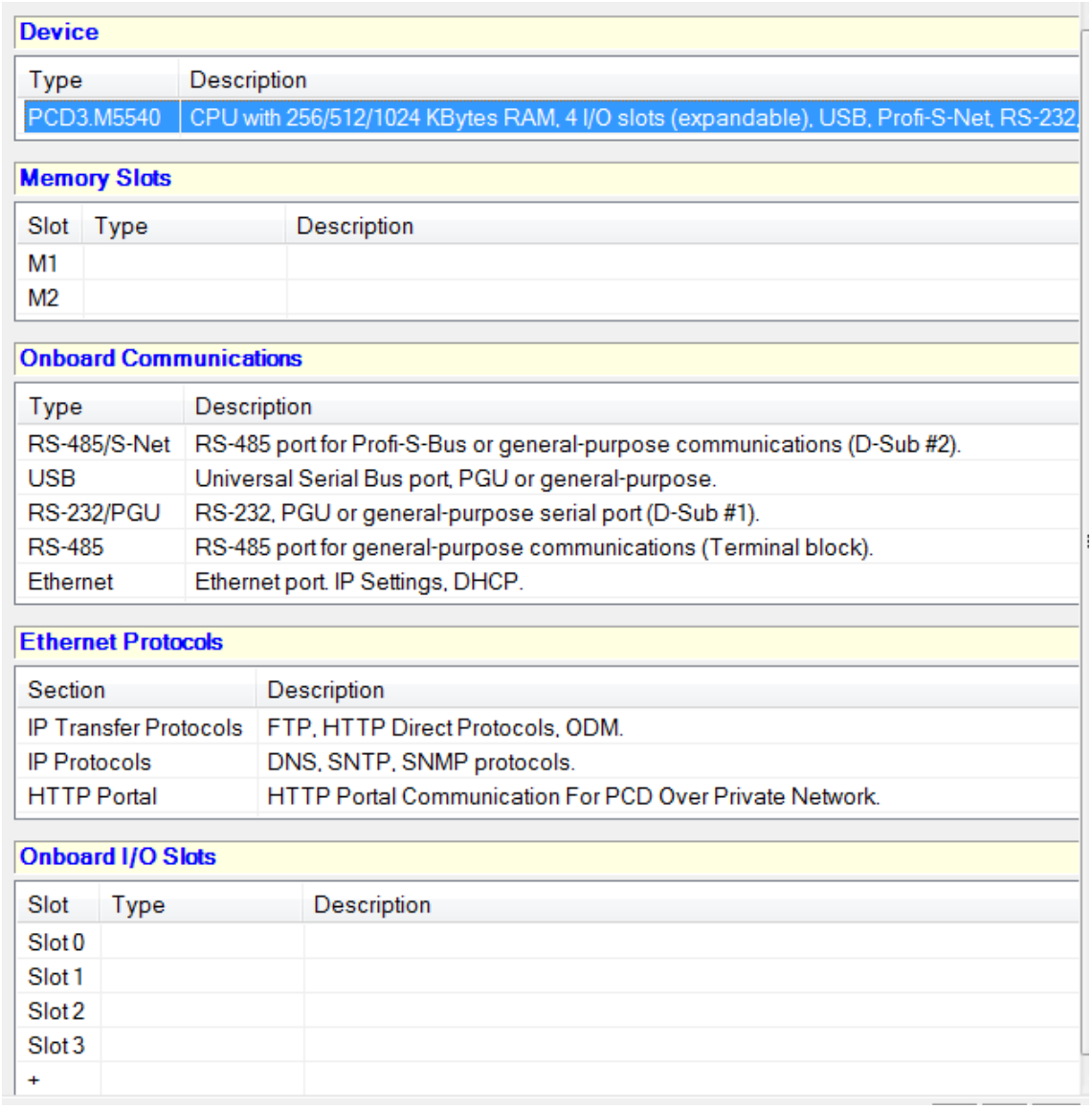
Zur Inbetriebnahme eines PCD-Steuerung ist es erforderlich, zumindest den PCD-Typ und seine Speicherkonfiguration zu definieren. Die weiteren Konfigurationen können anschließend entsprechend der erforderlichen Optionen wie z. B. Kommunikationsnetze oder Verwaltung der Ein-/Ausgänge vervollständigt werden.

Für die Benutzer der Versionen PG5 1.4 und älter ist der *Device Configurator* ein vollkommen neues Programm, das Konfigurationsparameter mit einer neuen Mensch-Maschine-Schnittstelle bietet, die auf gänzlich neuartige Weise aufgebaut ist. Obwohl nun neue Konfigurationsmöglichkeiten verfügbar sind, bleiben die Steuerung sowie deren Konfigurationsparameter die gleichen wie zuvor.

3.1 Einlesen der Steuerungparameter in den Konfigurator

Die einfachste Methode der Konfigurationserstellung besteht darin, eine Verbindung des Computers und des PCD mit einem USB-Kabel herzustellen und die bereits im Speicher des PCD vorhandene Konfiguration mit dem Menü *Online, Upload Configuration…* oder über die entsprechende Schaltfläche der *tool bar* einzulesen. Wenn der Speicher keinerlei Konfiguration enthält, übernimmt die PCD-Firmware die Übertragung der benötigten Informationen. Anschließend sind die Konfigurationen entsprechend Ihrer Anwendung zu kontrollieren.

3.2 Hauptansicht des Konfigurators



Device

Type	Description
PCD3.M5540	CPU with 256/512/1024 KBytes RAM, 4 I/O slots (expandable), USB, Profi-S-Net, RS-232,

Memory Slots

Slot	Type	Description
M1		
M2		

Onboard Communications

Type	Description
RS-485/S-Net	RS-485 port for Profi-S-Bus or general-purpose communications (D-Sub #2).
USB	Universal Serial Bus port, PGU or general-purpose.
RS-232/PGU	RS-232, PGU or general-purpose serial port (D-Sub #1).
RS-485	RS-485 port for general-purpose communications (Terminal block).
Ethernet	Ethernet port. IP Settings, DHCP.

Ethernet Protocols

Section	Description
IP Transfer Protocols	FTP, HTTP Direct Protocols, ODM.
IP Protocols	DNS, SNTP, SNMP protocols.
HTTP Portal	HTTP Portal Communication For PCD Over Private Network.

Onboard I/O Slots

Slot	Type	Description
Slot 0		
Slot 1		
Slot 2		
Slot 3		
+		

Die Hauptansicht des Konfigurators zeigt die verschiedenen Hardware-Elemente, aus denen der Automat besteht.

Device slot
Zeigt den Typ des aktuellen *Device* an. Wenn er ausgewählt ist, zeigt er die Eigenschaften des Steuerung an, wie z. B. die Größe des internen RAM/EPROM/Flash Speichers, das Passwort, den Gebrauch des S-Bus, die Verwaltung der Ein-/Ausgänge, die Optionen und den Stromverbrauch am Bus der E/A.
Um den Typ des *Device* zu verändern, muss die Maus auf das aktuelle *Device* gesetzt werden und das Kontextmenü *Change Device Type …* ausgewählt werden. Das Kontextmenü *Properties* ermöglicht die Anzeige des Fensters für die entsprechenden Eigenschaften.

Memory slots
Verfügbare Arbeitsspeicher-Steckplätze (memory slots) für Flash-Speicher-Erweiterungen entsprechend dem ausgewählten *Device*-Typ. Zur Konfiguration der Arbeitsspeicher-Steckplätze die entsprechenden Speichermodule auswählen und aus

dem Fenster *E/S Selector* an ihre Steckplätze verschieben. Zum Öffnen des Fensters *E/S Selector* das Menü *View, Selector Window* auswählen.

Onboard Communication slots
Die internen Kommunikationssteckplätze stellen die verfügbaren Kommunikationsschnittstellen dar. Je nach PCD-Typ ist die Anzahl der Kommunikationsschnittstellen unterschiedlich; sie sind nur dann konfiguriert, wenn sie vom PCD-Programm verwendet werden.

Bestimmte Steckplätze sind vordefiniert, andere frei konfigurierbar und benötigen eine Definition der entsprechenden Kommunikationsmodule durch Auswahl im Fenster *E/S Selector* und Verschieben zu einem Steckplatz.

Durch Auswahl eines Kommunikationssteckplatzes werden die Parameter der Kommunikationsschnittstelle im Fenster der Eigenschaften dargestellt und ihre Definition wird dadurch möglich.

Onboard slots
Stellt die verfügbaren Eingangs-/Ausgangs-Steckplätze des Steuerung dar. Die vorhandenen E/A-Module werden durch Auswahl im Fenster *E/S Selector* und Verschieben zu einem E/A-Steckplatz konfiguriert. Wie bei den anderen Steckplätzen zeigt die Auswahl einer E/A-Steckplätze die entsprechenden Konfigurationsparameter an.

Expansion slots
Die Erweiterungssteckplätze, die mit einem '+' gekennzeichnet sind, können ein Bus-Erweiterungsmodul aufnehmen, um zusätzliche E/A-Module hinzufügen zu können. Die Konfiguration eines Erweiterungsmoduls erfolgt durch Auswahl des entsprechenden Moduls im Fenster *E/S Selector* und Verschieben zu dem mit ‚+' gekennzeichneten Erweiterungssteckplatz.

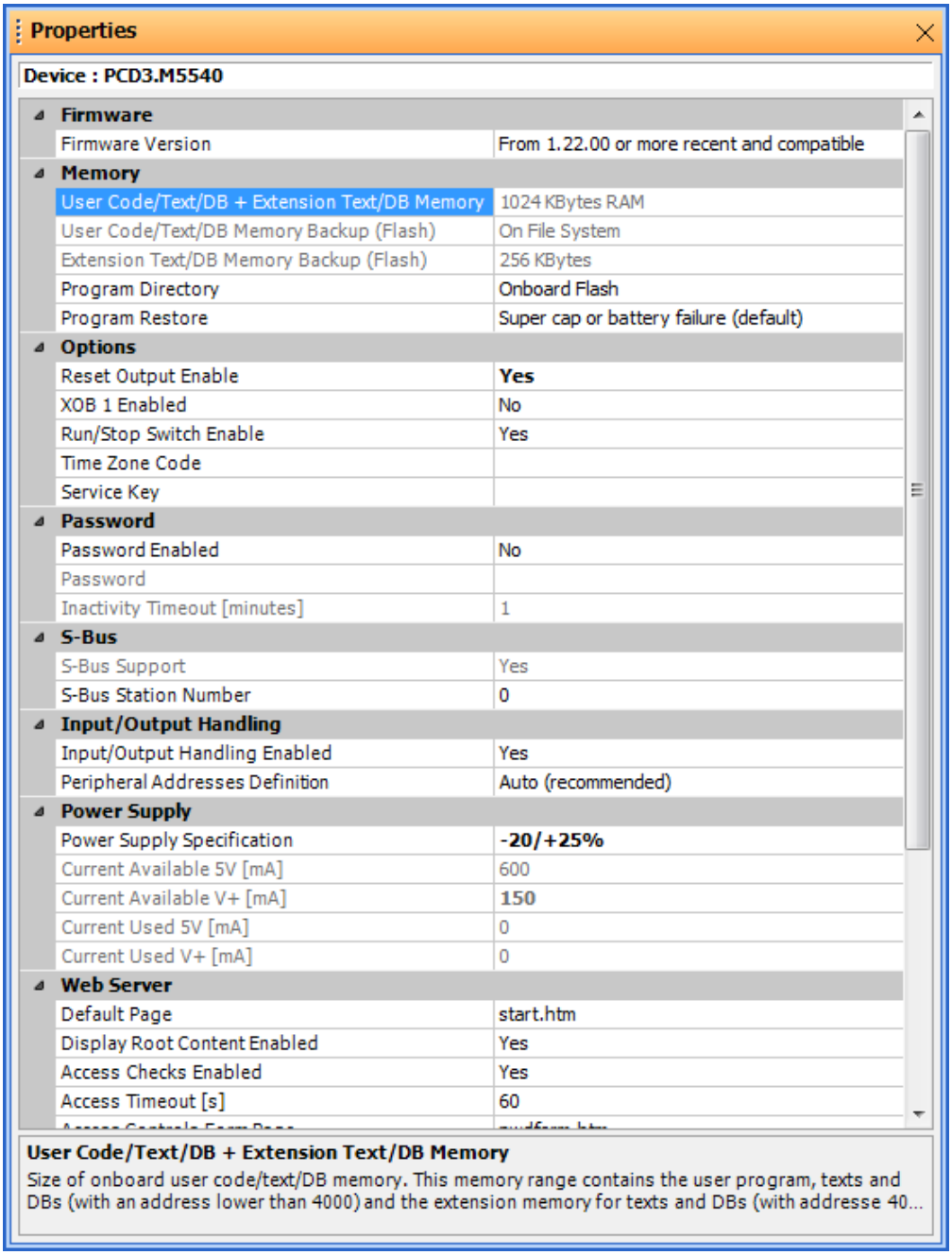
3.3 Laden der Parameter des Konfigurators in den Steuerung

Die Hardwarekonfiguration des Steuerung ist bei seiner ersten Anwendung erforderlich, aber auch bei allen Änderungen wie z. B. einer Speichererweiterung, des Einbaus eines Kommunikationsmoduls, usw.

Diese Parameter werden nicht nur im Device Configurator konfiguriert, sondern auch mit dem Menü *Download Configuration* oder über die entsprechende Schaltfläche in den Speicher des Steuerung geladen.

Achtung, das Laden des Programms über den *Project Manager* lädt die Konfigurationen nicht in den Steuerung. Die Konfigurationen müssen vom Device-Konfigurator aus geladen werden.

3.4 *Device properties*



Memory

Code/Text/Extension memory
Stellt im Steuerung verfügbaren RAM-Speicher dar, um das Programm sowie die Text- und Datenblöcke innerhalb der Adressen 0 …3999 empfangen zu können, aber auch die Texte und Datenblöcke mit Adressen über 3999.

Bei bestimmten älteren PCD kann dieser Speicher auf andere Weise organisiert werden. Der Erweiterungsspeicher ist vom *Code/Text*-Speicher getrennt. Es sind also zwei Parameter vorhanden:

Code/Text
Stellt den im Steuerung verfügbaren RAM- oder EPROM-Speicher dar, um die Programme sowie Text- und Datenblöcke innerhalb der Adressen # 0 …3999 empfangen zu können.

Bei bestimmten älteren PCD kann dieser Speicher auch vom Typ EPROM sein. Die neuen bieten jedoch an, ein Backup des Programms auf einen Flash-Speicher zu erstellen.

Extension Memory
Stellt den für die Speicheradressen > # 3999 verfügbaren RAM-Speicher dar.

User Program Memory Backup size (Flash)
Weist auf die Größe des internen Flash-Speichers des Steuerung hin; wenn dieser unzureichend ist, unterstützen bestimmte *device*-Typen eine Erweiterung des Flash-Speichers auf einem oder mehreren externen Steckplätzen. Siehe Konfigurationen unter *Memory Slots.*

Der Flash-Speicher wird zur Durchführung des Programm-Backups nach dem Laden in den Steuerung verwendet. Das Backup muss im Menü *Tools, Options* des *Project Manager* aktiviert werden oder mit der Menüauswahl *Online, Flash Backup/Restore* über den *Project Manager.*

Sollte ein Stromausfall einen Programmverlust im RAM-Speicher hervorrufen, stellt die Firmware des Steuerung das im Backup-Speicher vorhandene Programm automatisch wieder her.

Der Flash-Speicher unterstützt auch weitere Anwendungen wie die Sicherungskopie von Daten oder die Quelldateien des Projekts.

Password

Die Steuerung besitzen einen Schutzmechanismus mit Hilfe eines Passworts. Wenn sie durch ein Passwort geschützt sind, kann nur ein eingeschränktes Kommunikationsprotokoll verwendet werden. Es lässt nur den Zugang zu Registern, Timern, Zählern, Anzeigern, Eingängen, Ausgängen, Datenblöcken und der Uhr zu. Auf die übrigen Daten wie z. B. das Benutzerprogramm, die S-Bus-Konfiguration und die Verlaufstabelle kann nicht zugegriffen werden. Die Befehle Run, Stop, Step, … sind ausgeschaltet.

Das Herstellen der Kommunikation bei einem mit Passwort geschützten Steuerung zeigt vor der Erlaubnis zur Benutzung des kompletten Protokolls einen Dialog zur Eingabe des Passworts an. Die Auswahl der Schaltfläche *Cancel*' erstellt jedoch trotzdem die Kommunikation, aber mit den Einschränkungen des reduzierten Protokolls.

Wenn das Passwort vergessen wurde, muss der PCD-Speicher gelöscht und neu programmiert werden, um ihn ohne Passwort oder mit einem neuen Passwort konfigurieren zu können.

Löschen des Speichers
Für die RAM-Speicher ist dies sehr einfach. Der Automat muss lediglich ausgeschaltet und die Batterie für eine bestimmte Zeit herausgenommen werden.
Bei den EPROMs muss der Speicherchip aus dem Steuerung genommen und mit einer UV-Lampe gelöscht werden; anschließend muss es mit einem EPROM-Programmierer mit dem Benutzerprogramm und einem bekannten Passwort neu programmiert werden.
Bei den Flash-Speichern muss der Speicherchip aus dem Steuerung genommen und mit einem EPROM Programmer, der die Flash-Speicher-Chips unterstützt, gelöscht werden.

Beim Auslesen der *hardware settings* wird das Passwort des PCD nie angegeben. Es könnten sonst unbekannte Passwörter gelesen und anschließend in einen anderen PCD geladen werden, der damit unzugänglich würde, bis der Speicher herausgenommen und gelöscht wird.

S-Bus

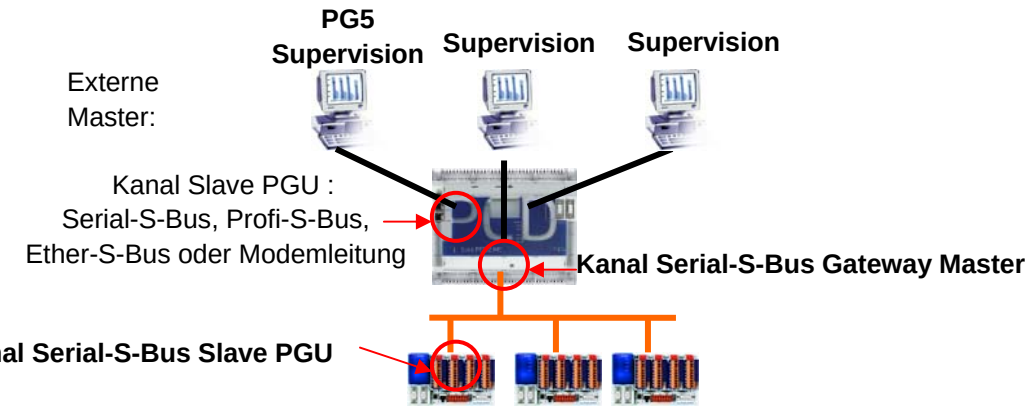
Wenn eine der unter *Onboard Communication* oder *Onboard Slots* definierten Kommunikationsschnittstellen benutzt wird, muss die S-Bus-Unterstützung aktiviert und die S-Bus-Stationsnummer definiert werden.
Die S-Bus-Stationsnummer gilt für alle Kommunikationskanäle des Steuerung.

Power Supply.

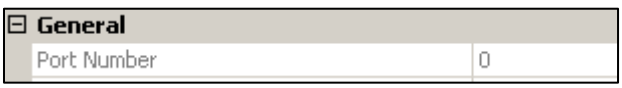
Die für die E/A-Module notwendige 5- und 24-Volt-Stromversorgung wird vom Bus des Steuerung durchgeführt. Es ist zu prüfen, ob die Konfigurationswerte der unter *Onboard Slots* des Konfigurators definierten E/A nicht höher sind als die maximal zur Verfügung stehenden Stromwerte.

3.5 Serial S-Bus Kommunikationseigenschaften

Serial S-Bus ist ein Master/Slave-Netzwerk, mit dem es möglich ist, die Steuerung im Netz an eine serielle Schnittstelle RS 485/232 anzuschließen, um Daten zwischen PCDs auszutauschen und die Supervision des Prozesses durchzuführen. Er unterstützt ebenfalls alle Funktionalitäten der Inbetriebnahme mit dem Saia PG5 Programmiertool (PGU) und eine entweder analoge oder ISDN-Modemleitung.

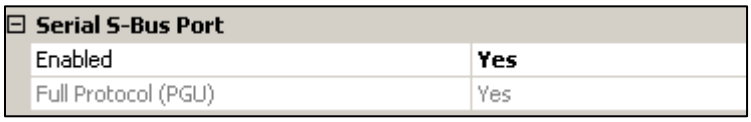


Das Fenster für die Eigenschaften ermöglicht je nach ausgewählten seriellen Kommunikationssteckplätzen die Konfiguration des Kanals Serial-S-Bus als Slave PGU für eine serielle Leitung oder als Slave PGU für eine Modem- oder Master Gateway Leitung. Die gewünschte Konfiguration aktivieren und anschließend die aktiven Parameter ergänzen.



Port Number
Nummer des Kommunikationskanals: Diese Nummer kann von den Programmanweisungen verwendet werden, um den Kanal für die Zuweisung (SASI) zu bestimmen oder um Datenaustauschtelegramme zu übertragen.

Full protocol (PGU) Serial-S-Bus (Slave PGU für eine serielle Leitung)

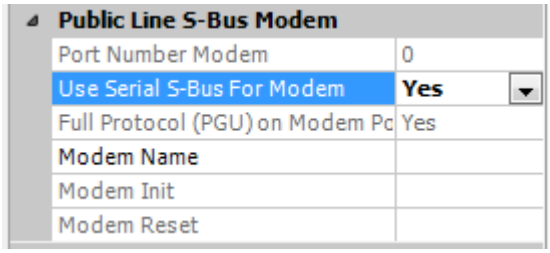


Zusätzliche Kommunikationssteckplätze ***Serial-S-Bus*** Slave können mit einem eingeschränkten Protokoll konfiguriert werden (ohne die Funktion PGU). Mit dem Device Konfigurator wird nur der PGU-Steckplatz konfiguriert, die Slave-Funktion - ohne PGU-Funktion - wird mit dem Programm Fupla/IL mit Hilfe des SASI-Befehls konfiguriert.

Anmerkung: die S-Bus-Adresse wird in den Eigenschaften des Device definiert.

Public Line S-Bus Modem (Slave PGU für eine Modemleitung)

Diese Konfiguration steht nur für die seriellen Kommunikationssteckplätze zur Verfügung, die alle notwendigen Kontrollleitungen für den Anschluss an ein Modem zur Verfügung stellen. Bereitstellung der gleichen Optionen wie bei *Full protocol (PGU) Serial-S-Bus*, jedoch mittels einer Telefonleitung und einem Analog- oder ISDN-Modem. Für eine einfachere Inbetriebnahme empfehlen wir die Verwendung der SBC Modems.



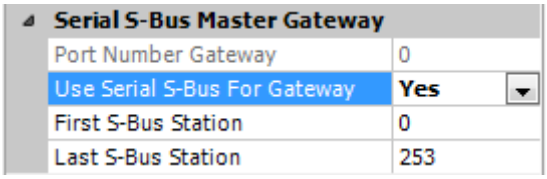
Modem Name
Ermöglicht die Auswahl des verwendeten Modemtyps. Es wird empfohlen, die Saia PCD Modems zu verwenden, denn sie sind bereits in dieser Liste aufgeführt, und die Parameter *Modem Init* und *Modem Reset* sind bereits konfiguriert und getestet.

Anmerkung: die S-Bus-Adresse wird in den Eigenschaften des Device definiert.

Serial S-Bus Master Gateway (gateway Master)

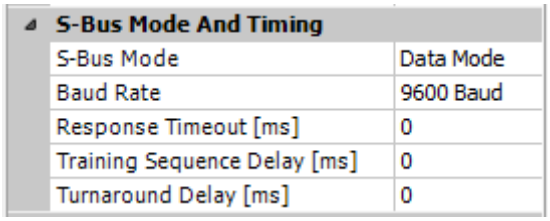
Die Funktion *Gateway* wird häufig verwendet, um zwei unterschiedliche Kommunikationsnetze zu verbinden, ein Saia PG5 Programmiertool, eine VisiPlus Kontrolle, eine Modemleitung im Serial-S-Bus-Netz anzupassen oder ein Multi-Master-Netz zu erstellen.

Alle vom externen Master empfangenen Telegramme, die nicht für die Station *gateway*-Station bestimmt sind, werden automatisch an den Kanal *gateway* Master umgeleitet.



First/Last S-Bus Station
Ermöglicht das Filtern der Telegramme, die über den *Gateway* entsprechend der Nummern der Empfängerstationen zu übertragen sind.

S-Bus Mode and Timings



Mode
Auswahl der von S-Bus angewandten Methode zur Angabe von Beginn und Ende der Mitteilungen; muss bei allen das Netzwerk bildenden Stationen identisch sein. Vorzugsweise ist der Modus Data auszuwählen. Diese Methode ist für alle Anwendungen geeignet, insbesondere für die Benutzung von Modems oder standardisierten Netzwerkeinrichtungen.

Baudrate
Kommunikationsgeschwindigkeit, muss bei allen Netzwerkstationen identisch sein.

Response Timeout
Wartezeit der Master-Station in Millisekunden für den Empfang der Antwort auf ein Telegramm. Dieser Zeitraum ist besonders wichtig, wenn der Slave nicht antwortet. Ist der Zeitraum zu lang, wird die Kommunikation stark verlangsamt. Ist er zu kurz, treffen die Antworten zu spät ein und verursachen das Verweigern der Annahme des vom Master versandten Telegramms.

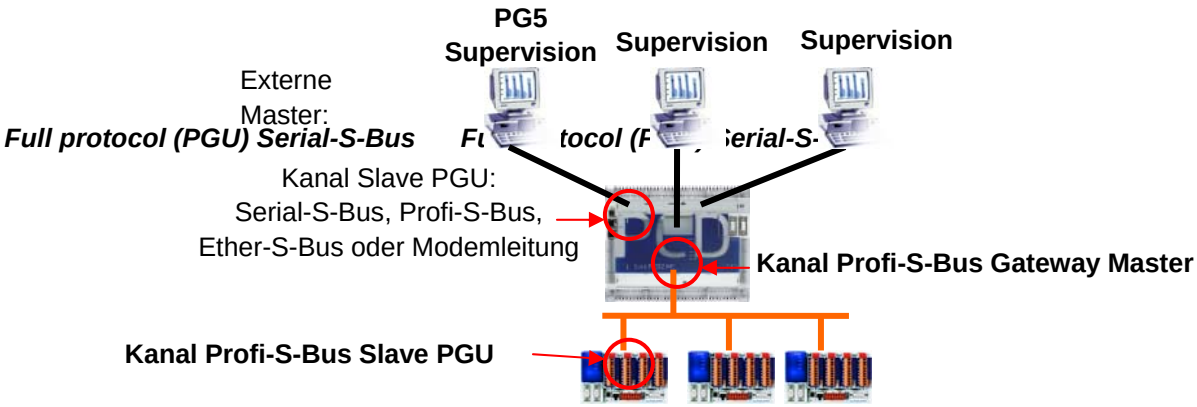
Der Standardwert für den Takt ist Null, das bedeutet, die Standardtakte werden angewandt. In der Praxis ändern wir diese nur bei der Verwendung von *Gateways* oder Modems: im Allgemeinen beschränken wir uns darauf, den *Timeout* bei den externen Mastern zu verlängern.

Training Sequence Delay (TS)
Zeitraum zwischen der Aktivierung des Signals RTS (*Request To Send*) und der Übertragung der Nachricht. Mit S-Bus ermöglicht die RTS-Aktivierung die Auswahl des Treibers RS-485 oder RS-422.

Turnaround Delay (TN)
Mindestzeit in Millisekunden zwischen dem Ende einer Antwort und der Übertragung des nächsten Telegramms. Dies gibt dem entfernten Posten die Zeit, nach dem Versand einer Nachricht die Leitung vom Sendemodus in den Empfangsmodus zu schalten. Der Zeitraum TN ist bei der Verwendung von PCD7.T100-Repeatern oder Modems für private Leitungen besonders wichtig.

3.6 Profi S-Bus Kommunikationseigenschaften

Profi-S-Bus ist ein Multi-Master Feldnetzwerk auf der Basis der Profibus FDL Standards und des SBC S-Bus-Protokolls. Es ermöglicht die Vernetzung der Steuerung zwecks Datenaustausch zwischen den PCD sowie die Kontrolle des Prozesses. Ebenfalls unterstützt werden alle Funktionen der Inbetriebnahme mit dem Programmiertool Saia PG5. (PGU)



Das dem S-Net Kommunikationssteckplatz entsprechende Fenster für die Eigenschaften ermöglicht die Konfiguration eines Profi-S-Bus Slave oder Gateway Master Kanals. Die gewünschte Konfiguration aktivieren und anschließend die aktiven Parameter ergänzen.

Profi-S-Bus (Slave)

Definiert den Profi-S-Bus-Kanal als Slave oder Slave-PGU. Diese Definition kann mit der Master-Funktion durch Hinzufügen eines SASI Master Befehls im Fupla- oder IL-Programm ergänzt werden.

Profi-S-Bus	
Channel Number Profi-S-Bus	10
Profi-S-Bus Enabled	**Yes**
Full Protocol (PGU) Profi-S-Bus	No
Slave	Yes
FDL Address	0
Use S-Net Configuration	No
S-Net File Name	
Baud Rate Profi-S-Bus	1.5 MBd
Bus Profile	S-Net

Full Protocol PGU

Slave PGU (Ja)
Unterstützt den Datenaustausch mit den Master-Stationen, Kontrollsystemen und Terminals. Unterstützt jedoch auch das Tool für die Programmierung und Inbetriebnahme PG5.

Slave (Nein)
Unterstützt nur den Datenaustausch mit den Master-Stationen, Kontrollsystemen und Terminals.

Address
Adresse der Station im Profibus FDL Netz. Um den Steuerung vollständig zu bestimmen, mit dem Telegramme ausgetauscht werden, sind zwei Adressen erforderlich, die Profibus-FDL-Adresse und die S-Bus-Adresse, die in den Eigenschaften des Device definiert ist.

Baudrate Profi-S-Bus
Kommunikationsgeschwindigkeit, muss bei allen Netzwerkstationen identisch sein.

Bus Profile
Die Übertragungstakte werden in drei Profilen zusammengefasst:

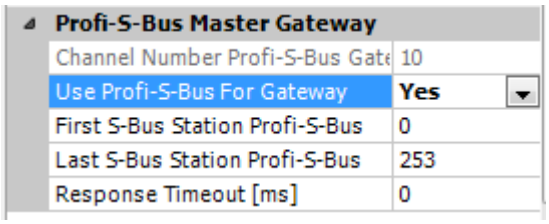
- *User-defined*: die Parameter werden vom Benutzer definiert
- *S-Net*: verwendet die am besten für das Saia PG5 S-Net geeigneten Werte
- *DP*: verwendet die am besten für das Profibus-DP-Netzwerk geeigneten Werte.

Das Profil muss bei allen Netzwerkstationen identisch sein.
Das S-Net-Profil ist bei der Verwendung von RIO PCD3.T76x im Netzwerk erforderlich.

Serial S-Bus Master Gateway. *(*gateway Master)

Die Funktion *Gateway* wird häufig verwendet, um zwei unterschiedliche Kommunikationsnetze zu verbinden und ein Saia PG5 Programmiertool, eine VisiPlus Kontrolle, eine Modemleitung im Profi-S-Bus-Netz anzupassen .

Alle von den externen Master empfangenen Telegramme, die nicht für die *gateway*-Station bestimmt sind, werden automatisch an den Kanal *gateway* Master umgeleitet.



First/Last S-Bus Station
Ermöglicht das Filtern der Telegramme, die über den *Gateway* entsprechend der Nummern der Empfängerstationen zu übertragen sind.

Response Timeout
Der Standardwert für *Timeout* ist Null, das bedeutet, dass der Standardtakt angewandt wird. In der Praxis ändern wir diesen nur bei der Verwendung von *Gateways* oder Modems: im Allgemeinen beschränken wir uns darauf, den *Timeout* bei den externen Mastern zu verlängern, aber niemals beim Gateway-Kanal.

Bus Parameters: *User defined*:

Bus Profile	User defined
Bus parameters	
Slot time	**300**
Min. Tsdr	**11**
Max. Tsdr	**150**
Quiet time	0
Setup time	**1**
Gap update factor	**10**
Highest station address	126
Max. retry limit	1

Die Bus-Eigenschaften enthalten alle Profi-S-Bus Kommunikationsparameter. Diese Werte können nur dann editiert werden, wenn das Bus-Profil *User-defined* ausgewählt wurde. Die Parameter müssen bei allen Netzwerkstationen die gleichen sein.

Alle Wartezeiten werden in Bits (Bit-Anzahl) angegeben.

Slot Time
Maximale Zeit, die ein Sender eines Daten- oder Tokentelegramms auf die Antwort des Empfängers wartet.

Min. Tsdr
Mindestwartezeit eines Slaves, bevor dieser nach einem Telegramm die Antwort zum Master sendet.

Max. Tsdr
Maximale Wartezeit eines Slaves, bevor dieser nach einem Telegramm die Antwort zum Master sendet.

Quiet Time
Wartezeit, bevor ein Sender nach dem Senden eines Datenblocks den Empfänger einschaltet.

Setup Time
Verstrichene Zeit zwischen dem Eintreffen eines Ereignisses und der Ausführung der entsprechenden Reaktion.

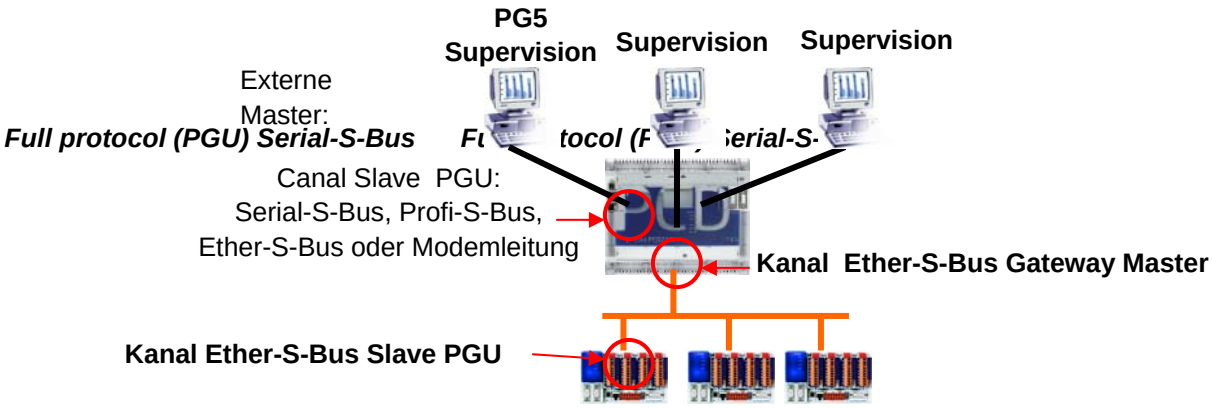
Gap Update Factor
Anzahl der Tokenumgänge zwischen zwei GAP Zyklen (Update).

Highest Station Address
Höchste vorhandene Stationsadresse (HSA) im Netzwerk.

Max. Retry Limit
Anzahl der Wiederholungen ohne Empfangsbestätigung (ACK) vor Versenden einer negativen Empfangsbestätigung (NAK).

3.7 Ether-S-Bus communication properties

Ether-S-Bus ist ein Multi-Master-Netzwerk auf der Basis der Ethernet Standards und des SBC S-Bus-Protokolls. Es ermöglicht die Vernetzung der Steuerung zwecks Datenaustausch zwischen den PCD und die Kontrolle des Prozesses. Ebenfalls unterstützt werden alle Funktionen der Inbetriebnahme mit dem Saia PG5 Programmiertool (PGU).



Das dem S-Net Kommunikationssteckplatz entsprechende Fenster für die Eigenschaften ermöglicht die Konfiguration eines Ether-S-Bus Slave oder Gateway Master Kanals. Die gewünschte Konfiguration aktivieren und anschließend die aktiven Parameter ergänzen.

Ether-S-Bus (Slave)

Definiert den Ether-S-Bus-Kanal als Slave oder Slave-PGU. Diese Definition kann mit der Master-Funktion durch Hinzufügen eines SASI Master Befehls im Fupla- oder IL-Programm ergänzt werden.

TCP/IP	
Channel Number	9
TCP/IP Enabled	Yes
Ethernet RIO Network	None
IP Address	0.0.0.0
Subnet Mask	255.255.255.0
Default Gateway	0.0.0.0
+ Access Control List	Hide

Ether-S-Bus	
Channel Number	9
Ether-S-Bus Enabled	**Yes**
IP Node	0
PGU Port	Yes
Slave	Yes
Network Groups	(Default)

IP Address
Adresse der Station im Ethernet-Netz. Um den Steuerung vollständig zu bestimmen, mit dem Telegramme ausgetauscht werden, sind zwei Adressen erforderlich, die Ethernet-Adresse und die S-Bus-Adresse, die in den Eigenschaften des Device definiert ist.

Subnet mask
Definiert den Teil der IP-Adresse, der zur Identifikation im Hostnetz gehört.
Alle dem Netzwerk entsprechenden Bits stehen auf 1, während die dem Host entsprechenden Bits auf 0 stehen. Die Netzwerkkennung ist das Ergebnis einer ET-Logikverknüpfung zwischen der IP-Adresse und der *subnet mask.* Jeder TCP/IP-Host verlangt eine *subnet mask*, selbst bei einem Netzwerksegment. Die Standard-*subnet mask* lautet daher 255.255.255.0 (Klasse C).

Default Router
Adresse des Default-Routers.

IP Node
Nummer des TCP/IP-Knotens. Der Knoten wird im Anwendungsprogramm des Steuerung verwendet, um die IP-Adresse der Slave-Station, mit der Daten ausgetauscht werden, zu referenzieren.

PGU port

Slave PGU (Yes)
Unterstützt den Datenaustausch mit den Master-Stationen, Kontrollsystemen und Terminals. Unterstützt jedoch auch das Tool für die Programmierung und Inbetriebnahme Saia PG5.

Slave (No)
Unterstützt nur den Datenaustausch mit den Master-Stationen, Kontrollsystemen und Terminals.

Network Groups
Zeigt das Dialogfenster zur Konfiguration der Steuerung als Client oder Server innerhalb der Netzwerkgruppe an.

*Serial S-Bus Master Gateway. (*gateway Master)

Die Funktion *Gateway* wird häufig verwendet, um zwei unterschiedliche Kommunikationsnetze zu verbinden und ein Saia PG5 Programmiertool, eine VisiPlus Kontrolle, eine Modemleitung im Ether-S-Bus-Netz anzupassen .

Alle von den externen Master empfangenen Telegramme, die nicht für die Station *gateway*-Station bestimmt sind, werden automatisch an den Kanal *gateway* Master umgeleitet.

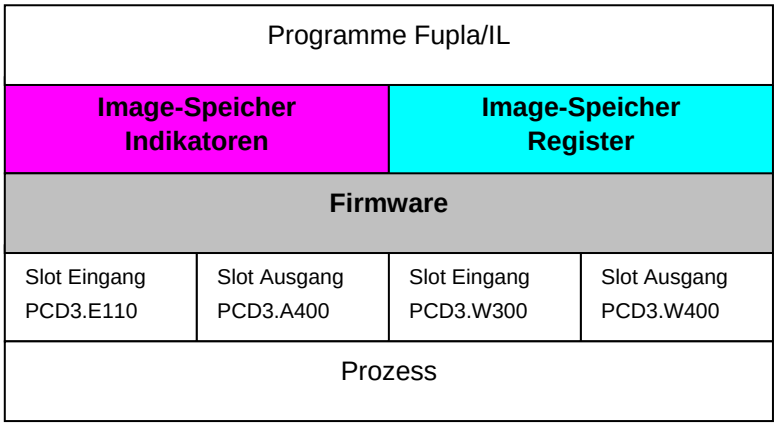
TCP/IP S-Bus Master Gateway	
Use TCP/IP For Gateway	**Yes**
First S-Bus Station	0
Last S-Bus Station	253
Response Timeout	0

First/Last S-Bus Station
Ermöglicht das Filtern der Telegramme, die über den *Gateway* entsprechend der Nummern der Empfängerstationen zu übertragen sind.

Response Timeout
Der Standardwert für *Timeout* ist Null, das bedeutet, dass der Standardtakt angewandt wird. In der Praxis ändern wir diesen nur bei der Verwendung von *Gateways* oder Modems: im Allgemeinen beschränken wir uns darauf, den *Timeout* bei den externen Mastern zu verlängern, aber niemals beim Gateway-Kanal.

3.8 *On board slots properties, Konfiguration des Image-Speichers.*

Die Verwaltung der Eingänge/Ausgänge des Steuerung kann mit Hilfe eines Image-Speichers erfolgen, der aus Indikatoren/Registern gebildet wird, die von der Firmware des Steuerung auf dem Laufenden gehalten werden. Die Fupla- oder IL-Programme greifen zum Auslesen/Schreiben der Ein-/Ausgänge nicht mehr direkt auf die Ein-/Ausgänge zu, sondern arbeiten mit dem Image-Speicher.



Der Image-Speicher kann mit den PCD1.Mxxx0, PCD2.Mxxx0 und PCD3 Steuerungen konfiguriert werden.
Es ist jedoch immer noch möglich, direkt auf die Eingänge/Ausgänge zuzugreifen, ohne den Image-Speicher zu konfigurieren. Dies ermöglicht es, die Kompatibilität mit den älteren Projekten der vorhergehenden PG5-Versionen und den Steuerung, die dies noch nicht unterstützen, zu garantieren wie PCD1.M1xx, PCD2.M1xx, PCD2.M480.

Device properties, notwendige Konfigurationen.

Wenn die Firmware des Steuerung den Image-Speicher unterstützt, kann die Verwaltung der E/A mit den nachstehenden Parametern aktiviert oder deaktiviert werden.

Input/Output Handling Enabled
Yes: alle E/A-Parameter des Moduls sind zur Unterstützung der Konfiguration des Image-Speichers verfügbar.

No: alle in den *Onboard slots* definierten Parameter haben keinen Einfluss auf das Benutzerprogramm.

Peripheral Address Definition
Automatische oder manuelle Definition der Adressenbereiche für den Image-Speicher jedes Moduls.

Onboard slots, Konfigurationen der E/A-Module.

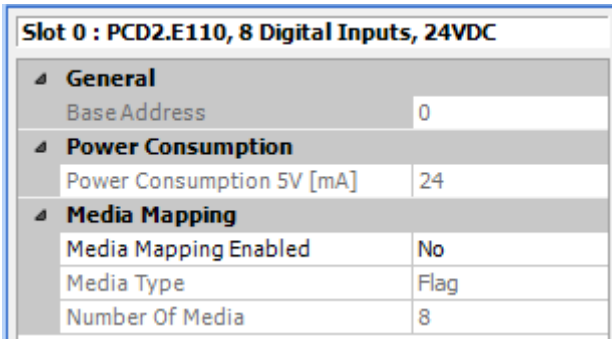
Ermöglicht die Konfiguration der im Steuerung vorhandenen E/A-Module durch Auswählen im Fenster *E/S Selector* und Verschieben zu einem E/A-Steckplatz.

Die Auswahl einer der E/A-Steckplätze zeigt die entsprechenden Konfigurationsparameter in einem Fenster für die Eigenschaften an.

Die Konfiguration der E/A-Steckplätze ist nicht notwendig, wenn der Image-Speicher von der PCD nicht unterstützt wird oder in den Eigenschaften des *Device* nicht aktiviert ist.

Eigenschaften der binären E/A.

Ermöglicht die Definition der für die Einrichtung des Image-Speichers notwendigen Konfigurationen.
Wenn die Auswahl *Media mapping* nicht angezeigt wird, bedeutet dies, dass die unter *Device* ausgewählte Firmware des Steuerung diese Funktion nicht unterstützt.



Base Address
Basisadresse des Moduls: 0, 16,32,…

Enabled Media Maping
Zyklische Aktualisierung des Image-Speichers (Register oder Indikatoren) mit den an den Steckplätzen des Steuerung vorhandenen Eingangs- und Ausgangswerten.

Media Type
Typ des verwendeten Mediums zum Speichern der Eingangs- oder Ausgangswerte. Bei analogen E/A handelt es sich immer um den Typ Register. Bei binären E/A handelt es sich standardmäßig um den Typ Indikator, kann aber auch Register sein.

Number of Media
Anzahl der notwendigen Medien für das Speichern der Werte. Für ein digitales Ausgangsmodul PCD3.E110 sind zum Beispiel 8 Indikatoren notwendig.

Symbol definition
Auswählen dieses Parameters, um die Schaltfläche anzuzeigen, die die Anzeige des Editors für die Symbole und die Änderung von Namen, Kommentaren und Adressen der den E/A entsprechenden Symbole ermöglicht.

Media Mapping

Slots / Symbols	Type	Address	Comments	Scope	Tags
PCD2.M5540, CPU with 1 MBytes RAM, 8 I/O slots (expandable), 3 communication slots, USB, Profi-S-Net, RS-232, Ethernet.					
Slot 0, PCD2.E110, 8 digital inputs, 15..30VDC, 8ms, current draw 12mA at 5V.					
S.IO.Slot0.DigitalInput	F [8]			Public	S_IO
IO.Slot0.DigitalInput0	F	S.IO.Slot0.DigitalInput + 0	Digital input 0	Public	S_IO
IO.Slot0.DigitalInput1	F	S.IO.Slot0.DigitalInput + 1	Digital input 1	Public	S_IO
IO.Slot0.DigitalInput2	F	S.IO.Slot0.DigitalInput + 2	Digital input 2	Public	S_IO
IO.Slot0.DigitalInput3	F	S.IO.Slot0.DigitalInput + 3	Digital input 3	Public	S_IO
IO.Slot0.DigitalInput4	F	S.IO.Slot0.DigitalInput + 4	Digital input 4	Public	S_IO
IO.Slot0.DigitalInput5	F	S.IO.Slot0.DigitalInput + 5	Digital input 5	Public	S_IO
IO.Slot0.DigitalInput6	F	S.IO.Slot0.DigitalInput + 6	Digital input 6	Public	S_IO
IO.Slot0.DigitalInput7	F	S.IO.Slot0.DigitalInput + 7	Digital input 7	Public	S_IO

Nach dem Build des Programms stehen diese Symbole zur Erstellung der Fupla- und IL-Programme zur Verfügung. Sie sind unter den Symbolen "All Publics" zu finden, und zwar unter *S.IO.Slot0, …*

Symbol Editor

Find: Type a substring to find

Symbol Name	Type	Address/Value	Comment	Tags	Scope
All Publics	ROOT				
IO	GROUP				
Slot0	GROUP				
DIGITAL_INPUT...	CONST	0	Address of digita...		Public
DigitalInput0	F	0	Digital input 0	S_IO	Public
DigitalInput1	F	1	Digital input 1	S_IO	Public
DigitalInput2	F	2	Digital input 2	S_IO	Public
DigitalInput3	F	3	Digital input 3	S_IO	Public

All Publics | System | Untitled3.fup

Eigenschaften der analogen E/A.

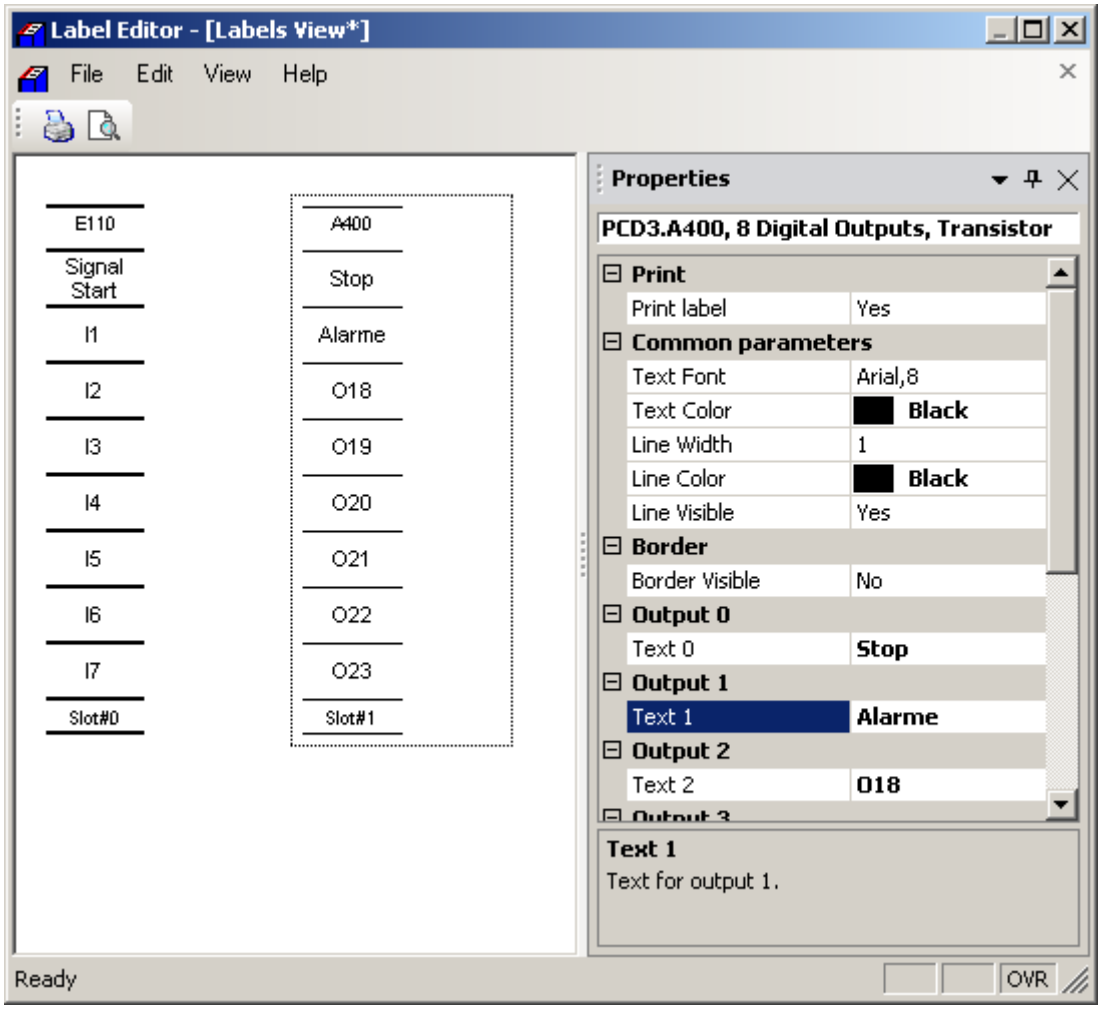
Werden wie die binären E/A-Module konfiguriert; es handelt sich um die gleichen Parameter wie zur Definition des *Media mapping* des analogen Moduls bei den Registern.

Analogue Output 1	
Output 1 Range	0..10V in mV or % resolution
Minimal Value Output 1	0
Maximal Value Output 1	10000
Reset Value Output 1	0

Ein neuer Abschnitt bietet jedoch die für die Konfiguration der E/A-Größen eines jeden Kanals notwendigen Parameter an.

3.9 Editieren der Etiketten für die E/A-Module.

Mit dem Menü *Tools, Label* ist es möglich, das nachstehende Fenster anzuzeigen, um die Etiketten vorzubereiten, die auf den PCD3 E/A-Modulen und PCD1.Mxxx0/PCD2.Mxxx0 Decke anzubringen sind. Etikett auswählen und im Fenster der Eigenschaften editieren.



Print Label
Aktiviert/deaktiviert den Druck des ausgewählten Etiketts

Common parameters
Parameter zur Definition des Seitenlayouts: Zeichensatz, Textfarbe, Größe und Farben der auf den Etiketten gedruckten Linien.

Border Visible
Zeichnet die Außenlinie des Etiketts an, um das Ausschneiden zu erleichtern.
Anmerkung: vorgeschnittene Blätter zum Ausdrucken der Etiketten stehen zur Verfügung.
Die Etiketten können doppelt so groß erscheinen, da es vorgesehen ist, diese längs zusammenzufalten. Sie erhalten dadurch mehr Stabilität in ihrer Halterung.

3.10 Erweiterung des Device Configurator mit neuen *devices* und E/A-Modulen

Für den Fall, dass neue Module oder Steuerung zur Verfügung stehen, ist die Installation einer neuen Version des Device Configurator oder PG5 zu ihrer Unterstützung nicht mehr notwendig. Es reicht aus, die Datei XML zu installieren, die die neue Hardware in einem Unterverzeichnis der PG5-Installation beschreibt, und anschließend die Software neu zu starten.

C:\Program Files\SBC\PG5_20**DeviceTemplates**

Inhaltsverzeichnis

4. PCD - Ressourcen

4.1. Einleitung

Dieses Kapitel enthält einen Überblick über alle verfügbaren Datentypen (Eingänge, Ausgänge, Indikatoren, Register, Zähler, Timer usw.), sowie deren Einsatzmöglichkeiten und Adressbereiche.

4.2. Hardware-Ressourcen

Jedes Programm besteht aus Funktionen, die dem Anwender erlauben verschiedene Arten von Ressourcen zu lesen, zu schreiben oder anderweitig zu manipulieren. Diese, dem Anwender zur Verfügung stehenden Ressourcen, werden Hardware-Ressourcen genannt.

4.2.1. Digitale Ein- und Ausgänge

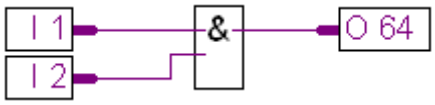
Ein- und Ausgänge haben logische Signale, welche zur PCD gehen (Eingänge) bzw. von der PCD kommen (Ausgänge). Eingänge zeigen den Status von Endschaltern, Druckknöpfen, Annäherungsschalter, Sensoren usw. Ausgänge können Ventile, Lampen, Motoren usw. ansteuern.
Ausgänge können geschrieben (setzen/rücksetzen) und gelesen werden. Eingänge können nur gelesen werden.
Eine PCD kann durch Einfügen von Ein- und/oder Ausgangsmodulen in die vorgesehenen Steckplätze erweitert werden. Die Startadresse eines Steckplatzes ist durch dessen Position definiert.

Das folgende Beispiel schaltet den Ausgang O 64 ein, wenn die Eingänge I 1 und I 2 = H sind. Eine andere Darstellung ist die boolsche Gleichung: O 64 = I 1 * I 2.

Instruction List (IL)
Programm:

```
COB   0
      0
STH   I 1
ANH   I 2
OUT   O 64
ECOB
```

Funktionsplan (**FUPLA**)
Programm:



FBox: *UND 2-10 Eingänge, (Binäre Funktionen*)

4.2.2. Hardware Uhr

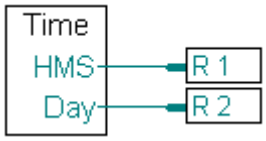
Die PCD haben eine eigene Uhr (RTC = Real Time Clock). Es kann mit einer Instruktion die Zeit und das Datum in je ein PCD-Register geladen werden.

Das Beispiel zeigt, wie die Uhr im Programm gelesen werden kann

Programm in IL (Instruction List):

```
COB    0
       0
RTIME  R 1
ECOB
```

Funktionsplan (FUPLA) Programm:



FBox: *Uhr lesen, (Zeitfunktionen)*

Dieses Programm liest die Uhr inkl. Datum und überträgt die Werte in 2 aufeinander folgende Register: R1 und R2.

R 1 = 093510	09 Uhr 35 Minuten und 10 Sekunden
R 2 = 073030210	Woche 07, Tag.-Nr. 3 (Mittwoch), 10. Februar (20)03

4.2.3. Interrupt-Eingänge

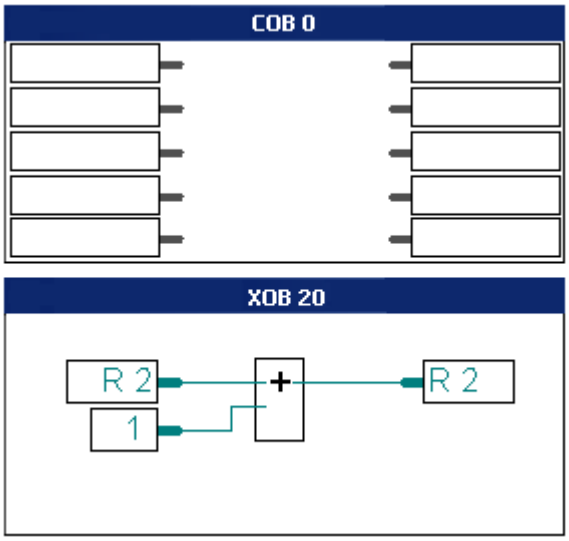
Einige PCD haben zwei Interrupt Eingänge, INB1 und INB2. Bei jeder ansteigenden Flanke an einem der beiden Eingänge wird der normale Programmablauf unterbrochen und die PCD führt einen speziellen Programm-Block aus. Die Programmblöcke sind XOB20 für INB1 und XOB25 für INB2. Diese Eingänge sind bis zu einer Frequenz bis zu 1000 mal pro Sekunde ausgelegt.

Das folgende Beispiel zeigt, wie die Impulse vom Eingang INB1 gezählt werden.

Instruction List (IL)
Programm:

```
COB    0      ; Programm
       0      ;

ECOB


XOB  20       ; INB1
INC    R 2    ; R2 wird
              ; inkrementiert
EXOB
```

Funktionsplan (FUPLA)
Programm:



Weitere Informationen siehe Saia PCD-Handbücher

Die Begrenzung durch den Eingangsfilter (schützt einen digitalen Eingang gegen Störspitzen und Prellen eines mechanischen Kontaktes) verhindert das Zählen eines Impulses mit einer Frequenz grösser 50Hz. Deshalb stellen Interrupt-Eingänge eine interessante Alternative für diese Art der Anwendung dar. Sie stellen einen Ersatz für den Einsatz von PCD2/3.H1xx Zählmodulen dar, die eine Zählfrequenz - abhängig vom Modultyp - von 10 bis 160kHz haben.

4.3. Interne (Software-) Ressourcen Einleitung

4.3.1 Flags

1 Informations-Bit (H/L)

F

NV (non volatile – nicht flüchtig)	
0	8191

Ein Flag speichert die Information von 1 Bit. Die Adressen der Flags sind F0 - F8191. Standardmässig sind die Flags nicht-flüchtig d.h., beim Aus und Wiedereinschalten der PCD bleibt der logische Zustand erhalten (vorausgesetzt, die Batterie ist i.O.). Flags können softwaremässig als flüchtig (volatile) definiert werden. Dies wird im Folgenden erläutert.

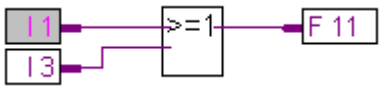
Das folgende Beispiel bringt das Flag F 11 in den logischen H-Zustand wenn die Eingänge 1 oder 3 = H sind. Die boolsche Gleichung lautet $F\ 11 = I\ 1 + I\ 3$

Verwendung von Flags in Programmen

Instruction List (IL)
Programm:

```
COB   0
      0
STH   I 1
ORH   I 3
OUT   F 11
ECOB
```

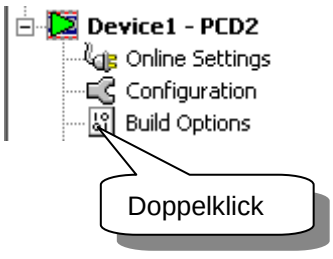
Funktionsplan (FUPLA)
Programm:

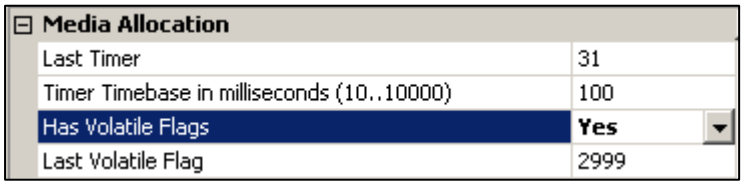


FBox: *Oder 2-10 Eingänge, (Binäre Funktionen)*

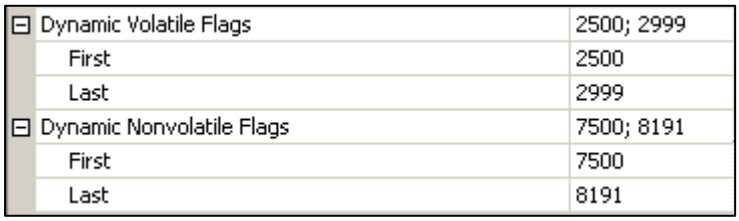
Standardmässig sind alle Flags nicht-flüchtig. Sollen die Flags flüchtig definiert werden, ist dies in den 'Software Settings' einzustellen (siehe Beispiel).

Setup flags





Media Allocation	
Last Timer	31
Timer Timebase in milliseconds (10..10000)	100
Has Volatile Flags	Yes
Last Volatile Flag	2999



Dynamic Volatile Flags	2500; 2999
First	2500
Last	2999
Dynamic Nonvolatile Flags	7500; 8191
First	7500
Last	8191

4.3.2 Register

32 Bit Wert
Integer : -2 147 483 648 to +2 147 483 647
Floating-Point : -9.22337E+18 to +9.22337E+18

R **NV (non volatile – nicht flüchtig)**
0 4095
16383 (PCD2.M480,PCD3.M)

Ein Register kann einen Integer-Wert (ganzzahlig) oder einen Floating-Point-Wert (Fliesspunkt) enthalten. Register werden für arithmetische Operationen oder im Zusammenhang mit analogen Werten, z.B. für Regelungsaufgaben verwendet. Es sind 4096 Register vorhanden. Die Register sind immer nicht-flüchtig.

Im FUPLA haben die Verbindungslinien, je nach Datenformat, verschiedene Farben: blau für Integer-Werte, gelb für Floating-Point-Werte. Blaue und gelbe Linien können nicht miteinander verbunden werden. Die Datenformate müssen für eine mathematische Operation ins gleiche Format gewandelt werden.

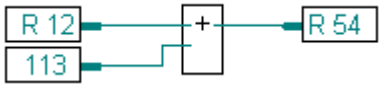
Verwendung von Registern im Programm

Das folgende Programm addiert die Zahl (Konstante) 113 zum Inhalt des Registers 12 und legt das Resultat in Register 54: R 54 = R 12 + 113

Instruction-List (IL):
Programm:

```
COB   0
      0
ADD   R 12
      K 113
      R 54
```

Funktionsplan (FUPLA)
Programm:

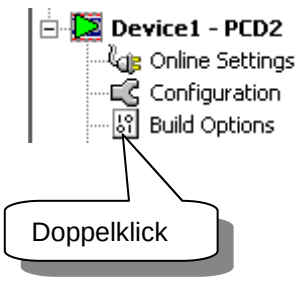


FBox: *Addition, (Ganzzahl Arithmetik*)

Setup Registers

Bei der der Festlegung von dynamischen Resourcen kann auf die feste Adressierung von Registern verzichtet werden. Dynamische Ressourcen werden verwendet, indem ein symbolischer Name für eine Ressource ohne feste Adresse definiert wird. Diese Einstellungen müssen nicht geändert werden, bis ein grosses Programm mit vielen Registern geschrieben wird.

Falls die Meldung *"Auto allocation overflow for type: R"* erscheint, ist die Anzahl dynamischer Register zu erhöhen.



Dynamic Registers	2000; 4095
First	2000
Last	4095

4.3.3 Konstanten

32 Bit Wert
Integer: -2 147 483 648 bis +2 147 483 647
Floating point: -9.22337E+18 bis +9.22337E+18

Konstanten sind feste Werte, die im Programm nicht verändert werden. Konstanten werden in Register geschrieben.

Beispiel: feste Koeffizienten wie z.B. π (PI) = 3.1415.

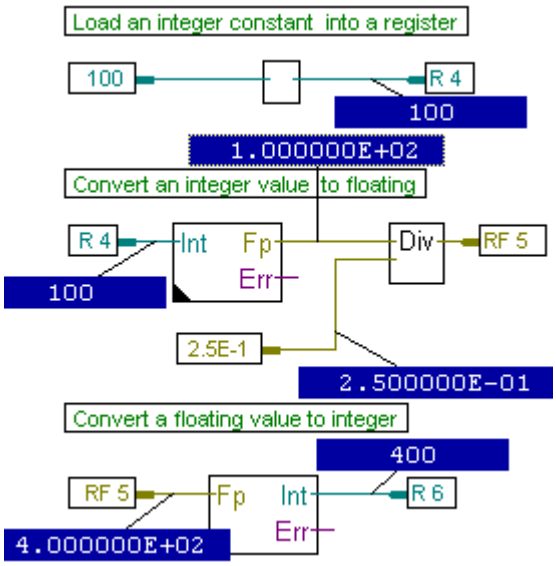
Das folgende Beispiel lädt die Konstante 100 in das Register 4. Danach soll R 4 durch 0.25 geteilt werden. Da R 4 einen Ganzzahlwert enthält, und dieser durch einen Floting-Point Wert (FP = Fliesspunkt) geteilt werden soll, muss R 4 ins FP-Format gewandelt werden. Es wird R 4 z.B. ins R 35 kopiert und ins FP-Format gewandelt. Danach wird R 35 durch 0.25 dividiert. Das Resultat der Division gelangt in R 5, dessen Inhalt in R 6 kopiert und wieder ins Ganzzahl-Format zurückgewandelt wird.

Verwendung von Konstanten im Programm

Instruction List (IL)Program:
Programm:

```
COB   0       ;Zykl. Organisations-
      0       ; Block
LD    R 4     ;Lade 100 ins R4
      100

COPY  R 4     ; Konvertiere Wert
      R 35    ; von Interger
IFP   R 35    ; zu Floating Point
      0
LD    R 36    ; Lade 0,25 ins R 36.
      2.5e-1
FDIV  R 35     ;Dividiere den Wert
      R 36    ; durch 0.25
      R 5     ; und lege Ergebnis
              ; in R5 ab.

COPY  R 5     ; Konvertiere
      R 6     ; das Ergebnis
FPI   R 6     ; zurück in Integer
      0
ECOB
```

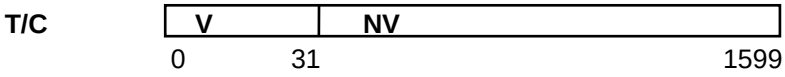
Funktionsplan (FUPLA)
Programm:



FBox: - *Move, (Ganzzahl Arithmetik)*
- *Ganzzahl zu Fliesspunkt, (Wandler)*
- *Division, (Fliesspunkt Arithmetik)*
- *Fliesspunkt zu Ganzzahl, (Wandler)*

4.3.4 **Timer und Counter** (Zeiten & Zähler)

31 Bit Wert (0 … 2 147 483 648)



Timer (T) und Counter (C) können Werte zwische 0 und 2 147 483 648 (31 Bits) annehmen. Die Voreinstellung für Timer sind die Adressen 0 bis 31 und Counter die Adressen von 32 bis 1599 für Counter. Diese Aufteilung kann den Bedürfnissen entsprechend angepasst werden. Die Voreinstellung für die Zeitbasis der Timer ist 100ms, d.h. dass eine Zeit alle 100 ms um einen Zählwert dekrementiert wird. Die Zeitbasis kann in den 'Software Settings' angepasst werden. Timer sind flüchtig, Counter nicht-flüchtig.

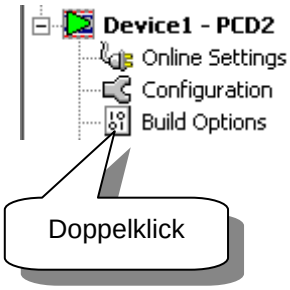
Timer und Counter können nur positive Werte annehmen. Der Wert kann durch das Laden eines neuen Wertes mit dem Befehl LD geändert werden. Timer werden nur dekrementiert, Counter können mit den Befehlen INC/DEC auf- und abwärts zählen (INC: ↑, DEC: ↓).

Timer und Counter können als binäre Elemente auf den logischen Zustand (H, L) abgefragt und verknüpft werden.

Enthält ein Timer oder ein Counter einen Wert grösser Null, ist der logische Zustand = H, ist der Wert gleich Null, ist der logische Zustand = L.

Die Aufteilung Timer – Counter kann in den 'Software Settings' gewählt werden. Hier kann auch die Zeitbasis für die Zeiten eingestellt werden.

Setup timers/counters



Last Timer	31
Timer Timebase in milliseconds (10..10000)	100

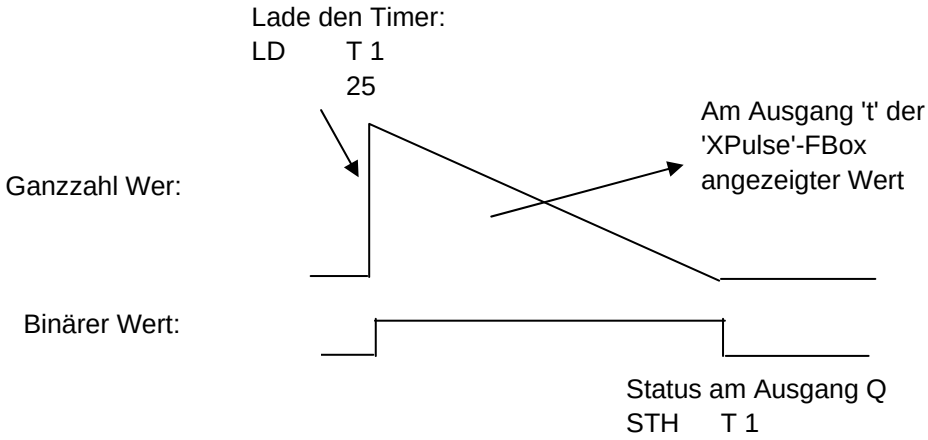
⊟ Dynamic Timers	5; 31
First	5
Last	31
⊟ Dynamic Counters	1400; 1599
First	1400
Last	1599

Technische Zusatzinformation

Je mehr Timer definiert werden und je kleiner die Zeitbasis gewählt wird, je grösser wird die Belastung der CPU. Dies ist bei der Aufteilung und bei der Wahl der Zeitbasis zu berücksichtigen.
Beispiel: 100 Timer benötigen 2% der CPU-Leistung.

Beispiel: Timer

Am Eingang 4 wechselt der Signalzustand von L nach H. Mit der ansteigenden Flanke dieses Signals soll am Ausgang 65 ein Signal von 2.5 sek. Länge ausgegeben werden.

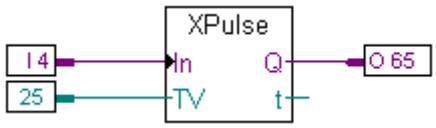


Lösung:

Instruction List (IL) Programm:

```
COB   0      ; Organisationsblock 0
      0      ; Time out Zeit
STH   I 4    ;Wenn am Eingang 4
DYN   F  12  ; steigende Flanke,
LD    T 1    ; lade den Timer1
      25     ; mit 2,5 Sekunden
STH   T 1    ; Übertrage den Timerstatus
OUT   O 65   ; zum Ausgang O65
ECOB
```

Funktionsplan (FUPLA) Programm:

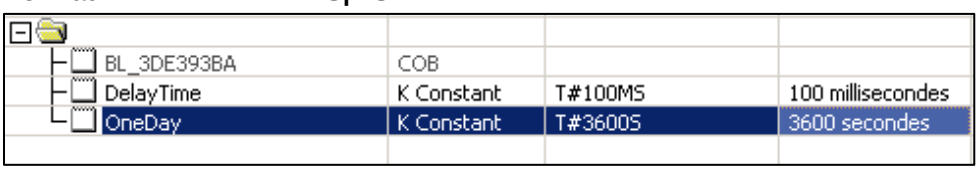


FBox: *Impulsfunktion (Zeitfunktionen)*

Technische Zusatzinformation

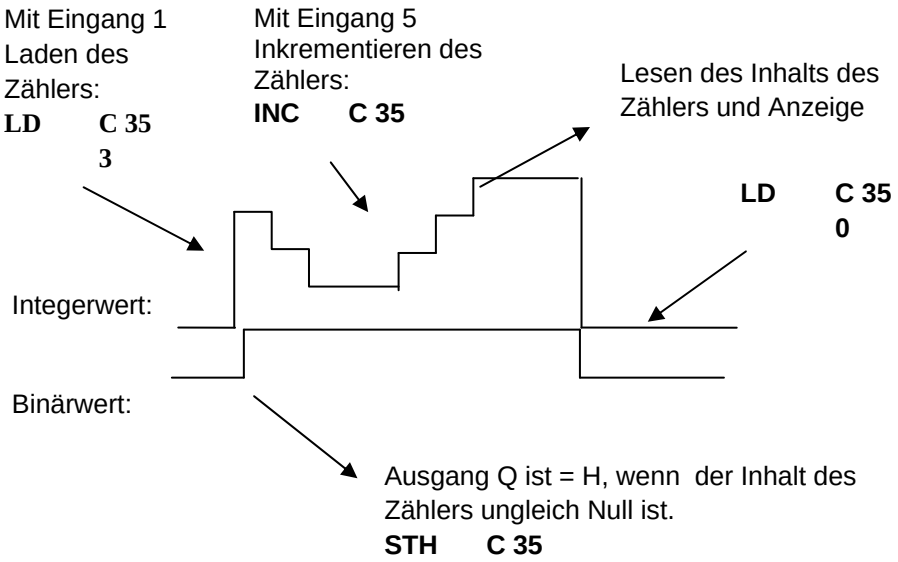
Timer werden in den SAIA PCD in Intervallen dekrementiert, die in den "Build Options" unter 'Timer' - 'Timebase' definiert wurden, normalerweise 100 ms. Wird die Zeitbasis verändert müssen alle im Programm verwendeten Zeitwerte angepasst werden. Um dies zu umgehen, kann der 'Time'-Datentyp zur Eingabe des Zeitwertes verwendet werden. Wird dieser 'Time'-Datentyp verwendet, berechnet der Linker den aktuellen Zeitwert gemäss der gewählten Zeitbasis.

Format: **T#nnnS|MS**



Beispiel: Counter

Ein Zähler soll so programmiert werden, dass dieser bei jeder Betätigung des Eingangs 5 einen Schritt aufwärts und bei jeder Betätigung des Eingangs 6 einen Schritt abwärts zählt. Der Zähler soll mit dem Eingang 1 auf 3 und mit dem Eingang 2 auf Null gesetzt werden.
Dies ist nur ein erstes Beispiel. Falls alles etwas verwirrlich wirkt, macht dies nichts, es wird alles viel klarer, wenn Sie selbst eine praktische Aufgabe programmieren werden.

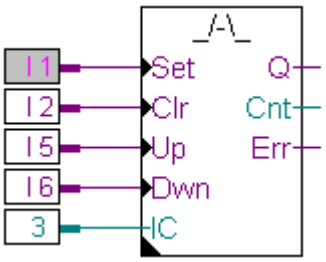


Lösung:

Instruction List (IL)Program:
Programm:

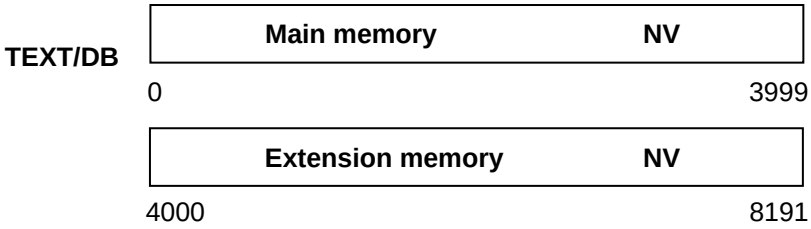
```
COB       0    ; Zyklischer Organisations-
            0  ; block
STH     I 1    ; Wenn Eingang 1 = H
LD      C 35   ; dann lade den Counter 35
            3  ;  mit 3
STH     I 2    ; Wenn Eingang 2 = H
LD      C 35   ; dann lade den Counter
            0  ; mit Null
STH     I 5    ; Wenn Eingang 5 = H
DYN     F 13   ; ansteigende Flanke
INC     C 35   ; dann Inhalt Counter 35 '+1'
STH     I 6    ; Wenn Eingang 6 = H
DYN     F 14   ; ansteigende Flanke
DEC     C 35   ; dann Inhalt Counter 35 '-1'
DSP     C 35   ; Display Inhalt Counter 35
ECOB
```

Funktionsplan (FUPLA)
Programm:



FBox: *Up/Down mit Vorwahl+Nullst. (Zähler)*

4.3.5 Texte und Datenblöcke



Texte und Datenblöcke (Data-Blocks, DB) sind nicht-flüchtig. Texte (Zeichenfolgen) sind Meldungen zu einer Anzeige (Display), Texte zum senden zu einem Pager, Initialstrings für die Kommunikation oder für Modems usw.
DBs werden für den Datenaustausch mit Registern, Datenspeicherung, Tabellen usw. verwendet.

Technische Zusatzinformation
Wo werden Texte/Datenblöcke (DB) gespeichert

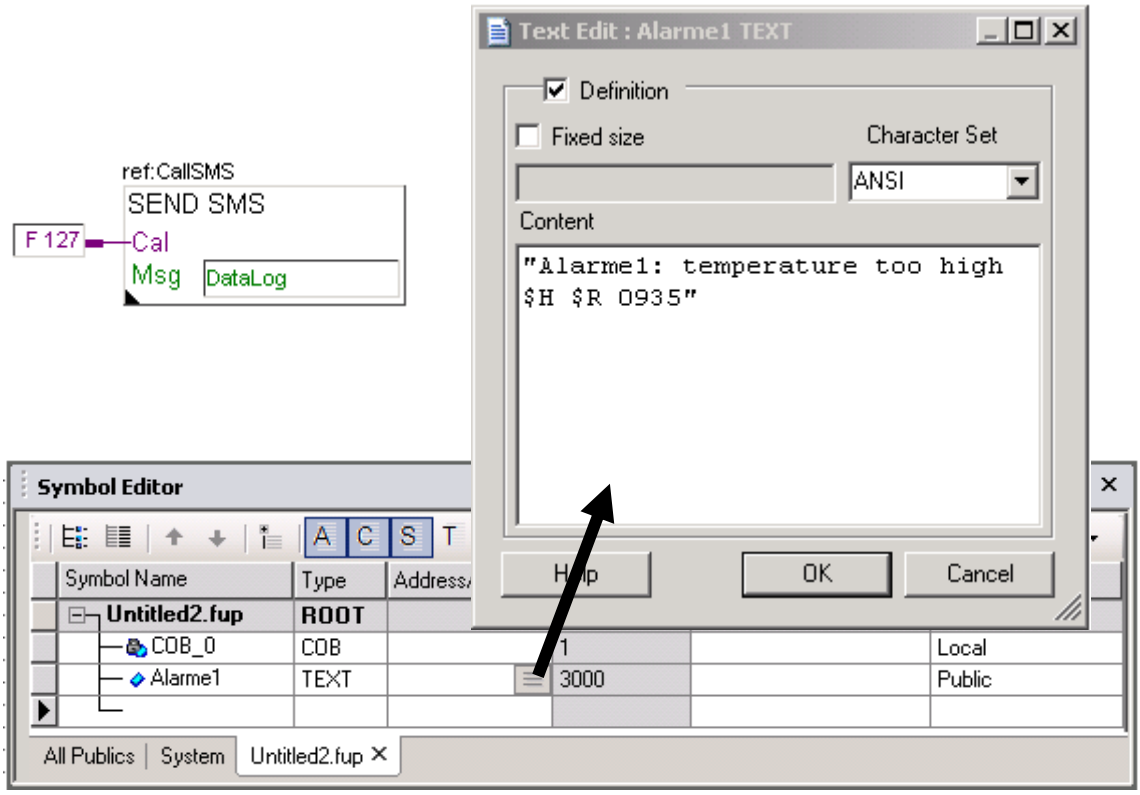
Register, Flags, Timer und Zähler werden vom System verwaltet und in einem kleinen RAM, getrennt vom Hauptspeicher, abgelegt.
DBs und Texte werden dagegen im Hauptspeicher, zusammen mit dem Anwenderprogramm, abgelegt. Wird der Hauptspeicher als Flash-EPROM oder als normales EPROM ausgelegt, ist zu beachten, dass Daten im RUN-Modus nur gelesen, nicht aber geschrieben werden können, womit die Daten in den DBs, z.B. für den Datenaustausch, nicht verändert werden können. In den meisten Fällen stört dies nicht. Sollen jedoch Daten auch geschrieben werden, so sind die DBs in ein sog. Extension-Memory (im Adressbereich grösser 4000) zu legen. Dieses Extension-Memory ist immer als RAM ausgeführt und kann somit gelesen und beschrieben werden.

Beispiel: Schreibweise von Texten und Datenblöcken (DB)

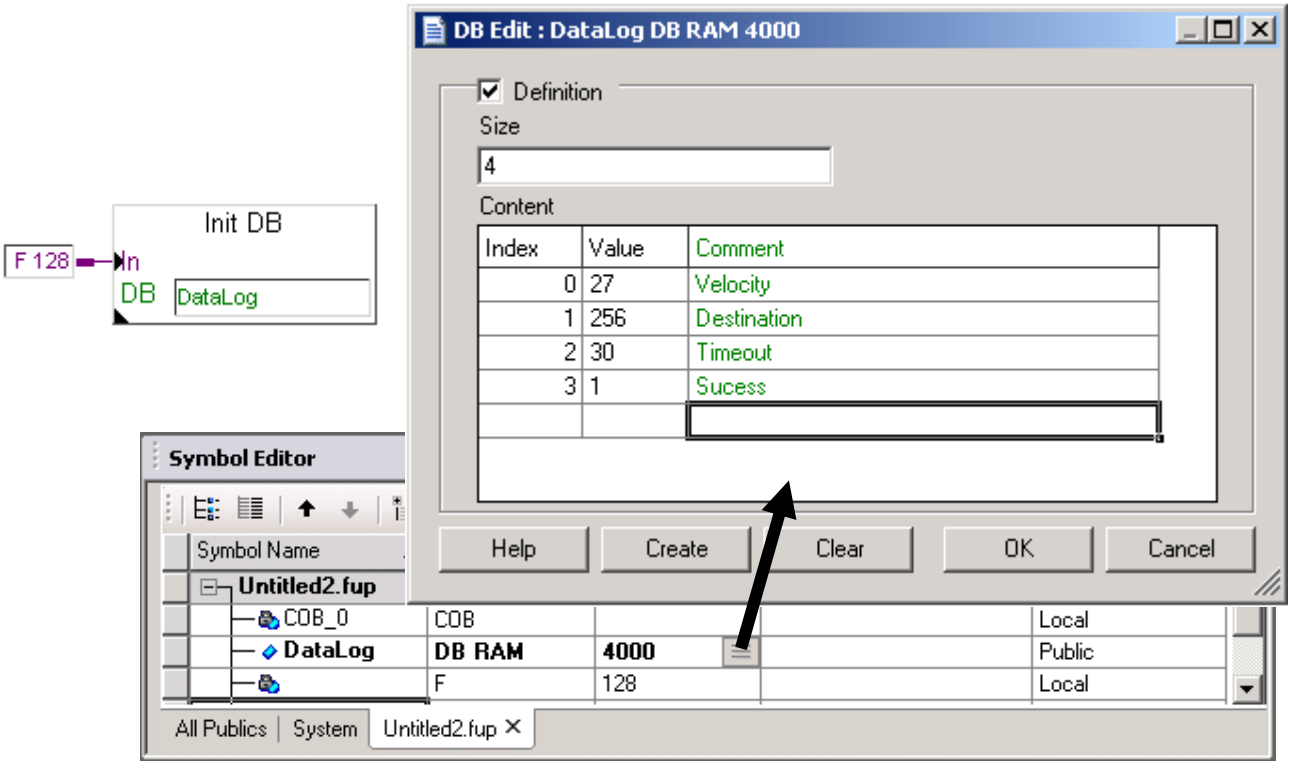
Das folgende Beispiel zeigt die Deklaration von Texten und DBs im
Instruction-List Programm:

```
TEXT 10 "Bonjour!"          ; Text Nummer 10 enthält den Klartext: Bonjour!

TEXT 11 [7]"Hello"          ; Text Nummer 11 ist 7 Zeichen lang, dabei sind
                            ;  die letzen 5 'Hello', die ersten 2 sind Leerschläge

DB 12 [] 45,46,78,999,0     ; DB Nummer 12 ([] = Länge wird durch Anzahl Werte
                            ;  bestimmt): Werte: 45, 46, 78, 999, 0

DB 13 [10]                  ; DB Nummer 13 mit 10 reservierten Werten
                            ;  zu Beginn sind alle Werte = 0

DB 14 [4] 2,3               ; DB 14 ist 4 Werte lang. Die ersten beiden Werte
                            ;  sind 2 und 3, die nächsten 2 sind = 0
```

Text Defintion mit dem Symbol Editor



Data Blok Defintion mit dem Symbol Editor



4.3.6. Zusammenfassung

Beschreibung	Media	Operand	Binär	Numerisch	Flüchtig
Inputs (Eingänge)	I	1) 0…8191	0,1		
Outputs (Ausgänge)	O	1) 0…8191	0,1		
Flags	F	0…16383	0,1		Nein
Registers	R	0…16383		-2 147 483 648...+2 147 483 647 -9.22337E+18...+9.22337E+18	Nein
Konstante	K			-2 147 483 648 à +2 147 483 647 -9.22337E+18 à +9.22337E+18	
Timers (Zeiten)	T	2) 0…31	0,1	0 … 2 147 483 648	Ja
Counters (Zähler)	C	2) 32…1599	0,1	0 … 2 147 483 648	Nein
Text	X	3) 0…3999 4) 4000 …		String von max. 3072 Zeichen	Nein
Data blocks (Datenblöcke)	DB	3) 0…3999 4) 4000 …		Max. 382 Elemente (Langsamer Zugriff) Max.16‘383 Elemente (Schneller Zugriff)	Nein

1) Abhängig von der PCD und deren Ein/Ausgangskonfiguration
2) Voreinstellung, konfigurierbar vom Menü "Device -> Software Settings"
3) Im gleichen Bereich gespeichert wie Anwenderprogramme (RAM / EPROM / FLASH)
4) Im erweiterten Speicherbereich (extension memory) gesichert (RAM)

Inhaltsverzeichnis

5 Symbol-Editor

5.1 Einleitung

Dieses Kapitel enthält einen Überblick über den Symbol-Editor und die Verwendung von Symbolen in Programmen.

5.2 Überblick

Ein Symbol ist ein Name, der die Adresse eines Ein- oder Ausgangs, einer Flag, eines Registers usw. angibt. Es wird empfohlen, Symbol-Namen statt der direkten Adresse einer Flag oder eines Registers bei der Erstellung eines Programms zu verwenden. Enthält das Programm aussagekräftige Namen, kann es einfacher gelesen werden. Sie können beispielsweise den Namen *Oil_Pump* für *O 32* vergeben und dann *Oil_Pump* als Adresse in Ihrem Programm verwenden.

Neben der Verwendung von Symbolen können Adressen oder Datentypen im *Symbol editor* korrigiert werden. Es ist nicht notwendig, alle Symbole im Programm auszubessern, es muss nur einmal im *Symbol editor* korrigiert werden. Die Änderungen an der Symbol-Definition werden automatisch an allen Stellen übernommen, an denen Symbole im Programm verwendet wurden. So besteht kein Risiko, dass Symbole an manchen Programmstellen nicht korrigiert werden und dadurch ein schwer zu findender Fehler entsteht.

Bevor Sie mit dem Schreiben eines Programms beginnen, können alle für das Programm benötigte Symbole festgelegt und im Tool *Symbol Editor* aufgelistet werden. Nach der Definition im Symbol-Editor können die Symbole von der PG5 verarbeitet werden. Dieses Tool ist bei der Suche nach Elementen in Programmdateien, bei der Berichterstellung zu Programmierfehlern oder bei der Fehlersuche sehr hilfreich.

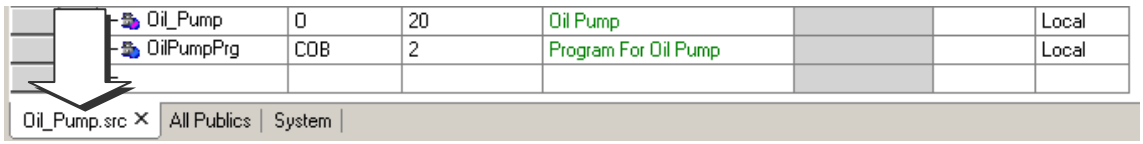
5.2.1 Bestandteile des Symbol-Editors

Öffnen des Symbol-Editors:

Die Symbol-Definitionen werden in den Programmdateien (IL/Fupla usw.) gespeichert. Wird eine Programmdatei in einem Editor geöffnet, öffnet sich ebenfalls der Symbol-Editor mit der entsprechenden Symbol-Liste.

Beispiel:

Mit dem Öffnen der Programmdatei *Oil_Pump.src* öffnet sich automatisch der Symbol-Editor mit demselben Namen.



Show Hide Symbols Editor

Das Fenster mit dem *Symbol editor* kann mit der Schaltfläche *Show/Hide Symbol Editor* oder über den Befehl *View/Symbol Editor* ein- und ausgeblendet werden.

Bestandteile des Symbol-Editors:

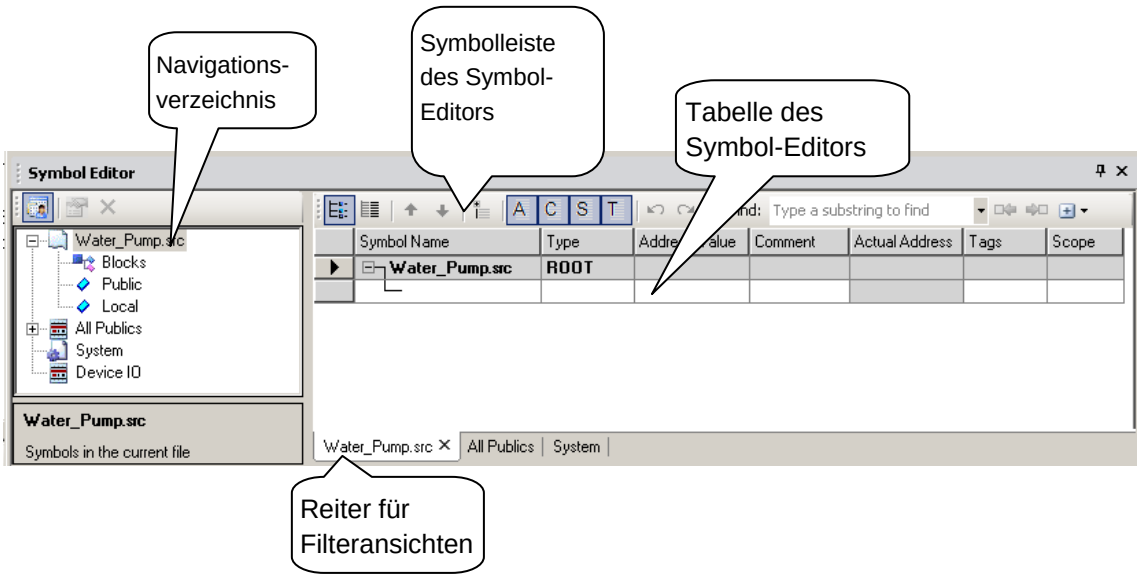


Tabelle im Symbol-Editor

Hierbei handelt es sich um einen Editor in Tabellenform, in dem Symbole definiert werden können. Am Ende befindet sich immer eine leere Reihe, in der neue Symbole festgelegt werden können. Über das Kontextmenü, das sich über einen Rechtsklick öffnen lässt, können verschiedene Befehle ausgeführt werden.

Symbolleiste im Symbol-Editor

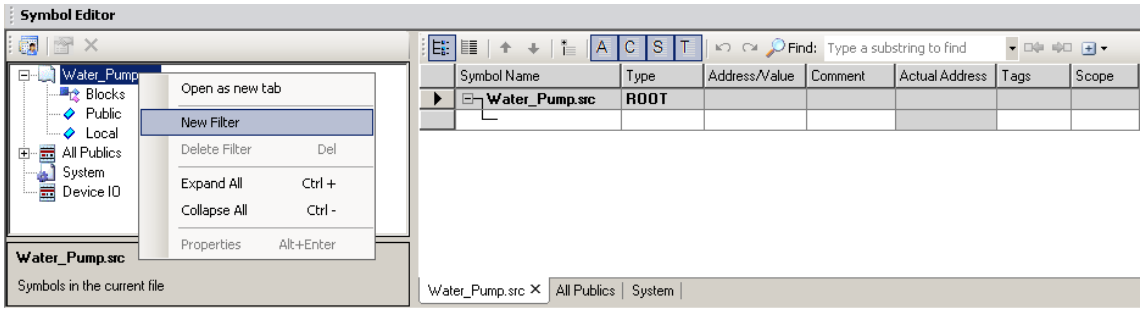
Über eine Reihe von Schaltflächen können Befehle im Symbol-Editor ausgeführt werden. Dazu gehören show/hide Navigator tree (Navigationsverzeichnis ein- und ausblenden), List view/Group view (Listen- und Gruppenansicht), Move up/down symbols (Symbole nach oben und unten scrollen), Expand/collapse (Erweitern/verkürzen), Show/hide columns in grid (Tabellenspalten ein- und ausblenden), Undo/Redo (Rückgängig/wiederherstellen), Find symbols (Symbole suchen) usw.

Reiter für Filteransichten

Jede Filteransicht wird in einem kleinen Fenster im Symbol-Editor geöffnet. Diese Fenster werden in den Reitern angezeigt. Sie können zwischen den einzelnen Reiteransichten wechseln.

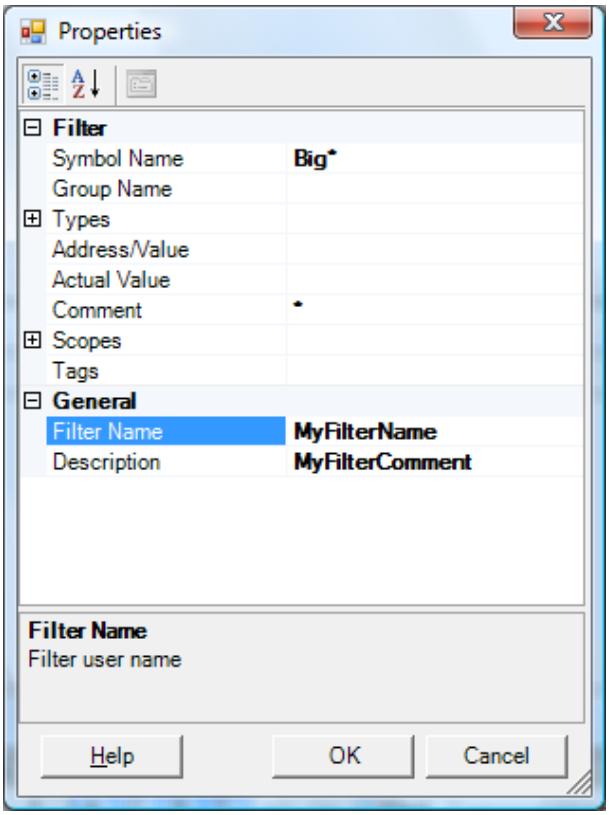
Navigationsverzeichnis

Dieses Verzeichnis weist die Struktur der vorgegebenen Filter auf, um die Symbole in der Tabelle gemäss dem ausgewählten Filter anzuzeigen. Das Navigationsverzeichnis kann über die Symbolleiste des Symbol-Editors ein- und ausgeblendet werden. Sie können ebenfalls individuelle Filterdefinitionen in das Navigationsverzeichnis aufnehmen. Die Filterregeln können in den Filtereigenschaften festgelegt werden. Neue Filter können Sie mit einem Rechtsklick über das Kontextmenü hinzufügen.



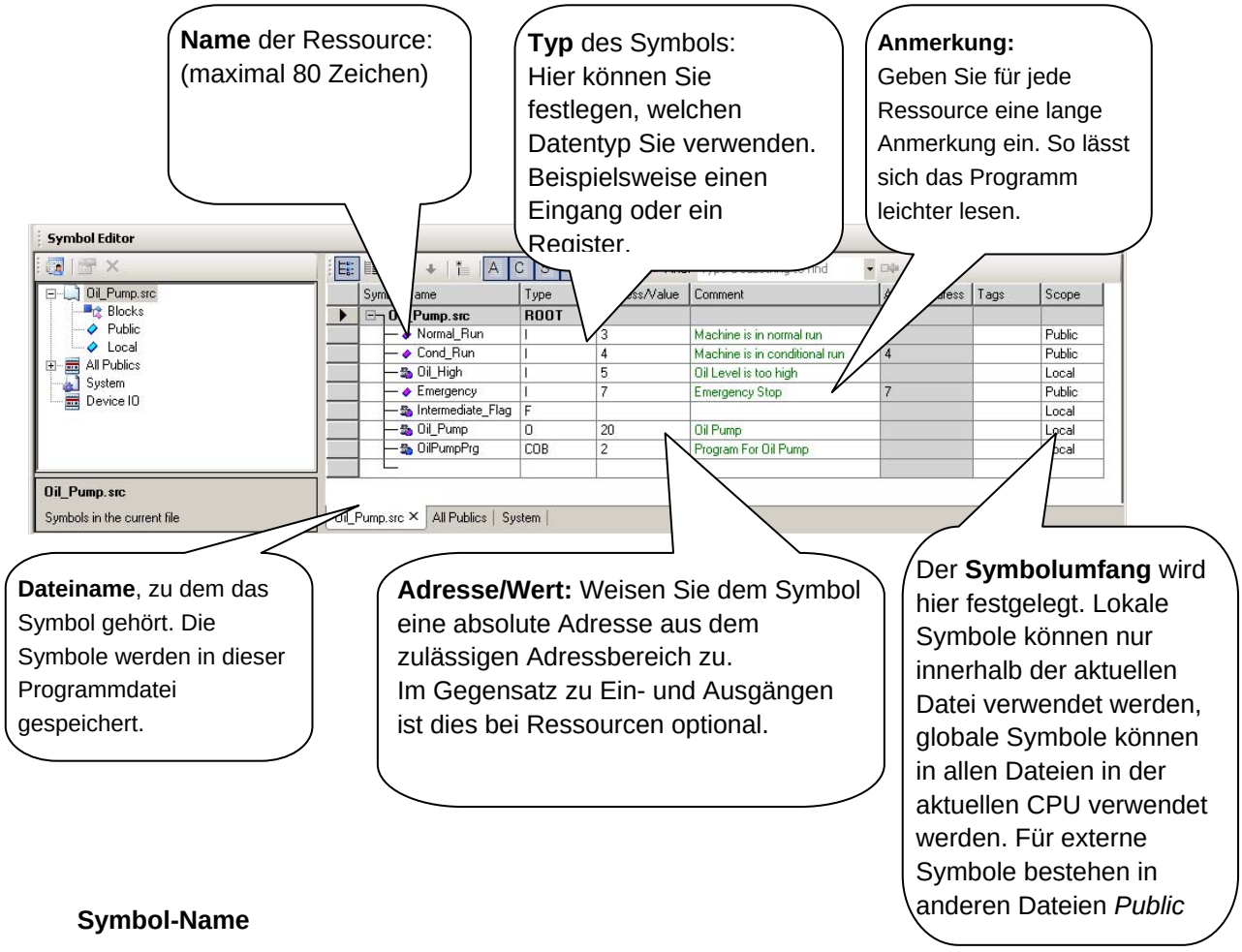
Filtereigenschaften

Die Symbole lassen sich mit Hilfe der vorhandenen Einträge in der Symbol-Tabelle filtern. Individuell erstellte Symbol-Filter können mit ihren Namen gespeichert werden. Um die Eigenschaften eines bestehenden Filters zu ändern, führen Sie einen Rechtsklick aus und wählen aus dem Kontextmenü die Option *Properties*.



5.2.2 Elemente eines Symbols

Symbole werden in Programmdateien festgelegt. Die Symbol-Tabelle der jeweiligen Programmdatei kann über den Reiter *Program file* eingesehen werden. Die Symbole können in diesem Reiter bearbeitet werden.



Symbol-Name

Das erste Zeichen ist immer ein Buchstabe. Danach folgen weitere Buchstaben, Ziffern oder ein Unterstrich. Bei Symbolen muss nicht auf Gross- und Kleinschreibung geachtet werden, sofern sie keine betonten Zeichen enthalten. „MotorOn“ ist identisch mit „MOTORON“, aber „GRÜN“ ist nicht identisch mit „grün“.

Reservierte Wörter können nicht als Symbol-Namen verwendet werden. Eine Liste mit reservierten Wörtern finden Sie am Ende dieses Kapitels.

Typ

Hier können Operandentypen wie Eingang (I), Ausgang (O), Register (R), Zähler (C), Timer (T), Text (X), DB usw. festgelegt werden.

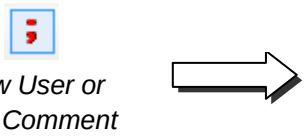
Adresse

Hier können absolute Adressen für Operanden festgelegt werden. Dazu gehören interne Daten wie Register, Flags usw. Wird keine Adresse eingegeben, wird diese automatisch vom System während des Build-Prozesses zugewiesen. Dabei handelt es sich um eine sog. Auto-Zuweisung. Für Ein- und Ausgänge müssen Adressen zwingend angegeben werden.

Anmerkung

Anmerkungen beziehen sich auf ein Symbol. Im IL-Editor können sie statt der Benutzeranmerkungen, die für alle Zeilen eines Programmcodes angegeben werden können, eingesehen werden.

Verwenden Sie dazu die Schaltfläche *View User* oder *Auto Comment.*



View User or Auto Comment

```
STH      Flag1      ;User Comments
```

```
STH      Flag1      ;Comments From Symbol Editor
```

Tatsächlicher Wert

Hierbei handelt es sich um eine Spalte, für die nur Lesezugriff besteht. Sie enthält die dynamischen Adressen, die das System dem Operanden automatisch nach dem Build-Prozess zuweist.

Tags

Tags können verwendet werden, um Public Symbole mit gemeinsamen Funktionen wie Network, HMI oder Supervision Symbols usw. auszuzeichnen.

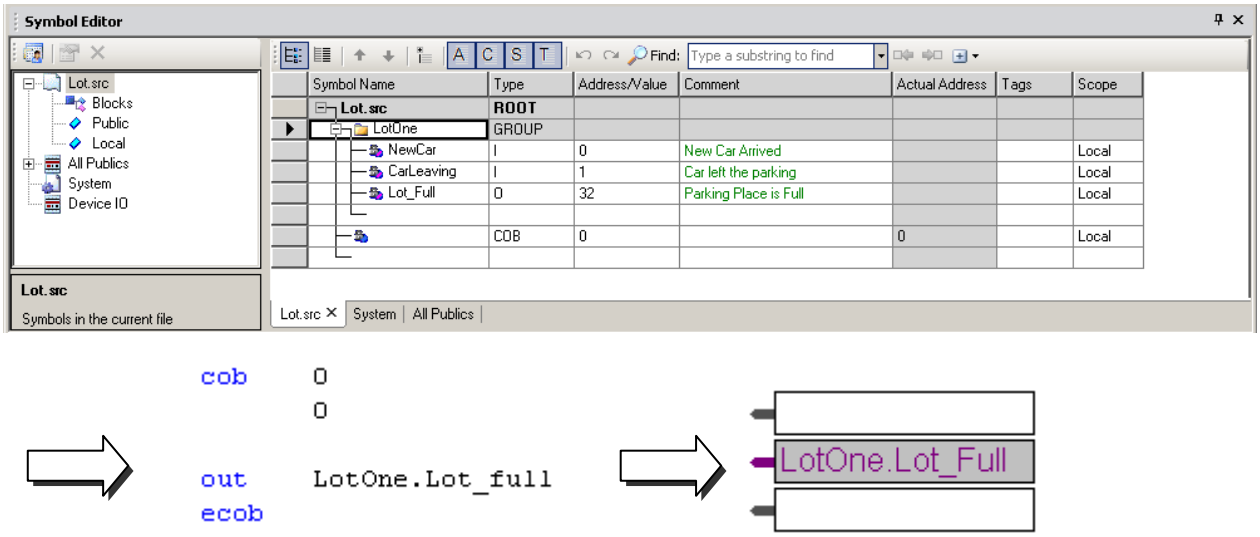
Umfang

Die Umfangskategorien *Local*, *Public* oder *External* können aus der Dropdown-Liste ausgewählt werden.

5.2.3 Gruppierung von Symbolen

Symbole können falls nötig gruppiert werden. So lässt sich das Programm leichter lesen. Mit der rechten Maustaste oder der Tastenkombination *STRG+G* können Sie im Symbol-Editor eine neue Gruppe hinzufügen und dann die benötigten Symbole definieren oder in den Ordner ziehen. Gruppennamen können verschachtelt sein (bis zu 10 Ebenen), z.B. *Group1.Group2.Group3.Symbol*.

Beispiel: Die Gruppe mit dem Namen *LotOne* enthält mehrere Symbole:

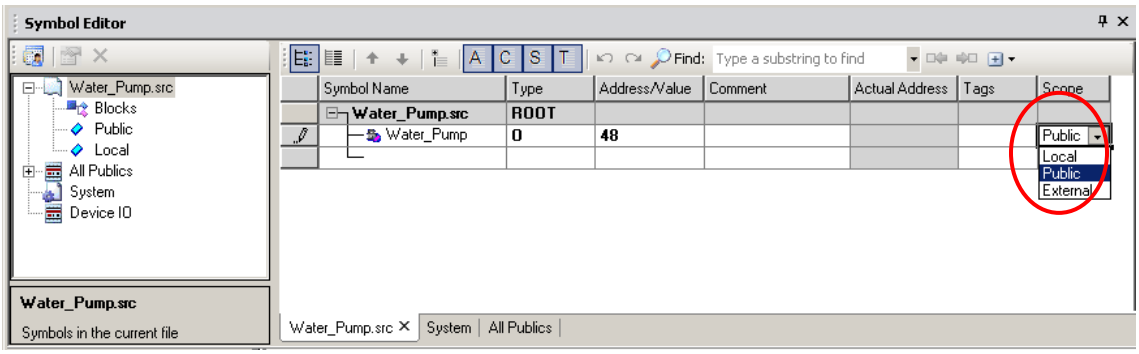


Im Programm steht der Gruppenname *LotOne* vor dem Symbol-Namen *Lot_full*, die Namen sind durch einen Punkt voneinander getrennt.

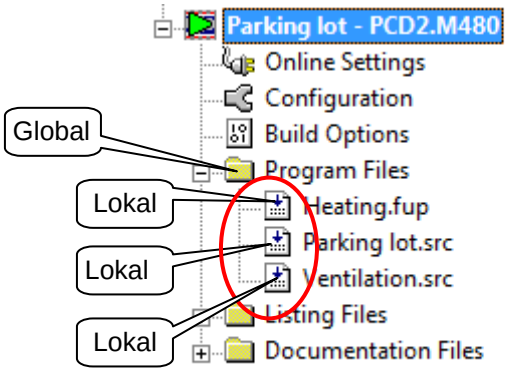
In sehr umfangreichen Programmen treten womöglich viele Symbole mit ähnlichen Namen auf, insbesondere wenn sich derselbe oder ähnlicher Code mehrmals wiederholt. Statt vielen Symbolen mit leicht unterschiedlichen Namen, z.B. *Motor_A_Symbol1*, *Motor_B_Symbol1* usw., können Sie denselben Symbol-Namen verwenden, ihn jedoch in einer unterschiedlichen Gruppe anlegen, z.B. *MotorA.Symbol1*, *MotorB.Symbol1*.

5.2.4 Symbol-Umfang

Der Symbol-Umfang kann aus der Spalte *Scope* ausgewählt werden. Es stehen drei Symbol-Kategorien zur Verfügung: *Local, Public* und *External*.



Beispiel: Gerät mit drei Programmdateien



Lokale Symbole

Lokale Symbole können nur in der Datei verwendet werden, in der sie definiert wurden. Beispielsweise kann ein Symbol mit dem Umfang „lokal“, das in *Parking_Lot.src* definiert wurde, ausschliesslich in *Parking_Lot.src* verwendet werden.

Public Symbole

Public Symbole können in einer beliebigen Datei innerhalb eines Geräts verwendet werden. Beispielsweise können *Public* Symbole, die in der Datei *Parking_Lot.src* definiert wurden, in *Heating.fup* oder *Ventilation.src* verwendet werden, da all diese Dateien im selben Gerät vorliegen.

Externe Symbole

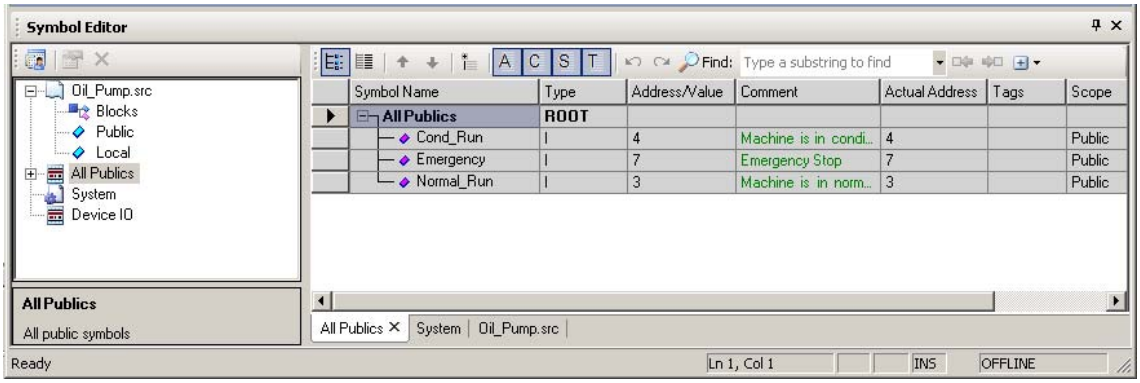
Externe Symbole verweisen auf eine Definition von *Public* Symbolen in einer anderen Programmdatei. Um ein *Public* Symbol zu verwenden, das in einer anderen Programmdatei definiert wurde, muss die Definition des externen Symbols in der

aktuellen Programmdatei vorliegen. Ziehen Sie das *Public* Symbol aus dem Filter *Publics* in den Programm-Editor. Die Definition des externen Symbols wird automatisch in die aktuelle Datei aufgenommen. (Weitere Einzelheiten zur Verwendung von *Public* Symbolen finden Sie im nächsten Abschnitt „Einsatz von Symbolen“).

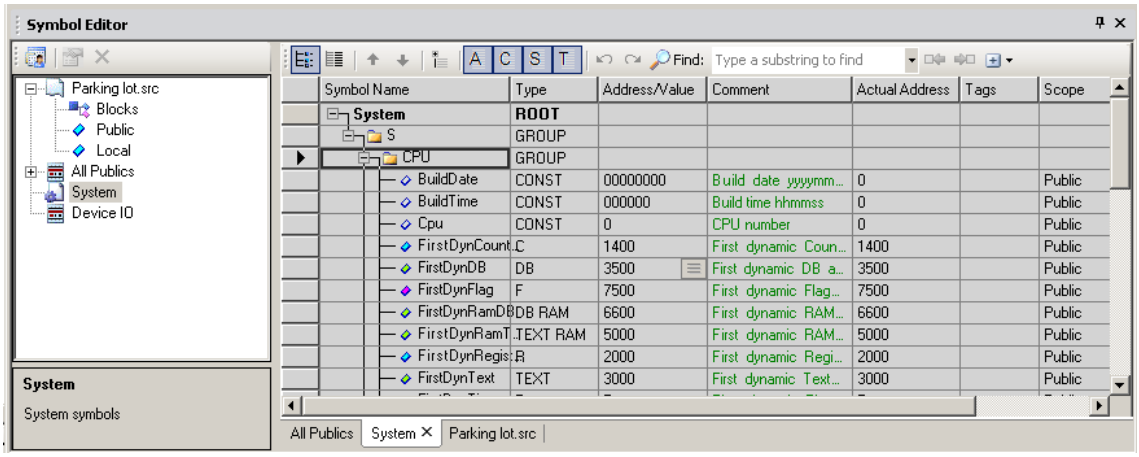
5.2.5 Filteransichten

Die Filter können im Navigationsverzeichnis aufgerufen werden. Die einzelnen Filteransichten können ausserdem in einem neuen Reiter geöffnet werden. Die folgenden Abbildungen zeigen die Ansichten für einige vorgegebene Filter. Eine graue Symbol-Tabelle zeigt an, dass die Symbole in diesen Ansichten nicht bearbeitet werden können. Die Symbol-Definition kann nur in der Quelldatei geändert werden, in der auch die entsprechende Symbol-Definition abgelegt ist.

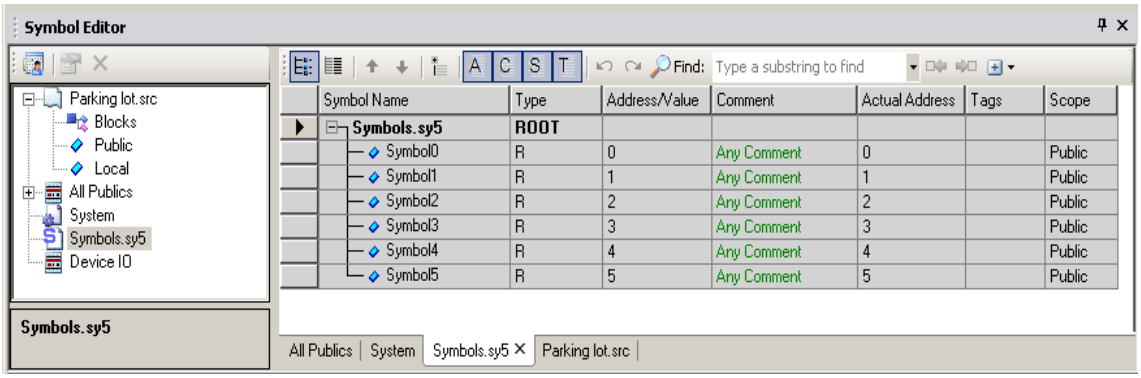
Der vorgegebene Filter *Publics* zeigt alle Publicen Symbole, die im aktuellen Gerät vorliegen.



Der vorgegebene Filter *System* zeigt alle Systemsymbole, die im aktuellen Gerät vorliegen.



Enthält das Gerät die Datei *.sy5*, erscheint der Filter mit dem *File Name* im Navigationsverzeichnis, und alle Symbole aus der *.sy5*-Datei werden in der Tabelle angezeigt.

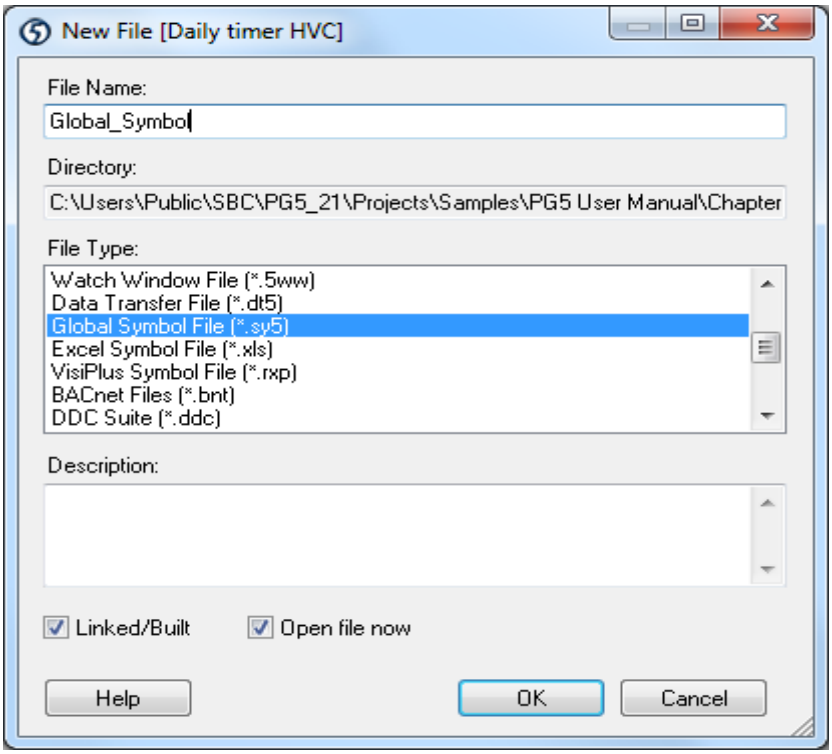


5.2.6 Symbol-Definition in der .sy5-Datei

Wie bereits in den oben stehenden Abschnitten erwähnt, sind die Symbol-Definitionen in Programmdateien (IL/Fupla usw.) abgespeichert. Alternativ können *public* Symbole an einer zentralen Stelle der *.sy5*-Datei definiert und in Programmdateien des aktuellen Geräts verwendet werden.

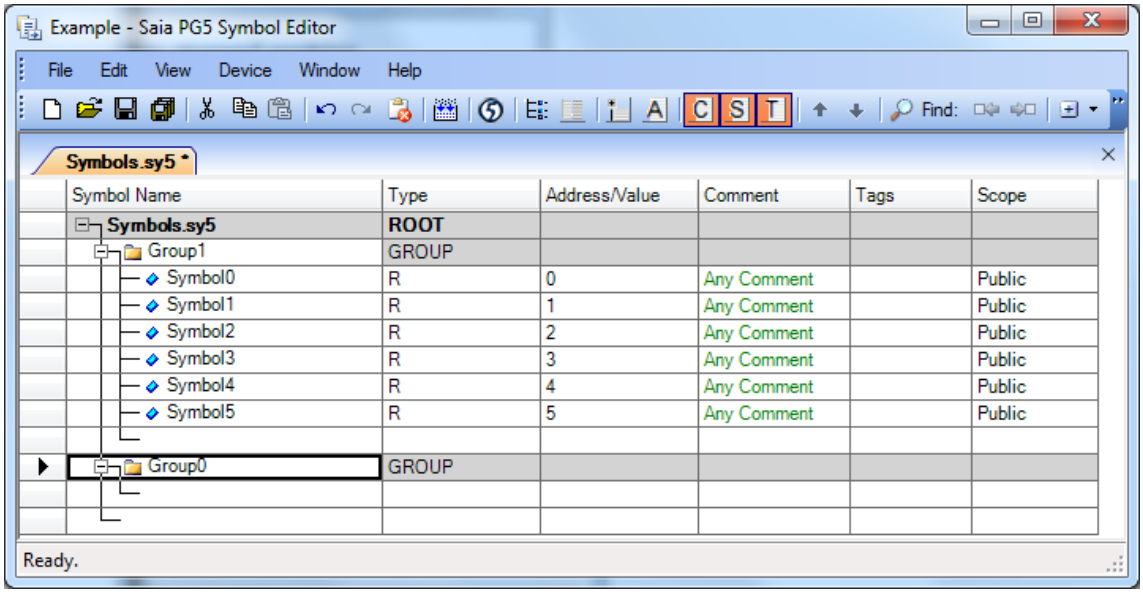
Wenn *Public* Symbole in Programmdateien (IL/Fupla usw.) definiert werden, gehen sie nicht verloren, wenn Programmdateien von einem Gerät zu einem anderen kopiert werden. Daher ist es vorteilhaft, die *Public* Symbole in den entsprechenden Programmdateien, anstatt an einer Stelle in der *.sy5*-Datei zu definieren.

Neue *.sy5*-Dateien können erstellt und über die Auswahl des Dateityps *.sy5* zum Projekt hinzugefügt werden. Fügen Sie die *.sy5*-Datei aus dem Project Manager ähnlich wie eine Programmdatei hinzu, und die *.sy5*-Datei öffnet sich im Symbol-Editor.



Geöffnete .sy5-Datei im Symbol-Editor

Diese Datei mit der *Public* Symbol-Definition wird im Symbol-Editor geöffnet, die Symbol-Definitionen können in der Datei hinzugefügt werden.



Globale Symbole aus PG5-Versionen 1.4 und älter

Wird ein Projekt mit PG5-Version 1.4 in der PG5-Version 2.0 wiederhergestellt, werden die *Public* Symbol-Definitionen in der Datei *Globals.sy5* gespeichert und können im Symbol-Editor eingesehen werden.

5.2.7 Symbol-Definitionen in den Dateiformaten .xls, .txt und .rxp

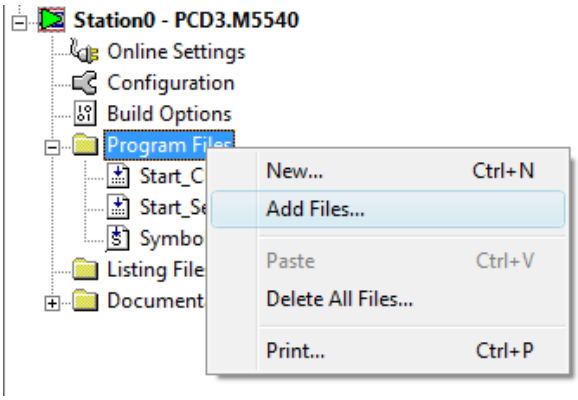
Symbol-Definitionen können auch in anderen Anwendungsdateien wie *.xls, .txt* usw. definiert werden. Diese Dateien können dann zum Gerät hinzugefügt werden. Nachdem die Build-Symbole in diesen Dateien definiert wurden, stehen sie im aktuellen Gerät zur Verfügung. Dadurch ist es nicht mehr notwendig, externe Symbol-Dateien zu importieren und diese mit den internen Symbolen im Geräteprogramm zusammenzuführen. Wir ziehen eine Verknüpfung der externen Dateien einem Import vor! Dieser Prozess ist besser und einfacher. Darüber hinaus können neue Versionen dieser Dateien ganz einfach hinzugefügt werden.

Wenn jedoch noch Symbole importiert werden müssen, können jederzeit die Symbole aus den Originaldateien wie *.xls, .txt, .rxp, oder .sy5* kopiert und in die Symbol-Tabelle eingefügt werden. (Weitere Einzelheiten in Abschnitt 5.3.16).

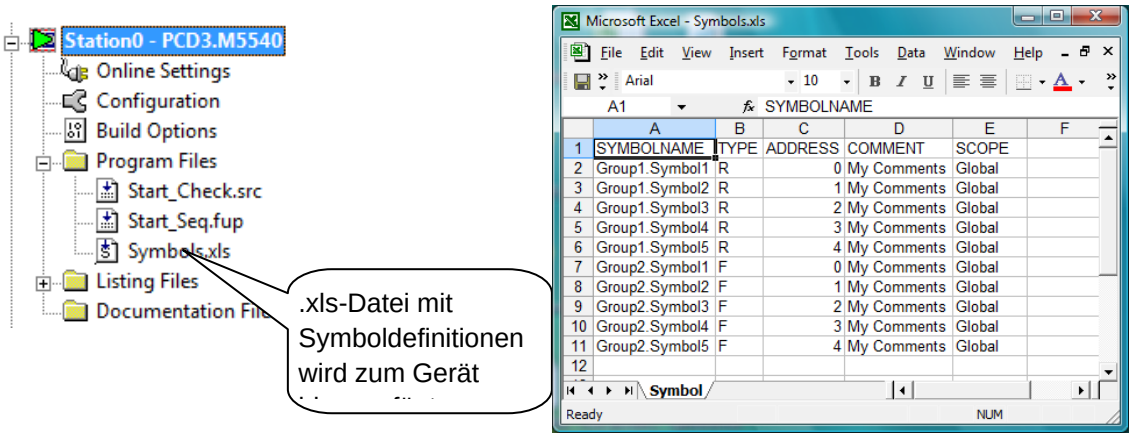
Wurde die Symbol-Datei bereits bearbeitet, kann sie durchsucht und über einen Rechtsklick auf den Ordner *Program Files* und über einen Klick auf *Add Files* hinzugefügt werden. Es kann ebenfalls eine neue Symbol-Datei (*.xls, .txt* usw.) erstellt und hinzugefügt werden. Führen Sie dazu einen Rechtsklick auf *New* in *Program files* aus und wählen Sie im Dialogfenster *New File* den entsprechenden Dateityp.

Anmerkung:
Public Symbol-Dateien mit der Endung *.sy5, .xls* und *.rxp* werden in dem Programm bearbeitet, das ihrer Endung entspricht, und im originalen Dateiformat gespeichert.

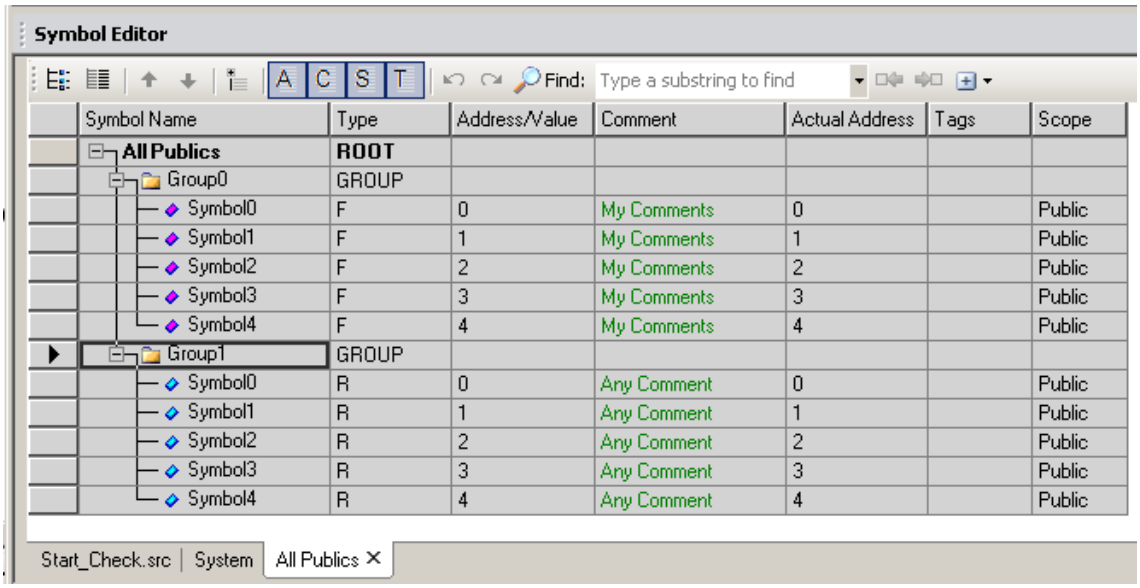
Symbol-Definitionsdatei hinzufügen



Symbol-Definitionen aus Excel-Datei (*.xls) hinzugefügt



Nach dem Build-Prozess stehen die in der Excel-Datei definierten *Public* Symbole im Filter All *Publics* zur Verfügung und können dann in den Programmen verwendet werden.



5.2.8 Definieren von Symbolen für die Kommunikationsnetzwerke

Der Datenaustausch zwischen zwei verschiedenen PCDs ist komplizierter als der Informationsaustausch zwischen zwei Dateien. Zwischen zwei PCDs ist eine Netzwerkverbindung erforderlich. Diese Netzwerkverbindung kann in unserem Netzwerk-Editor eingerichtet werden (derzeit werden Netzwerke vom Typ SBus, Profibus DP und LON unterstützt). Für die Datenübertragung zwischen PCDs können Netzwerksymbole in Programmen eingesetzt werden. Diese Symbole werden als *Public* Symbole angezeigt.

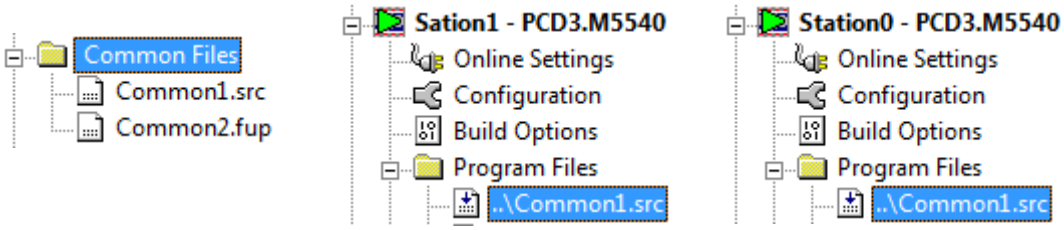
5.2.9 Symbol-Definition in gemeinsam genutzten Dateien

Wenn dieselbe Datei in mehreren verschiedenen Geräten verwendet werden soll, wird sie im Ordner *Common Files* des Projekts abgelegt. Bei diesen Dateien kann es sich um Programmdateien wie *.src, .fup, .sfc* oder *Public* Symbol-Dateien wie *.sy5, .xls, .txt, .rxp* usw. handeln.

Public Symbole, die in Programmdateien definiert wurden, oder *Public* Symbol-Dateien, die in den Ordner *Common Files* des Projekts gelegt wurden, können für mehrere verschiedene Geräte genutzt werden. Wird die gemeinsam genutzte Datei mit dem aktuellen Gerät verbunden, stehen alle *Public* Symbole, die in der gemeinsam genutzten Datei definiert wurden, im aktuellen Gerät zur Verfügung.

Beispiel:

Common1.src wird sowohl in Station0 als auch in Station1 referenziert. Alle in *Common1.src* definierten *Public* Symbole stehen sowohl in *Station0* als auch in *Station1* zur Verfügung.



5.3 Einsatz von Symbolen

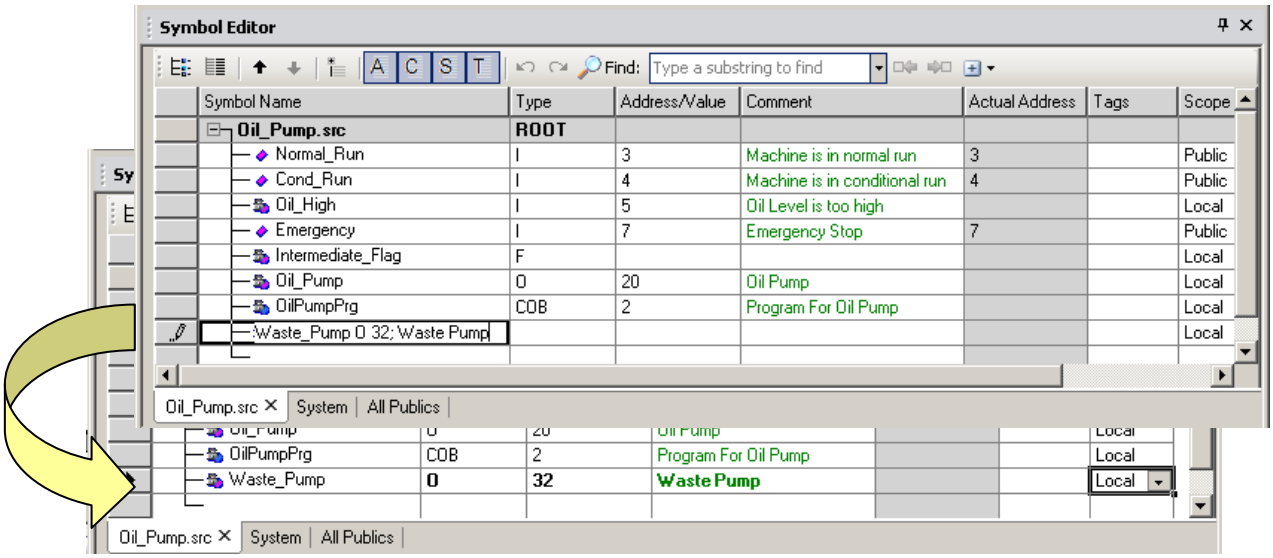
5.3.1 Hinzufügen von Symbolen zur Symbol-Liste

Einfache Methode

Öffnen Sie die Datei, mit der Sie arbeiten wollen. Es öffnet sich ebenfalls der Symbol-Editor. Im Symbol-Editor wird am Ende immer eine leere Zeile für neue Symbol-Einträge angezeigt. Um ein neues Symbol hinzuzufügen, geben Sie *Symbol Name* ein und bestätigen Sie mit der Eingabetaste. Wählen Sie jetzt den *Type*, geben Sie *Address/Value* und *comments* ein und wählen Sie bei *scope* den Umfang des Symbols. Wenn die Symbol-Definition hinzugefügt wird, wird am Ende automatisch eine neue leere Zeile für die nächste Symbol-Definition eingefügt.

Schnelle Methode

Sie können ebenfalls Variablen für die verschiedenen Informationsfelder aus dem Feld *Symbol Name* eingeben. Dieser Vorgang ist praktischer und schneller. Siehe nachfolgendes Beispiel.



Es muss folgende Syntax eingehalten werden:
Symbol_Name Typ Adresse ;Anmerkung

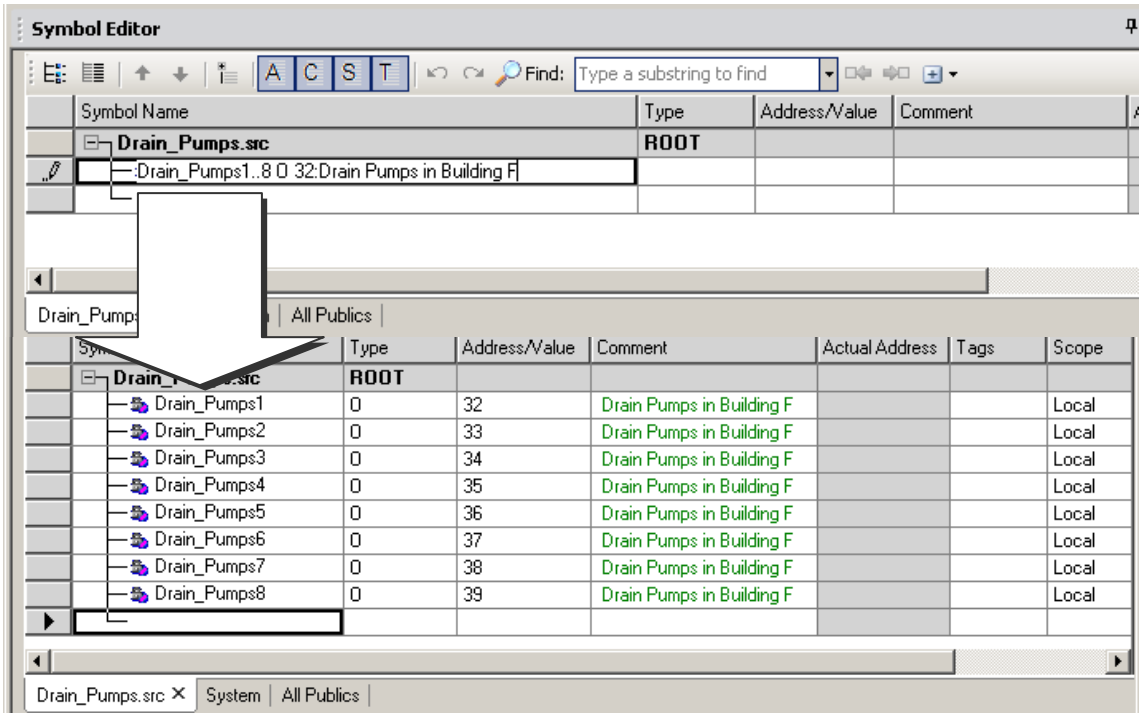
Wenn die oben angegebene Syntax für das neue Symbol verwendet wurde, werden die Informationen durch Drücken der Enter-Taste automatisch in die korrekten Felder übernommen.

Eingabe von Symbol-Teildefinitionen

Es können Symbol-Teildefinitionen gespeichert werden. So können Sie beispielsweise Symbol-Namen und -typen eingeben, die Arbeit im Symbol-Editor unterbrechen und Adressen, Anmerkungen und Tags zu einem anderen Zeitpunkt eingeben. Ihre Arbeit im Symbol-Editor können Sie dann später fortsetzen und die Zwischenergebnisse speichern.

5.3.2 Hinzufügen mehrerer Symbole zum Symbol-Editor

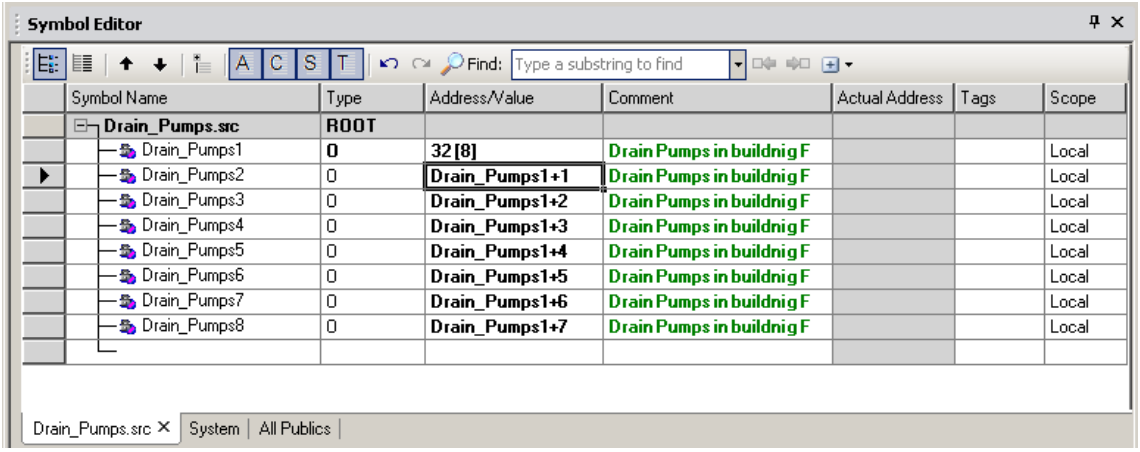
Sie können eine Ihrer Liste Reihe von Symbolen hinzufügen, falls notwendig. Geben Sie dazu den Symbol-Namen mit der ersten und letzten Elementnummer wie abgebildet ein (Drainpumps1..8 O 32 ;Pumpen in Gebäude F). 8 steht für die Anzahl der Symbole, O steht für Ausgang und 32 ist der Beginn des Adressbereichs, den Sie eingeben. Drücken Sie *Enter*, und der Symbol-Editor vervollständigt die Liste. Es können 100 Symbole gleichzeitig hinzugefügt werden. Sind mehr als 100 Symbole angegeben, werden nur die ersten 100 Symbole übernommen.



5.3.3 Referenzierte Symbole

Ein Symbol kann über eine Referenz zu einem anderen Symbol statt des Adressenwerts definiert werden. Wenn eine Symbol-Reihe vorliegt, können die Symbol-Adressen mit der Basisadresse der Reihe verknüpft werden. (Die Symbol-Adresse wird über ein anderes Symbol der Tabelle definiert).

Wenn die physischen Adressen von mehreren Ein- und Ausgängen in der Software geändert werden müssen, kann dies problemlos über die referenzierten Symbole vorgenommen werden. Sie müssen dabei nur das erste ändern, die Änderungen werden dann für alle anderen übernommen. Im folgenden Beispiel besteht für die referenzierten Symbole *Drain_Pumps2* bis *Drain_Pumps8* eine Adressreferenz auf das erste Symbol *Drain_Pumps1*.



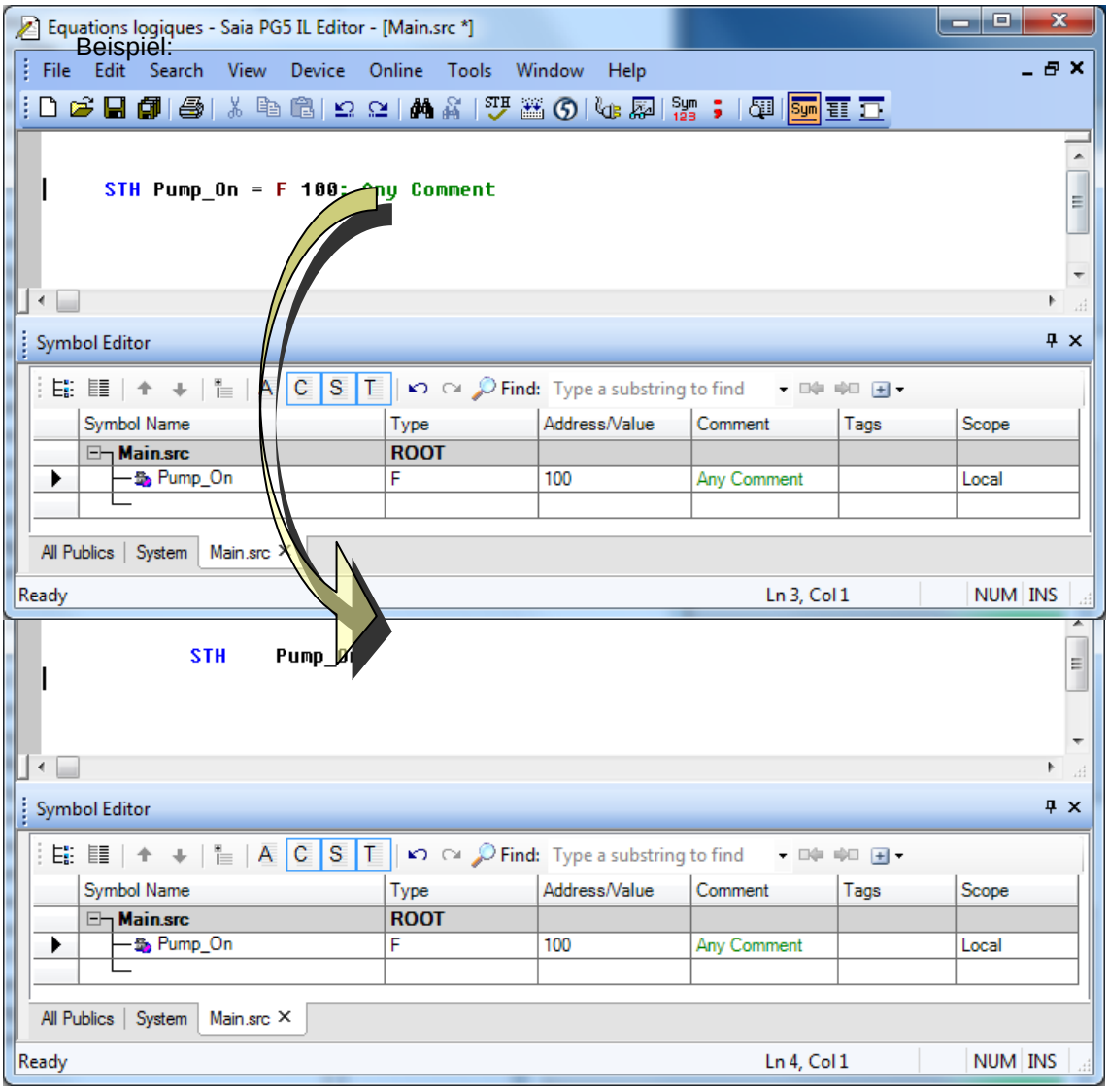
5.3.4 Hinzufügen eines Symbols im IL-Programm

Neue Symbole können auch während der Programmbearbeitung hinzugefügt werden. Bearbeiten Sie dazu eine Programmcodezeile mit dem mnemotechnischen Code und dessen Operanden. Geben Sie für den Operanden den Symbol-Namen und die Definition in der folgenden Syntax an:
Symbol_Name = Typ Adresse ;Anmerkung

Nach Drücken der *Enter*-Taste werden die neuen Symbole automatisch in die *Symbols*-Liste übernommen, aber nur, wenn die Symbol-Definition korrekt ist und wenn die Option *Automatically add entered type/value to the Symbol Table* ausgewählt wurde (Menü *Tools, Options* im IL-Editor).

Anmerkung:
Alle neuen Symbole, die direkt über den IL-Editor festgelegt werden, werden als lokale Symbole hinzugefügt. Falls nötig kann der Umfang dann zu einem späteren Zeitpunkt im Symbol-Editor geändert werden.

Beispiel:

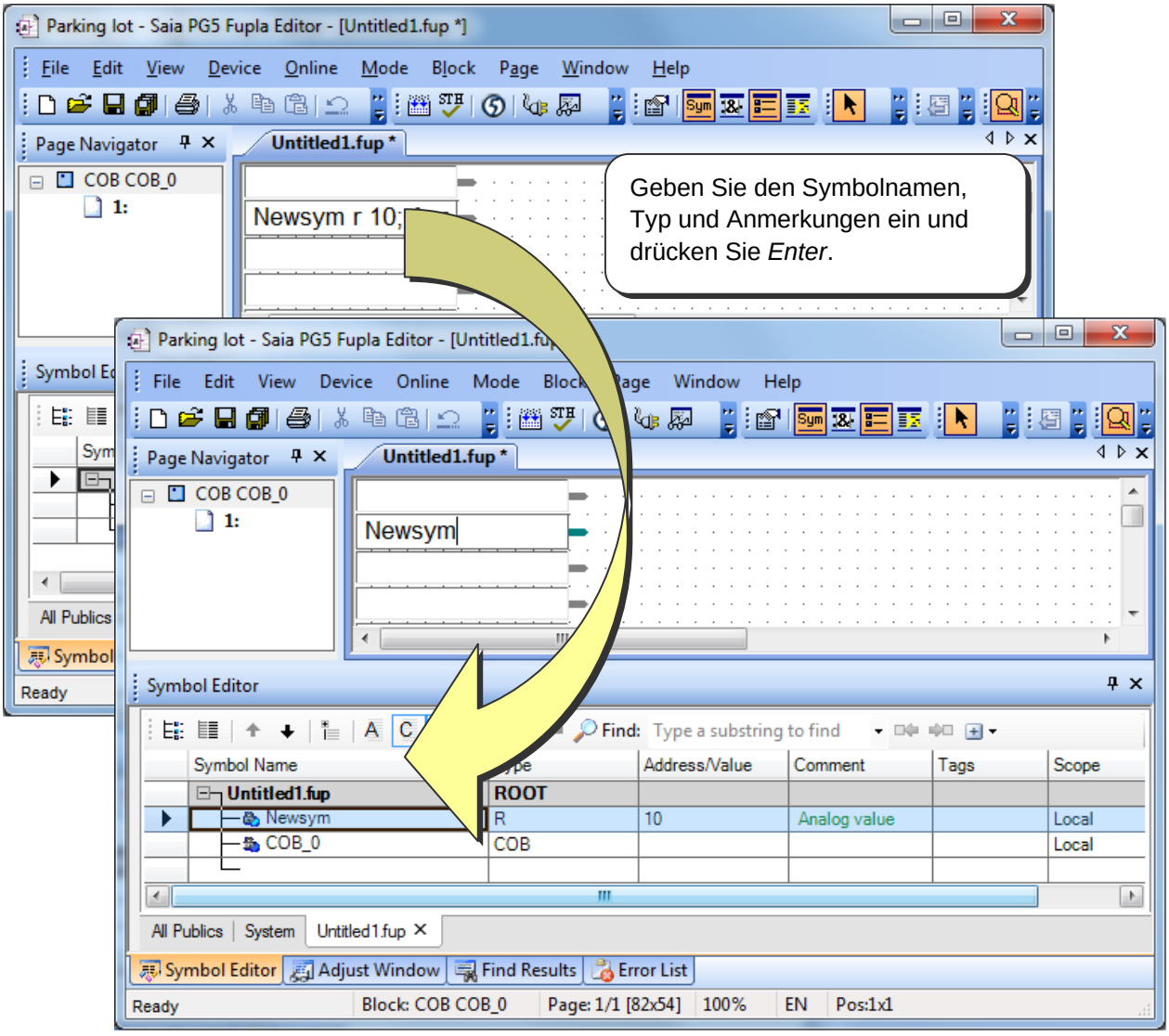


5.3.5 Hinzufügen eines Symbols in Fupla

Der Fupla-Editor funktioniert genau so wie der IL-Editor. Sie können neue Symbole direkt über das Eingangs-/Ausgangsfeld in Fupla in der Liste des Symbol-Editors eingeben.

Syntax: *Symbol-Name Symbol-Typ [Adresse] [; Anmerkung]*

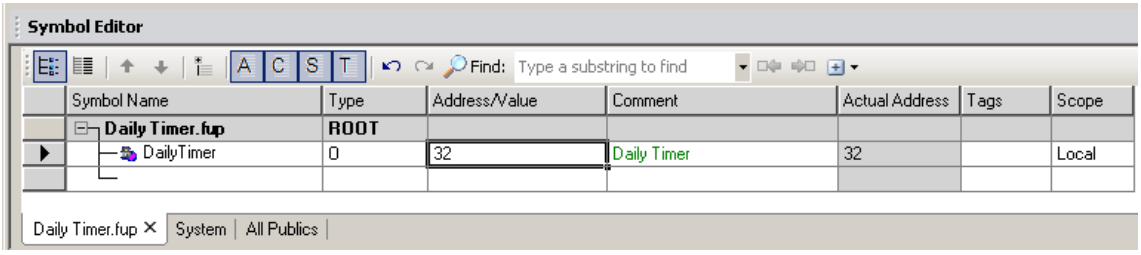
Anmerkung:
Alle neuen Symbole, die direkt über den Fupla-Editor festgelegt werden, werden als lokale Symbole hinzugefügt. Falls nötig kann der Umfang dann zu einem späteren Zeitpunkt im Symbol-Editor geändert werden.



5.3.6 Symbol-Adressierungsarten

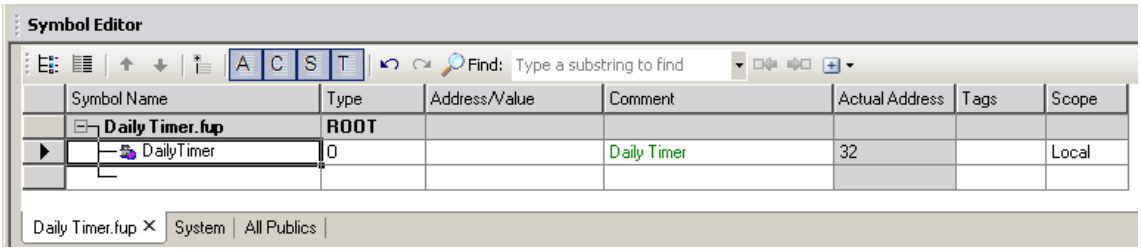
Eine Symbol-Definition enthält nicht zwingend alle unten angegebenen Informationen. Wir unterscheiden zwischen drei verschiedenen Adressierungsarten:

Symbol-Namen



Die Daten werden über einen Symbol-Namen, Typ, Adresse und optionale Anmerkungen definiert. Die Korrektur eines Symbols, des Typs oder der Adresse wird von der Symbol-Liste unterstützt. Alle Benutzerprogrammanschlüsse werden automatisch aktualisiert, wenn ein Symbol geändert wird.

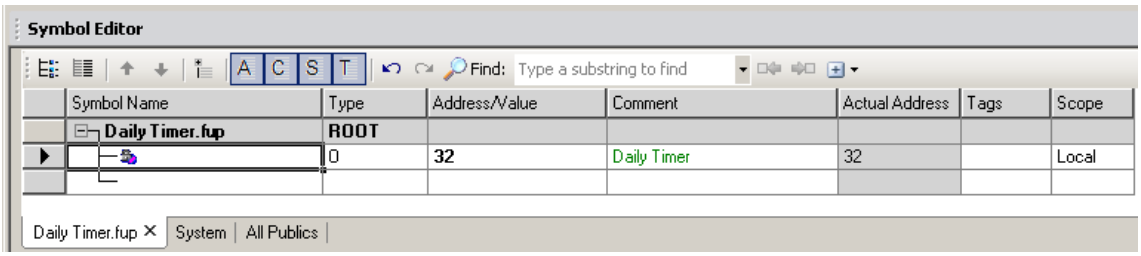
Dynamische Adressierung



Hierbei handelt es sich um eine Form der symbolischen Adressierung, bei der die Adresse nicht festgelegt wird. Die Adresse wird automatisch während des Build-Prozesses zugewiesen. Sie wird dem Adressbereich entnommen, der in den *Build Options* angegeben wurde. (Siehe Project Manager). Nach Abschluss des Build-Prozesses wird die dynamische Adresse in der Spalte *Actual Address* angezeigt.

Anmerkung: Die dynamische Adressierung kann für Flags, Zähler, Timer, Register, Texte, DB, COB, PB, FB und SB genutzt werden. Für Eingänge, Ausgänge und XOB müssen jedoch immer die absoluten Adressen angegeben werden.

Absolute Adressen



Die Daten werden nur mit einem Typ und einer Adresse (z.B. 32) und einer optionalen Anmerkung definiert. Werden absolute Adressen direkt im Programm verwendet, kann sich das als Nachteil bei der Änderung des Typs oder der Adresse

herausstellen. Im Benutzerprogramm werden die Änderungen, die in der Symbol-Liste vorgenommen wurden, nicht übernommen. Die Änderungen müssen dann manuell für die Anschlüsse des Programms vorgenommen werden. Daher wird empfohlen, Symbol-Namen mit optionaler dynamischer Adressierung zu verwenden.

5.3.7 Einsatz von Symbolen in Programmen

Bei der Bearbeitung eines Programms können die bereits im Symbol-Editor definierten Symbole auf verschiedene Arten eingesetzt werden. Alle der folgenden Methoden können sowohl im IL- als auch im Fupla-Editor verwendet werden.

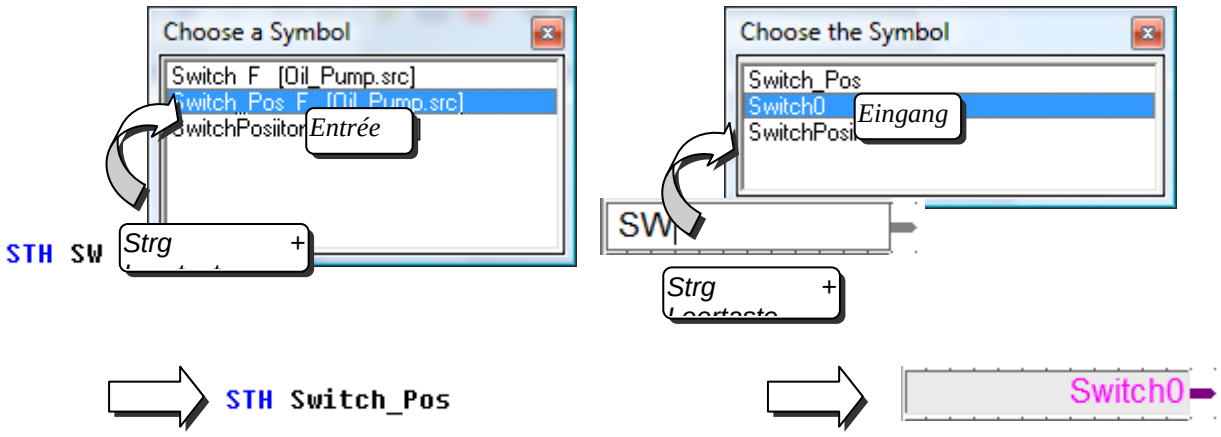
Symbol-Eingabe über die Tastatur

Der Symbol-Name wird vollständig über die Tastatur für alle Befehle eingegeben, in denen das Symbol auftritt. Mit dieser Methode ergeben sich womöglich Rechtschreibfehler im Symbol-Namen, was erst beim Build-Prozess offensichtlich wird.

Symbol-Eingabe über selektives Suchen

Wenn Sie lange Symbol-Namen verwenden, ist Ihr Programm einfacher zu lesen. Es ist jedoch unpraktisch, einen langen Symbol-Namen an den notwendigen Stellen immer wieder neu eingeben zu müssen. Dies umgehen Sie, indem Sie einfach die ersten Buchstaben eines Symbols eingeben und dann Strg+Leertaste drücken. Dann erscheinen alle Symbole, die der Suchanfrage entsprechen. Das benötigte Symbol kann entweder mit der Maus oder den Pfeiltasten der Tastatur (↑, ↓) ausgewählt und mit *Enter* bestätigt werden.

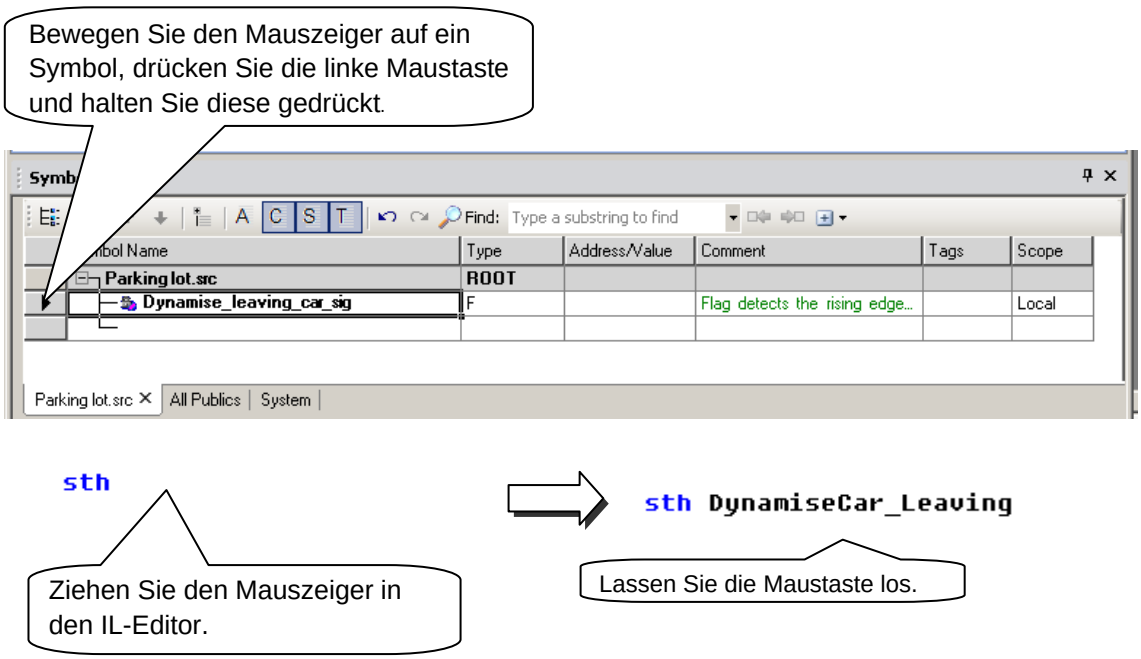
Beispiel:



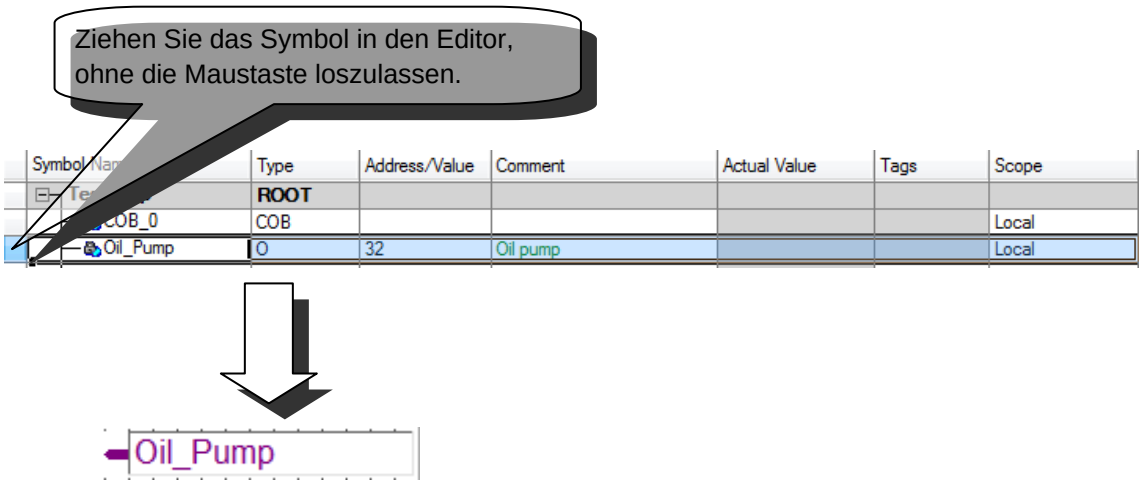
Ziehen von lokalen Symbolen in den Programm-Editor

Wurden die Symbole im Symbol-Editor definiert, können sie in den Programm-Editor gezogen und im Programm verwendet werden, ohne dass dabei der Symbol-Name eingegeben werden muss. So werden Tippfehler umgangen. Gehen Sie im Fenster *Symbols* mit dem Mauszeiger auf die Definitionszeile eines Symbols, drücken Sie die linke Maustaste und halten Sie diese gedrückt. Ziehen Sie den Mauszeiger in den Programm-Editor und lassen Sie die Maustaste los. Das ausgewählte Symbol wird automatisch dort eingefügt, wo sich der Mauszeiger befindet.

Beispiel 1

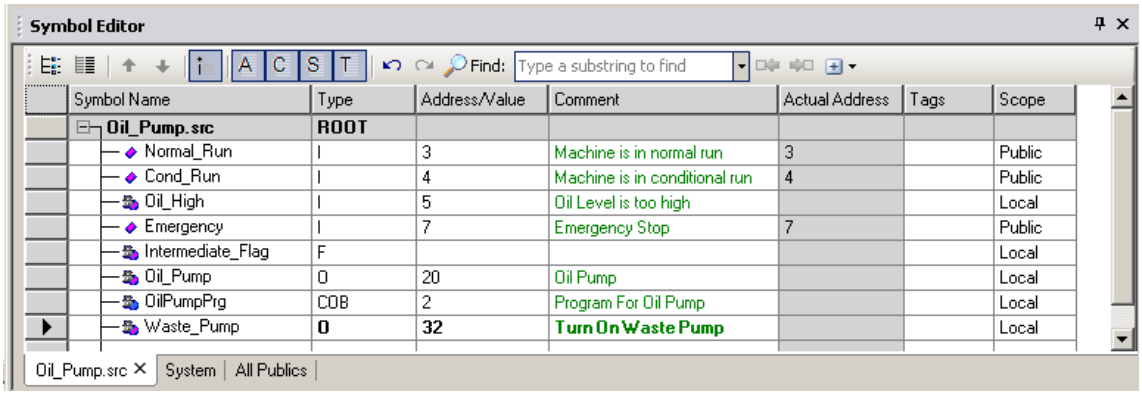


Beispiel 2



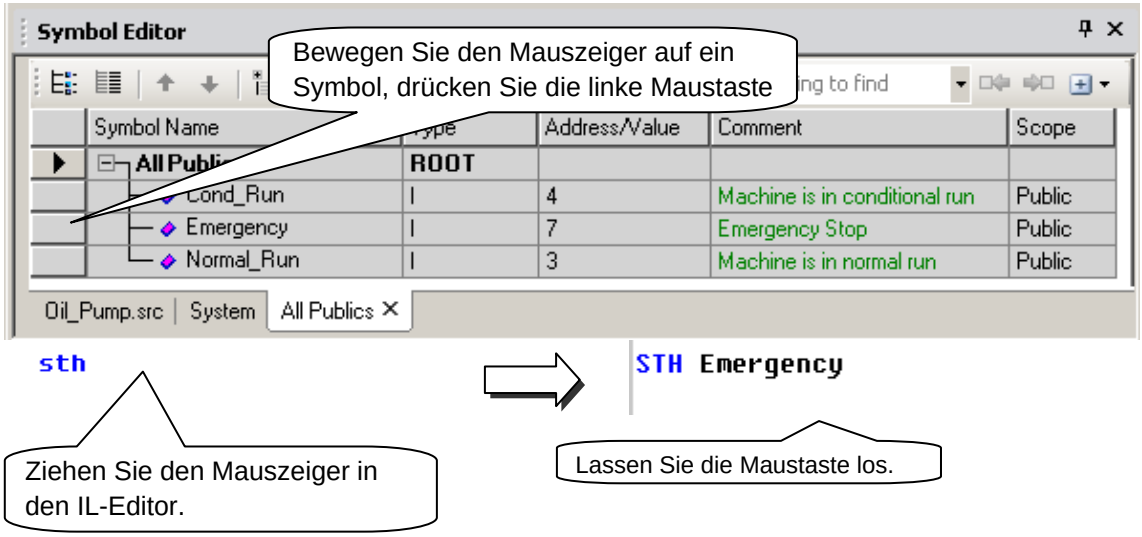
Ziehen von *Public* Symbolen in den Programm-Editor

Public Symbol *Emergency* definiert in *Oil_Pump.src*

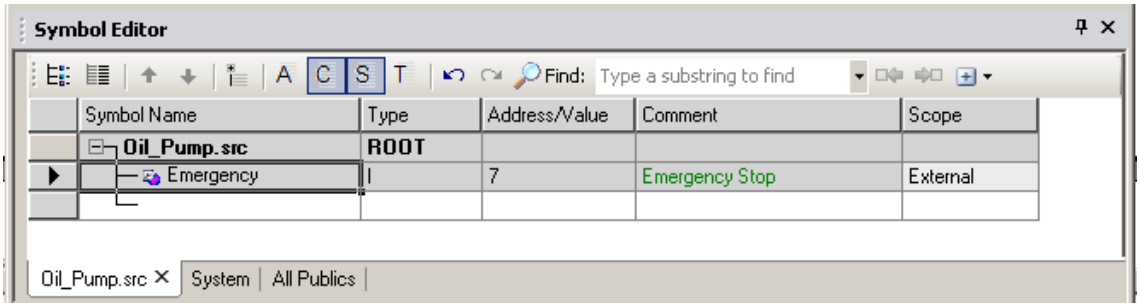


Die *Public* Symbole, die in *Oil_Pump.src* definiert wurden, werden im Filter *All Publics* der Dateien im aktuellen Gerät angezeigt, z.B. *Parking_Lot.src*. Um das *Public* Symbol zu verwenden, kann es aus dem Filter *All Publics* in den Programm-Editor ***gezogen*** werden. Wenn Sie z.B. das *Public* Symbol *Emergency* in den Programm-Editor ziehen, wird automatisch das externe Symbol in die Datei *Parking_Lot.src* aufgenommen.

Zeigen Sie die *Public* Symbole in *Parking_Lot.src* an.



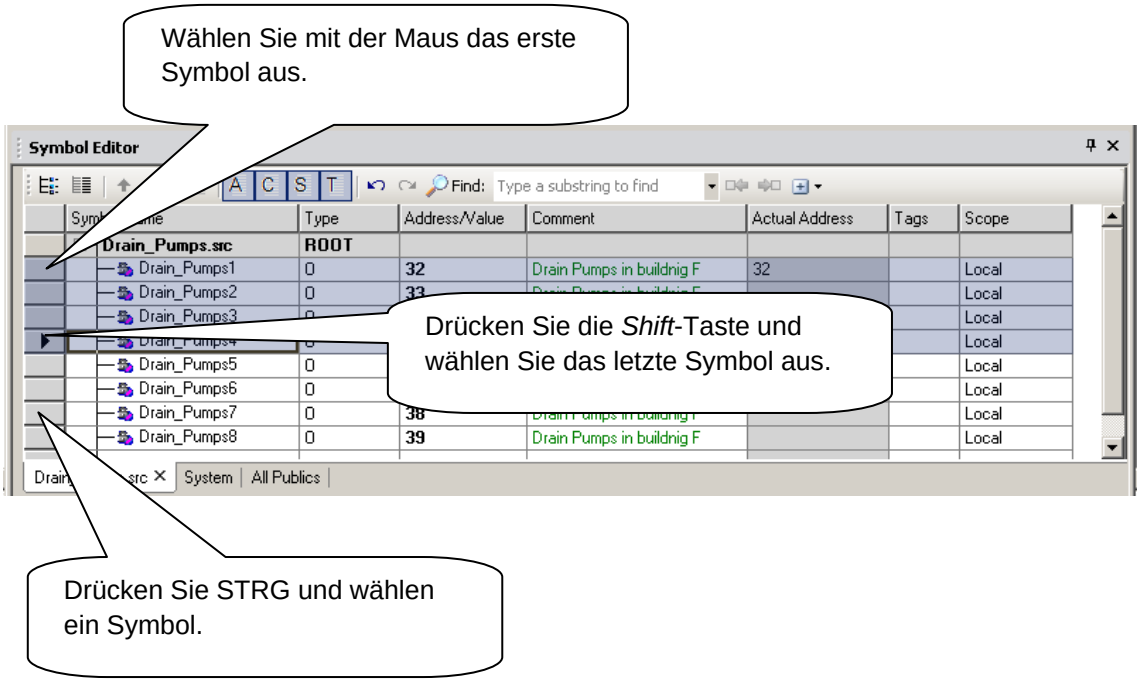
Das externe Symbol wurde automatisch in Parking_Lot.src hinzugefügt.



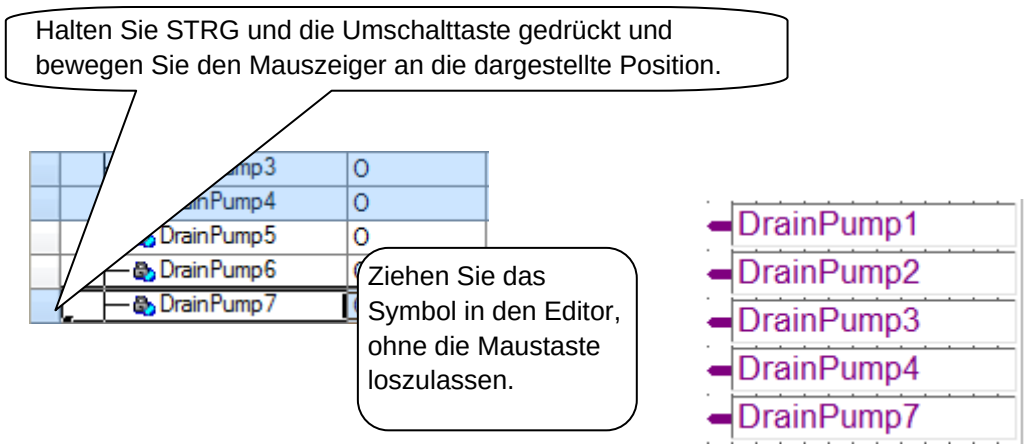
Ziehen von mehreren Symbolen in den Programm-Editor

Damit Symbol-Namen nicht mehrmals in ein Programm eingegeben werden müssen (Gefahr von Tippfehlern), können ein oder mehrere Symbole aus dem Symbol-Editor ausgewählt und in das Fupla- oder IL-Programm gezogen werden.

Beispiel: Auswahl mehrerer Symbole



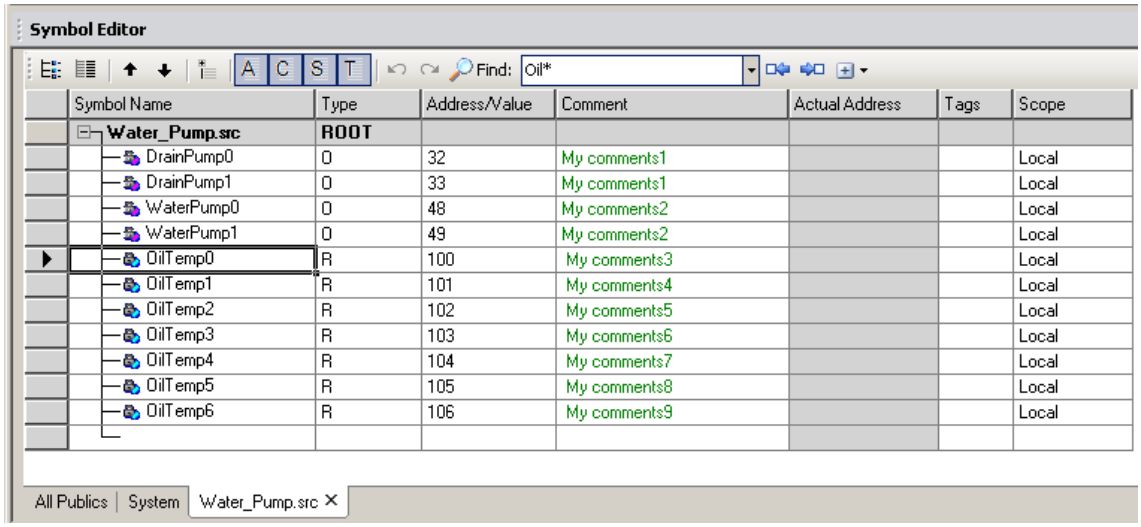
Beispiel: Symbol, das in den Fupla- oder IL-Editor gezogen wurde



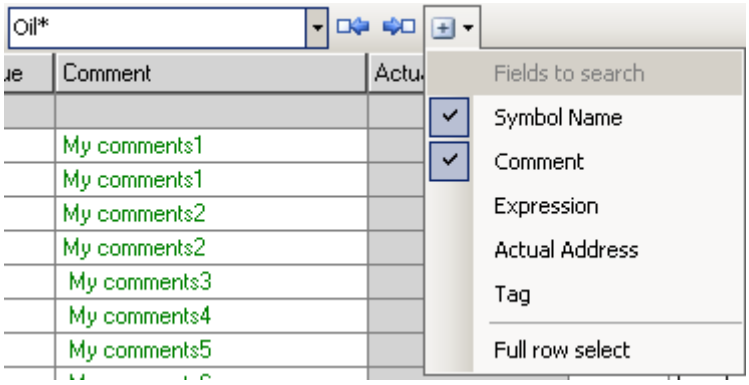
5.3.8 Suche nach Symbolen

Sind sehr viele *Public* Symbole in der Tabelle enthalten, kann über die Suchfunktion das Symbol in der Tabelle gesucht werden. Wurde das Symbol gefunden, kann es in den Programm-Editor gezogen werden.

Über die Funktion „Symbol suchen“ können Symbole gemäss ihrem regulären String gesucht werden. Die Schaltflächen „Weiter/Zurück“ stehen ebenfalls zur Verfügung. Bei der Suche können auch Platzhalter wie „*“ oder „?“ verwendet werden.



In den erweiterten Optionen können zu durchsuchende Spalten aktiviert und deaktiviert werden. Die Suche wird nur in den ausgewählten Spalten durchgeführt.



5.3.9 Auto-Zuweisung

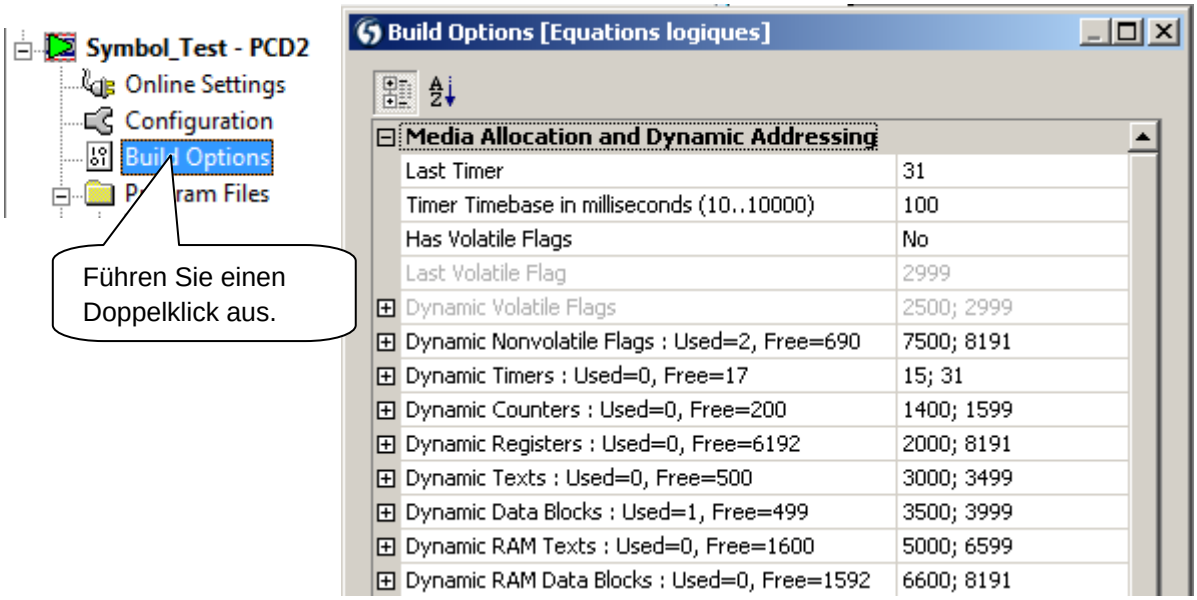
Bisher wurden Elemente immer folgendermassen festgelegt:

	Symbolname	Typ	Adresse	Anmerkung
Beispiel:	Pumpspeed	R	2000	;Geschwindigkeit in l/min

Das Festlegen einer Adresse ist nur bei den Symbol-Typen Ein- und Ausgang erforderlich. Wenn Sie einen Symbol-Typ ausser Ein- oder Ausgang eingeben, müssen Sie dafür keine Adresse festlegen. Wenn Sie keine Adresse eingeben, weist die PG5 ihrem Element während des Build-Prozesses automatisch eine Adresse zu. Dies wird als automatische (oder dynamische) Zuweisung bezeichnet. Die PG5 prüft den Adressbereich für das jeweilige Element, der in den *Software Settings* konfiguriert wurde, und weist während des Build-Prozesses eine Adresse zu.

Beispiel:	Pumpspeed	R	;Geschwindigkeit in l/min

Definition eines Registers im Programm ohne Angabe einer Adresse:

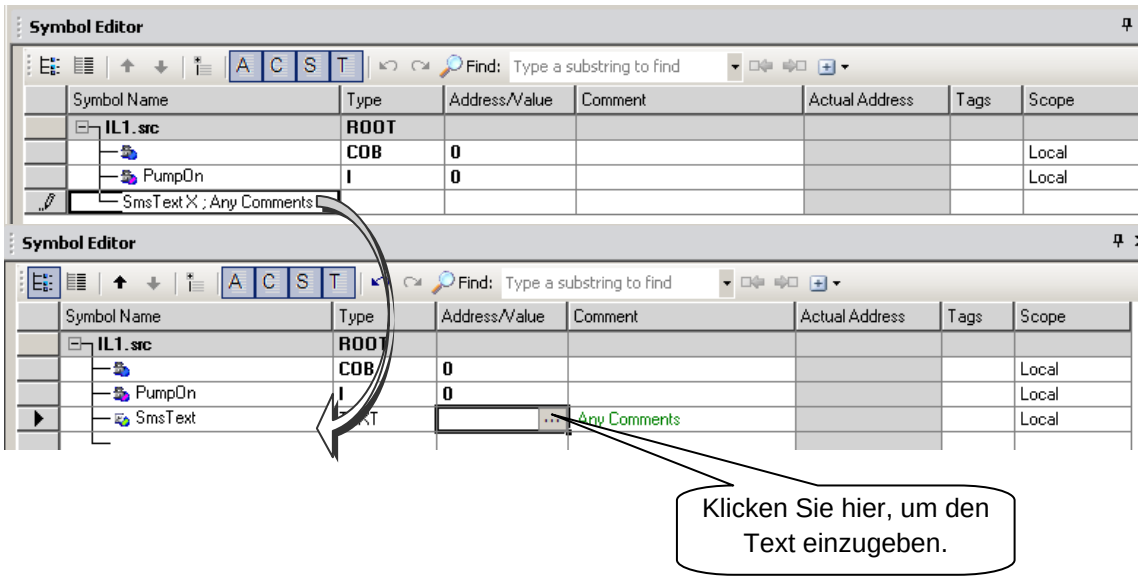


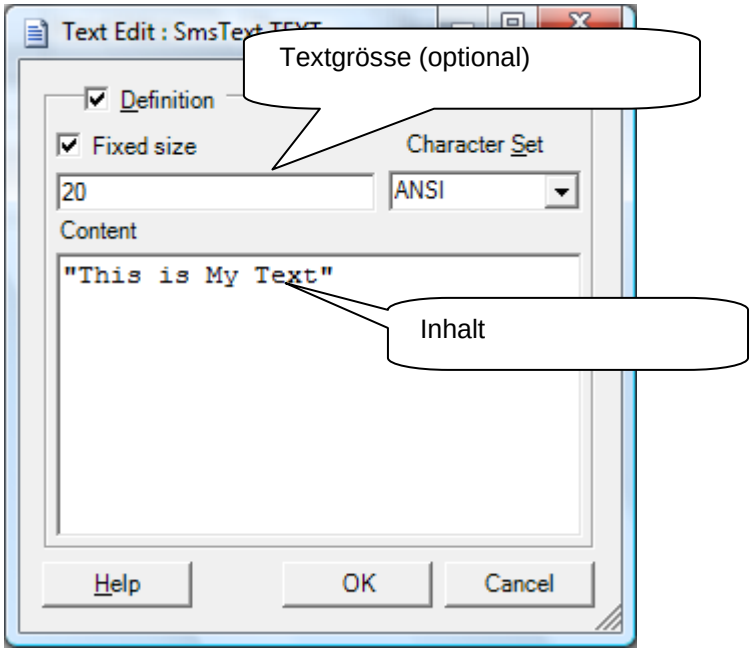
Dem Register wird während des Build-Prozesses eine Zahl zwischen 3500 und 4095 zugewiesen. Dies ist darauf zurückzuführen, dass der dynamische Bereich zwischen 3500 und 4095 für Register in den *Software Settings* festgelegt wurde. Nach dem Build-Prozess wird die zugewiesene Adresse in der Spalte *Actual Address* des Symbol-Editors angezeigt.

5.3.10 Texteingabe

Wenn Sie einen Text zu Ihrer PCD hinzufügen wollen, muss dieser Text zunächst als solcher ausgewiesen werden. Dazu wählen Sie im Symbol-Editor den Symbol-Typ „TEXT“. Alternativ kann der Datentyp „X“ nach dem Symbol-Namen wie unten dargestellt eingegeben werden.

Beispiel:

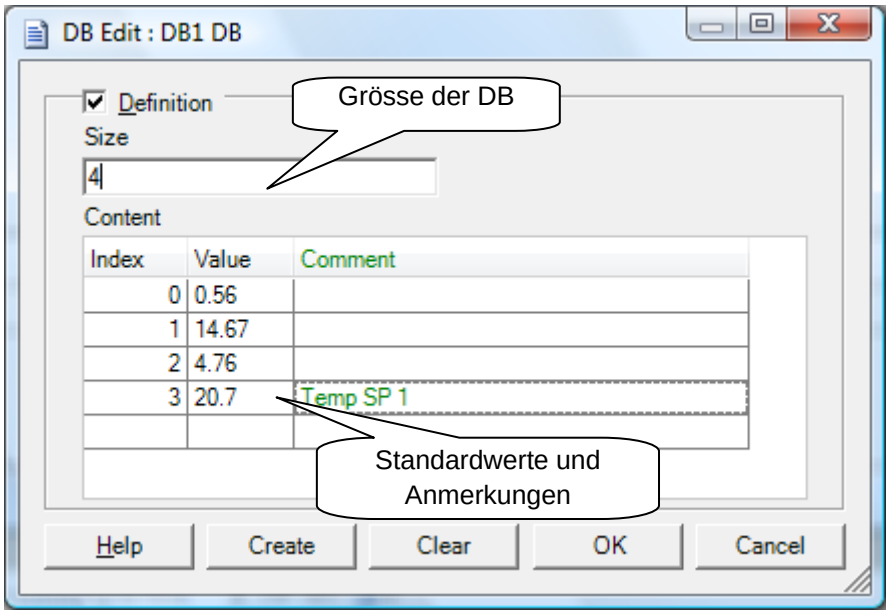




Vergessen Sie nicht, " " zu verwenden, sonst ist der Text nicht gültig.

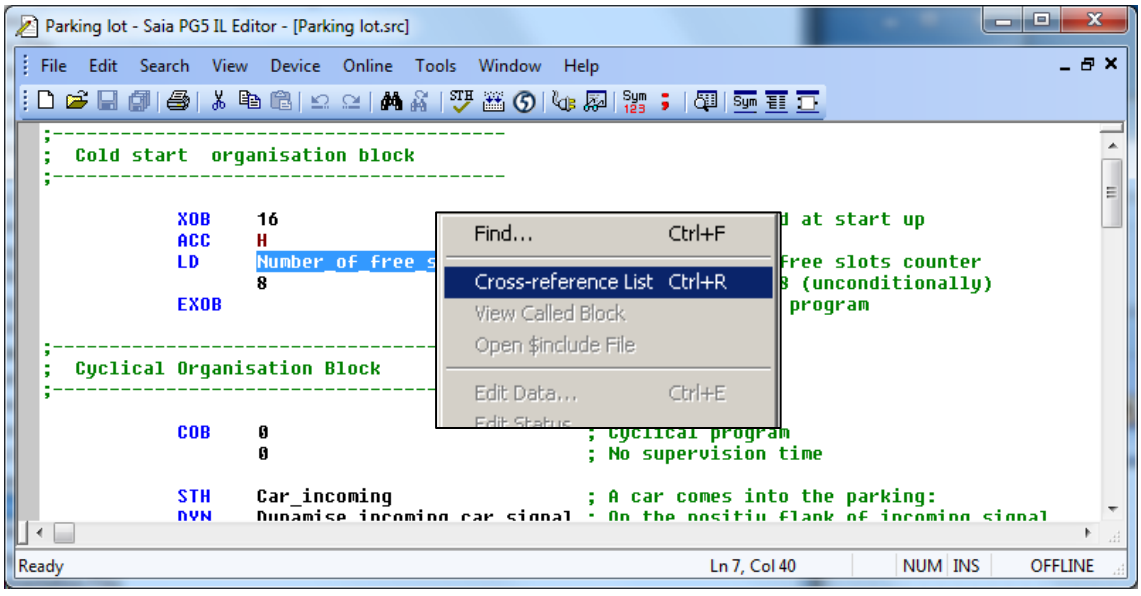
5.3.11 Eingabe von DBs

Auch für DBs gibt es einen speziellen Editor. Definieren Sie das Symbol mit dem Typ „DB“ und klicken Sie auf die Schaltfläche in der Spalte *Address/Value*, um den Editor zu öffnen. Geben Sie die Grösse ein (Anzahl der DB-Elemente) und klicken Sie auf die Schaltfläche *Create*. Es können darüber hinaus Standardwerte und Anmerkungen angegeben werden.

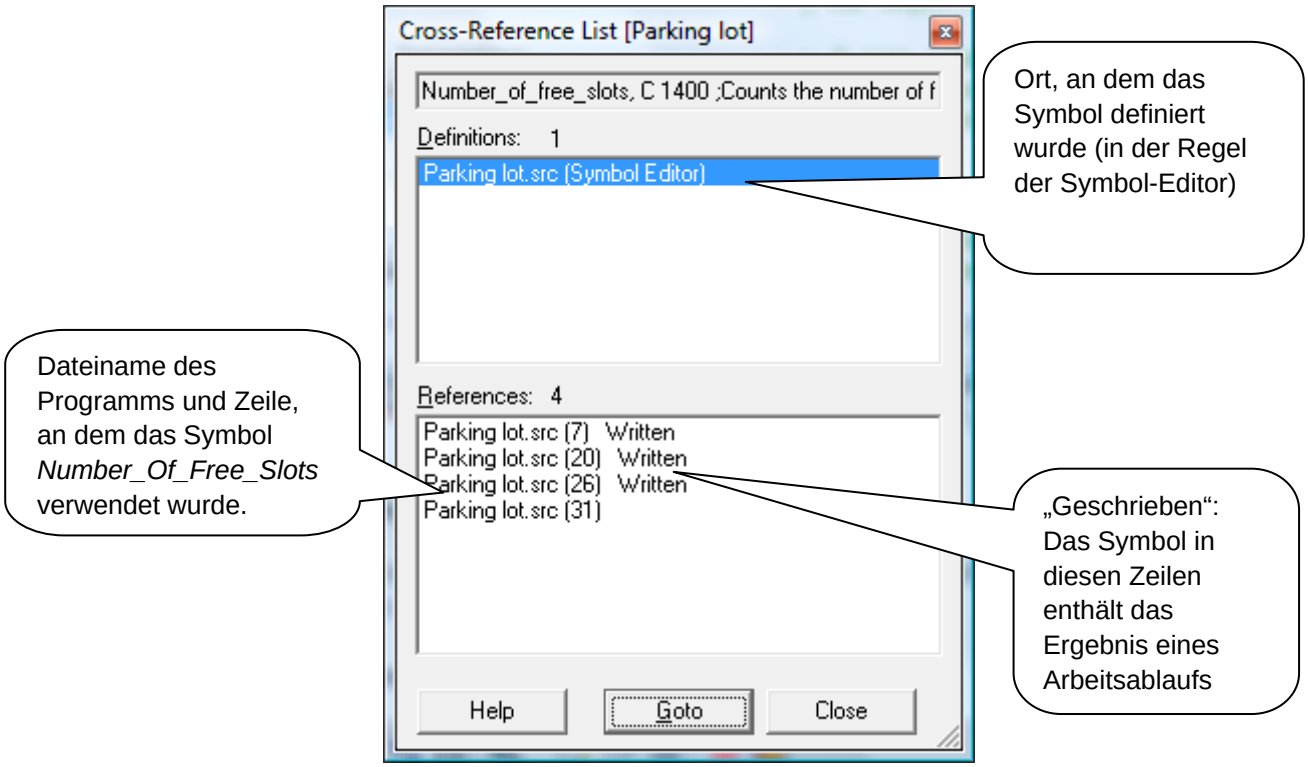


5.3.12 Symbol-Verweis

Ein Symbol wird häufig mehrmals innerhalb einer Programmdatei oder sogar in verschiedenen Dateien verwendet. Nach dem Speichern der Dateien, führen Sie auf einem Symbol einen Rechtsklick mit der Maus aus und rufen Sie die Funktion *Cross reference List* auf. Die Verweisoption kann ebenfalls über einen Rechtsklick und dem Kontextmenü im Symbol-Editor aufgerufen werden.

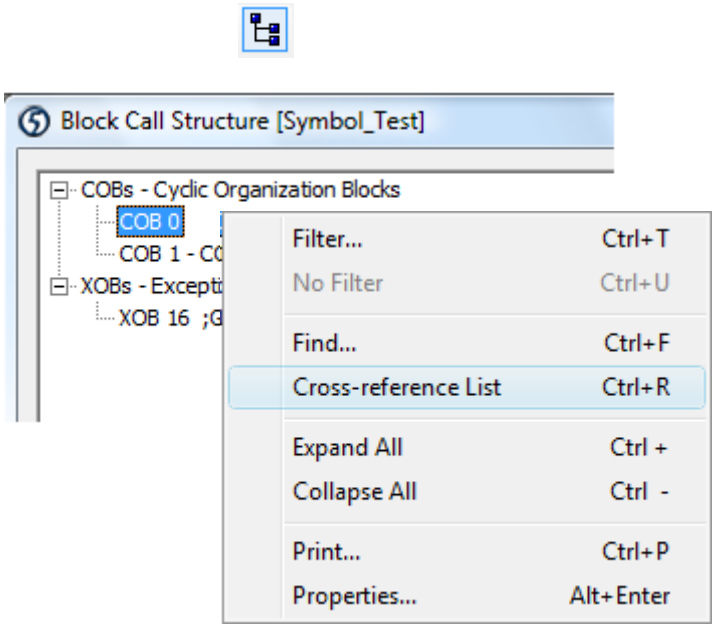


Die Verweisfunktion zeigt den Dateinamen und die Zeilenzahl an. Es wird ebenfalls angegeben, wie oft ein bestimmtes Symbol eingesetzt wurde. Führen Sie einen Doppelklick auf eine Position in der Verweisliste aus, um die Programmdatei zu öffnen. Der Cursor muss sich dabei auf dem jeweiligen Symbol befinden.



Das Tool *cross-reference* kann nicht nur im S-Editor und Fupla, sondern auch in den verschiedenen Ansichten verwendet werden, die im Project Manager zur Verfügung stehen.

Beispiel: Ansicht mit Blockstruktur



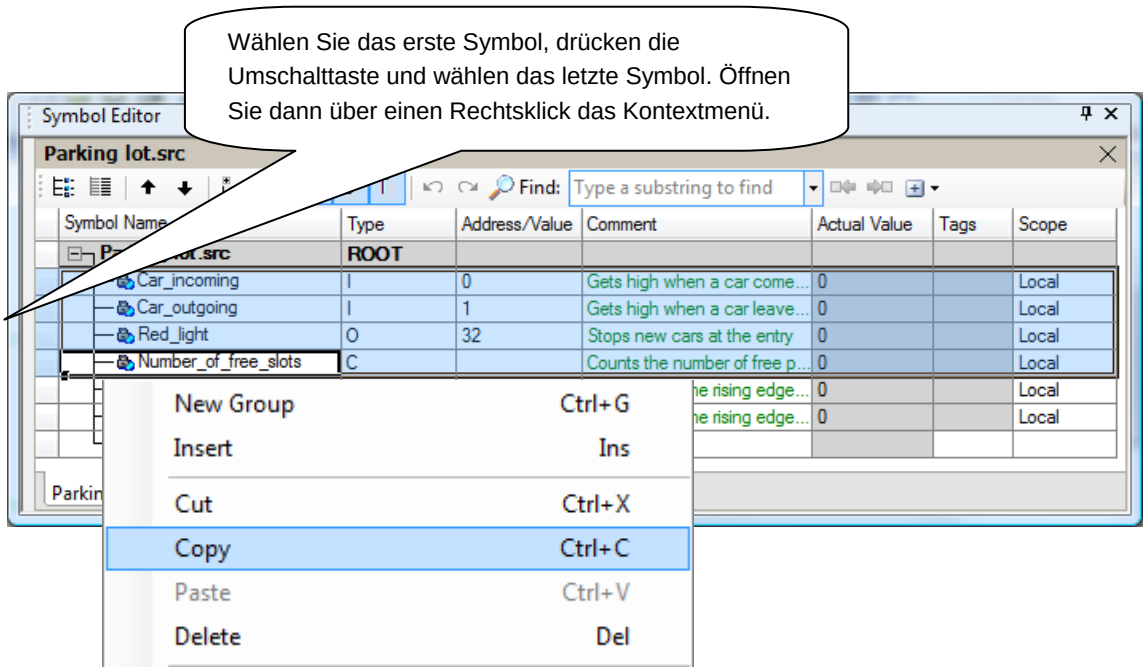
5.3.13 Bearbeiten von Bereichen in Symbol-Tabellen

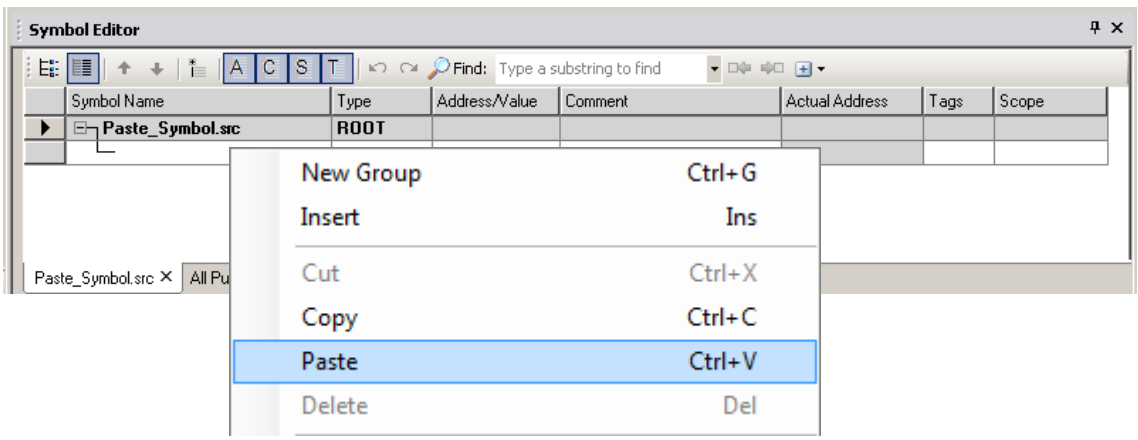
Es können Bereiche der Symbol-Tabelle ausgewählt und kopiert/eingefügt oder ausgeschnitten/eingefügt und weitere Bearbeitungsschritte ausgeführt werden. Die Bearbeitungsschritte sind in derselben Symbol-Tabelle oder in unterschiedlichen Symbol-Tabellen möglich. Mit dieser Funktion können Symbol-Definitionen schnell von einer Datei in eine andere übertragen/kopiert werden.

Zu den Bereichen, die in der Symbol-Tabelle für Bearbeitungsschritte ausgewählt werden können, gehören:
- Symbol-Definitionen (ganze Reihe)
- Teile von Symbol-Definitionen Beispiel: - Ausschliesslich Symbol-Namen
- Mehrere Symbol-Definitionen
- Teilbereich der Tabelle Beispiel: - Ausschliesslich Symbol-Namen und Typen
- Symbol-Gruppen usw.

Um die Symbole zu bearbeiten, wählen Sie die Symbole bzw. Bereiche und führen einen Rechtsklick aus, um die Funktionen aus dem Kontextmenü aufzurufen. Gehen Sie zur entsprechenden Stelle in derselben Tabelle oder in einer anderen Datei und führen einen Rechtsklick aus, um die Funktionen aus dem Kontextmenü aufzurufen.

Beispiel: Symbol-Definitionen kopieren und einfügen

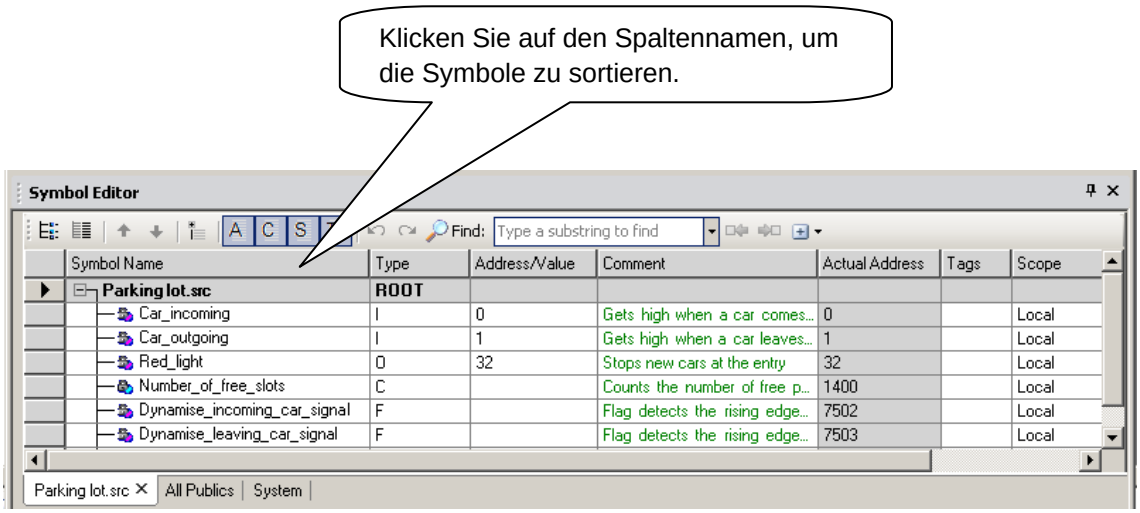




5.3.14 Sortieren der Symbol-Liste

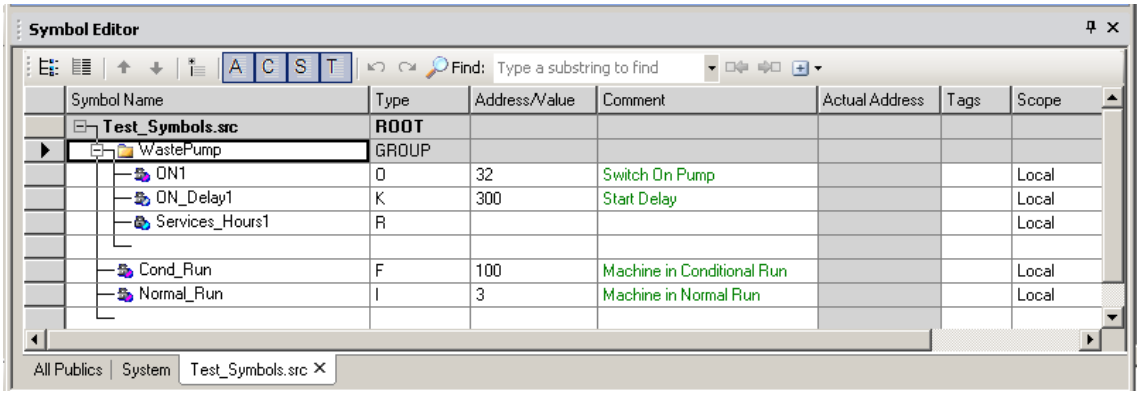
Die Symbol-Liste kann in der Symbol-Tabelle sortiert werden, indem Sie auf den Spaltennamen klicken. Symbol-Listen können nach Symbol-Namen, Typ, Adresse, Anmerkungen, tatsächlichem Wert, Tags oder Umfang sortiert werden.

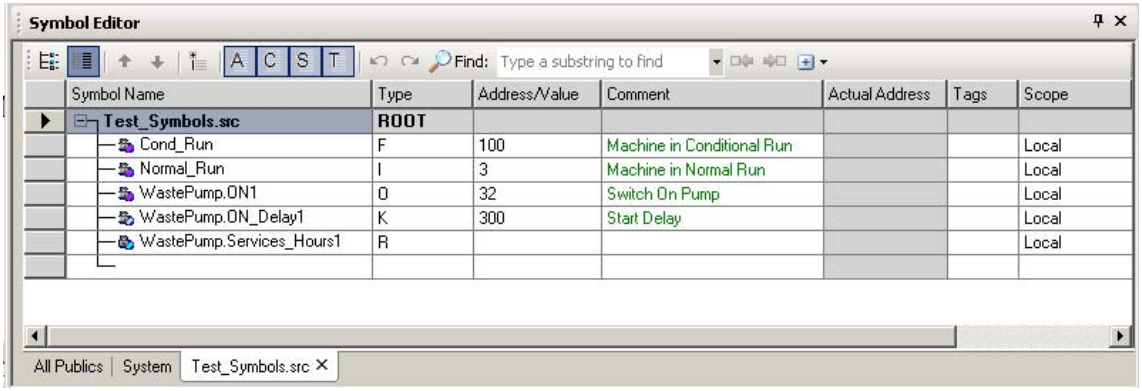
Nach Symbol-Namen sortierte Symbole:



Die Symbole können ebenfalls in *List view, Switch to List view* sortiert werden. Klicken Sie auf eine der Spalten, um sie zu sortieren: nach Namen, Typ, Adresse oder Anmerkung.

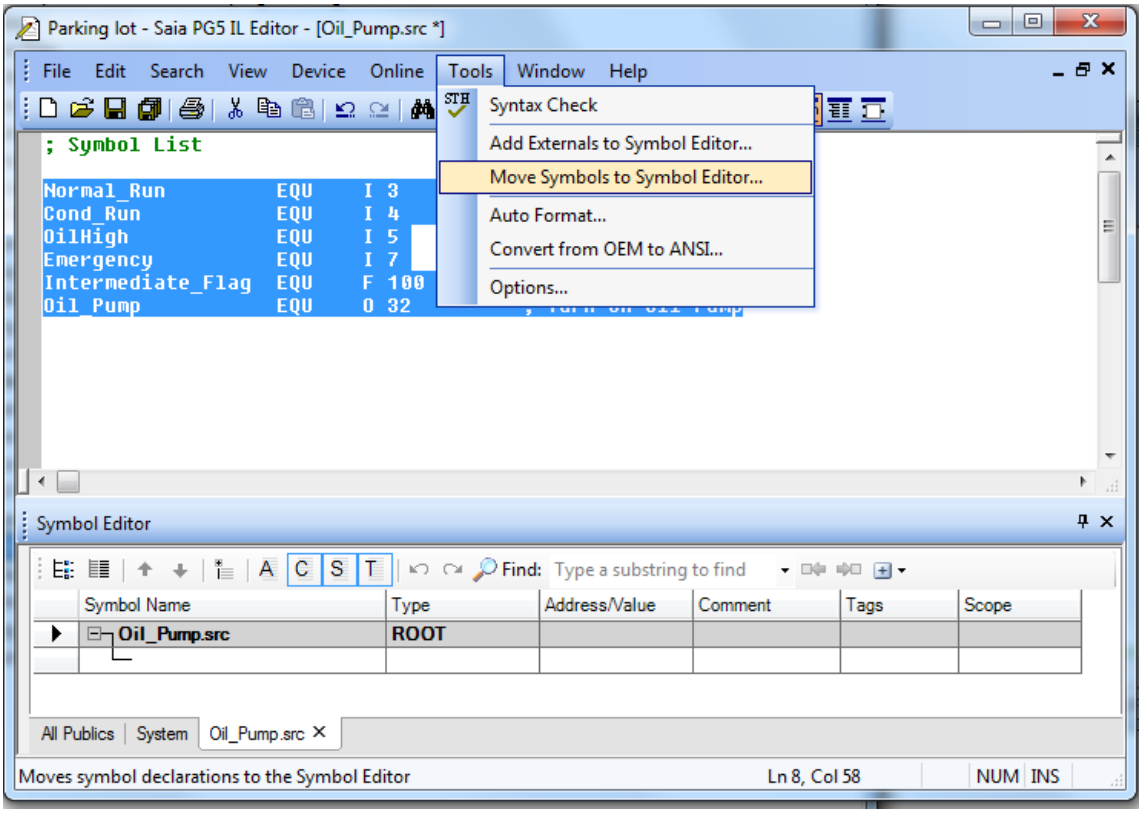
Gruppe/ Liste auswählen





5.3.15 Importieren von Symbolen aus den EQUATE-Angaben

Wenn Sie über alte PG4/3 Instruktionslistendateien mit EQU- oder DOC-Angaben verfügen, markieren Sie einfach die Angaben und importieren die entsprechenden Symbole über den Menüpfad: *Tools*, *Move Symbols to Symbol Editor*. Die Symbole werden dann von der Programmdatei in die Symbol-Liste verschoben.

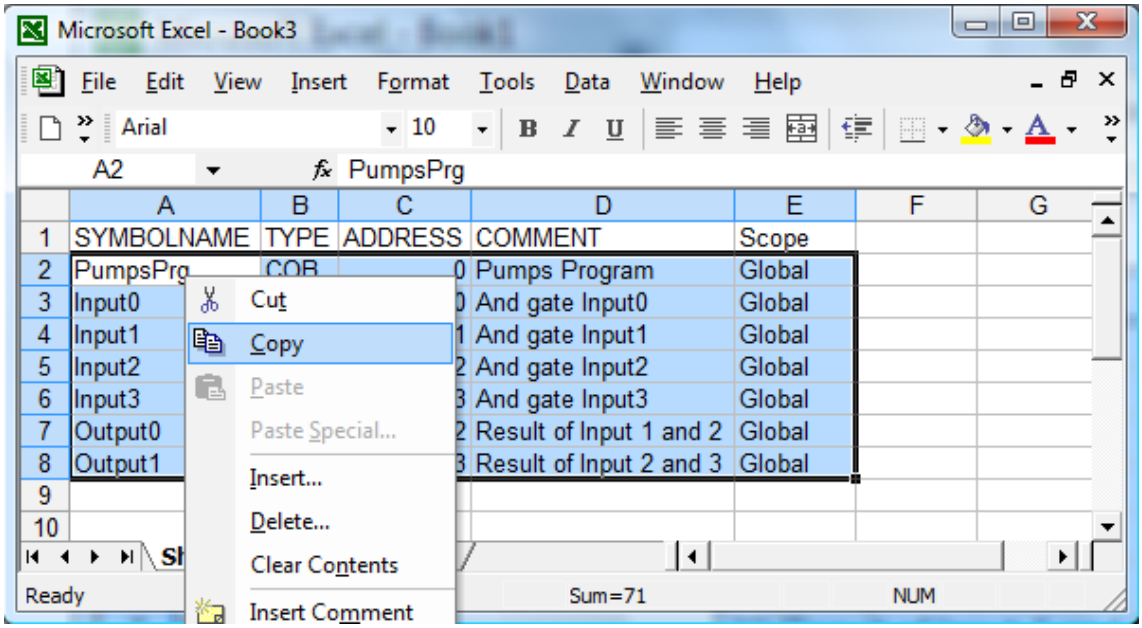


5.3.16 Importieren/Zusammenführen von Symbolen

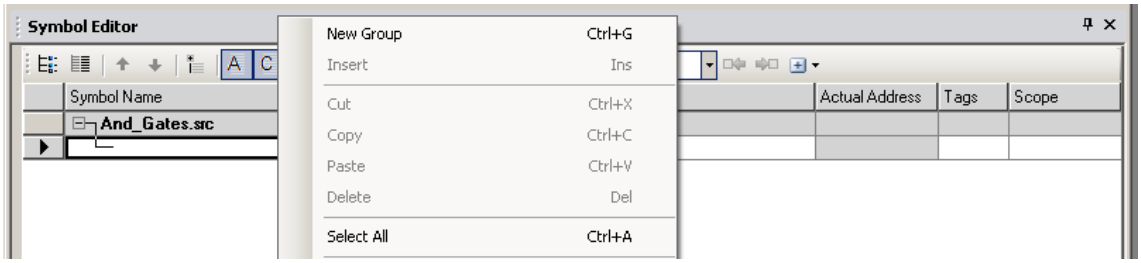
Symbole können aus einem anderen Programm (Electro CAD, VisiPlus, usw.) importiert und in Ihrem Projekt verwendet werden. So erhält Ihre Dokumentation für das ganze Projekt eine einheitliche Struktur, und die Bezeichnungen in ihren Plänen entsprechen dann denjenigen im Programmcode. Verwenden Sie einfach die Exportfunktion in CAD, um die Symbole in eine Textdatei zu exportieren und kopieren Sie dann die Symbole und fügen Sie sie im Symbol-Editor ein.

Das Kopieren/Einfügen von Symbolen aus anderen Programmen wie Excel, Word usw. kann für das Importieren/Zusammenführen von Symbolen verwendet werden. Schreiben Sie dazu eine Liste mit Symbolen in Excel, WinWord oder einen beliebigen anderen Text-Editor, kopieren Sie sie und fügen Sie die Liste direkt im Symbol-Editor ein. Bearbeiten Sie beispielsweise eine Symbol-Datei wie unten dargestellt, kopieren Sie die Symbol-Definition aus Excel und fügen Sie sie im Symbol-Editor ein. Wenn der Symbol-Editor bereits eine Symbol-Liste enthält, werden durch das Einfügen der Symbole am Ende der Tabelle die bestehenden Symbol-Definitionen nicht verändert oder überschrieben.

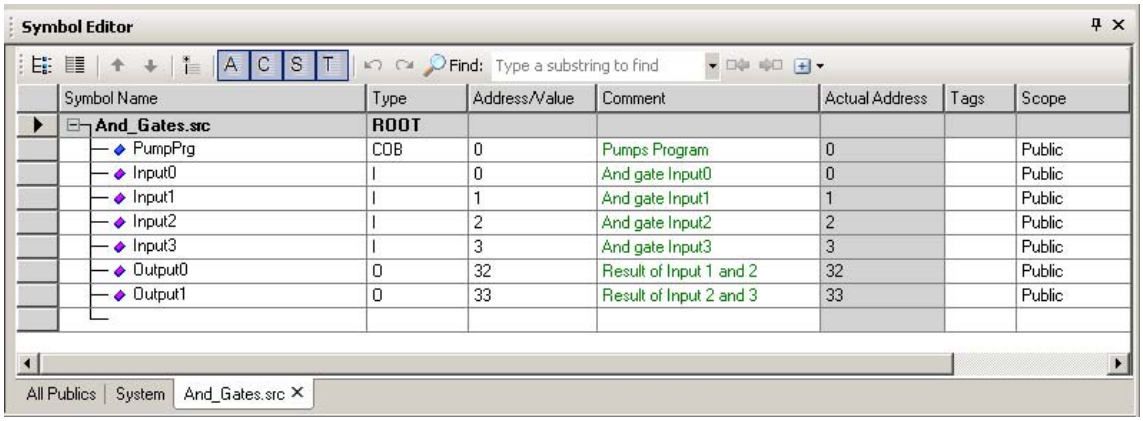
Beispiel: Kopieren und Einfügen eines Symbols aus einer Excel-Tabelle in den Symbol-Editor



Öffnen Sie mit einem Rechtsklick das Kontextmenü, um die Symbole einzufügen



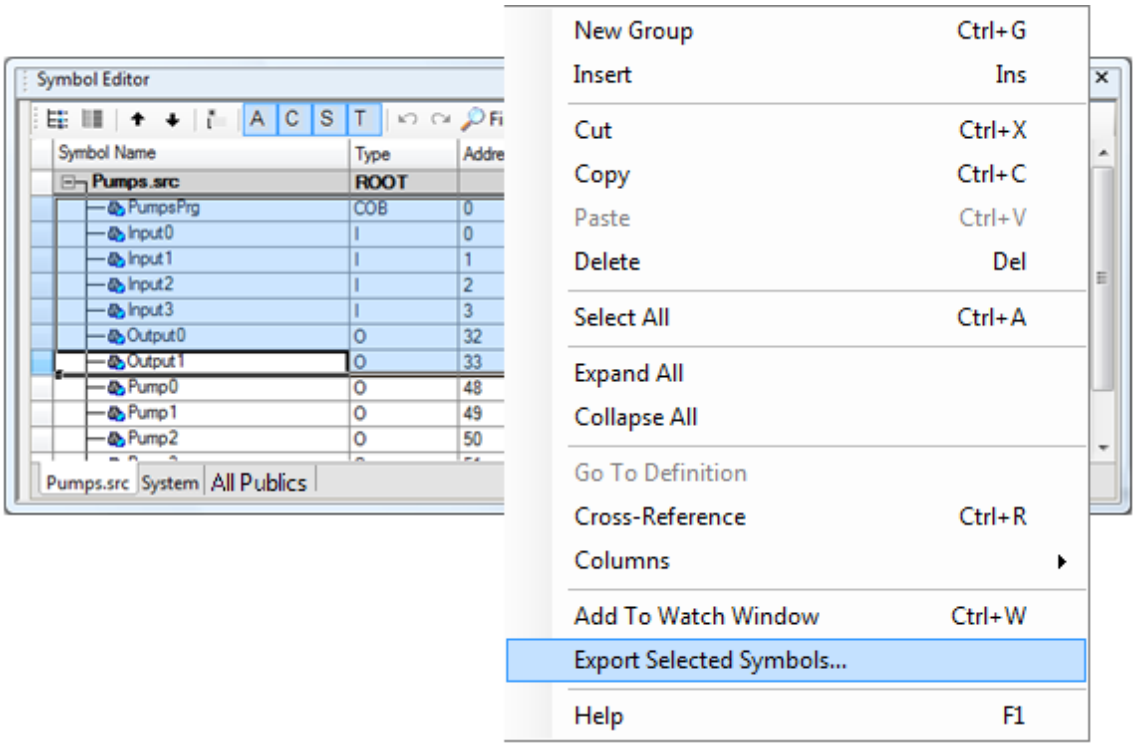
Symbole nach dem Einfügevorgang



5.3.17 Exportieren von Symbolen

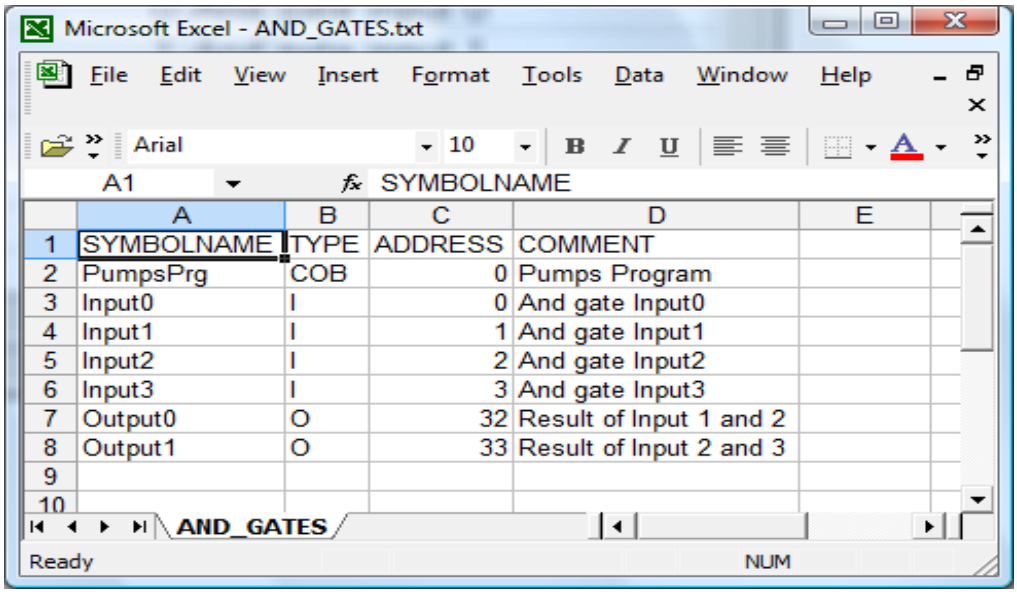
Die Symbol-Liste eines Programms kann in andere Anwendungen exportiert werden (z.B. Excel, VisiPlus oder Word), um beispielsweise den Betriebsbericht zu erstellen. Symbole können ebenfalls aus dem Symbol-Editor in andere Anwendungen wie Excel, Word oder einen Text-Editor kopiert und eingefügt werden.

Beispiel: Export von Symbolen in Excel:
Wählen Sie die zu exportierenden Symbole und dann aus dem Kontextmenü im Symbol-Editor die Option *Export Selected Symbols.*



Wenn Sie eine Symbol-Liste nach Excel exportieren möchten, empfehlen wir dringend das Format *Tab separated Text file (*.txt)*. So erhalten Sie bessere Ergebnisse als mit dem *Excel*-Dateiformat *(*.xls).*

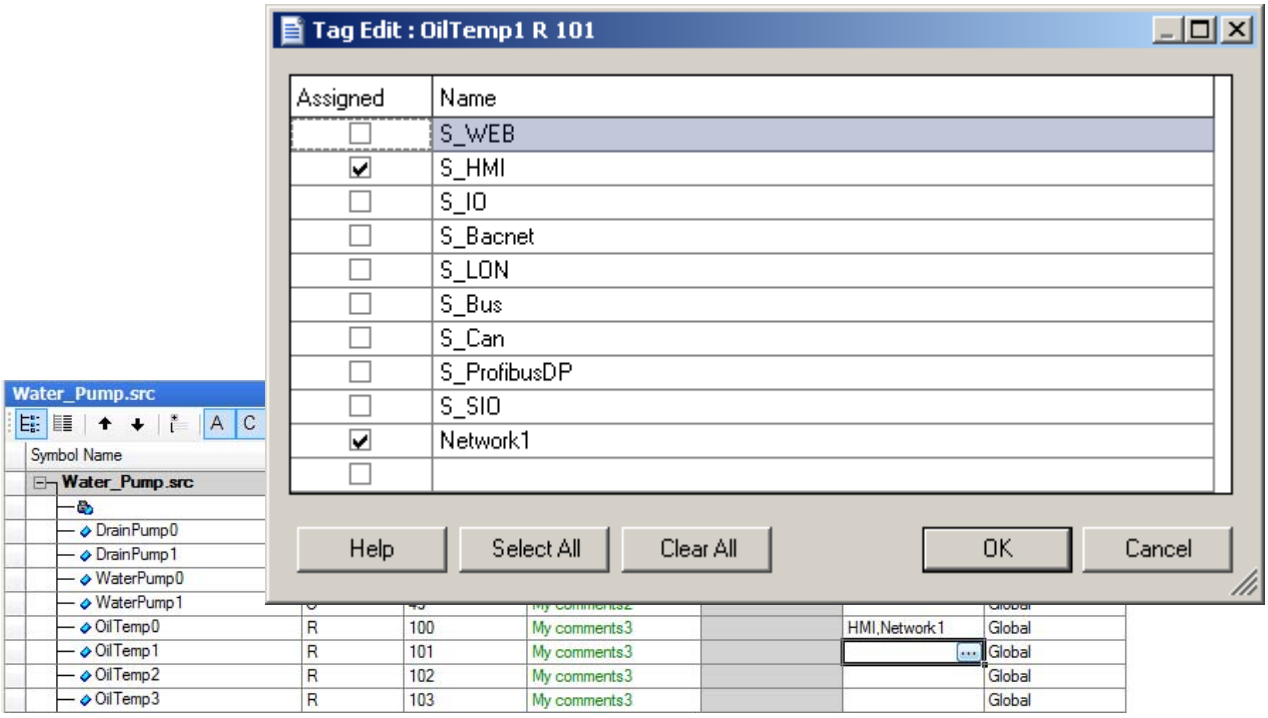
Starten Sie Excel und öffnen Sie die Textdatei mit den exportierten Symbolen.



5.3.18 Symbol-Tags

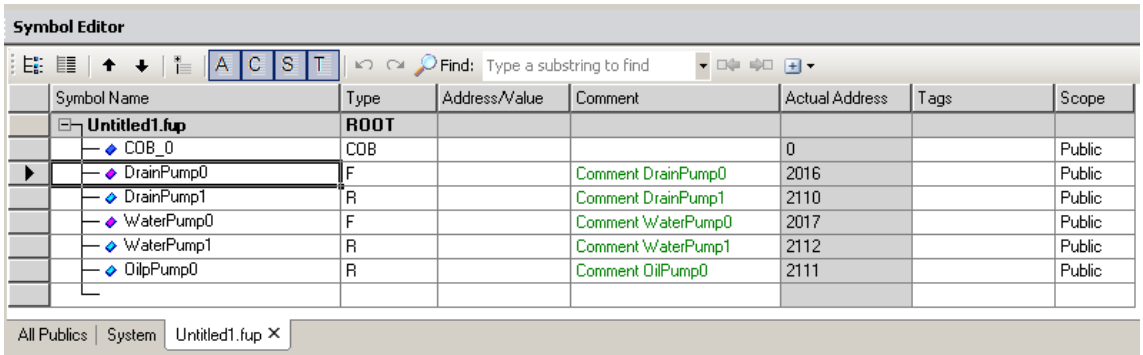
Tags können verwendet werden, um Symbole mit gemeinsamen Funktionen wie Network, HMI oder Supervision Symbols usw. auszuzeichnen. Neue Tags werden dem Symbol über die Auswahl in der Prüfliste zugewiesen. Neue Tags können aus der Tag-Zelle hinzugefügt werden. Mit Tags lassen sich gemeinsame Symbole in derselben Ansicht filtern. Damit können ausgewählte gemeinsame Symbole in andere Anwendungen wie Excel oder .rxp exportiert werden.

Tags beeinflussen das Programm nicht. Hierbei handelt es sich nur um ein Tool zur Symbol-Verwaltung. Um ein Tag zu bearbeiten, wählen Sie die Zelle aus und geben einen oder mehrere Namen mit einem Komma als Trenner ein oder öffnen Sie das Dialogfenster.



5.3.19 Tatsächlicher Wert

Nach dem Build-Prozess werden in der Spalte mit den tatsächlichen Werten die dynamischen Symbol-Adressen angezeigt. Der dynamische Adressbereich für die einzelnen Datentypen wird in den *Build Options* des Projekt Managers festgelegt.



5.3.20 Initialisieren von Symbolen

Symbole, die von der PCD verwendet werden, können auf zwei verschiedene Arten initialisiert werden:

- Initialisierung während eines PLC-Kaltstarts (Aufstarten)
- Initialisierung, wenn das Programm in die PCD geladen wird

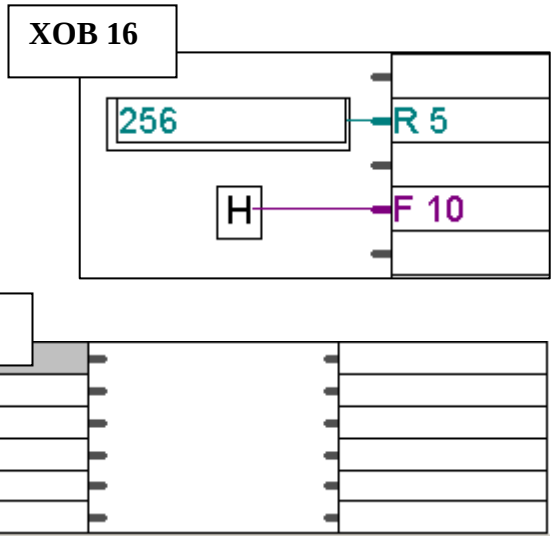
Während eines Kaltstarts
Die Initialisierung von Symbolen während eines Kaltstarts wird in XOB 16 vorgenommen. Dieser Funktionsblock wird nur einmal während eines PCD-Kaltstarts bearbeitet. Zur Initialisierung von Symbolen in XOB 16 muss entsprechender IL-Code geschrieben werden.

Beispiel: Initialisierung eines Flags und Registers während eines PCD-Kaltstarts

Programm in IL

```
XOB  16      ;Kaltstartblock
LD   R 5     ; R 5 = 256
     256
SET  F 10    ;F 10 = 1

EXOB

COB  0       ;zyklischer Blo
     0
…
;Ihr Programm
…
ECOB
```

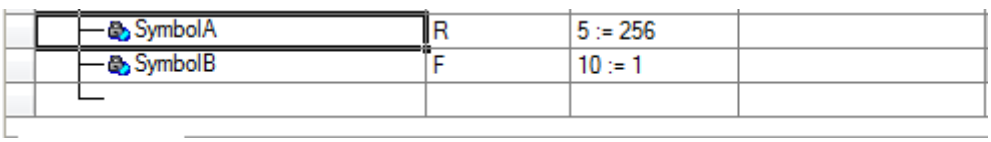
Programm in Fupla



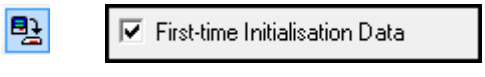
Weitere Einzelheiten zu COB- und XOB-Blöcken finden Sie in Kapitel 7 dieses Dokuments.

Beim Laden des Programms
Um ein Symbol zu initialisieren, wenn das Programm in die PCD geladen wird, muss auf die Adresse ein **:=** (Doppelpunkt, Gleichheitszeichen) folgen, auf das wiederum der Initialisierungswert folgt.

Beispiel:



Vorsicht:
Die folgende Option muss beim Laden des Programms aktiviert sein:



5.3.21 Reservierte Wörter

Die folgenden Wörter sind reserviert und können nicht als Symbol-Namen verwendet werden:

- Assembler-Befehle: PUBL, EXTN, EQU, DEF, LEQU, LDEF, MACRO, ENDM, EXITM…,
- Codebefehle und Abkürzungen der verschiedenen Datentypen der PCD: I, O, F, R, C, T, K, M, COB, FB, TEXT, X, SEMA, DB,
- Spezialbefehle MOV: N, Q, B, W, L, D,
- Konditionscodes: H, L, P, N, Z, E,
- Alle mnemotechnischen Befehle,
- Vorgegebene Symbole,
- Interne Symbole, die für die automatische Ressourcenzuweisung reserviert wurden, beginnen mit einem unterstrichenen Zeichen. Beispiel: _____TEXT, _____F
- Internes Symbol __CSTART__, wird mit $$ verwendet.

5.3.22 Fehler- und Warnmeldungen

Fehler- und Warnmeldungen werden angezeigt, wenn nicht zulässige Einträge im Symbol-Editor gemacht wurden. Es folgen einige Beispiele:

Symbol-Name ist zu kurz:

Wird nur ein Zeichen als Symbol-Name eingegeben, erscheint im Editor die Fehlermeldung „Symbol-Name zu kurz (min. 2 Zeichen)“.

Kopieren von Symbol-Namen:

Wird ein neues *Public* Symbol hinzugefügt, dessen Name bereits für ein anderes *Public* Symbol vergeben wurde, wird im Editor eine Fehlermeldung angezeigt und beide Symbole erscheinen in roter Schrift.

Wird ein neues lokales Symbol hinzugefügt, dessen Name bereits für ein anderes lokales Symbol im selben Modul bzw. in derselben Datei vergeben wurde, wird im Editor eine Fehlermeldung angezeigt und beide Symbole erscheinen in roter Schrift.

Wird ein neues lokales Symbol mit einem Namen hinzugefügt, der bereits für ein *Public* Symbol verwendet wurde, erscheint im Editor eine Warnung.

Um die Fehler- und Warnmeldungen auszublenden, muss einer der Symbol-Namen geändert werden.

Inhaltsverzeichnis

6 FUPLA-Programmierung

6.1 Einleitung

Fupla ist die einfachste und schnellste Möglichkeit, PCD Steuerungen zu programmieren.

Fupla, "*FUnction PLAn*" ist ein graphisches Programmierwerkzeug, welches dem Anwender erlaubt, ein Programm bzw. einen Prozess mit Hilfe von hunderten vordefinierter Grafischer Elemente (FBox, Function Box) zu zeichnen.
Diese FBoxen sind in verschiedenen Bibliotheken abgespeichert. Mit den in der Basis-Bibliothek enthaltenen Basis FBoxen kann bereits ein Grossteil der Programmfunktionen realisiert werden. Zusätzlich sind noch Bibliotheken für spezielle Anwendungsgebiete verfügbar. Die HLK Bibliothek für den Bereich Heizen, Lüften Klima; Die Modem Bibliothek enthält FBoxen für den Datenaustausch via Modem, (analog, ISDN, GSM, GPRS), die Handhabung von SMS, Pager und DTMF Meldungen. Verschiedene andere Bibliotheken wie z.B. für LON oder EIB Kommunikation, für die Anbindung von Belimo Produkten ermöglichen es, praktisch jede Applikation mit Fupla zu realisieren.

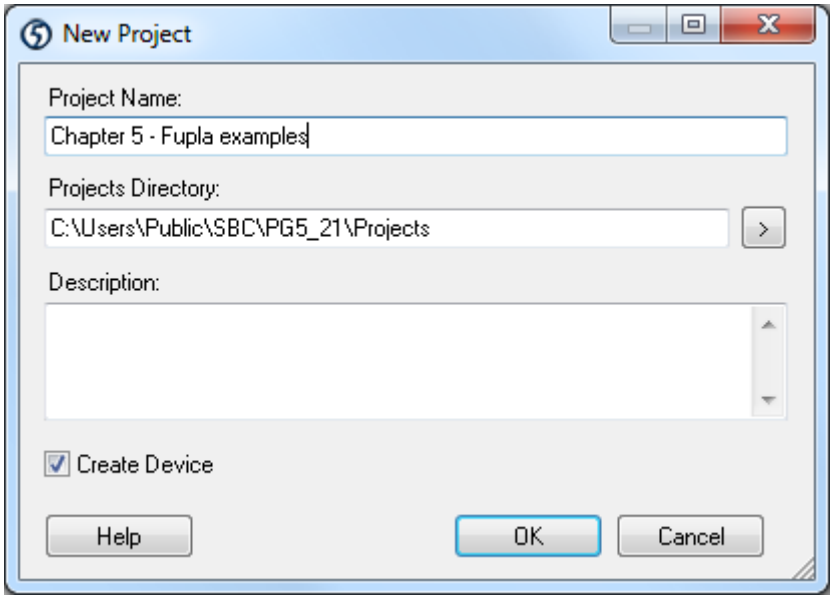
Der grosse Vorteil von Fupla liegt darin, dass der Anwender ein PCD Programm erstellen kann, ohne dass er eine Linie Programmcode schreiben muss, und dass er etwas über die Programmcode kennen muss.

6.2 Erstellen eines Fupla Projektes

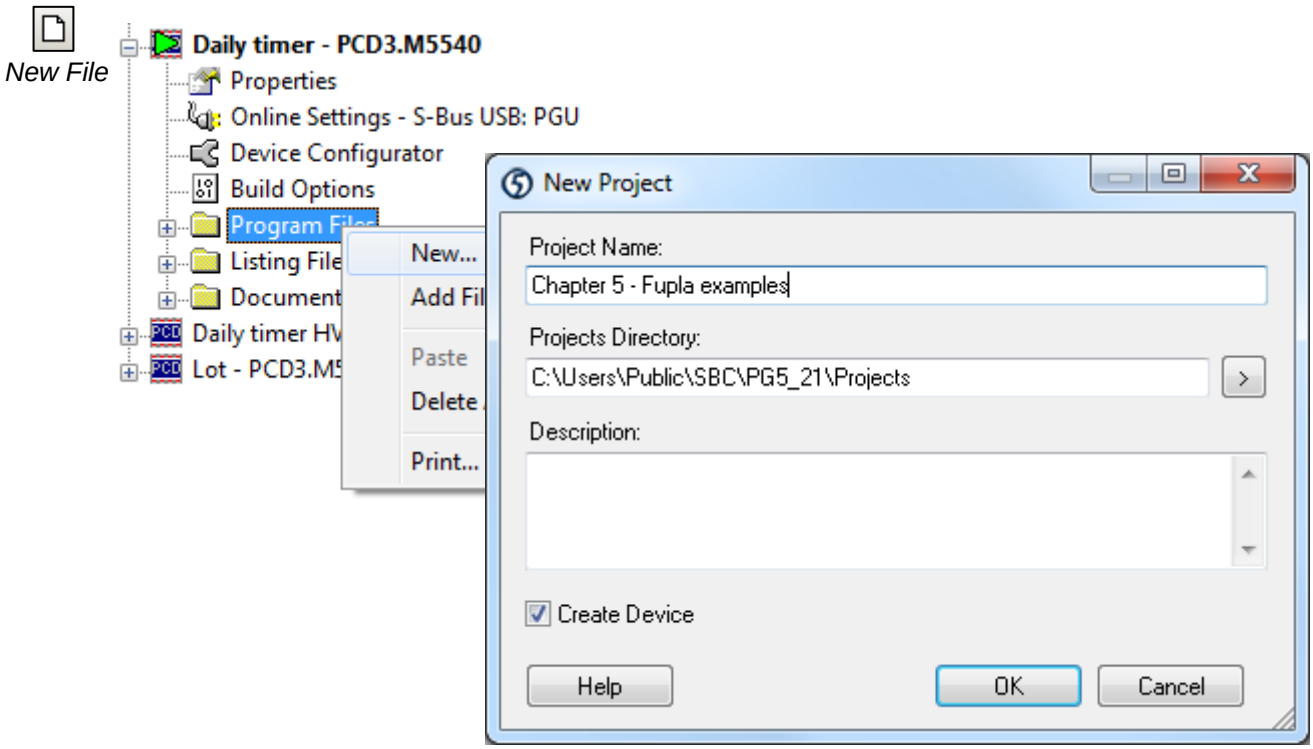
Das Fupla Beispiel wird in einem neuen PG5 Projekt abgespeichert.

6.2.1 Neues Projekt erstellen

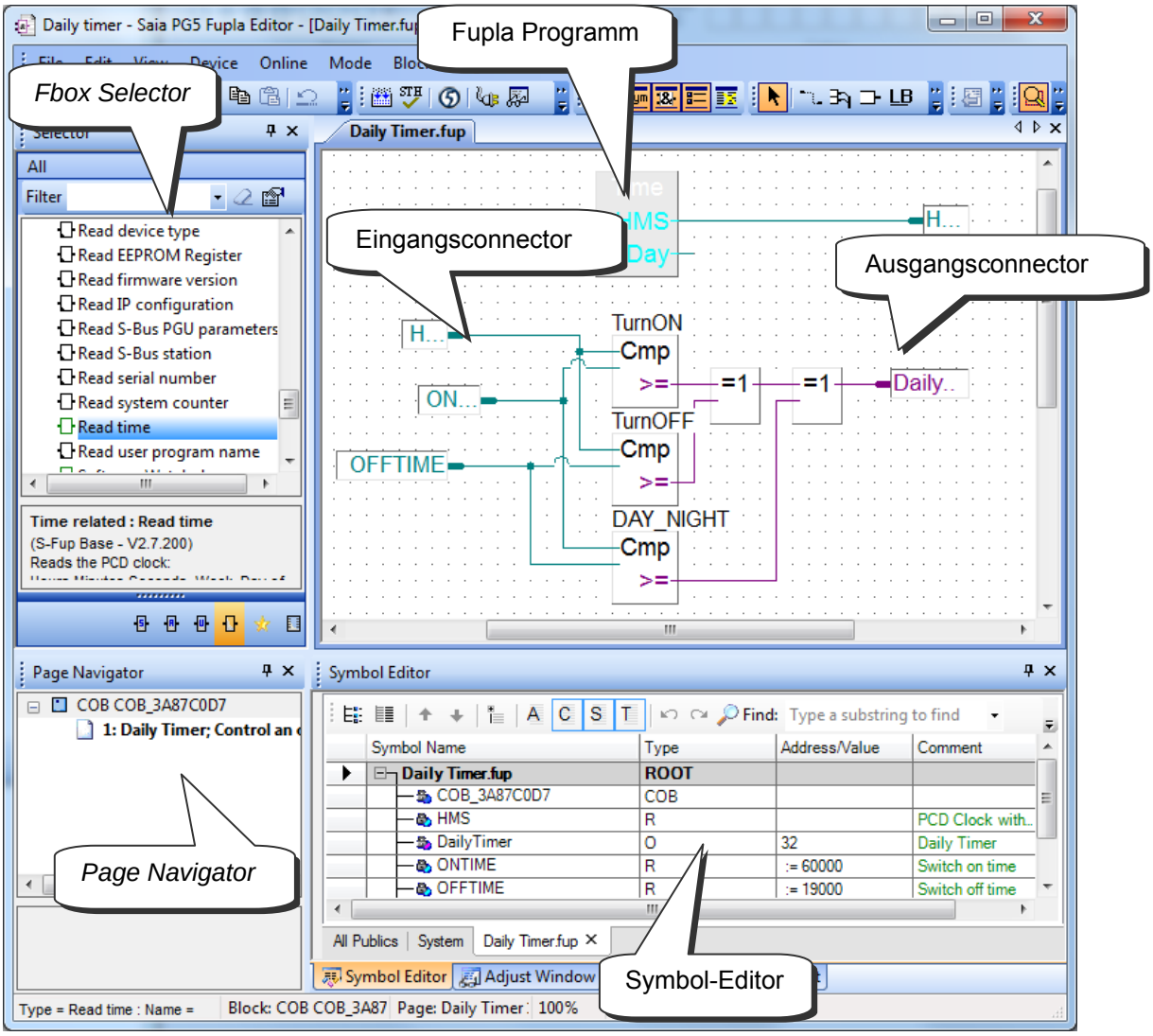
Im *SAIA PG5 Project Manager* Fenster, Auswahl von *File, Project, New…* und neues Projekt erstellen.



Zum Erstellen einer neuen Programmdatei ist "New File" oder die rechte Maustaste anzuklicken.



6.3 Aufbau einer Fupla Seite



Abarbeitungsweise eines Fupla Programmes

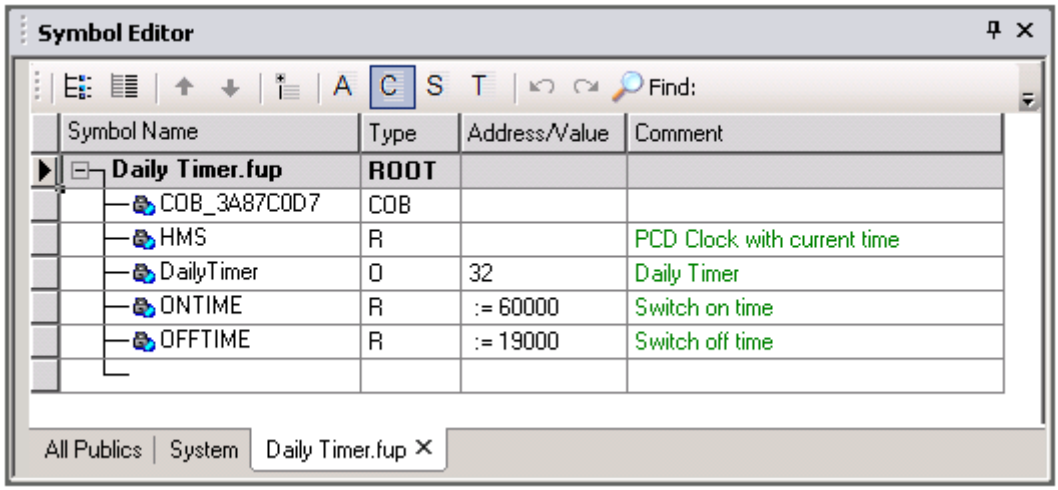
Die PCD liest die Informationen welche durch die, am Rand „Eingang“ dargestellten Eingangs- Symbole des Fupla Programmeditors definiert sind, verknüpft diese gemäss dem Programm und schreibt das Ergebnis auf die Ausgangs-Symbole am Rand „Ausgang“. Die verwendeten Symbole werden im Symbol Editor aufgelistet. Ausser den Datentypen *input* und *constant* können alle Datentypen als Eingangs- oder Ausgangs Symbole verwendet werden. Die Datentypen *input* und *constant* können nicht beschrieben werden und dürfen deshalb nur im Rand „Eingang“ verwendet werden.

In der Mitte des Fupla Programmeditors ist das eigentliche Programm welches aus verschiedenen grafischen Funktionen (FBoxen) besteht. Die FBoxen werden aus dem *FBox selector* Fenster ausgewählt. Die Verbindungslinien zwischen den FBoxen stellen die auszutauschenden Daten zwischen den FBoxen dar. Die Farbe der Verbindungslinien ist abhängig vom Datentyp. Binäre Verbindungen (Boolean) sind rot, Ganzzahlige (Integer) sind blau und Fliesskomma (floating-point) sind gelb. Unterschiedliche Datentypen können grafisch nur miteinander verbunden werden, wenn sie zuvor zu einem einheitlichen Datentypen konvertiert worden sind. (FBox: *Wandler*).

Bei Programmen mit mehreren Fupla Seiten kann mit dem *Page Navigator* Fenster, rasch zwischen den Fupla Seiten gewechselt oder die Programmstruktur navigiert werden.

6.4 Symbols-Fenster

Show Hide Symbol Editor



Das *Symbols Editor* Fenster enthält eine Liste aller Symbole eines Programms. Sie kann mit dem *Show/Hide Symbol Editor* Knopf oder mit dem Menü-*Befehl View/Symbol Editor* aufgerufen werden. Jede Zeile zeigt alle Informationen die zu einem bestimmten Eingang, Ausgang, Register gehören und bildet gleichzeitig ein Symbol:

Symbol Name
Ein Symbol ist ein Name der die Adresse eines Eingangs, Ausgangs, Flags, Registers…bezeichnet Es ist ratsam Symbol-Namen innerhalb eines Programms zu verwenden und nicht die absolute Adresse eines Flags oder Registers. So kann eine Adresse oder ein Datentyp im *Symbols*-Fenster leicht abgeändert werden. Anstatt die Änderung an jeder betroffene Stelle im geändert werden. Das Risiko, eine Stelle im Rand zu vergessen oder einen schwer auffindbaren Fehler zu produzieren, entfällt.

Syntax für Symbol-Namen
Das erste Zeichen ist immer ein Buchstabe, gefolgt von weiteren Buchstaben, Ziffern, oder dem Underscore-Zeichen. Sonderzeichen (ö,è,ç,…) sind zu vermeiden. Gross/Kleinschreibung hat keinen Einfluss: MotorOn und MOTORON ergeben die gleichen Symbole.

Type
Bestimmt den Typ des Symbols Input(I), Output(O), Register(R), Counter(C), Timer(T), text(X), DB, …

Address/Value
Jeder Symbol-Typ besitzt einen bestimmten Bereich verfügbarer Adressen:
Ein-/Ausgänge: abhängig von den in der PCD gesteckten Modulen

Flags:	F	0…16383
Register:	R	0…16383
Timer/Counter:	T/C	0…1599

Comment
Dieser Kommentar ist mit seinem Symbol verbunden und kann auf der Fupla-Seite angezeigt werden. Platzieren Sie die Maus auf dem Connector, um seine vollständige Symboldefinition in einer Textfahne anzuzeigen.

```
STH    Flag                ;Copy the Flag state into the accu
       STH    Flag                ;Control the incrementation
```

Scope

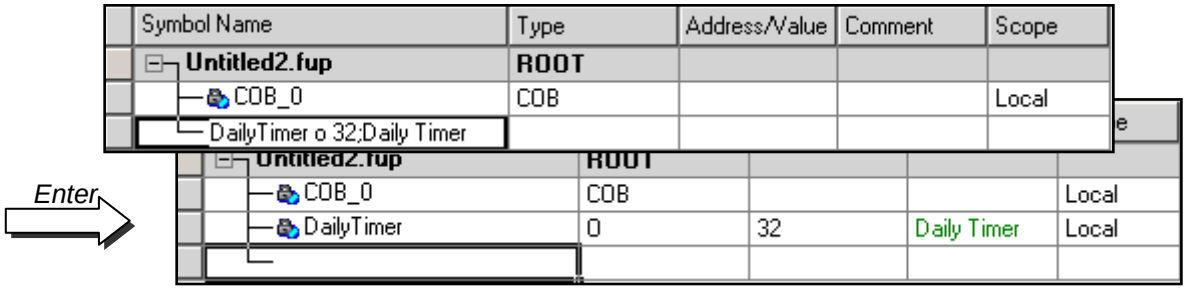
Das Feld *Scope* definiert die Zugänglichkeit des Symbols.*Local* = Das Symbol wird nur in der aktuellen Datei verwendet. *Public* = Es kann von anderen Dateien aus auf das Symbol zugegriffen werden, wenn das Symbol dort als *Public* ausgewiesen ist.*External* = Das Symbol ist in einer anderen Datei als *Public* definiert.

6.4.1

6.4.2 Neue Symbole der *Symbols* Liste hinzufügen

Einfache Methode:
Klicken Sie auf das Fenster *Symbol Editor*, um der Liste ein Symbol hinzuzufügen.Es steht immer eine leere Zeile zur Verfügung. Wählen Sie die Felder *Symbol Name*, *Type*, *Address/Value, Comment* und *Scope* aus und füllen Sie diese aus. Mit der Tabulator- oder der Enter-Taste gelangen Sie in das nächste Feld.

Schnelle Methode 1:

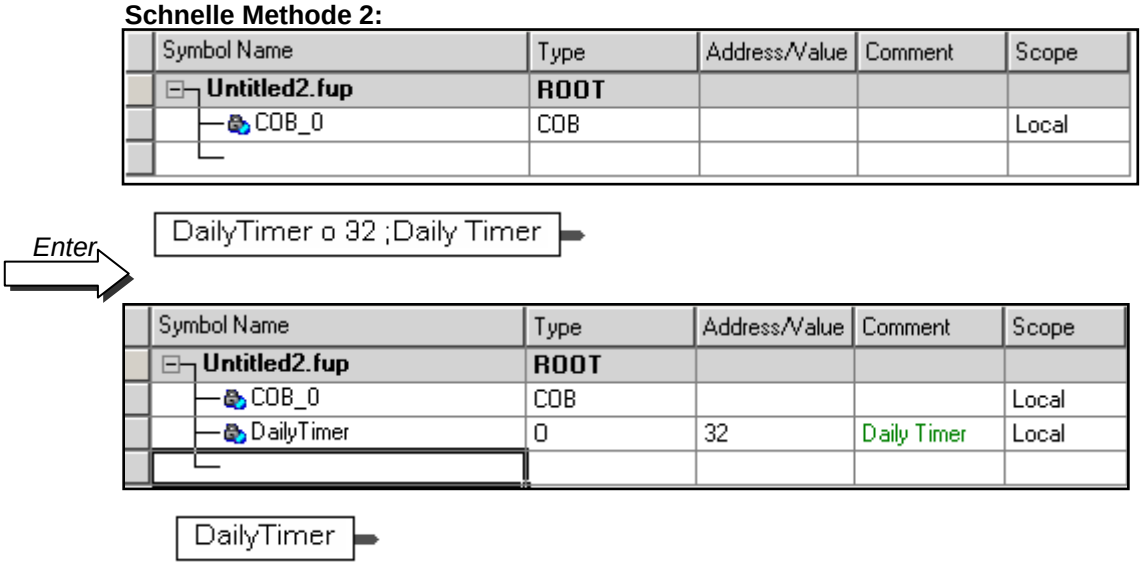


Eine weitere Möglichkeit ist die Eingabe von Variablen für die verschiedenen Informations- Felder vom *Symbol Name* Feld. Das ist praktischer und schneller. Siehe Beispiel unten.

Folgende Syntax einhalten: symbol_name, type, address; comment

Wurde das neue Symbol nach obiger Syntax festgelegt, *Enter* Taste auf der Tastatur betätigen und die Informationen werden automatisch in die richtigen Felder geschrieben.

Schnelle Methode 2:



Neue Symbole könne auch während des Programmierens hinzugefügt werden. Dazu ds Symbol im Rand mit Mnemo-Kode und Operand editieren. Für den Operanden Symbol-Name und Definition nach folgender Syntax eingeben:
symbol_name, type, address; comment

Enter Taste auf der Tastatur betätigen und das neue Symbol wird automatisch der *Symbols* Liste hinzugefügt.

6.4.3 Symbolen Adressierung

Eine Symbol-Definition muss nicht unbedingt all die Information wie unten enthalten. Drei Typen der Adressierung werden unterschieden:

Absolute addresses

Symbol Name	Type	Address/Value	Comment	Scope
Untitled2.fup	**ROOT**			
	0	32	Daily Timer	Local

Es sind nur der Typ und die Adresse (z.B. 32) definiert und optional ein Kommentar. Wird direkt im Programm die absolute Adresse benutzt, ist dies beim ändern von Typ oder Adresse von Nachteil. Das Anwenderprogramm übernimmt die in der Symbol-Liste durchgeführten Änderungen nicht. Änderungen müssen manuell an jeder betroffenen Stelle im Rand durchgeführt werden. Deshalb sind Symbol-Namen vorzuziehen, optional mit dynamischer Adressierung.

Symbol-Namen

Symbol Name	Type	Address/Value	Comment	Scope
Untitled2.fup	**ROOT**			
DailyTimer	0	32	Daily Timer	Local

Die Daten sind durch Symbol-Name, Typ, Adresse und optionalem Kommentar definiert. Änderung des Symbols, Typs oder der Adresse wird von der Symbol-Liste unterstützt und jede betroffene Stelle im Rand wird automatisch geändert.

Dynamische Adressierung

Symbol Name	Type	Address/Value	Comment	Scope
Daily Timer.fup	**ROOT**			
HMS	R		PCD Clock w...	Local

Dies ist eine Form von symbolischer Adressierung bei der die Adresse nicht definiert wird. Die Adresse wird automatisch beim Verarbeiten (build) des Programms hinzugefügt. Die Adresse wird von einem Adressbereich genommen, der in den *Software Settings* definiert ist. (Siehe Project Manager.)

Bemerkung:
Dynamische Adressierung kann für Flags, Counter, Timer, Register,Texte, DBs, COBs, PBs, FBs und SBs angewendet werden. Eingänge, Ausgänge und XOBs müssen jedoch absolut adressiert werden.

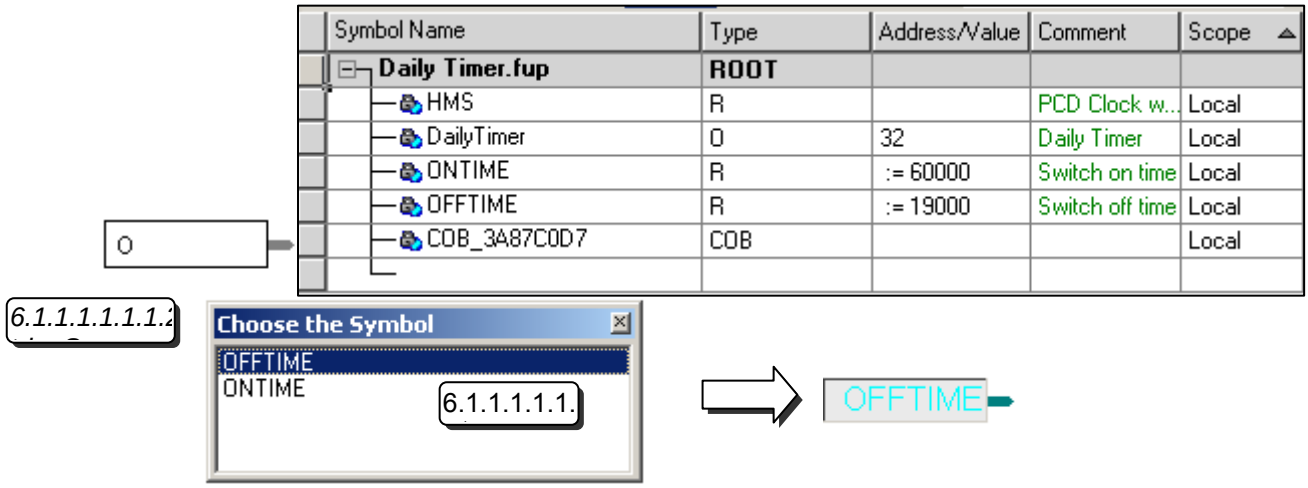
Verwenden Sie die dynamische Adressierung nicht für Daten, auf die über ein Netzwerk oder ein Leitsystem zugegriffen wird, da sich die dynamischen Adressen beim Wiederaufbau eines Programmes gelegentlich verändern.

6.4.4 Benutzen der Symbole aus der *Symbols*-Liste im Fupla Programm

Nach erfolgter Programmierung können die im *Symbols Editor* Fenster festgelegten Symbole auf verschiedene Arten benutzt werden:

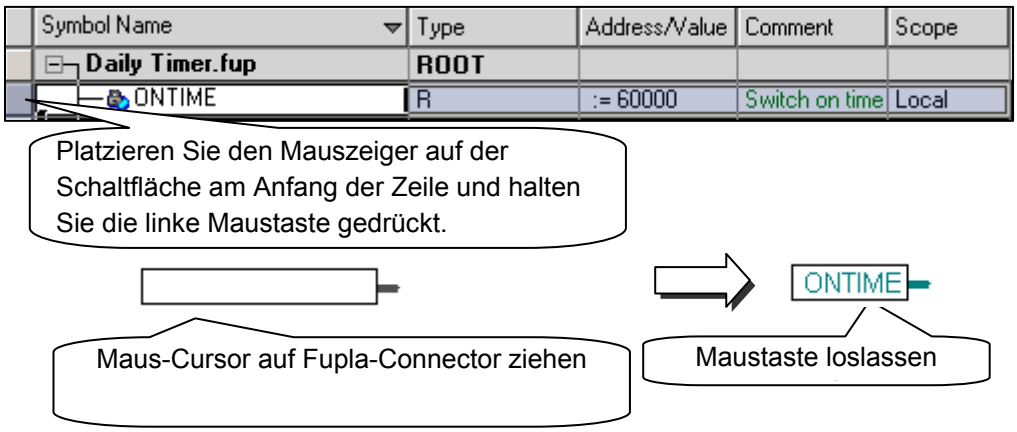
Symbol eingeben über die Tastatur
Der Symbol-Name wird für jede Instruktion, die diesen benutzt, vollständig über die Tastatur eingegeben. Diese Methode birgt die Gefahr von Schreibfehlern, die erst bei der Verarbeitung des Programms bemerkt werden.

Symbol eingeben durch selektive Suche



Nur einige wenige Zeichen des Symbol-Namens auf der Tastatur eingeben und dann die *Ctrl+Space* Tasten drücken hat zur Folge, dass ein Fenster mit einer Liste all der Symbole erscheint, die mit der eingegebenen Buchstabenkombination beginnen. Das benötigte Symbol kann mit der Maus oder den Pfeiltasten auf der Tastatur ausgewählt (↑ ↓) und mit *Enter* übernommen werden.

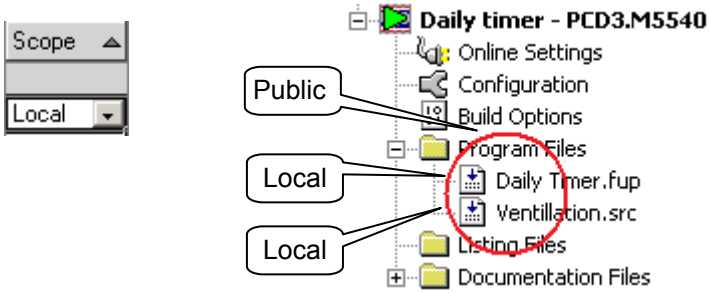
Symbol eingeben mit Drag-and-Drop



Diese Methode der Symbol-Eingabe schliesst alle Schreibfehler aus. Im *Symbols*-Fenster den Maus-Cursor auf der Zeile der Symbol-Definition positionieren, linke Maustaste drücken und gedrückt halten. Ziehen Sie den Mauszeiger über einen leeren Connector und lassen Sie die Maustaste los. Das ausgewählte Symbol wird automatisch zu dem durch den Mauszeiger markierten Connector hinzugefügt.

Sie können das Symbol auch auf einen leeren Platz auf der Fupla-Seite ziehen. Dadurch können Connector und Symbol automatisch in einem einzigen Arbeitsschritt hinzugefügt werden.

6.4.5 Local, Public und External Symbole



Die Zugänglichkeit eines Symbols wird über dessen Reichweite definiert:

Lokale Symbole sind nur innerhalb der Datei zugänglich, in der die Definition des Symbols enthalten ist. So sind beispielsweise *Public* Symbole nur in *Daily timer.fup* aus allen Dateien im *device* zugänglich.

Public Symbole werden von den beiden Dateien *Parking lot.fup* und *Ventilation.src* im *Device Daily timer* genutzt, unabhängig davon, welche Datei das Symbol definiert.

Normalerweise werden im Symbol Editor drei Reiter angezeigt: ein Reiter mit dem Namen der geöffneten Fupla-Datei, ein Reiter namens *All Publics* und ein Reiter namens *System*.

In dem Reiter mit dem Namen der geöffneten Datei können sämtliche in der Datei verwendeten Symbole bearbeitet werden. Die Definitionen der public und lokalen Symbole sowie Referenzen auf externe Symbole werden ebenfalls auf dieser Seite vorgenommen und in einer Datei, beispielsweise *Daily Timer.fup*, gespeichert. Das Feld *Scope* definiert, ob ein Symbol *Local*, *Public* oder *External* ist:

In Fupla sind neue, mit dem Symbol Editor oder dem Fupla Editor erstellte Symbole standardmässig entweder *Local* oder *Public*. Das ist davon abhängig, welche Option unter *View, Options, Symbols, Add symbols with Public scope* in Fupla definiert wurde.

Der Reiter *All Publics* zeigt alle public Symbole im Device an. Der Reiter *System* zeigt alle Systemsymbole an. Die Symbole auf den Seiten *All Publics* und *System* werden aktualisiert, wenn eine Datei gespeichert wird oder ein *Build* beendet wurde. Diese Seiten nutzen die Ergebnisse des Build, um alle public und Systemsymbole aus allen Dateien im Programm des Device zu sammeln und sie alle in einer Tabelle anzuzeigen.

Soll ein public oder ein Systemsymbol im Programm platziert werden, wählen Sie das Symbol auf der Seite aus und ziehen Sie es mit der Drag-and-Drop-Funktion auf die Seite. Der Verweis auf das public Symbol wird auf der Symbolseite der Datei mit der Reichweite *External* angegeben. Damit wird angezeigt, dass das Symbol in einer anderen Datei definiert ist.

Die Symbole auf der Seite *All Publics* können nicht bearbeitet werden. Die Definitionen public Symbole können nur innerhalb der Datei bearbeitet werden, in der sie definiert sind. Mit dem Befehl *Goto Definition* im Kontextmenü können Sie die Datei öffnen, die Symbol definiert. Die Spalte *File* zeigt den Namen der Datei an, in

der das Symbol definiert wird. Diese Datei muss zur Veränderung des Symbols geöffnet werden.

6.5 Editieren der Connectors

Die Eingangs- und Ausgangsconnectors können an einer beliebigen Stelle auf den Fupla-Seiten platziert werden und können die Symbole enthalten, die für die von den FBoxes beschriebenen Programmfunktionen notwendig sind.

Standardmässig kann jede neue Seite bereits über Ränder mit Connectors links und rechts verfügen. Wenn Sie neue Seiten lieber ohne diese Connectors anzeigen lassen wollen, so dass Sie die Connectors nach eigenem Belieben setzen können, deaktivieren Sie bitte die entsprechende Option im Menü: *View, Options…, Workspace , New pages with side connectors.*

Um Connectors zu entfernen, die sich am linken oder rechten Rand der Seite befinden, wählen Sie folgendes Menü: *Page, Remove Unused Connectors.*

Um Connectors erneut auf einer leeren Seite zu platzieren, wählen Sie folgendes Menü: *Page, Add Side Connectors*.

6.5.1 Platzierung der Connectors

Add Connector

Um einen Connector und sein Symbol zu einer Fupla-Seite hinzuzufügen, drücken Sie die Taste *Place Connectors* auf der Werkzeugleiste und platzieren den Mauszeiger auf der Fupla-Seite. Ein „Read“-Eingangsconnector kann durch Drücken der linken Maustaste hinzugefügt werden. Ein „Write“-Ausgangsconnector kann durch gleichzeitiges Drücken der Umschalttaste und der linken Maustaste hinzugefügt werden. Der Connector, den Sie gerade hinzugefügt haben, ist einsatzbereit. Ihm kann jetzt ein Symbol zugewiesen werden. Im Connector wird ein Mauszeiger angezeigt. Wenn Sie das Symbol im Connector nicht sofort bearbeiten möchten, drücken Sie die Escape-Taste und platzieren den nächsten Connector.

6.5.2 Editieren des Symbols in einem Connector

Um ein bereits auf der Fupla-Seite vorhandenes Connectorsymbol zu editieren oder zu ändern, wählen Sie den Connector mit einem Doppelklick aus. Im Connector wird nun ein Mauszeiger angezeigt. Sie können nun das Symbol einschliesslich seiner vollständigen Definition eingeben.

Beachten Sie bitte, dass neu in Connectors eingegebene Symbole automatisch zur Symbolliste hinzugefügt werden, die im *Symbols Editor* Fenster angezeigt wird.

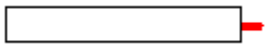
6.5.3 Schnellverfahren für die Platzierung eines Symbols und seines Connectors

Symbole, die bereits im *Symbols* Fenster vorhanden sind, können auf einen leeren Platz auf der Fupla-Seite gezogen werden. Dadurch entsteht ein neuer Connector, der dieses Symbol enthält.
Wenn das Symbol an einen FBox-Eingang oder -Ausgang gezogen wird, wird ein Eingangs- oder Ausgangsconnector direkt mit der FBox verbunden.

6.5.4 Ziehen, Kopieren/Einfügen, Löschen eines Symbols

Die Auswahl des rot markierten Bereichs betrifft nur das Symbol. Sie können das Symbol mit der Maus auswählen und es in einen anderen Connector ziehen, kopieren/einfügen oder löschen. Durch Drücken der rechten Maustaste wird ein Kontextmenü mit allen verfügbaren Optionen angezeigt.

6.5.5 Kopieren/Einfügen, Löschen eines Connectors

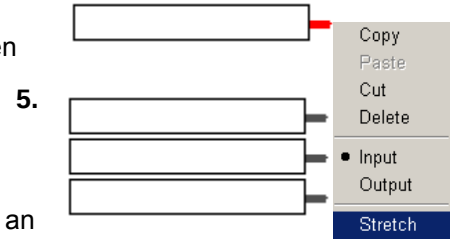


Die Auswahl des weiss markierten Bereichs betrifft den Connector und das enthaltene Symbol. Durch Drücken der rechten Maustaste wird ein Kontextmenü mit allen verfügbaren Optionen angezeigt.

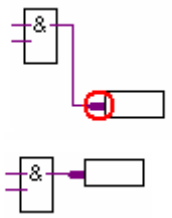
6.5.6 Strecken von Connectors

Connectors können gestreckt werden. Das bedeutet, dass die Anzahl an Connectors durch ein vertikales Ziehen des Mauszeigers bestimmt werden kann.

Wählen Sie die Taste *Select Mode.*
Wählen Sie einen Connector in einem rot markierten Bereich.
Rufen Sie das Kontextmenü durch Drücken der rechten Maustaste auf.
Auswahlmenü: *Stretch*
Ziehen Sie den Mauszeiger vertikal, um die Anzahl an Connectors zu bestimmen.
Drücken Sie die linke Maustaste.

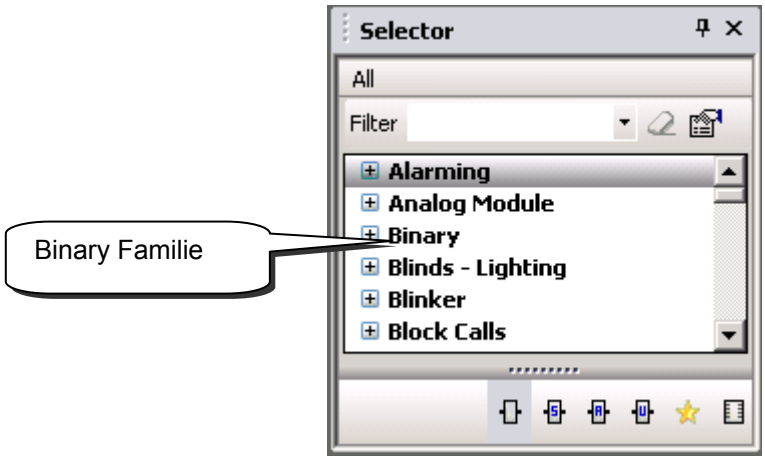


6.5.7 Vertikale Verschiebung eines Connectors



Um den Connector zu verschieben, platzieren Sie den Mauszeiger auf dem roten Kreis.
Drücken Sie die Umschalttaste und halten Sie sie gedrückt.
Drücken Sie die linke Maustaste und halten Sie sie gedrückt.
Ziehen Sie den Mauszeiger vertikal auf einen leeren Platz auf der Seite.
Lassen Sie Maus- und Umschalttaste los.

6.6 Fbox Selector



Show/Hide Selector Window

Das Fenster FBox *Selector* zeigt alle FBoxen, die in den verfügbaren FBox-Bibliotheken enthalten sind, an. Sie sind in *Families* mit ähnlichen Anwendungsdomainen organisiert. Hier sind einige der Hauptfamilien:

Binary	FBoxen zur Lösung logischer Gleichungen
Integer	Ganzzahlige Grundrechenarten
Floating Point	Grundrechenarten mit Gleitkommazahlen
Counter	Zählen von Aufgaben
Time related	Zeitabhängige Aufgaben
Analogue Module	Steuerung analoger Module
Communication	Datenaustausch in S-Bus- oder Ethernet-.Netzwerken
Converter	Umwandlung von binär in integer, integer in Gleitkomma usw.

…

Die FBox-Bibliotheken bieten fast alle Arbeitsprozesse, die das Programm braucht. Da es zahlreiche *Families* und *FBoxes* gibt, kann es schwierig werden, die richtige zu finden. Daher stehen Ihnen verschiedene Suchfunktionen zur Verfügung.

Ist eine Familie von FBoxen ausgewählt, gelangt man durch das Drücken einer Buchstabentaste automatisch zu der Familie, die mit dem entsprechenden Buchstaben beginnt. Ist ein Familienzweig geöffnet, gelangt man durch das Drücken einer Buchstabentaste automatisch zur nächsten FBox in dieser Familie, die mit dem entsprechenden Buchstaben beginnt.

In der Wekzeugleiste des *Selector*-Fensters gibt es ein Feld *Filter*, in das ein Filterstring eingegeben werden kann. Wenn Sie beispielsweise ADD eingeben und die *Enter*taste drücken, werden im Fenster *Selector* nur FBoxen angezeigt, die das Schlagwort ADD enthalten, nämlich Floating Point und Integer. Klicken Sie auf die Schaltfläche *Clear Filter,* um wieder alle FBoxen anzuzeigen.

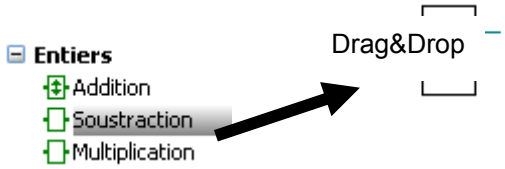
Der Zweig *Libraries* im Project Manager zeigt lokale Bibliotheken des aktuellen Projekts an sowie alle Bibliotheken, die mit PG5 installiert wurden. Entfernen Sie das Häkchen vor den Bibliotheken, die Sie nicht verwenden möchten. Dies verringert die Anzahl der im Fenster *Selector* angezeigten Bibliotheken.

FBoxen, die regelmäßig verwendet werden, können in die Liste *Favorites* aufgenommen werden. Wählen Sie die FBox, öffnen Sie das Kontextmenü und verwenden Sie den Befehl *Add To Favorites*.

Favorites

Um die *Favorites* anzuzeigen, klicken Sie unten im Fenster *Selector* auf die Schaltfläche.

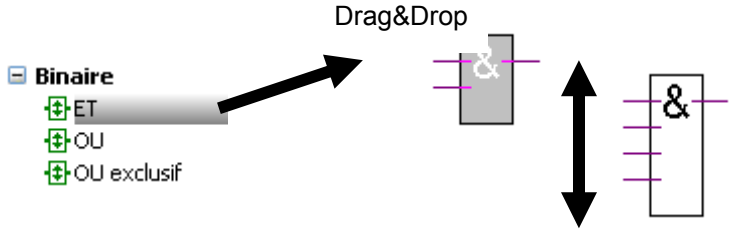
6.6.1 Editieren einer FBox



Die für das Schreiben eines Programms benötigten Funktionen können im FBox Selector ausgewählt und dann per Drag&Drop in das Fupla-Programm eingefügt werden.

6.6.2 Editieren von erweiterbaren (ausziehbaren)FBoxen

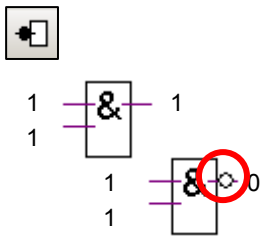
Verschiedene FBoxes sind erweiterbar, das heisst, dass die Anzahl der Ein- oder Ausgangsanschlüsse der FBox durch die vertikale Bewegung der Maus definiert werden kann.



Wählen Sie eine erweiterbare FBox aus.
Ziehen Sie diese per Drag&Drop auf die Fupla-Seite.
Bewegen Sie die Maus vertikal, bis die korrekte Anzahl der Ein- oder Ausgänge angezeigt wird.
Drücken Sie die linke Maustaste zum Beenden.

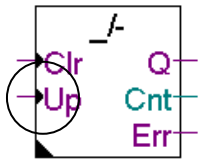
6.6.3 Invertierung binärerer Signale

Auswahl des *Invert Connector* Knopfes in der Werkzeugleiste.
Positionierung des Mauszeigers auf den binären Ein- oder Ausgangsanschluss, danach linke Maustaste anklicken.



6.6.4 Dynamisierung (Flankentrigger)

Die Eingänge einiger FBoxen sind dynamisiert. (Flankengetriggert). Diese Eingänge reagieren nur auf die ansteigende Flanke des binären Signals. Dynamisierte Eingänge werden mit einem kleinen schwarzen Dreieck gekennzeichnet.



Fbox: *Zähler, Up mit Nullstellung*

Zum Beispiel kann ein Impulszähler nicht ständig inkrementiert werden, wenn das Eingangssignal "*UP*" der FBox den Wert "1" hat. Wenn dies nicht so wäre, so würde der Impulszähler ständig inkrementiert, solange der "*UP*" Eingang auf 1 ist.
Für diese Aufgabenstellung müssen digitale Eingänge dynamisiert werden.
Dieses dynamisieren bewirkt, dass nur die positive Flanke des "*UP*" Eingangssignals den Impulszähler inkrementiert.

Teilweise ist es nötig, die Ein- oder Ausgangssignale einer FBox zusätzlich zu dynamisieren.
Dies ist mit der FBox *Binäre Funktionen, Flanke* möglich.

6.6.5 Kommentare

Kommentare können beliebig direkt im Fupla Programm integriert werden:

1. Auswahl des *Place comment* Knopfes in der Werkzeugleiste.
2. Positionierung des Kommentars in der Fupla Seite, danach linke Maustaste anklicken.
3. Kommentar schreiben.
4. *Enter* Taste betätigen.

6.6.6 FBox Hilfe

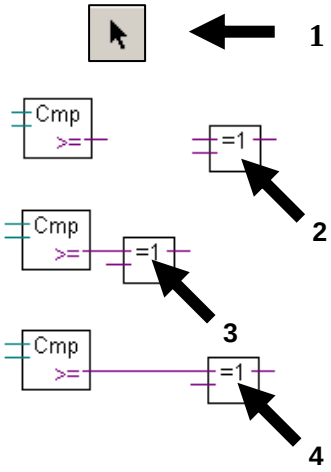
Um eine vollständige Beschreibung einer Funktion zu erhalten, wählen Sie im Fenster *Selector* oder auf der Fupla-Seite die FBox aus und drücken Sie dann die Taste F1.

Eine unbekannte FBox, die in einem Programm gefunden wird, können Sie schnell identifizieren, indem Sie das Fenster *Selector* öffnen, mit dem Mauszeiger auf die unbekannte FBox zeigen und einmal die linke Maustaste drücken. Der *Selector* wählt dann die FBox aus.

6.7 Editieren von Verbindungen zwischen den FBoxen und Connectors

6.7.1 Verbindung durch Verschieben einer FBox

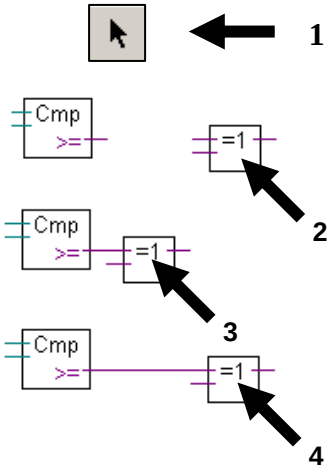
1. Auswahl des *Select Mode* Knopfes in der Werkzeugleiste.
2. Mauszeiger zur FBox positionieren und linke Maustaste betätigen.
3. Linke Maustaste gedrückt halten und FBox zur gewüschten Stelle ziehen.
4. Die Verbindungen zwischen den FBoxen werden erstellt, sobald sich zwei Kontaktstellen berühren.



6.7.2 Verbindung mit automatischem Routing

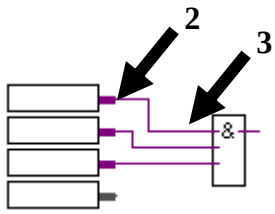
1 Klicken Sie auf die Taste *Line Mode* auf der Werkzeugleiste.
2 Platzieren Sie den Mauszeiger auf dem Ausgangsort und drücken Sie die linke Maustaste.
3 Platzieren Sie den Mauszeiger auf dem Zielort und drücken Sie die linke Maustaste.

Anmerkung:
Es können auch Zwischenstationen ausgewählt werden.
Um die Arbeit an der Verbindung zu unterbrechen, drücken Sie die rechte Maustaste.



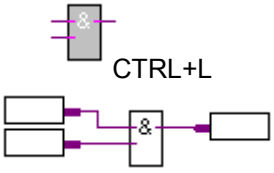
6.7.3 Mehrfachverbindung mit automatischem Routing

1 Wählen Sie das Menü *Mode, Connect Bus* oder (CTRL+B).
2 Wählen Sie mit der Maus einen Ausgangspunkt.
3 Wählen Sie nun einen Zielpunkt.



6.7.4 Verbindung aller Eingänge/Ausgänge einer FBox mit Connectors

Platzieren Sie den Mauszeiger auf einer FBox. Drücken Sie die rechte Maustaste, um das Kontextmenü aufzurufen: *Connections, Connect to Side Connectors.*

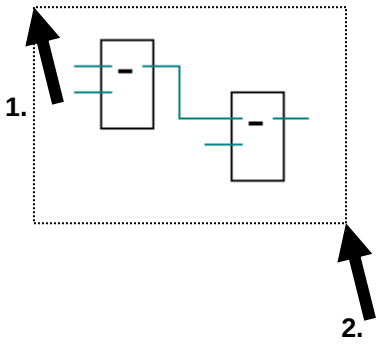


6.7.5 Löschen von Linien, FBoxes, Connectors oder Symbolen

Drücken Sie zunächst die Taste *Delete Object* auf der Werkzeugleiste, dann die Verbindungen, FBoxes oder Symbole, die Sie löschen möchten.

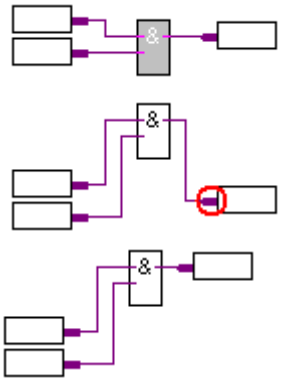
Im Schnellverfahren können Sie einen Bereich markieren und ihn löschen.

1 Drücken Sie die Maustaste.
2 Ziehen Sie den Mauszeiger, ohne die Maustaste loszulassen.
3 Lassen Sie die Maustaste los.
4 Wählen Sie *Delete* im Menü *Edit.*



6.7.6 Vertikale Verschiebung einer FBox/eines Connectors ohne Aufhebung der Verbindungen

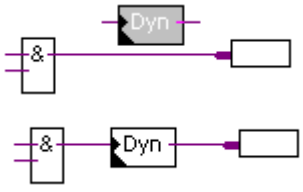
Platzieren Sie den Mauszeiger auf einer FBox.
Drücken Sie die Umschalttaste und halten Sie sie gedrückt.
Drücken Sie die linke Maustaste und halten Sie sie gedrückt.
Ziehen Sie den Mauszeiger vertikal auf einen leeren Platz auf der Seite.
Lassen Sie Maus- und Umschalttaste los.



Um den Connector zu verschieben, platzieren Sie den Mauszeiger auf dem roten Kreis und wiederholen Sie das oben beschriebene Verfahren.

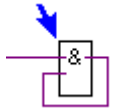
6.7.7 Einfügen einer FBox ohne Aufhebung der Verl

Wählen Sie im *Selector* die FBox, die Sie einfügen möchte
Platzieren Sie sie über der Verbindung.



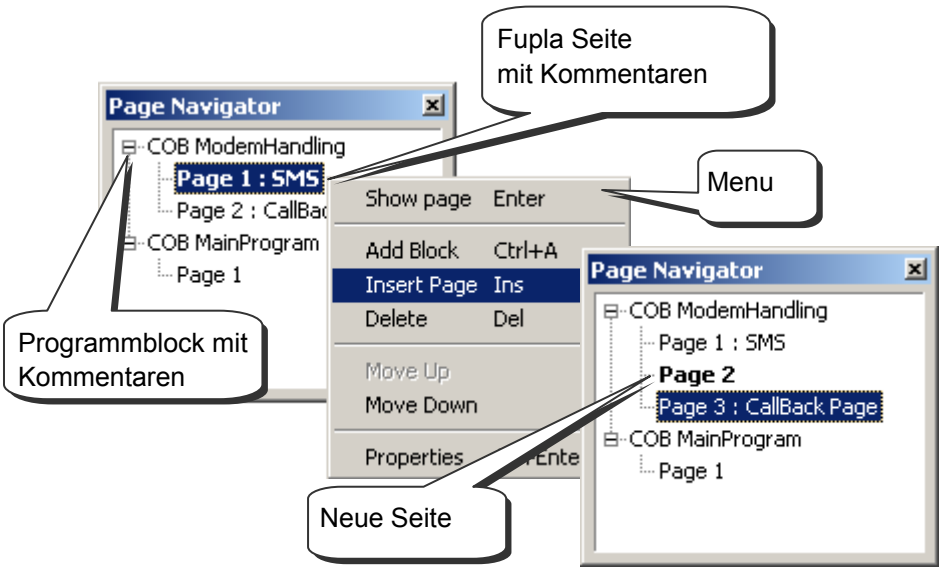
6.7.8 Wichtige Regeln

Es sind keine Schleifen gestattet. Wenn eine Schleife angelegt wird, wird eine Fehlermeldung angezeigt: *Page 1: Error 55: Loop back detected*



Es sind keine direkten Verbindungen zwischen Eingangs- und Ausgangsconnectors erlaubt. Es muss mit einer FBox gearbeitet werden: *Binary, Direct transfer* oder *Integer, Direct transfer.*

6.8 Editieren von FUPLA-Seiten



Show/Hide Page Navigator

Im *Page Navigator* Fenster sind die verwendeten Programmblöcke und Fupla Seiten ersichtlich. Jede Fupla Datei kann bis zu 200 Fupla Seiten beinhalten welche in folgende Programmblöcke gruppiert werden können: COBs, PBs, FBs, oder SBs. Die Abarbeitungszeit von Fupla wird schneller, wenn nicht zu viele Fupla Seiten in einer Fupla Datei enthalten sind. Standardmässig werden Fupla Seiten in einen COB Programmblock integriert. Mehr Informationen über Programmblöcke und deren Verwendung sind im Kapitel ProgrammStrukturen dieses Handbuches ersichtlich.

6.8.1 Fupla Seite einfügen

Insert Page

Page Navigator Fenster öffnen, selektieren der gewünschten Seite durch „Doppelklick" und danach betätigen *Insert Page* Knopfes in der Werkzeugleiste
Es ist auch möglich eine Seite einzufügen indem die *Insert Page* Taste der Tastatur betätigt wird oder über den Menüpunkt *Page* in der Werkzeugliste: *Page Insert After (Page Insert Before)*

6.8.2 Fupla Seite löschen

Page Navigator Fenster öffnen, selektieren der zu löschenden Seite, rechte Maustaste anklicken und Auswahl von *Delete* aus dem Menü

6.8.3 Seitennavigation

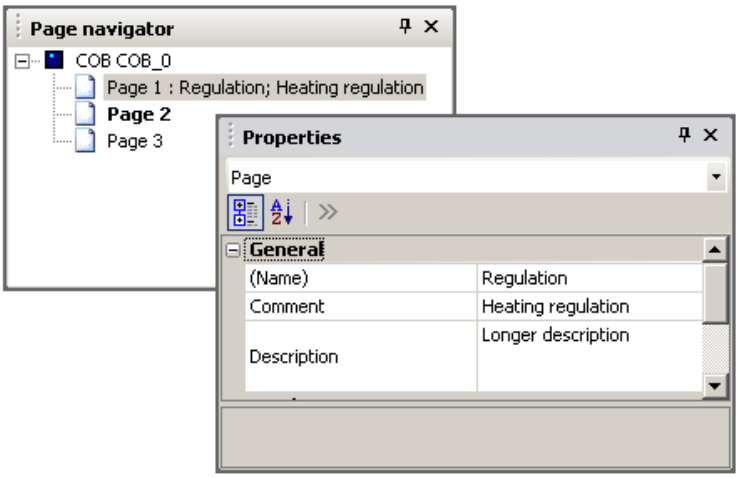
Goto Next Page

Page Navigator Fenster öffnen, Mauszeiger über der gewünschten Seite positionieren und linke Maustaste Doppelklicken.
Es ist ebenfalls möglich mit den Knöpfen der Werkzeugleiste *Go to Previous Page* and *Go to Next Page* von einer Seite zur nächsten Seite des Fupla Programmblockes zu navigieren. Sollte einer der beiden Knöpfe grau sein, so befinden Sie sich bereits auf der ersten oder letzten Seite des Programmblockes.

6.8.4 Fupla Seiten-Dokumentation

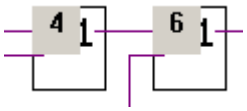
Es wird ausdrücklich empfohlen, jede Fupla Seite zu dokumentierten. Der Name und der Kommentar jeder Fupla Seite werden im *Page navigator* Fenster angezeigt und erleichtert das navigieren zwischen den Seiten. Im Eingabefeld *Description* können nützliche Informationen über das Programm eingegeben werden welche den Unterhalt und die Pflege des Programmes erleichtern.

Wählen Sie den *Page Navigator* aus, öffnen Sie das Kontextmenü und verwenden Sie den Befehl *Properties*, um die Seite *Properties* anzuzeigen.



6.8.5 Abarbeitung des Fupla Programmes in der PCD

In der PCD wird das Fupla Programm wie folgt abgearbeitet:
Die Fupla Seiten eines Programmblockes werden nacheinander von oben links der ersten Seite bis unten rechts der letzten Seite abgearbeitet. Die genaue Reihenfolge der Abarbeitung der FBoxen in der PCD kann mit der Menüfunktion *Page, Show FBox priorities* der Werkzeugleiste angezeigt werden.

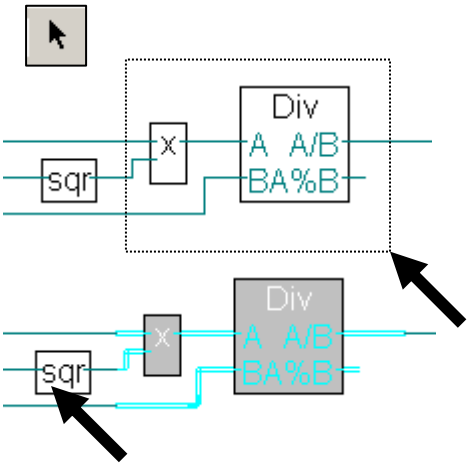


6.9 Kopieren und Einfügen

Oftmals werden gleiche Fupla-Programmsequenzen mehrmals im Anwenderprogramm verwendet. In diesen Fällen ist es nicht nötig, die Fupla-Programmsequenzen jedesmal neu zu erstellen. Viel effizienter ist, wenn die Fupla-Programmsequenzen durch Kopieren und Einfügen dupliziert werden, und danach die duplizierten Fupla-Programmsequenzen nach Bedarf adaptiert werden.

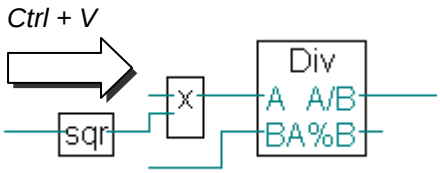
6.9.1 Kopieren und Einfügen von Programmteilen

1. Anklicken des Select Mode Knopfes in der Werkzeugleiste.
2. Markieren des zu kopierenden Bereiches:
   - Linke Maustaste drücken.
   - Bewegen der Maus bei gedrückter Maustaste.
   - Linke Maustaste loslassen.
3. Zufügen einer FBox oder Verbindung zum ausgewählten Bereich:
   - *Ctrl* Taste der Tastatur drücken.
   - Mit gedrückter *Ctrl* Taste, Auswahl der Verbindung, Rand und FBoxen welche dem ausgewählten Bereich zugefügt werden soll.



Kopieren des ausgewählten Bereiches mit dem *Edit Copy* Menu der Werkzeugleiste, oder mit den *Ctrl + C* Tasten.

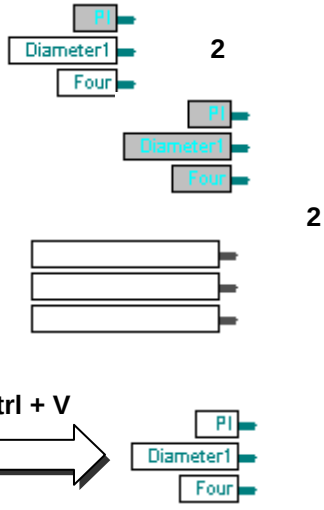
Einfügen des kopierten Bereiches mit dem *Edit Paste* Menu der Werkzeugleiste, oder der *Ctrl + V* Tasten.



Positionieren des kopierten Bereiches auf der Fupla Seite:

- Positionierung des Mauszeigers auf die Mitte des kopierten Bereiches.
- Linke Maustaste drücken.
- Bewegen der Maus bei gedrückter Maustaste.

6.9.2 Kopieren und Einfügen von Symbolen



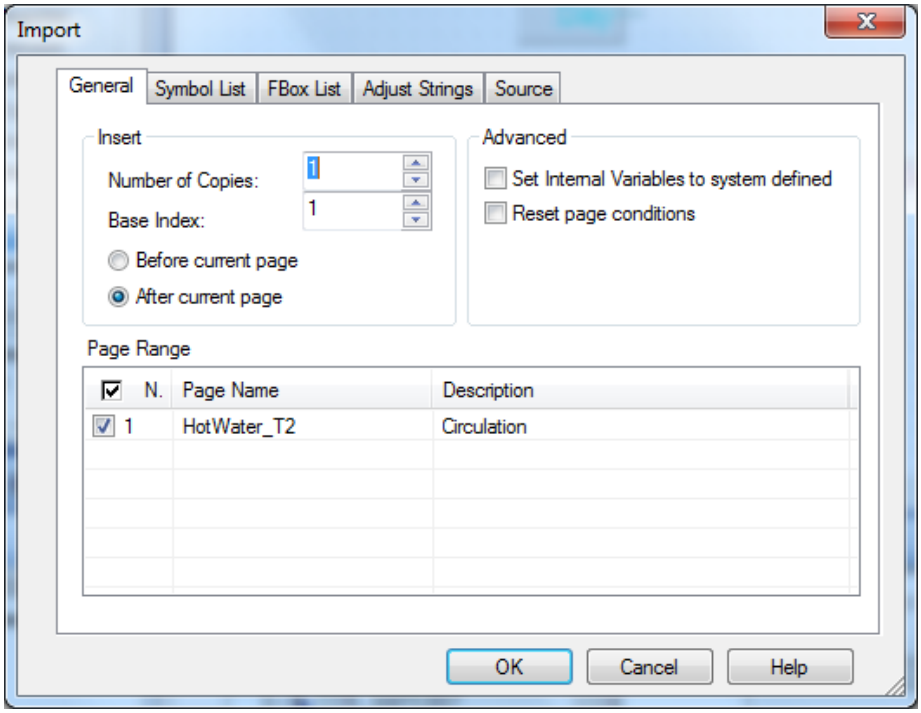
1. Anklicken des Select Mode Knopfes in der Werkzeugleiste.
2. Markieren einer Liste von Symbolen:
   - Positionierung des Mauszeigers auf das erste Symbol.
   - Linke Maustaste anklicken.
   - Positionierung des Mauszeigers auf das letze Symbol.
   - *Shift* Taste drücken. *)
   - Mit gedrückter *Shift* Taste, linke Maustaste anklicken.
3. Kopieren des ausgewählten Bereiches mit dem *Edit Copy* Menu der Werkzeugleiste, oder mit den *Ctrl* + *C* Tasten.
4. Positionieren des the Mauszeigers auf einen freien Bereich des Randes.
5. Einfügen des kopierten Bereiches mit dem *Edit Paste* Menu der Werkzeugleiste, oder der *Ctrl* + *V* Tasten.

*) Die *Ctrl* Taste ermöglicht, nicht aufeinander folgende Symbole auszuwählen

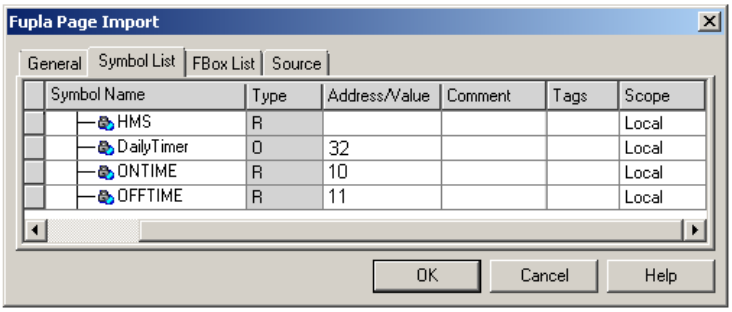
6.9.3 Importieren von Templates

Templates können in allen Projekten wiederverwendet werden. Wählen Sie die Vorlage im Fenster *Template Selector* aus und ziehen Sie sie auf die Fupla-Seite, um die Seitensequenz mit ihren FBoxen, Links, Symbolen, Parametereinstellungen usw. in die Datei einzufügen.

Ein Dialogfenster wird angezeigt, in dem die Namen und Adressen der importieren Symbole sowie andere Parameter geändert werden können. Diese Funktionalität ist einem Makro sehr ähnlich oder auch Funktionen mit Parametern, die wie eine Bibliothek verwendet werden können.



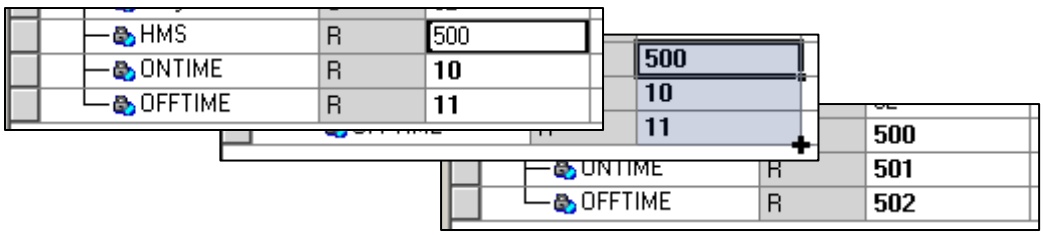
Anzahl der Kopien, Basisindex
Wenn von einer Vorlage mehrere Kopien benötigt werden, können Sie definieren, wie häufig die Vorlage eingefügt werden soll und welcher Basisindex dem Symbol und dem Gruppennamen hinzugefügt wird.
Vor/Nach aktueller Seite
Damit wird die Sequenz vor oder nach der momentan in Fupla aktiven Seite importiert.
Interne Variablen auf systemdefinierten Wert setzen
Einige FBoxen verfügen über *Adjust Variables*, deren Name vom Nutzer oder dem System (*Static symbols*) definiert wird. Mit dieser Einstellung können Sie Adressen und statische Symbole vom Autor definieren lassen oder standardmässig dynamische Adressen und interne Symbole wiederherstellen.
Seiteneinstellungen zurücksetzen
Im Fenster *Properties* auf einer Fupla-Seite kann die Durchführung einer *Condition* für jede Seite definiert werden. Mit dieser Einstellung können beim Import von Seiten die Einstellungen entfernt werden.
Seitenreichweite
Hier werden einzelne Seiten einer Vorlage oder eine komplette Vorlage zum Import ausgewählt.



Die Seite *Symbol List* zeigt alle Symboldefinitionen und –referenzen an, die in eine Vorlage importiert werden. Die Symbole können dann mit anderen Namen, Adressen, Kommentaren und Reichweiten neu definiert werden.

Das Markieren von Symbolen, um sie in eine Gruppe zu verschieben, ist die schnellste Möglichkeit, die Namen aller importierten Symbole zu ändern. Mit dem Kontextmenübefehl *Insert Pre-group* werden die ausgewählten Symbole in die Gruppe mit dem gewünschten Namen verschoben.

Ordnen Sie die Symbole nach Typen, indem Sie die Schaltfläche *Type* am Kopf der Spalte anklicken, um die Adressen der Symbole zu verändern. Wählen Sie die erste Adresse aus und bearbeiten Sie diese. Ziehen sie dann das kleine Quadrat in der unteren rechten Ecke der ausgewählten Zelle nach unten, bis alle gewünschten Adresse ausgewählt sind und lassen Sie dann die Maustaste los. Die ausgewählten Adressen werden, von der ersten Adresse ausgehend, fortlaufend neu nummeriert.



Beachten Sie beim Import mehrerer Kopien derselben Vorlage die Parameter auf der Seite *General*. Es ist hilfreich, mit dem Zeichen # einen Index in die Namen der Symbole oder Gruppen einzufügen. Dieses Zeichen wird automatisch durch den Basisindex ersetzt, der sich für jede Kopie der Vorlage um 1 erhöht. Die Symbole können auch mit Hilfe des Kontextmenübefehls *Indexing* ausgewählt werden.

Die Seite *FBox List* zeigt eine Liste aller FBoxen an, deren Symbole mit einem Namen definiert sind. Auch diese Namen können mit dem Zeichen # verändert werden.

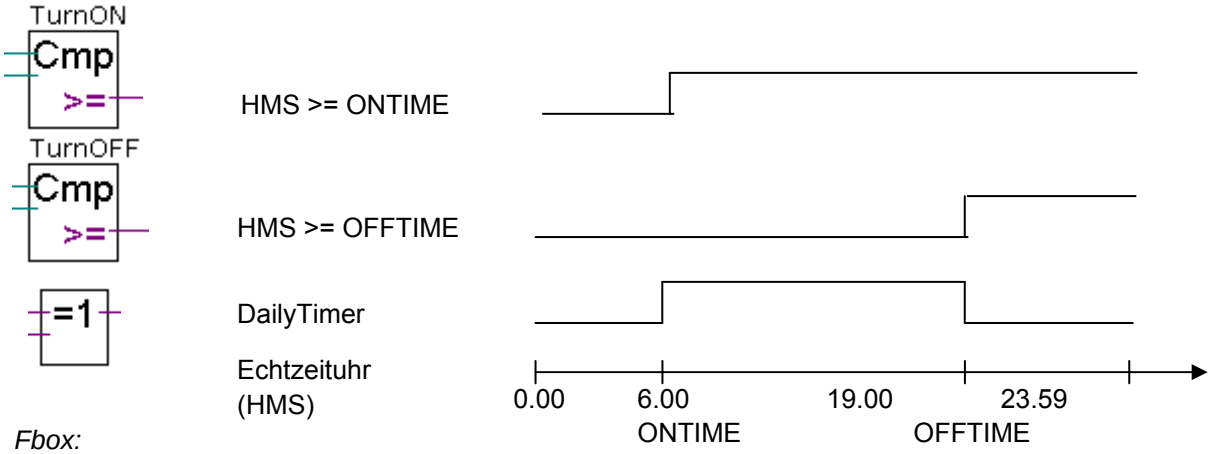
6.10 Erstellen des ersten Fupla Programmes

6.10.1 Ziel

Nachdem die Werkzeuge bekannt sind, besteht der nächste Schritt darin, ein komplexeres Programm, als die zuvor aufgezeigten logischen Verknüpfungen zu erstellen. Wir wollen eine Zeitschaltuhr programmieren welche jeden Tag einen digitalen Ausgang (Ausgang 32) um 06.00 Uhr ein- und um 19.00 Uhr ausschaltet. Obschon diese Funktionalität in der HLK Bibliothek mit einer FBox verfügbar ist, werden wir die Funktionalität mit Standard FBoxen realisieren.

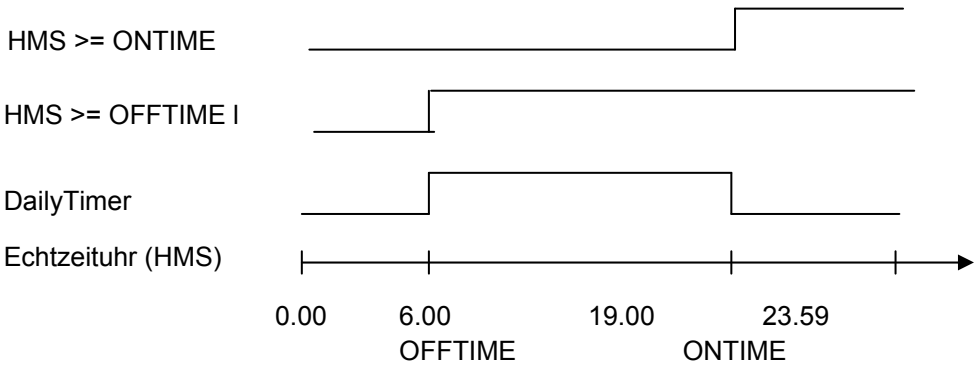
6.10.2 Vorgehensweise

Bevor mit dem Schreiben des Programms begonnen werden kann, muss eine Logik gefunden werden mit welcher die gestellte Aufgabe mit möglichst einfachen Funktionen gelöst werden kann.
Für dieses Zeitschaltuhrbeispiel werden wir zwei Vergleichsfunktionen verwenden. Mit dem ersten Vergleich wird verglichen ob die Echtzeituhr (z.B. die Zeit unserer Uhr oder die Echtzeituhr in der PCD) in *HMS* (Stunden, Minuten, Sekunden) gleich oder grösser der Einschaltzeit *ONTIME* ist. Mit dem zweiten Vergleich wird verglichen ob die Echtzeituhr kleiner oder gleich der Ausschaltzeit *OFFTIME* ist. Wenn die Ergebnisse beider Vergleiche mit einer logischen Verknüpfung (exklusiv oder) verknüpft werden, muss der Digitale Ausgang 32 *DailyTimer* eingeschaltet werden.



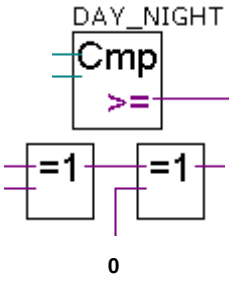
Fbox:
- Ganzzahl Arithmetik, Wert grösser/gleisch
- Binäre Funktionen, Xor 2-10 Eingänge

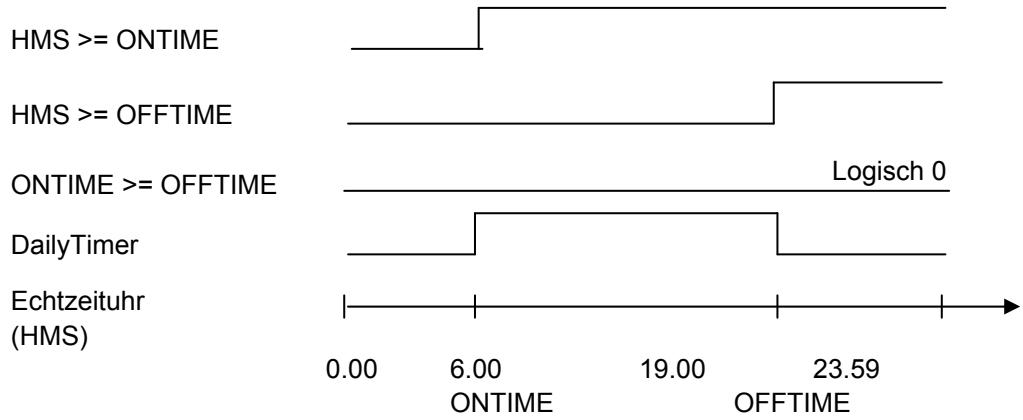
Diese FBox Kombination ist eine Lösung für die Aufgabe, hat aber einen kleinen Fehler. Was geschieht, wenn die Einschaltzeit grösser ist als die Ausschaltzeit? (z.B. falls der Ausgang während der Nacht eingeschaltet werden soll). Die Untenstehende Grafik zeigt dass der PCD Ausgang invers zum gewünschten Verhalten geschaltet wird.



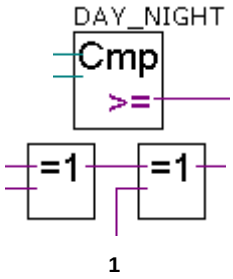
Es ist deshalb nötig, einen dritten Vergleich in unsere Logik einzufügen. In diesem Vergleich wird geprüft, ob die Einschaltzeit grösser oder gleich der Ausschaltzeit ist. Die Endgültige Lösung der Aufgabe sieht wie folgt aus:

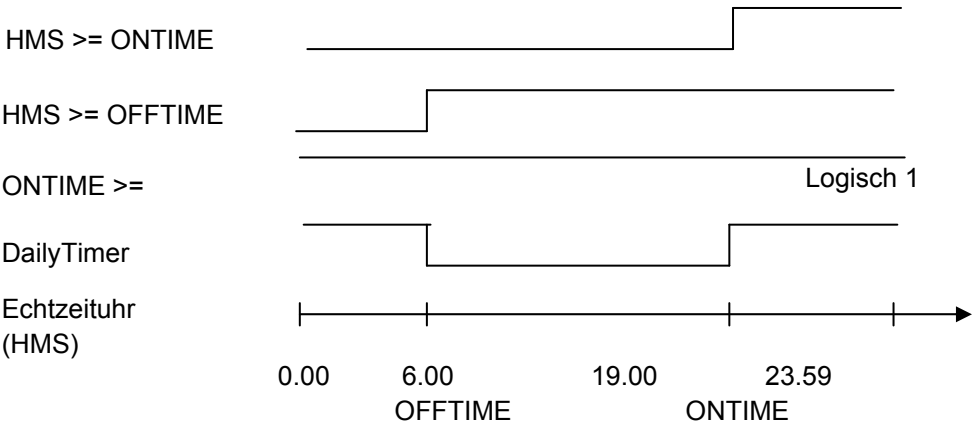
Falls Ausgang bei Tag geschaltet wird





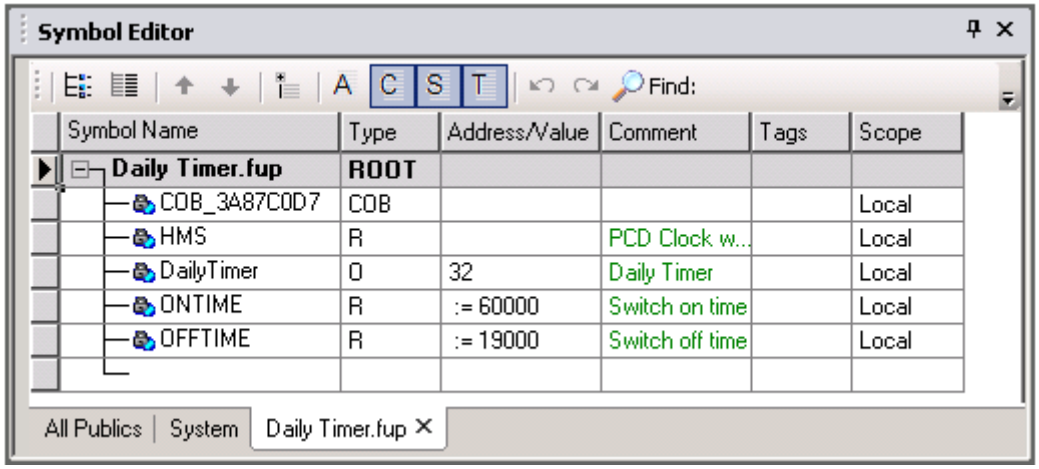
Falls Ausgang bei Nacht geschaltet wird



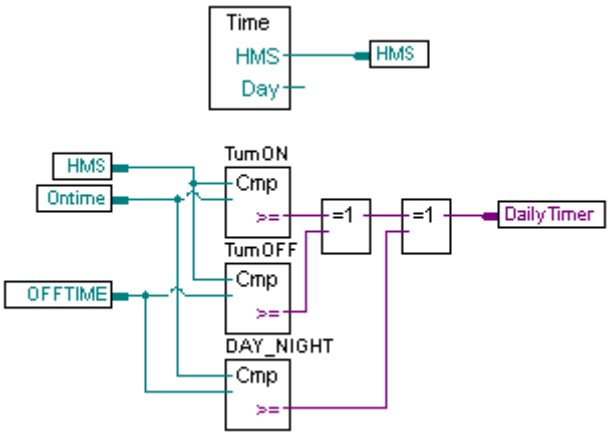


6.10.3 Programmierung

Es ist nun an der Zeit das Programm zu erstellen. Zu Beginn dieses Kapitels haben wir ein Projekt definiert welches eine Datei mit dem Namen *DailyTimer.fup* enthält. In diese Datei wird das Programmbeispiel geschrieben.



Zuerst wird die Symbolliste erstellt. Die Echtzeituhr der PCD wird dynamisch in einem Register mit dem Namen HMS gespeichert. Die absolute Adresse des Registers wurde nicht definiert da diese vom PG5 automatisch beim verarbeiten (Build) des Programms generiert wird.
Das gleiche gilt auch für die Einschalt- und Ausschaltzeiten (*ONTIME, OFFTIME)*, ausser dass der Ausdruck «:=6000» nicht einer absoluten Registeradresse entspricht, sondern dies der Initialwert ist, mit welchem die Symbolvariable beim laden des Programms in die PCD initialisiert wird. (:=6000 bedeutet 6 Stunden 00 Minuten 00 Sekunden).
N.B.: Bei einem Kaltstart der PCD wird die Symbolvariable nicht neu mit dem Initialwert initialisiert. Die Initialisierung einer Symbolvariable mit ihrem Initialwert erfolgt nur beim laden des Programms vom PG5 in die PCD!



Alle benötigen FBoxen sind in der *Standard* Gruppe des *Selector* Fenster enthalten:

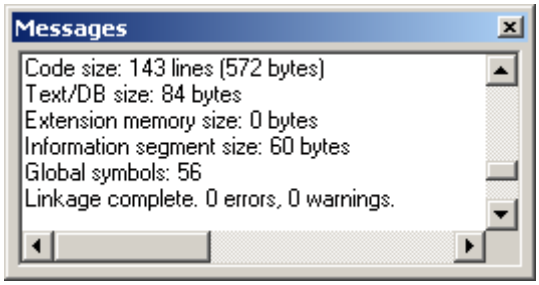
- *Zeitfunktionen, Uhr lesen*
- *Ganzzahl Arithmetik, Wert grösser/gleich*
- *Binäre Funktionen, Xor 2-10 Eingänge*

6.11 Verarbeitung (build) des Programms

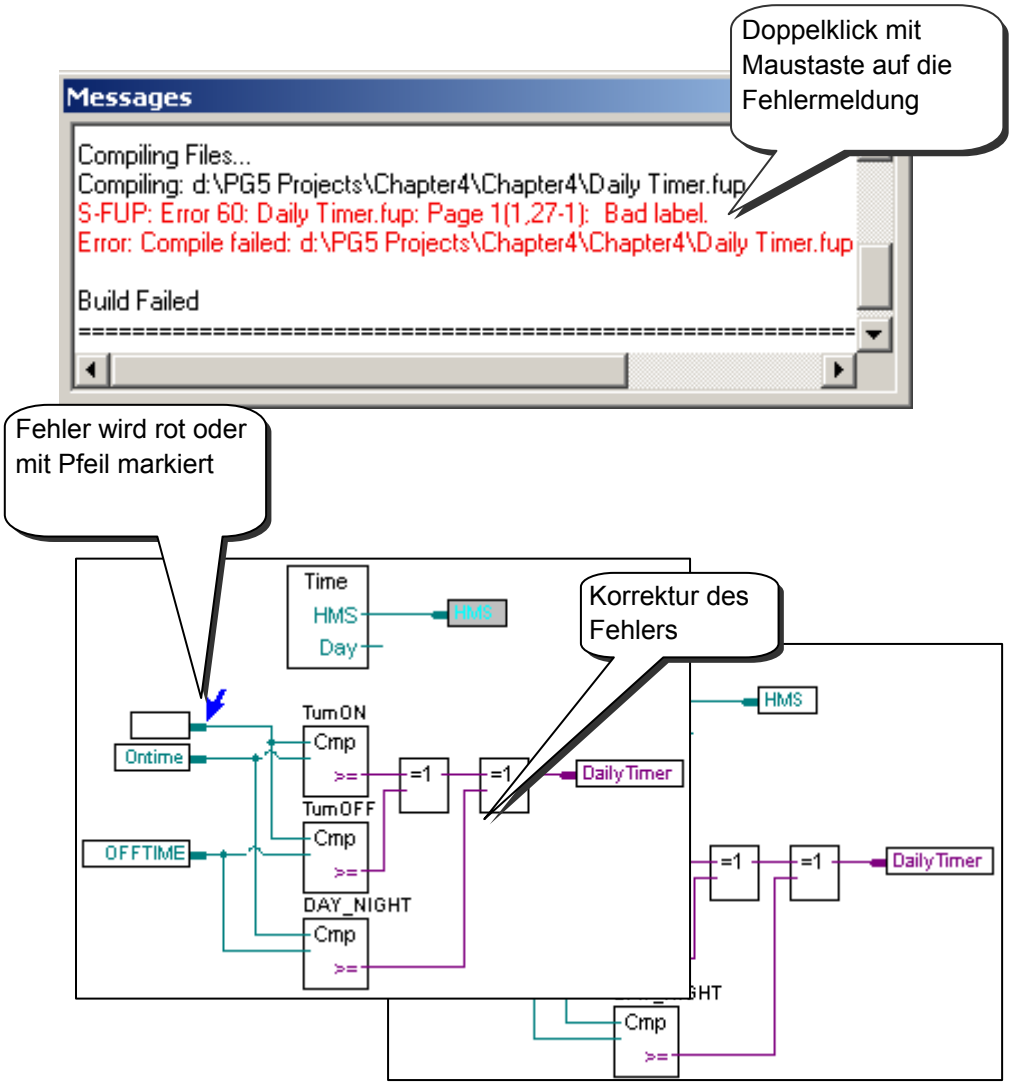
Build All

Bevor das erstellte Programm in der PCD abgearbeitet werden kann, muss es verarbeitet “build” werden. Dazu kann die Menüfunktion *Device, Build Changed Files* or *Rebuild All Files* Knopf in der Werkzeugleiste verwendet werden.

Das *Message* Fenster zeigt die Ergebnisse des während des Verarbeitungsprozesses (build) durchlaufenen Prozesse auf: (Compiling, Assembling, Linking etc.).
Wenn das Programm korrekt erstellt worden ist, wird der Verarbeitungsprozess (build) mit folgender Meldung beendet:
Build successful. Total errors 0 Total warnings:



Wurden Fehler in der Zeichnung entdeckt, werden diese in rotem Text angezeigt. Es kann nun auf diesen roten Text geklickt werden um zu den Fehlerstellen im Programm zu gelangen.



6.12 Laden des Programms in die PCD

Download Program

Das Anwenderprogramm ist nun erstellt und verarbeitet und kann vom PC in die PCD geladen werden. Dies erfolgt mit dem *Download Program* Knopf in der Werkzeugleiste oder über das *SAIA PG5 Project Manager* Fenster, *Online* Menu, *Download Program.*
Sollten Kommunikationsprobleme zwischen PC und PCD auftreten so sind die Kommunikationseinstellungen unter *Settings Online* and *Settings Hardware* sowie das Schnittstellenkabel zwischen PC und PCD (PCD8.K111, USB) zu prüfen.

6.13 Programmfehler finden und korrigieren (Debug)

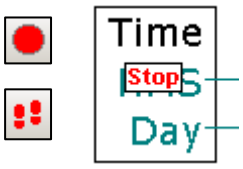
Die erste Version eines Programms ist oftmals noch nicht perfekt. Deshalb muss das Programm eingehend getestet werden. Der Programmtest erfolgt mit dem gleichen Fupla Editor welcher zur Programmerstellung verwendet wurde.

6.13.1 Go On/Offline – Run – Stop - Step-by-step

1. *Go On /Offline* Knopf betätigen
2. Zum Programmstart *Run* Knopf betätigen

Achten Sie gleichzeitig auf die *RUN* LED der PCD.
Nachdem der *Run* Knopf betätigt wurde, sollte die *RUN* LED der PCD eingeschaltet werden. Dies bedeutet, dass die PCD das Anwenderprogramm abarbeitet.

3. Bei Betätigung des *Stop* Knopfes hält die PCD die Abarbeitung des Anwenderprogramms an und die *RUN* Lampe der PCD wird ausgeschaltet.
4. Bei jeder Betätigung des *Step-by-step* Knopfes oder des *F11* Funktionsknopfes wird die PCD eine einzelne FBox abarbeiten. (Schrittweises Abarbeiten)



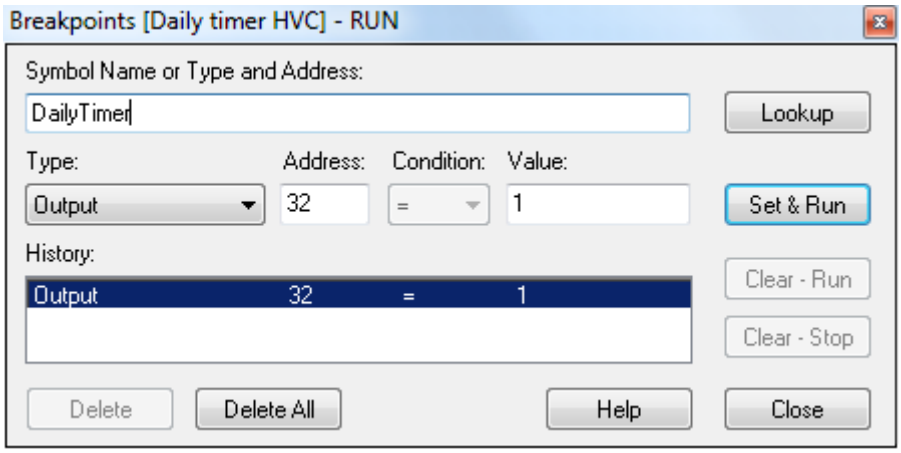
Die im nächsten Schritt abgearbeitete Fbox wird mit dem *Stop* Symbol angezeigt.

6.14 Anhalte-Punkte (Breakpoints)

Anhalte-Punkte ermöglichen Ihnen das Anhalten eines Programms bei bestimmten Ereignissen, die sich auf FBoxes oder Symbole beziehen können:
Niedriger oder hoher Status (low/high) eines Eingangs, Ausgangs, Flags oder Statusflags
Wert liegt im Register oder Zähler vor

Anhalte-Punkt auf einem Symbol
Die für ein Anhalten notwendige Bedingung kann mit Hilfe des Menüs *Online Breakpoints* definiert werden.

Set or Clear Breakpoints



In oben gezeigtem Fenster werden Symboltyp und -adresse/-nummer definiert. Ein Symbol kann einfach vom Symbol-Editor in das Feld *Symbol Name* gezogen werden. Anschliessend werden die Anhalte-Punkt-Bedingung und der Status/Wert definiert.

Durch Drücken der Taste *Set & Run* wird die PCD in den Modus ‚Conditional Run' geschaltet. Die *Run* LED der PCD blinkt und die *Run* Taste ist wechselweise grün und rot.

Die PCD schaltet automatisch in Stop-Modus, wenn die Anhalte-Punkt-Bedingung erfüllt ist. Wenn eine Anweisung beispielsweise den Ausgangswert verändert, steht der Status von 32 auf ‚high'. Die letzte von der PCD bearbeitete FBox wird rot angezeigt. Sie können das Programm entweder im Step-by-Step-Modus oder unter Verwendung eines weiteren Anhalte-Punkts bearbeiten.

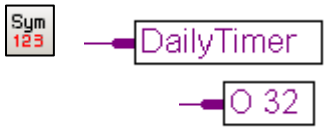
Bei Bedarf kann der Conditional-Run-Modus unterbrochen werden:
Die Taste *Clear-Run* schaltet die PCD in RUN-Modus. Die *Run* LED der PCD leuchtet auf und die *Run* Taste ist grün.
Die Taste *Clear-Stop* schaltet die PCD in Stop-Modus. Die *Run* LED der PCD geht aus und die *Run* Taste leuchtet rot.

Wenn mehrere bedingte Anhalte-Punkte definiert sind, werden sie alle im Feld *History* aufgezeichnet. Es kann ein beliebiger Anhalte-Punkt mit der Maus ausgewählt und mit der Taste *Set & Run* aktiviert werden.

Anhalte-Punkt an einer Programm-FBox

Wählen Sie eine beliebige FBox im Programm und führen Sie das Menü oder Taste *Online, Run to, Fbox* aus, um das Programm an der gewählten FBox anzuhalten. Setzen Sie Ihre Arbeit im Step-by-Step-Modus fort.

6.14.1 Anzeige von Symbolnamen oder absoluten Adressen



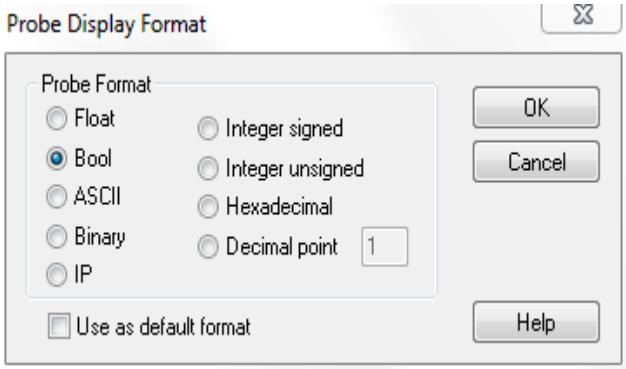
Mit *Change Symbol/Resource view* Knopfes der Werkzeugleiste können die Eingangs- und Ausgangsvariablen welche am Eingang/Ausgang Rand des Fupla-Editors verwendet worden sind, entweder mit dem Symbolnamen oder mit der absoluten Adresse dargestellt werden. Sollte ein Symbolname nicht mit der absoluten Adresse ersetzt werden, so bedeutet dies, dass die absolute Adresse erst bei der Programmverarbeitung (build) dem Symbolischen Namen zugewiesen wird.

6.14.2 Anzeige des Symbolstatus in Fupla

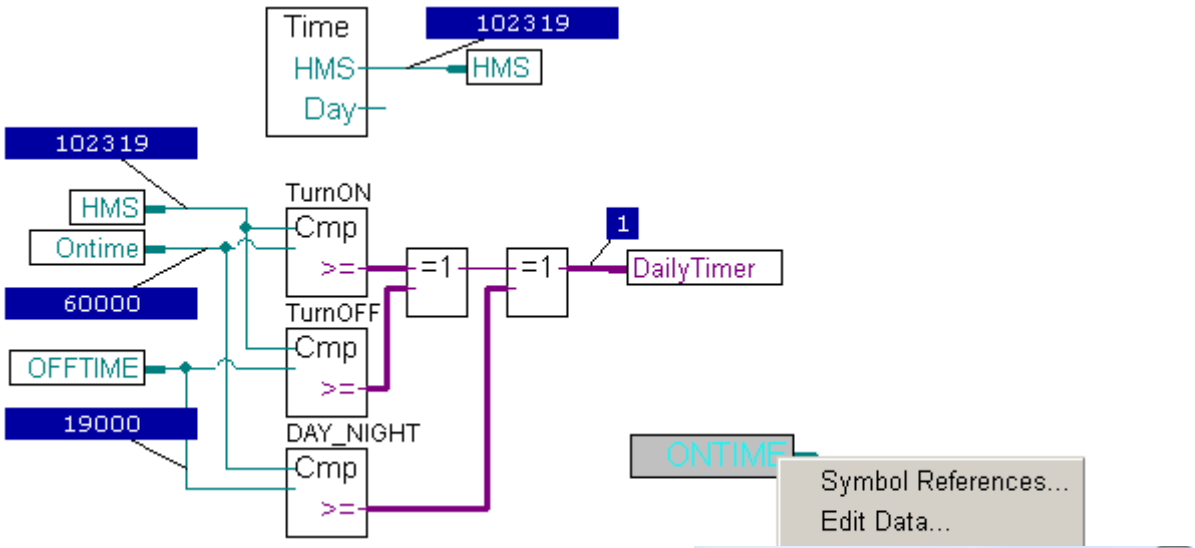
Falls sich der Fupla Editor *Online* mit der PCD befindet und die PCD in *RUN* geschaltet ist, kann der Zustand jeder, im Programm verwendeten Ressource angezeigt werden:

Add Probe

Der logische Zustand binärer Ressourcen wird durch schmale oder breite Linie angezeigt (Breite Linie = 1; schmale Linie = 0)
Der Zustand aller anderer Ressourcen kann durch anklicken der entsprechenden Ressource mit der linken Maustaste angezeigt werden.

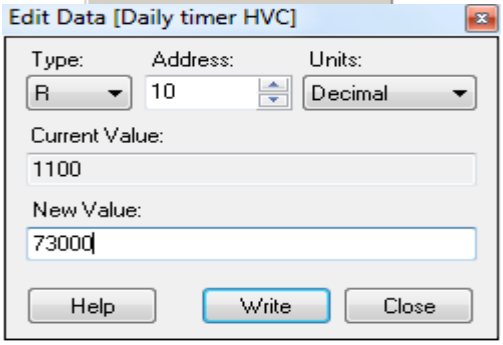


Ein Doppelklick auf eine Probe öffnet das *Probe Display Format* Fenster. Damit kann die Darstellung des Anzeigeformates der gewählten Ressource zwischen Integer, Hexadezimal, Binär, Fliesskomma, Boolean oder ASCII geändert werden



6.14.3 Editieren von Symbolen online

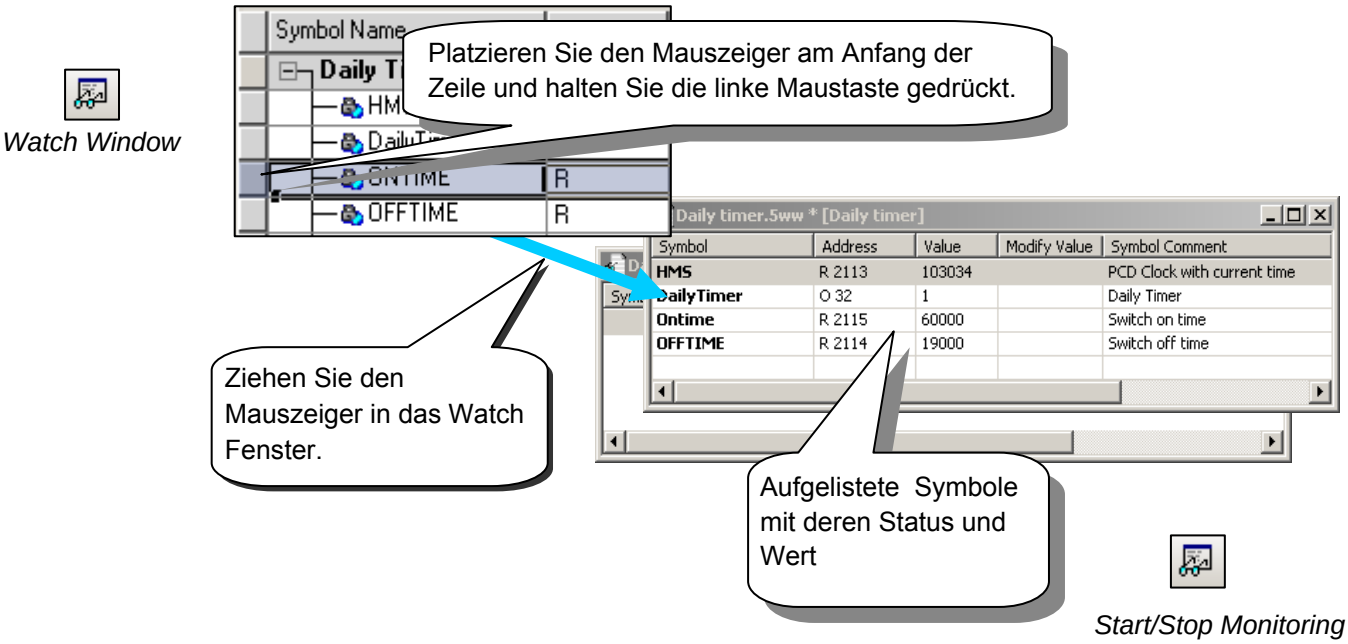
Wenn Sie das Programmverhalten unter bestimmten Nutzungsbedingungen prüfen, kann es manchmal hilfreich sein, den Status/die Werte von in den Eingangsconnectors oder *Symbol Editor* vorhandenen Symbolen zu ändern.



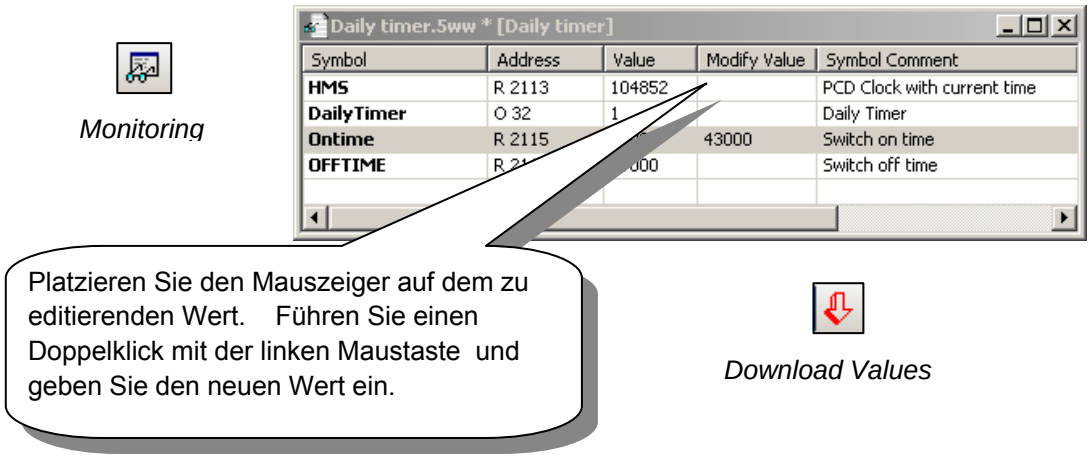
Wählen Sie mit der Maus einen Eingangsconnector aus und drücken Sie die rechte Maustaste, um das Kontextmenü aufzurufen.
Mit dem Kontextmenü *Edit Data* können Sie den Status/Wert eines Symbols in einem Connector ändern.

6.14.4 Anzeige des Symbolstatus mit *Watch window*

Eine andere Möglichkeit den Zustand der in unserem Beispiel verwendeten Symbole zu testen ist die Verwendung vom *Watch Window* Fenster. Dazu muss der *Watch Window* Knopf auf dem *Project Manager* Fenster betätigt werden. Danach müssen die Symbole vom Symboleditor in das *Watch window* Fenster kopiert werden:



Um die korrekte Funktion unseres Zeitschaltprogrammbeispiels zu testen, werden wir nun die Ein- und Ausschaltzeiten (*ONTIME* und *OFFTIME)* modifizieren und den Zustand des Ausganges *DailyTimer* beobachten. Um die Ein- und Ausschaltzeiten zu modifizieren bitte wie folgt vorgehen:

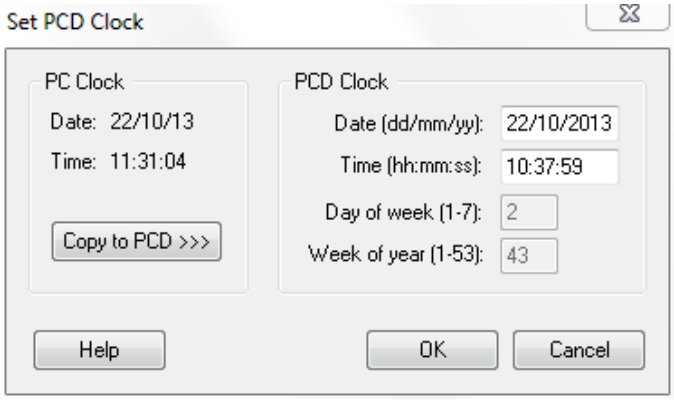


6.14.5 Echtzeituhr der PCD einstellen

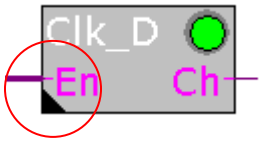
Um die Echtzeituhr der PCD auf die korrekte Zeit einzustellen muss wie folgt vorgegangen werden:

1. Anklicken des *Online configurator* Knopfes im *Project Manager* Fenster. Danach Auswahl des *Clock* Knopfes.

2. Kopieren der PC-Echtzeituhr zur PCD-Echtzeituhr mit *Copy to PCD*>>> Knopfes, oder Modifikation der PCD-Echtzeituhr in den entsprechenden Feldern von *SAIA PCD Clock*.



6.15 FBox Einstellfenster

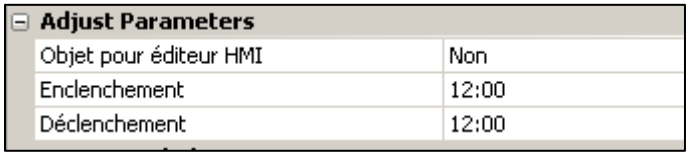


Fbox:
HLK Uhren, Uhr Tagesprogramm

Einige FBoxen verfügen über zusätzliche *Adjust Parameters*. Sie sind durch ein schwarzes Dreieck in der linken unteren Ecke gekennzeichnet. Mit diesen Parametern können besondere Eigenschaften der FBox konfiguriert werden. Das kann auch online geschehen.

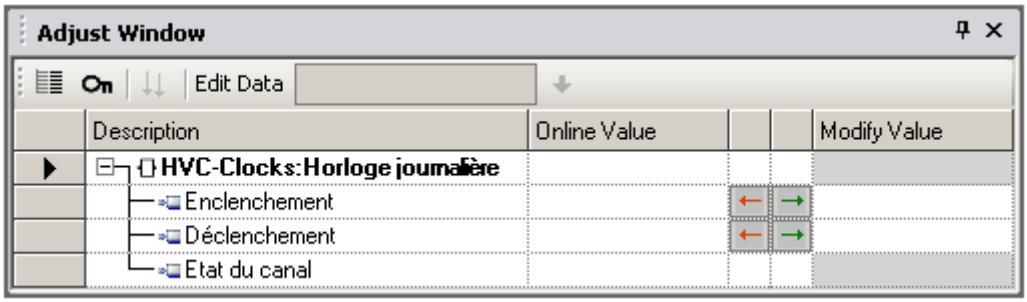
Wenn das Fenster *Properties* bereits offen ist, klicken Sie einfach auf die FBox, um deren Eigenschaften anzuzeigen. Dann werden die *Adjust Properties* angezeigt. Ist das Fenster noch nicht offen, klicken Sie mit der rechten Maustaste auf die FBox, um das Kontextmenü einzublenden und wählen Sie den Befehl *Properties.*

Offlinebearbeitung von *Adjust Parameters*



Die Offlinebearbeitung von *Adjust Parameters* erfolgt über das Fenster *Properties.* Die Werte des Parameters werden in der Fupla-Datei gespeichert. Bevor die PCD die neuen Parameter verwendet, muss das Programm und heruntergeladen werden.

Onlinebearbeitung von *Adjust Parameters*



Die Onlinebearbeitung von *Adjust Parameters* erfolgt über den Befehl *View, Adjust Window*, welcher das Fenster für die Onlineanpassung öffnet, in dem sowohl die tatsächlichen als auch die bearbeiteten Werte angezeigt werden. Die bearbeiteten Werte werden direkt in den Speicher der PCD geschrieben und nicht in der Ausgangsdatei von Fupla aktualisiert.

Wenn Fupla online ist, wird automatisch die Onlineversion von *Adjust Window* statt des Fensters *Properties* angezeigt.

6.15.1 Initialisierung der HLK FBoxen

Fbox: CVC Init,
Initialization CVC 7

Falls der Anwender FBoxen aus der HLK Bibliothek verwenden will, so muss er zwingend am Beginn der Fupla Datei eine HLK-Initialisierungs-FBox platzieren. Mit dieser Initialisierungs-FBox werden verschiedene, gemeinsame Eigenschaften der HLK-FBoxen, wie z.B. das verhalten der FBox nach den laden oder das Aufstartverhalten beim einschalten der PCD eingestellt.
Mit dem *Res* Eingang und den unten aufgeführten Parametriedaten der *Heavac 6* Initialisierungs-FBox kann die Funktionalität der anderen HLK FBoxen nach einem laden des Anwenderprogramms oder nach einem Kaltstart der PCD massgebend beeinflusst werden.

Reset	
Automatic Reset ...	Activated
Evaluate Reset Input	At startup

Zusammenhang zwischen dem Laden des Anwenderprogramms + Parametrierdatenpunkt *Automatic Reset*

Parametrierdatenpunkt *Automatic Reset* hat den Wert *Activated:*
Alle HLK FBoxen werden beim Laden des Anwenderprogramms mit den, im Anwenderprogramm definierten Werten initialisiert (Off-line Parametrierwerte werden verwendet).

Parametrierdatenpunkt *Automatic Reset* hat den Wert *Not activated:*
Alle HLK FBoxen werden beim Laden des Anwenderprogramms mit den in der PCD gespeicherten Werten initialisiert (On-line Parametrierwerte werden verwendet).
Zusammenhang zwischen Eingang *Res* + Parametrierdatenpunkt *Evaluate Reset Input*
Wenn der Eingang *Res* auf logisch 1 gesetzt wird, so werden die Parametrierdatenpunkte aller HLK FBoxen mit den im Anwenderprogramm definierten Werten initialisiert (Off-line Parametrierwerte werden verwendet).

Abhängig von der Einstellung des Parametrierdatenpunktes *Evaluate Reset Input* wird der Zustand des Eingangs *Res* nie, nur beim Kaltstart (aufstarten) der PCD oder immer in Betracht gezogen.

Grüne/rote LED der FBoxen
Einige FBoxen besitzen eine dreifarbige LED.

- LED ist grau: PCD ist nicht on-line
- LED ist rot oder grün : PCD ist on-line
- LED ist grün: FBox funktioniert einwandfrei
- LED ist rot: Fehler in der Fbox. Generell bezieht sich dieser Fehler auf unpassende Eingangssignale oder falsch verwendeten Parametriedaten der Fbox.

Eine detailliertere Beschreibung über das Verhalten der FBox und der LED ist in der entsprechenden FBox Hilfe ersichtlich.

Bemerkung:
In der HLK Bibliothek sind verschiedene Initialisierung FBoxen enthalten. (*Initialization HVC 4, …7*) Die FBox *Initialization 6* ist die neuste. Bei neuen Applikationsprogrammen wird die Verwendung der FBox *Initialisierung HLK 7* empfohlen.

6.15.2 FBoxen mit Parametrierdaten

FBox: HEAVAC clocks, Daily clock

Die Funktionalität des zuvor in diesem Kapitel programmierten Beispiels „Zeitschaltuhr“ kann mit nur einer FBox *ClK_D* aus der HLK Bibliothek realisiert werden.
Der Ausgang der FBox wird abhängig von den FBox Parametriedaten ein- oder ausgeschaltet.
Der Parametrierdatenpunkt *Objects for HMI Editor* wird nur im Zusammenhang mit den HMI Terminals verwendet. Wenn kein HMI Terminal verwendet wird, so muss dieser Parametrierdatenpunkt auf *No* gesetzt werden.
Mit dem Eingang *En* kann die Zeitschaltuhr deaktiviert werden. Wenn der Eingang *En* logisch 0 ist, wird der Ausgang *Ch* immer auf logisch 0 gesetzt.

Adjust Parameters	
Objet pour éditeur HMI	Non
Enclenchement	12:00
Déclenchement	12:00

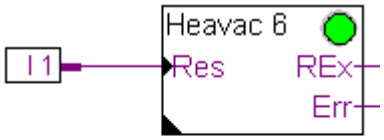
6.15.3 Kleinst HLK Applikation

Um die Funktionalität der Parametrierdaten zu testen, werden wir das Beispiel „Zeitschaltuhr“ welches wir am Beginn dieses Kapitels programmiert haben nochmals programmieren. Diesmal werden wir aber die FBoxen der HLK Bibliothek benutzen. Es werden nur zwei FBoxen benötigt. Erstellen Sie das Programm wie unten dargestellt, verarbeiten (build) und laden Sie das Programm in die PCD, gehen Sie on-line.

Rebuild All Files

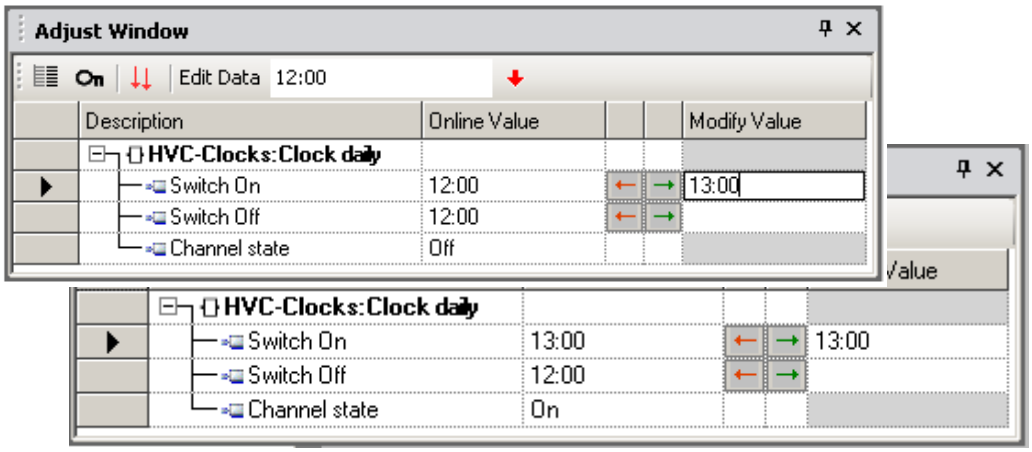
Download Program

Go Online



Bemerkung:
Die Fbox *Initialization HVC 7* muss unabhängig von der Anzahl verwendeter HLK FBoxen nur einmal auf der ersten Fupla Seite programmiert werden.

6.15.4 Modifikation der Einstell-Parameter, wenn online



Die Online-Modifikation der Einstell-Parameter wird durch die Ansicht *Adjust Parameters* unterstützt. Es zeigt die Parameter der ausgewählten FBox in einem Fenster, welches ähnlich wie das Watch-Fenster funktioniert.

Die Spalte "Beschreibung" beschreibt die Einstell-Parameter. Die Online Spalte zeigt den Wert des Parameters aus dem Speicher der PCD. Die "Wert ändern“ Spalte ermöglicht das eingeben neuer Werte, welche einzeln oder gleichzeitig in die PCD geschrieben

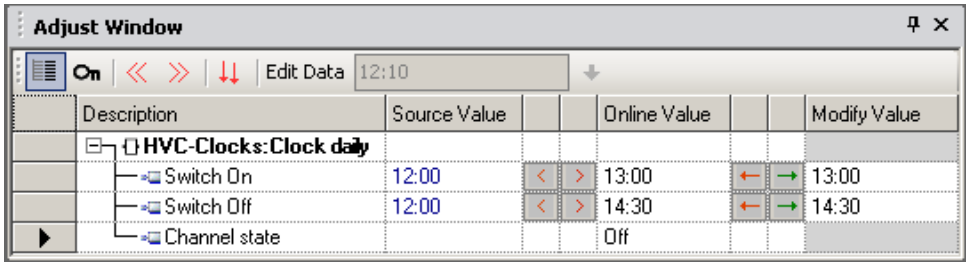
Schreibt einen einzigen Parameter in die PCD

Schreibt alle geänderten Paramter gleichzeitig.

Es ist auch möglich, einen Parameter aus zu wählen und ihn im *Edit Data* Feld in der Symbolleiste zu ändern.

Die Werte der veränderten Parameter werden direkt in den PCD Speicher geschrieben, sie werden den Inhalt der ursprünglichen Fupla Datei nicht ändern.

6.15.5 Wiederherstellen der originell parameters aus der Fupla Datei



Nach online Modifikationen der Einstell-Parameter, ist es möglich, die original Werte aus der Fupla Datei wieder her zu stellen. Die *Show Source Value*-Taste füllt die Spalte "Source Value" mit den original Werten aus der Fupla Datei. Siehe das FBox *Properties window*.

 Stellt einen einzelnen Parameter wieder her.

 Stellt alle Parameter für die aktuelle Fbox aus der Fupla Datei wieder her.

Sie können auch die Menübefehle Online, Schreiben FBox Einstellen der Parameter-Download aus dem Einstellen der Parameter für Fupla Datei benutzen

6.15.6 Speichern der Online-Parameter in Fupla Datei

Wenn die Parameter, welche online bearbeitet worden sind, passen, können diese auch in die Fupla Datei gespeichert werden.

 Speichern eines einzehlnen Parameters

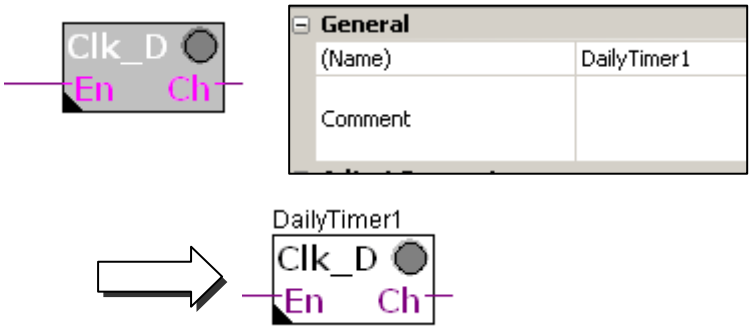
 Speichern alle Parameter von der aktuellen FBox

Sie können auch den Menübefehl Online, Lesen FBox Einstellen-Parameter, für das Hochladen der Einstellen-Parameter von der PCD benutzen und diese in die Fupla Datei speichern.

6.15.7 Definition von Symbolnamen für die Referenzierung von FBoxen

Teilweise ist es erforderlich, dass Parametrierwerte der FBoxen im Fupla Programm, von SCADA Systemen oder anderen Geräten gelesen oder geschrieben werden müssen.
Dies ist möglich wenn ein Symbolname für die FBox definiert worden ist, welcher die Flags und Register referenziert die hinter den Parametrierwerten verwendet werden.
Um diesen Symbolnamen zu definieren muss der Mauszeiger auf die Mitte der FBox positioniert werden und durch betätigen der rechten Maustaste das Menü geöffnet werden.
Auswahl von Fbox Properties…
Definition eines Symbolnamens welcher dann die Referenz zu den in der FBox verwendeten Parametrierwerten darstellt.

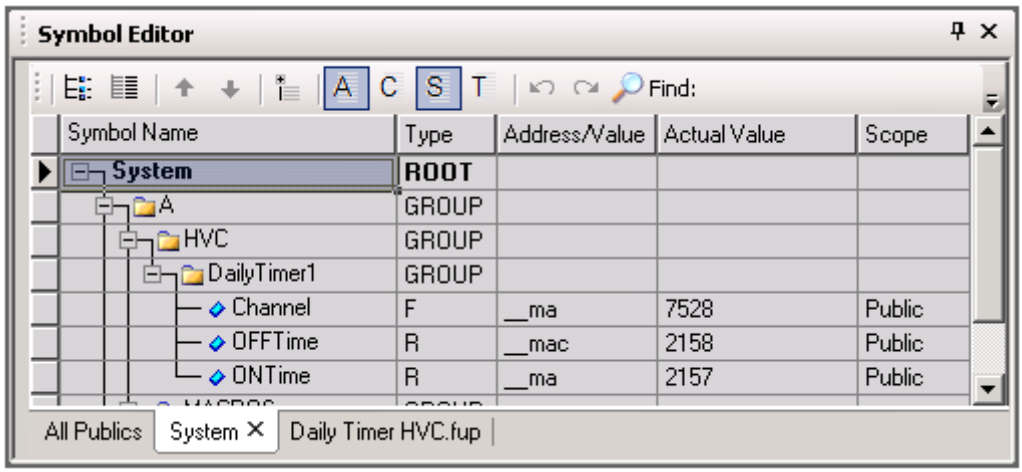
Öffnen Sie das Fenster *Properties* der FBox und füllen Sie das Feld *Name* im Bereich *General* aus, um diese Symbole zu definieren.



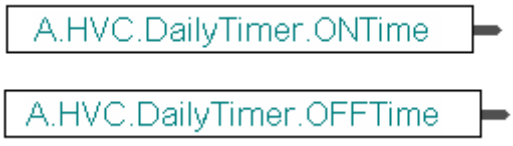
Verarbeiten Sie das Program und öffnen Sie den Symbol Editor. Öffnen Sie die Symbolseite *System*.

Show or Hide Symbol Editor

Im Zusammenhang mit der HLK Bibliothek entsprechen die Systemsymbole den Parametrierwerten der FBoxen. Diese sind in der Gruppe A.HVC.Name gruppiert, wobei der Name dem Namen der FBox entspricht.

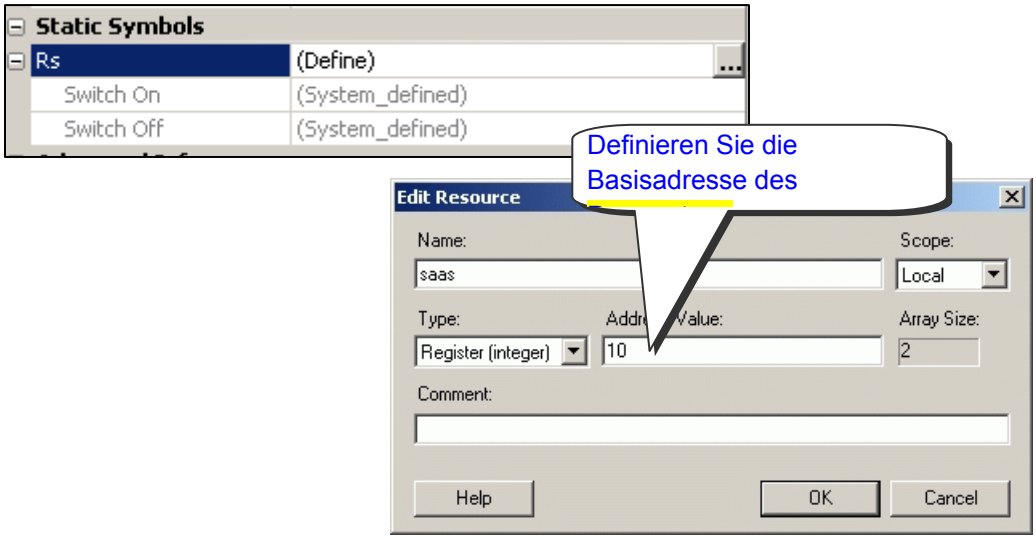


Nun müssen nur noch diese Systemsymbole in Fupla verwendet werden.



6.15.8 Definition der absoluten Adressen für die Parametrierdaten

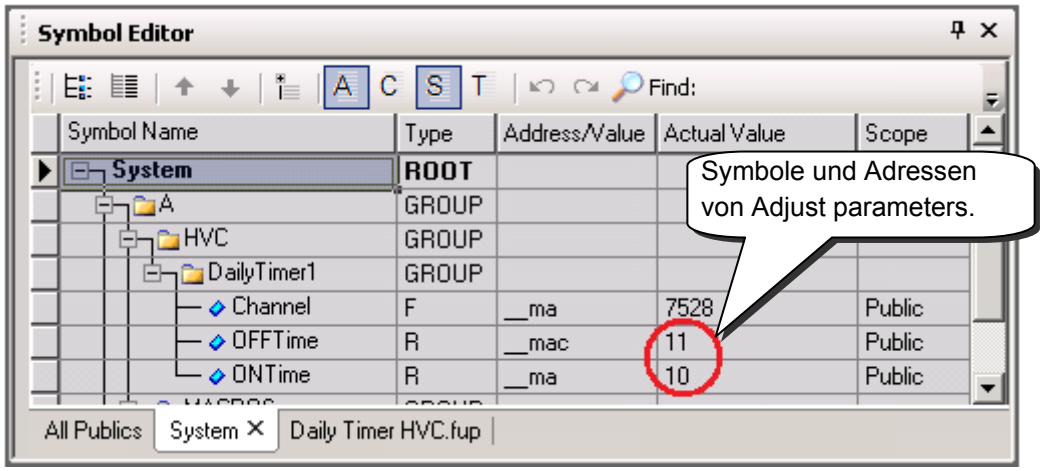
Definieren Sie das Systemsymbol für die Adjust Parameters wie oben beschrieben und fügen Sie die Adresse aus den *Properties* der FBox hinzu. Wählen Sie die Zeile (*Define*) und klicken Sie auf die Schaltfläche … am Ende der Zeile.



Verarbeiten Sie das Program und öffnen Sie den Symbol Editor. Den Systemsymbolen wurden die unten angezeigten Registeradressen zugeordnet.

Rebuild All Files

Show or Hide Symbol Editor



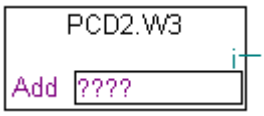
6.16 Inbetriebnahme eines Analogmoduls

Zum Lesen und Schreiben analoger Werte ist ein kleines Programm für jedes analoge Modul erforderlich. Damit werden die Verstärkung der Kanäle und die A/D- oder D/A-Umwandlung gesteuert. Das Programm wird entweder von einer FBox geliefert oder durch den vom *Device Configurator* erstellten Image-Speicher.

6.16.1 Erfassen einer Analogmessung

Die bisher vorgestellten Abtastprogramme arbeiten mit digitalen Ein- und Ausgängen, die ihre Adressen oder Symbole an den Rand des FUPLA-Editors setzen.

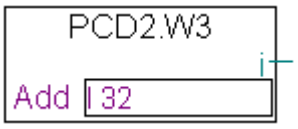
Mit analogen Ein-/oder Ausgangsmodulen muss eine FBox zum Erfassen des Analogwertes benutzt werden. Diese FBoxen stehen mit Bibliotheken zur Verfügung: *Analogmodulen oder HEAVAC-Analog* .



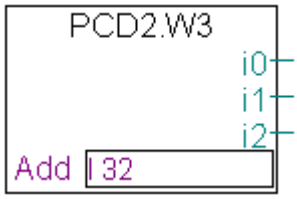
Diese Bibliotheken bieten eine grosse Vielzahl an FBoxen, wobei jede einem Analogmodul entspricht. Der im *FBox Selector* erscheinende Name stimmt mit der Modulreferenznummer überein.

Analoge Fboxen sind erweiterbar. Der Benutzer kann die Anzahl der von einer Applikation benötigten Messkanäle definieren. Auch wenn einige Messkanäle nicht benutzt werden oder wenn ein zusätzlicher Kanal hinzugefügt wird, kann man mithilfe des Kontextmenüs *Resize FBox* seine Abmessungen einstellen. Allerdings kann man eine Fbox auch mit der maximalen Anzahl von Kanälen definieren, selbst wenn nicht alle benutzt werden.

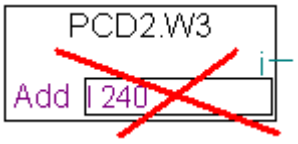
Das Feld *Add* ermöglicht die Basisadresse des zu definierenden Analogmoduls. Diese Adresse gibt an, wo das Modul im PCD eingesetzt worden ist: 0, 16, 32, …



Analogmessungen sind in den FBox-Eingängen i 0 bis i 7. möglich. Diese können direkt mit anderen FBoxen verbunden werden oder die Werte können in einem Register gespeichert werden. Das Speichern eines Wertes in ein Register ist eine gute Lösung, vor allem, wenn der Wert auf vielen verschiedenen Seiten des Programms oder Graftec Step und Transitionnen benutzt wird.

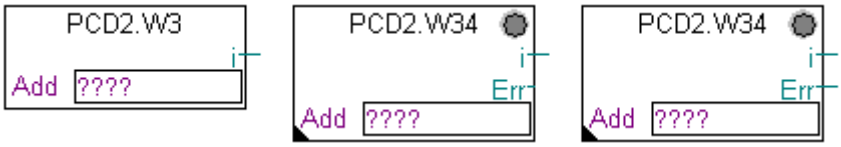


Achtung:
Achten Sie darauf, dass Sie niemals mehr als eine FBox pro Analogmodul definieren und dass Sie nie das Analogmodul in der PCD-Watchdog-Adresse (255) einsetzen. Ansonsten kann der vom Modul gelieferte Wert falsch sein.



6.16.2 Beispiel für PCD2.W340 Analog-Eingangsmodule

Wenn der PCD mit einem PCD2.W340 ausgestattet ist, der 8 universelle Eingangskanäle besitzt, kann der Benutzer eine der folgenden FUPLA- FBoxen benutzen und die erforderliche Anzahl der Messkanäle einrichten.



FBoxen: PCD2.W3, PCD2.W34, PCD2.W34 with error

Die Messeinheiten hängen vom Modul, der FBox und den gewählten Einstellungsparametern ab.

Das PCD2.W340 ist ein Universalmodul. Es unterstützt die Messung der Bereiche 0..10V, 0..2.5V, 0..20 mA und Pt/Ni 1000 Temperatursonden. Zum Festlegen der Messreihe muss eine Brücke auf dem Modul gewählt werden. Die Auflösung beträgt 12 Bit, was mit 4095 gemessenen Werten gleichzusetzen ist. (Einzelheiten zu diesen Modulen finden Sie in Ihrem PCD-Hardwarehandbuch).

Die *PCD2.W3* FBox bietet eine Rohmessung. Bei diesem Modul mit einer Auflösung von 12 Bits, die einem gemessenen Wert zwischen 0 und 4095 entspricht, muss der Benutzer die Messungen in eine physikalische Standardeinheit umrechnen.

Die *PCD2.W34* FBox ist etwas fortgeschrittener. Hier können in einem Parametrierfenster die Messeinheiten für jeden einzelnen Kanal festgelegt werden. Die FBox-LED leuchtet rot, sobald eine der Messungen den eingestellten Bereich überschreitet: Kurzschluss oder Bruch im Sondenkabel. Der Fehler kann über die Schaltfläche *Acknowledge* im Parametrierfenster zurückgestellt werden.

Adjust Parameters	
Configuration channel 0 t	
Ch 0 / Mode or sensor type	mV
Ch 1 / Mode or sensor type	Ni 1000
Ch 2 / Mode or sensor type	uA

Die *PCD2.W34 with error* FBox bietet dieselben Funktionen zum Umrechnen der Masseinheiten, hat aber zusätzlich noch einen Fehlerausgang, der den fehlerhaften Kanal anzeigt sowie einen zusätzlichen Regulierungsparameter zum Festlegen eines Standardwertes im Störungsfalle.

Adjust Parameters	
Output when in error	Last value

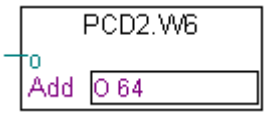
6.16.3 Beispiel für PCD2.W610 Analog-Ausgangsmodule

Hier gilt dasselbe Prinzip wie für die Eingangsmodule: Der Benutzer setzt eine dem Ausgangsmodul entsprechende FBox auf die FUPLA-Seite, zieht es zur Nummer der Ausgangskanäle und legt die Basisadresse des Moduls fest.

Anders als bei den Eingangs-FBoxen, wird der Sollwert der Analogausgänge auf der linken Seite der FBox angezeigt.

Diese Eingänge können direkt mit anderen FBoxen oder mit Registern verbunden werden, die am Rand „Eingang“ der FUPLA-Seite festgelegt sind.

Wenn der PCD mit einem PCD2.W610-Modul ausgerüstet ist, der über 4 universelle Analogausgänge verfügt, kann die nachstehende FBox für einen Stromausgang von 0…20 mA oder einer Spannung von 0…10 V benutzt werden.



FBox: PCD2.W6

Zum Festlegen der Ausgangsreihe muss eine Brücke auf dem Modul gewählt werden. Die Auflösung dieses Moduls beträgt 12 Bit und ist mit 4095 gemessenen Sollwerten gleichzusetzen. Der Ganzzahlwert am FBox-Eingang bestimmt die Ausgangsspannung oder den Ausgangsstrom des Kanals:

Eingangswert an der Fbox	Ausgangsspannung [V]	Ausgangsstrom [mA]
0	0	0
2047	5	10
4095	10	20

Andere FBoxen verfügen über ein Parametrierungsfenster zum Einstellen der Sollwertbereiche, die auf den FBox-Eingang angewendet werden (z.B. die FBox für das PCD2.W605-Modul das über 6 elektrisch isolierte Ausgänge zwischen 0…10V verfügt):

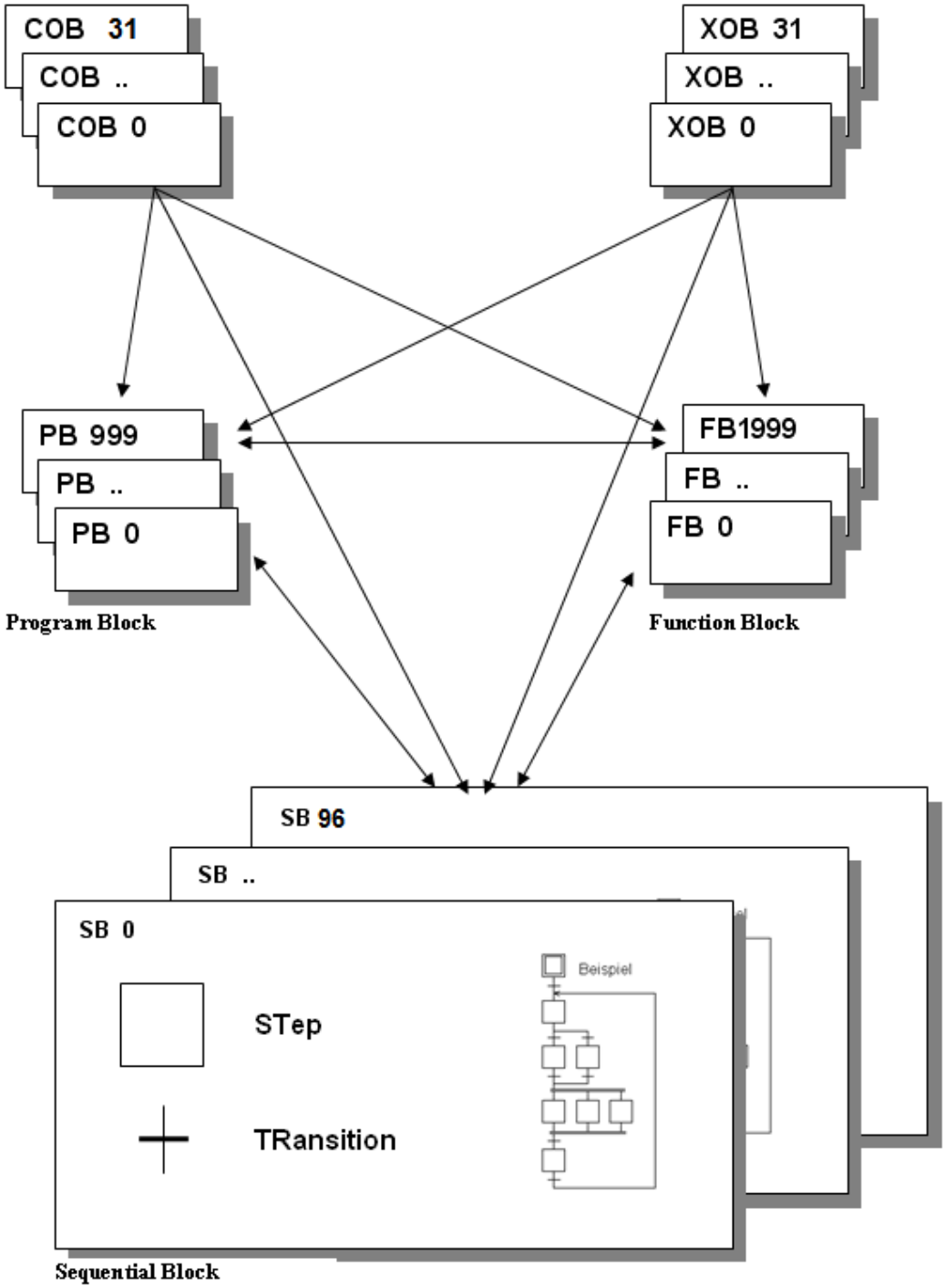
Mit den Parametern *User scaling 0 und 100%* kann man Werte für die minimalen und maximalen Kanalspannungen festlegen, die auf den FBox-Eingang angewendet werden.
Der *Reset value*-Parameter entspricht dem auf dem Kanal angewendeten Wert, wenn der PCD unter Spannung steht.

Inhaltsverzeichnis

7 Programmstrukturen

Der Erfolg eines guten Programms ist verbunden mit seiner Struktur. Sie macht das Programm einfach, schnell zu unterhalten und zu entwickeln.

Die Programmiersprache SAIA PCD ist eine strukturierte Sprache. Sie schlägt verschiedene Organisationsblöcke vor, in denen der Benutzer die Instruktionen seiner Anwendung einbringt. Jeder Block bietet dem Benutzer einen besonderen Dienst an. Die verfügbaren Organisationsblöcke sind folgende: Zyklische Organisations-Blöcke (COB), Funktions-Blöcke (FB), Programm Blöcke (PB), Ausnahme Blöcke (XOB) und die Sequenziell Blöcke (SB).



7.1 Cyclic Organization Block (COB 0 to 31)

Zyklische Organisationsblöcke (COBs) sind die “Aufgaben” des Programms, die kontinuierlich ohne Programmschleifen und ohne auf Vorgänge zu warten laufen. Wenn die PCD startet, führt sie der Reihe nach in einer kontinuierlichen Schleife die Instruktionen in jedem COB aus. Jedes Programm muss mindestens einen COB haben. Da jeder COB zyklisch ausgeführt wird, kann er regelmässig nach wichtigen Vorgängen wie beispielsweise Eingangssignalen, Endschaltern, Nothaltetastern usw. suchen.

Es ist wichtig, das Sie das Konzept der COBs verstehen. Da alle COBs ununterbrochen laufen müssen, sollten sie keine Warteschleifen oder Verzögerungen enthalten, da diese die regelmäßige Abwicklung von Vorgängen verhindern würden.

Wenn Sie den Fupla Editor (S-Fup) verwenden, wird automatisch ein COB erstellt. Fupla Programme sind Funktionsplan-Programme, die zyklisch ausgeführt werden und daher gut für COBs geeignet sind.

Wenn das Programm in Instruktionsliste geschrieben ist, beginnt der Block mit einer COB-Anweisung und endet mit ECOB (End COB). Der Code des Blocks steht zwischen diesen beiden Instruktionen. Am Anfang jedes COBs ist der Akkumulator (ACCU) immer hoch (1). Wie Sie noch sehen werden, ist das bei zyklischen Programmen von großer Bedeutung.

Die Instruktion des COBs hat zwei Operanden. Der erste ist die COB-Nummer, der zweite die Überwachungszeit des COBs. Beträgt die Überwachungszeit 0, wird die Ausführungszeit des COBs nicht überwacht. Beträgt die Überwachungszeit nicht 0, wird das Timeout in Zehntelsekunden dargestellt (z.B. 10 = 100mS, 100 = 1S). Wird der COB nicht innerhalb dieses Zeitraums beendet, wird der Exception Organisationsblock XOB 11 aufgerufen. Am Ende von XOB 11, wird der unterbrochene COB an der Stelle aufgenommen, an der der Abbruch stattfand, und die Überwachungszeit beginnt von Neuem. Die Leuchtdiode „Error“ leuchtet nicht, da der Fehler vom Programm behoben wurde.

Wenn der XOB 11 nicht programmiert ist, leuchtet die Leuchtdiode “Error” der PCD und die Ausführung wird mit dem nächsten COB fortgesetzt, der seine eigene Überwachungszeit startet. Im nächsten Programmzyklus beginnt der unterbrochene COB an der Stelle, an der der Abbruch stattfand und die Überwachungszeit beginnt von Neuem.

In Fupla-Programmen wird die Überwachungszeit mit dem Befehl *Block, Properties* konfiguriert.

Anmerkung: Jeder COB hat seinen eigenen *Index Register*.

7.1.1 Creating a block

Eine Fupla-Datei kann mehrere Blöcke enthalten, die mit dem Menü *Block* hinzugefügt, gelöscht oder bearbeitet werden können.

General	
(Name)	COB_3A87C0D7
Type	COB
Comment	
Number	
Scope	File
COB Supervision Time	0

Der Befehl *Block, Properties* öffnet das Fenster, welches links zu sehen ist.

Name	Symbolname des Blocks
Type	Blocktyp:COB, PB, FB, XOB
Comment	Freitextfeld
Number	Blocknummer. COBs sind beispielsweise von 0 bis 31 durchnummeriert, PBs von 0 bis 999. Standardmässig ist die Blocknummer, außer bei XOBs, leer (dynamische Zuordnung). Wird die Blocknummer dynamisch zugeordnet, wird die tatsächliche Blocknummer während der Vorbereitung zugewiesen.
Scope	Reichweite des Symbols (local oder global). Ein globales Symbol ist auch für andere Dateien zugänglich.
COB Supervision Time	Abbruchzeit des COBs, in Hundertstelsekunden

7.1.2 Beispiel

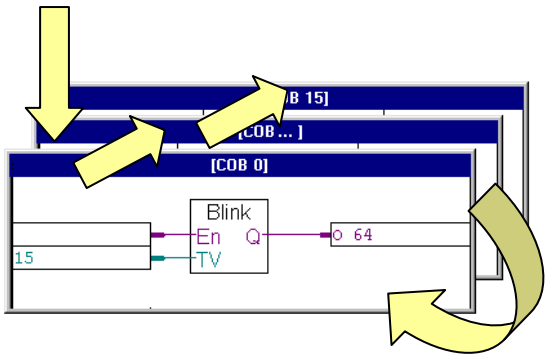
Nachfolgend ist ein Beispiel-Programm in sowohl IL als auch FUPLA abgebildet, dass den Ausgang 64 mit einer Frequenz von 1,5 sec blinken lässt. Das Programm ist in COB 0 hinterlegt und wird folglich vor allen nachfolgenden COBs 1-15 ausgeführt.

IL Programm

```
COB   0     ; Start COB 0
      0     ; Ueberwachungszeit
STL   T 1   ; Wenn Timer T1 = 0,
LD    T 1   ; Lade ihn mit 1.5
            ;   sec.
      15
COM   O 64  ; Und Ausgang 64
            ;    wechseln
ECOB        ; COB 0 endet hier


COB   15    ; Nächsten COB
            ;    ausführen
      0
      NOP
ECOB
```

Fupla Programm

START UP



Fbox: *Blinker, Blinker symmetrisch*

7.2 Programmblöcke (PB) und Funktionsblöcke (FB)

Die Programmiersprache ermöglicht die Arbeit mit Programmblöcken (PB 0 bis 999) und Funktionsblöcken (FB 0 bis 1999). Damit können Struktur und Hierarchie des Programms leicht organisiert werden.

Der einzige Unterschied zwischen PBs und FBs besteht darin, dass ein FB mit Parametern aufgerufen werden kann, was bei einem FB nicht möglich ist.

FBs sind eine ideale Lösung für die Entwicklung von Bibliotheken, die in vielen Projekten verwendet werden können. So wird die Entwicklungszeit gesenkt.

PBs und FBs müssen von anderen Blöcken aus aufgerufen werden (COB, PB, FB, SB oder XOB).

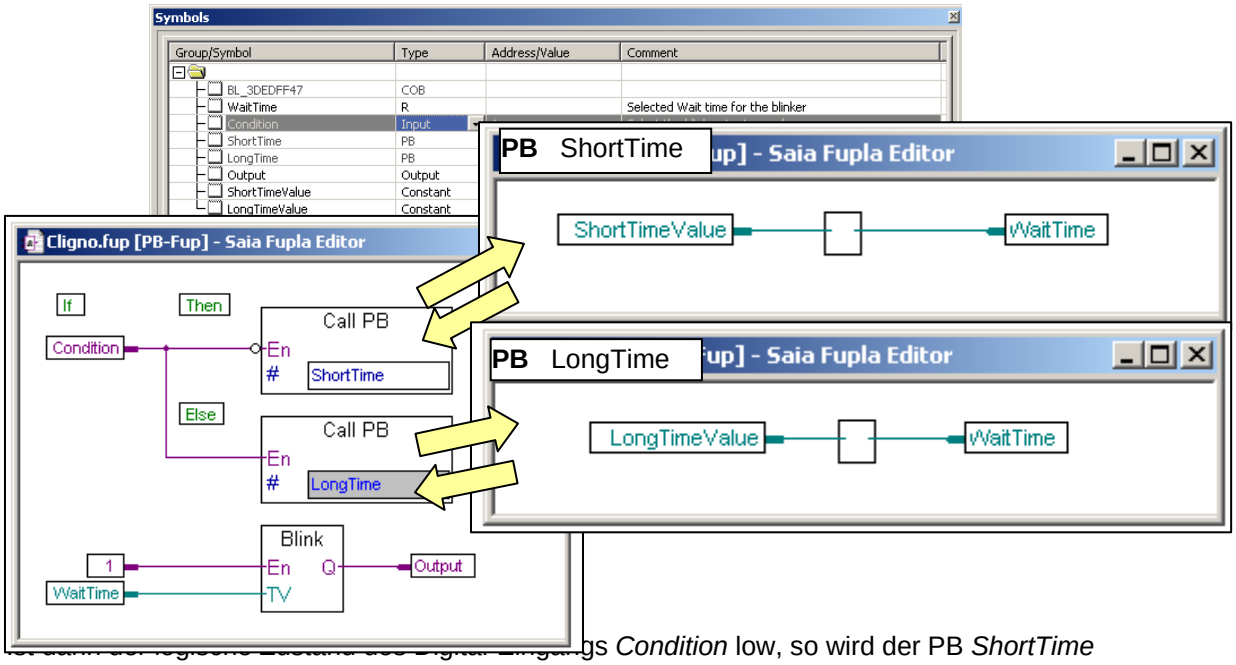
Es gibt zwei Arten von Verbindungen: Solche die nicht immer weitergeleitet werden müssen und solche, die immer weitergeleitete werden müssen. Ersteres ist vom Ergebnis der logischen Operation abhängig, letzteres ist unabhängig von Einstellungen. In einem Programm können dieselben PBs und FBs mehrfach aufgerufen werden.

Ein PB/FB kann auch andere PBs/FBs aufrufen. Das Maximum beträgt 31 Aufrufe. Wird die maximale Anzahl überschritten, wird der Exception Organisationsblock XOB 10 aufgerufen, wenn dieser programmiert wurde.

7.2.1 Beispiel

Nachfolgend ist ein Beispiel-Programm in FUPLA abgebildet, das anhand eines Eingangswertes die Blinkgeschwindigkeit eines Ausgangs verändert.

Fupla Programm:



ist dann der logische Zustand des Digital-Eingangs *Condition* low, so wird der PB *ShortTime* ausgeführt und die Konstante *ShortTimeValue=5* in das Register *WaitTime* übertragen. Anderenfalls erfolgt der Aufruf von PB *LongTime*, wodurch die Konstante *LongTimeValue=15* nach *WaitTime* übertragen wird, dessen Wert das Blinkintervall des PB *Blink* bestimmt. Die Ausführung von *Blink* nach den beiden PB-Calls sichert hierbei die korrekte Intitialisierung von *WaitTime* nach einem Kaltstart.

Erstellen Sie mit dem Menübefehl *Block, New* einen neuen Block und übernehmen Sie den Namen des Blocks aus dem Fenster *Properties*, wenn das Programm im Fulpa-Editor bearbeitet wird.
Die FBoxen *Call PB* befinden sich in der Familie *Block Call* im Fenster *FBox Selector*.

IL Programm:

```
;Two-speed Blinker
LongTime            EQU    PB 1
ShortTime           EQU    PB 2
ShortTimeValue      EQU    K 5        ;0,5s
LongTimeValue       EQU    K 15       ;1,5s
Condition           EQU    I 1
Output              EQU    O 32
WaitTime            EQU    T

            COB    0
                   0
            STH    Condition      ;IF Condition = High)
            CPB    L ShortTime    ;  THEN Call PB ShortTime
            CPB    H LongTime     ;  ELSE Call PB LongTime
            ECOB

            PB     ShortTime
            STL    WaitTime       ;IF WaitTime = Low
            LD     WaitTime       ;  load it with a short value
                   ShortTimeValue
            COM    Output         ;  Invert the output
            EPB

            PB     LongTime
            STL    WaitTime       ;IF WaitTime = Low
            LD     WaitTime       ;  load it with a long value
                   LongTimeValue
            COM    Output         ;  Invert the output
            EPB
```

Wenn das Programm in Instruktionsliste geschrieben ist, beginnt der Block mit einer PB- oder FB-Anweisung, dem Blocknamen oder einer Nummer als Operand. Das Ende des Blocks wird durch EPB oder EFB angezeigt. Der Programmcode befindet sich zwischen diesen beiden Instruktionen.

Am Beginn jedes Blocks ist der *ACCU* immer *High*. Wenn ein PB oder FB aufgerufen wird, werden die Inhalte des *ACCUs* gespeichert, auf High für den Beginn des aufgerufenen Blocks gesetzt und wiederhergestellt, wenn der Aufruf zurückkommt. Damit können logische IF...THEN...ELSE-Strukturen programmiert werden, in denen die Aufrufe der Blöcke unter bestimmten Bedingungen weitegeleitet werden.

Anmerkungen:
Die Statuskennzeichen (Fehler, Negativ, Positiv und Null) werden nicht gespeichert und wiederhergestellt, wenn der PB eines FBs aufgerufen wird. Es wird lediglich der Status des *ACCUs* gespeichert. Wenn die Statuskennzeichen gespeichert werden sollen, müssen sie in normale Kennzeichen kopiert werden.
PBs und FBs sollten, genau wie COBs, keine Warteschleifen und Verzögerungen beinhalten. Sprünge aus dem Block sind unzulässig. Es sollten generell keine Sprünge nach hinten verwendet werden.

7.2.2 Funktionsblöcke mit Parametern

Das folgende Beispiel zeigt den FB einer Blink-Applikation. Der Aufruf des FBs erfolgt dabei zweimal, um beim ersten Mal die „Blinkrate“ des Ausgangs 64 auf 1,5 sec zu stellen und beim zweiten Mal die des Ausgangs 65 auf 3 sec abzuändern.

```
            FB    1           ; Start des FB 1

tempo       LDEF  = 1         ; [T] Adresse Timer
delay       LDEF  = 2         ; [W] Adresse Pausenzeit zwischen
                              ;      2 Blinkzyklen
Blinker     LDEF  = 3         ; [O] Adresse Blinker

            STL   = tempo     ; Wenn Timer tempo abgelaufen ist
            LDL   = tempo     ; Dann lade Timer tempo mit
                              ;    Pausenzeit delay
                  = delay
            COM   = blinker   ; Und invertiere den Ausgang
                              ;     Blinker
            EFB               ; FB-Ende


            COB   0
                  0

            CFB   1           ; Erster Aufruf des FB  1
                  T 1
                  15
                  O 64

            CFB   1           ; Zweiter Aufruf des FB 1
                  T 2
                  30
                  O 65
            ECOB
```

As already stated, the only difference between PBs and FBs is that FBs can be called with parameters. The CFB instruction is followed by the list of parameters, numbered from 1 to a maximum of 255. Inside the black, parameter numbers can optionally be defined with symbol names, which are local to the block.

Symbols for FB parameters are indicated by '=' followed by the parameter number. For example, `STL = 1`. Or you can define a symbol with a value "`= 1`", as shown in the example above.

Note: Fupla programs do not support calling FBs with parameters.

> Fast alle Instruktion können sich auf FB-Parameter beziehen. Nur die Instruktion LD (Load) bildet eine Ausnahme, weil ein 32-bit-Operand benötigt wird und FB-Parameter 16-bit-Operanden sind. Ein 32-bit-Wert kann mit Hilfe von zwei LDH- und LDL-Instruktionen (Load High Word und Load Low Word) übertragen werden, wobei jeder Transfer 16 Bits umfasst.

Bei geschachtelten FB-Aufrufen können Parameter direkt von einem zum nächsten Aufruf weitergeleitet werden:

```
COB   1
      0
CFB   1
      R 1  ;parameter 1 = R 1
EFB
```

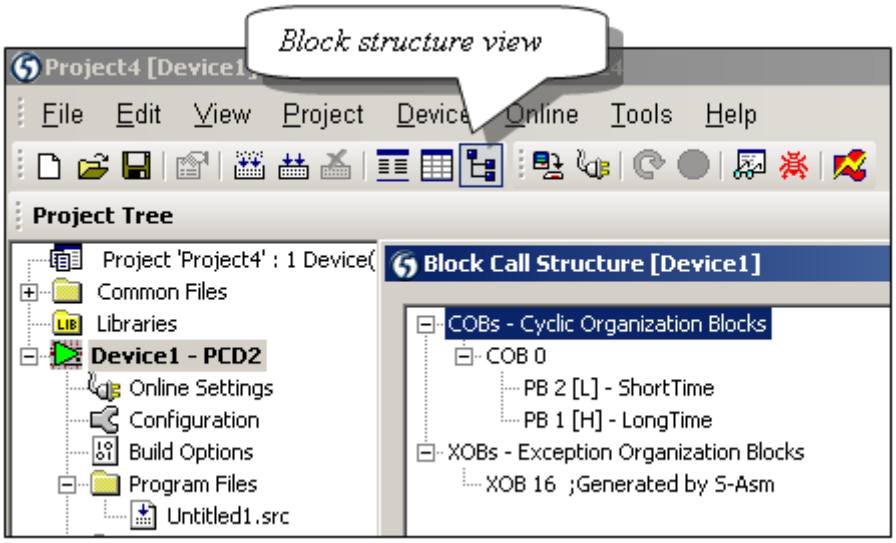
```
FB    1
CFB   2
      =1 ;call FB 2 with FB 1's parameter 1
EFB
```

```
FB    2
INC   =1  ;FB 1's parameter 1
CFB   3
      =1  ;call FB 3 with FB 1's parameter 1
EFB
```

```
FB    3
DEC   =1  ;FB 1's parameter 1 = R 1
EFB
```

7.3 Ansicht der Block-Struktur

Die Struktur eines Programms lässt sich mit Hilfe des „Block-Struktur Ansicht“- Icons in der Saia PG5 Project Manager Werkzeugleiste anzeigen. Hiermit lässt sich z.B. überprüfen, welcher PB, FB oder SB von welchem COB aufgerufen wird. Das unten aufgeführte Beispiel zeigt z.B., dass der FB 1 zweimal von COB 0 aufgerufen wird.

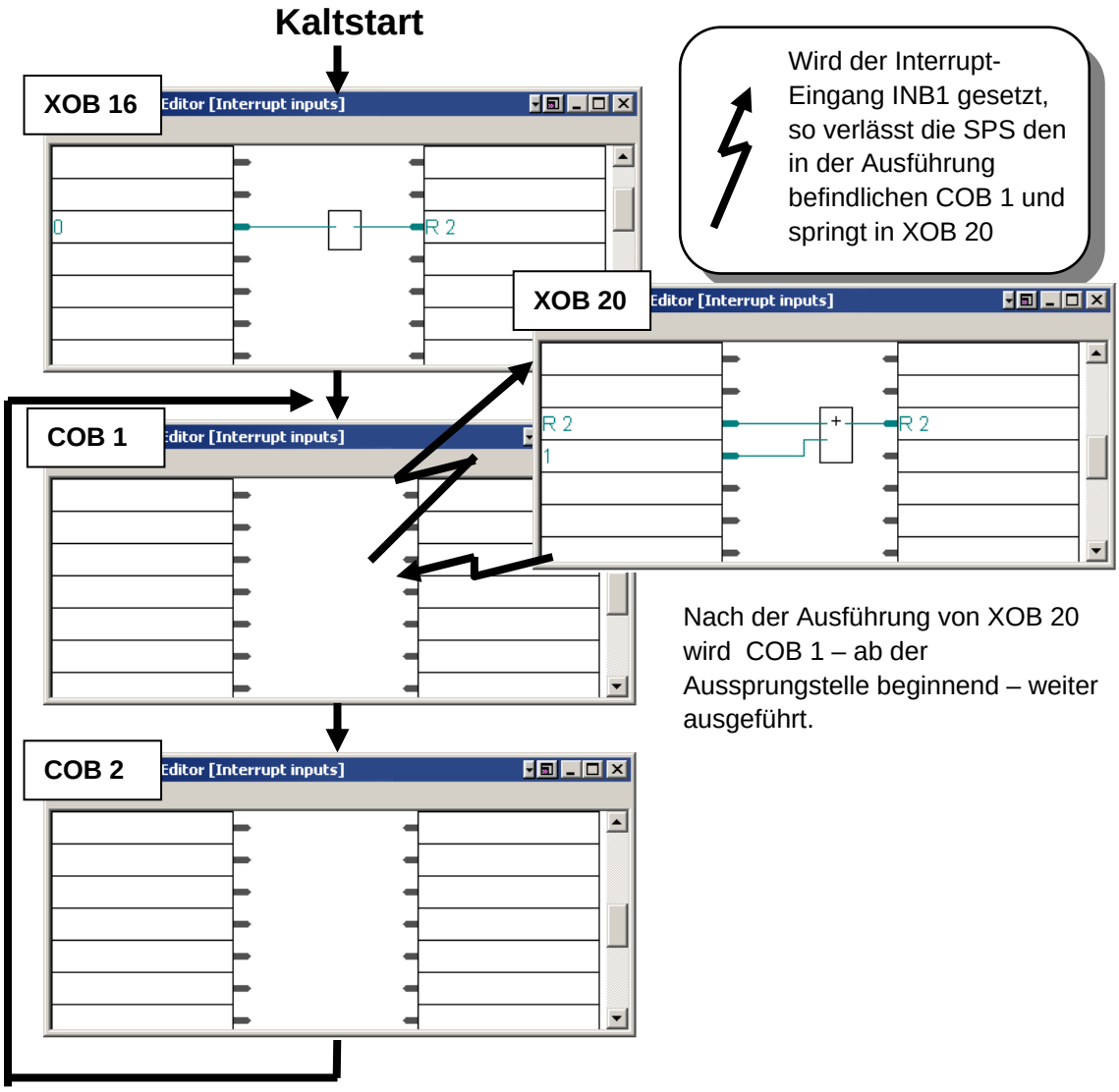


7.4 **Exception Block** (XOB)

XOBs sind eigenständige Programmblöcke, die automatisch beim Auftreten von Hard- und Software-Fehlern (internen Events), wie auch externen Events ausgeführt werden. Jedem Event ist dabei ein spezieller XOB zugeordnet, dessen freiprogrammierbarer Inhalt die Aktion festlegt. Die Zuordnung zwischen Event und XOB-Nummer kann vom Benutzer nicht geändert werden kann.

Beispiel:
Nach dem Kaltstart ist Register 2 mit Null zu initialisieren, das anschliessend Impulse am Interrupt-Eingang INB1 zählt.

Hinweis: für diese Aufgabe werden keine Programmteile in den COBs benötigt.



Nach der Ausführung von XOB 20 wird COB 1 – ab der Aussprungstelle beginnend – weiter ausgeführt.

Beispiel:
Wurde die Batterie bei ausgeschalteter PCD entfernt, so leuchtet die Error-Led bei Programm-Start auf, sofern XOB 2 nicht programmiert ist. Anderenfalls wird XOB 2 ausgeführt und die Error-Led bleibt dunkel.

7.4.1 Alle XOBs der PCD-Familie in einer Kurzübersicht

XOB	Kurzbeschreibung	Priorität
0	Spannungsversorgungs-Problem in der PCD (PCD6) oder Watchdog (PCD1/2)	4
1	Spannungsversorgungs-Problem im Erweiterungsgehäuse (PCD 6)	2
2	Batterie-Pegel ungenügend	2
4	Paritäts-Fehler auf dem Adress-Bus (PCD6)	1
5	I/O-Modul antwortet nicht (PCD4/6)	1
7	Systemüberlastung aufgrund zuvieler Events	3
8	Falsche Instruktion	4
9	Zuviele aktive Graftec-Verzweigungen	1
10	Mehr als 7 verschachtelte PB/FB-Aufrufe	1
11	COB-Monitor-Zeit überschritten	3
12	Index-Register-Überlauf	1
13	Error-Flag gesetzt	1
14	Zyklischer Interrupt	3
15	Zyklischer Interrupt	3
16	Kalt-Start	4
17	Interrupt via S-Bus	3
18	Interrupt via S-Bus	3
19	Interrupt via S-Bus	3
20	Interrupt via INB1	3
21	Interrupt Eingang	3
22	Interrupt Eingang	3
23	Interrupt Eingang	3
25	Interrupt via INB2	3
26	Zyklischer Interrupt	2
27	Zyklischer Interrupt	2
28	Zyklischer Interrupt	2
29	Zyklischer Interrupt	2
30	Keine Verbindung zum RIO	1

Tritt ein Event auf und der zugehörige XOB ist nicht programmiert, so wird das Anwender-Programm fortgesetzt und die Error-Led auf der Frontplatte der PCD eingeschaltet.

Ist hingegen der XOB programmiert, so wird dieser ausgeführt und die Error-Led bleibt dunkel.

Eine Prioritätssteuerung trägt dafür Sorge, dass stets die XOBs mit der höchsten Prioritätsstufe 4 vor denen mit niedrigeren Stufen ausgeführt werden.

XOBs zur Fehlerdiagnose und ihre Programmierung wird in den folgenden Kapiteln beschrieben. XOBs können nicht direkt vom Nutzerprogramm aufgerufen werden.

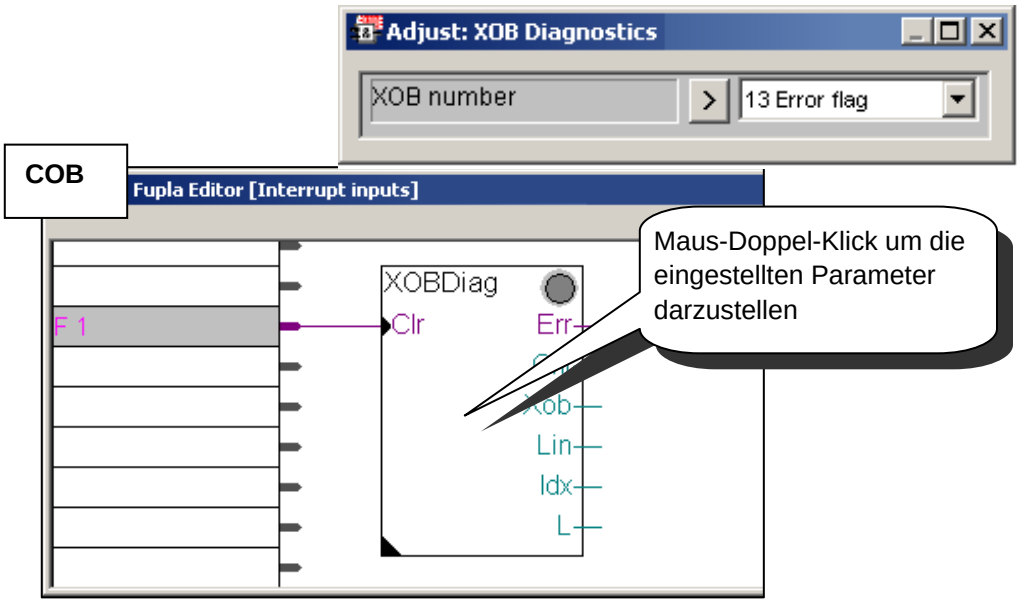
7.4.2 Anwendung der XOBs

Folgende Fehler in der Programm-Struktur lassen sich durch spezielle XOBs einfach herausfinden:

- Fehler in Programmzeilen
- Überschreitung der Verschachtelungstiefe von grösser 7
- Überschreitung der max. 32 aktiven Transitionen in Graftec Programmen
- Endlosschleifen
- Arithmetische Fehler
- Kommunikations-Fehler

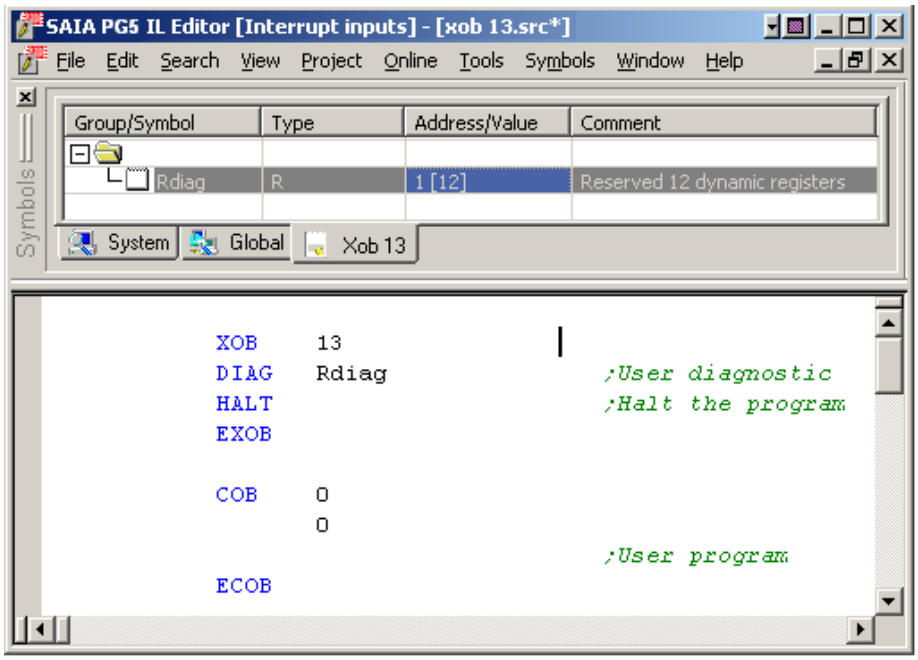
Beispiel in Fupla:

Beispiel zur Lokalisierung von Programm-Fehlern mit Diagnostic XOBs in Fupla: Diagnostic XOBs werden über die Fbox *Spezialfunktionen, XOB-Diagnose* in das Fupla-Programm eingefügt. Hierüber lassen sich Diagnose-Informationen über Ausgänge von Funktionen, Fehler-Zähler, XOB-Nummer, Programmzeilen-Nummer, etc. generieren.



Gleiches Beispiel in IL:

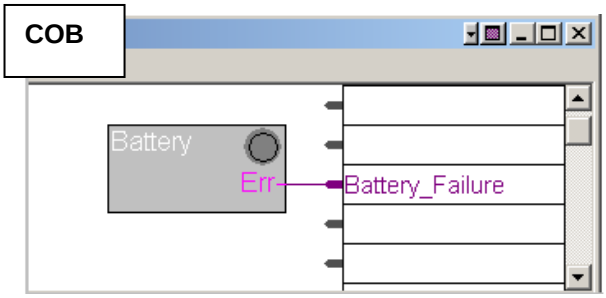
Die IL Programm Diagnose stellt – identisch zum oben abgebildeten Beispiel – die Diagnose-Informationen in den Registern Rdiag + 0 … +12 zur Verfügung.



Überwachung der SPS:

Überwachung der PCD-Batterie (Muss ca. alle 3-4 Jahre gewechselt werden)

Beispiel in Fupla:
In Fupla wird der XOB 2 –Block zur Batterieüberwachung automatisch über die Fbox *Spezialfunktionen, Batterie* erzeugt. Bei Batterie-Problemen wird der dort vorhandene Battery_Failure-Ausgang auf High gesetzt.

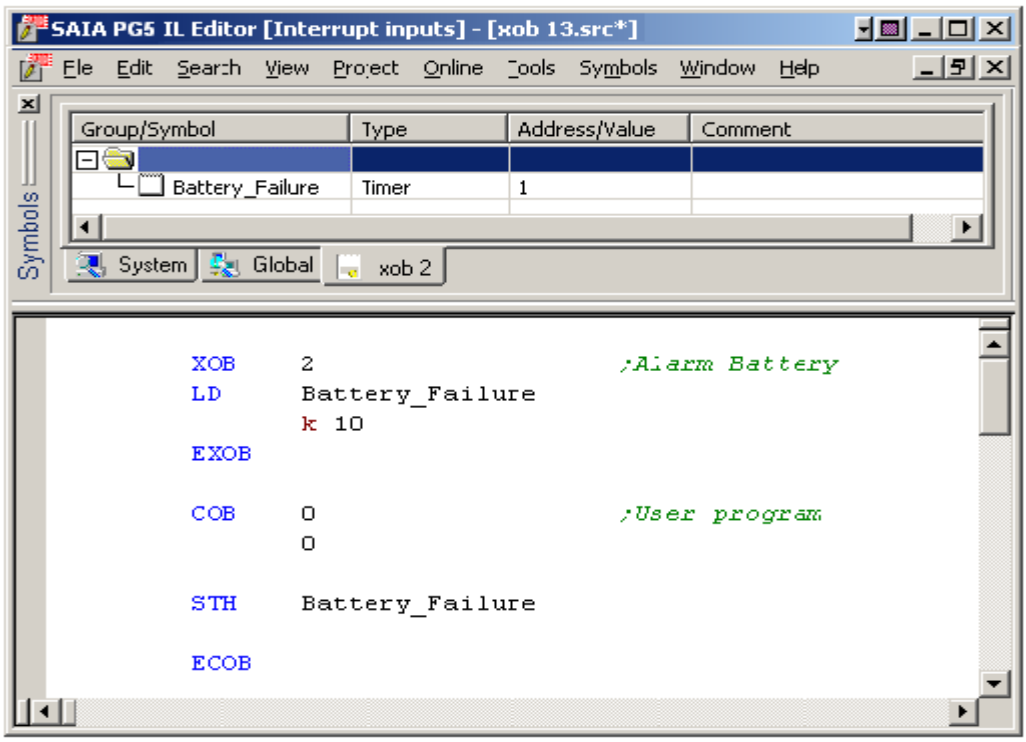


Beispiel in IL:
Im Fall eines Batterie-Fehlers wird die Batterie-Led auf dem PCD-Gehäusedeckel eingeschaltet und XOB 2 zyklisch aufgerufen.

In diesem Beispiel lädt XOB 2 einen Timer mit einer Verzögerungszeit von einer Sekunde. Da dieser Block zyklisch aufgerufen wird, erfolgt die Timer-Initialisierung ständig, womit der Timer-Wert nicht Null abfällt. Folglich ist im Fall eines Batterie-Fehlers *Battery_Failure = 1*, wobei der Abfall auf *Battery_Failure = 0* erst eine Sekunde nach Behebung des Fehlers erfolgt.

Überwachung von Spezial-Events oder schneller externer Signalen, wie:

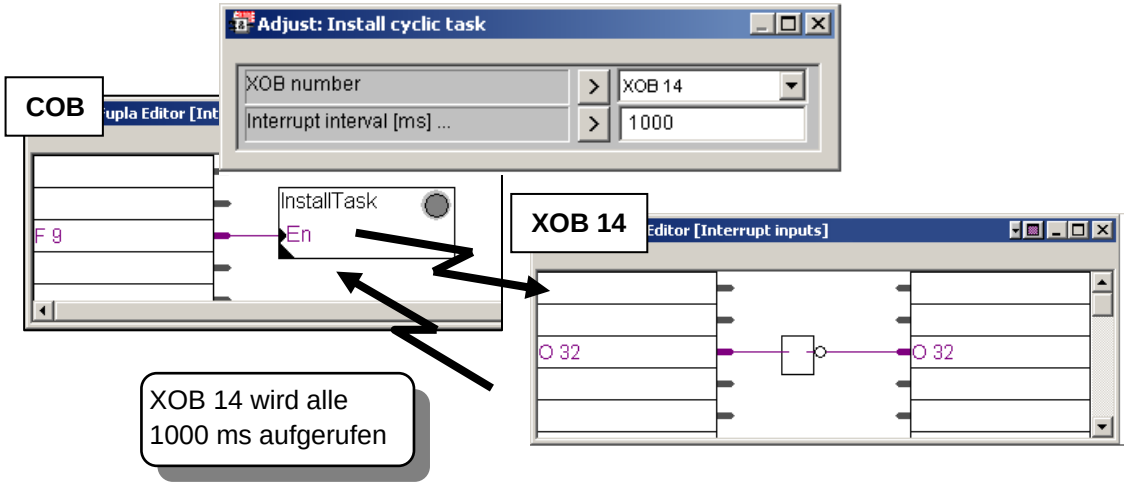
- Interrupt-Eingänge
- Zyklische Programm-Unterbrechungen
- Programm-Unterbrechung bei Telegramm-Eingang



- Kalt Start und Werte-Initialisierung

Beispiel in Fupla:

Das folgende Beispiel zeigt einen blinkenden digitalen Ausgang in Fupla, der die Funktionen *Spezialfunktionen, XOB, Zykl. Aufgabe installieren* und *Binäre Funktionnen, Move* verwendet.



Das Beispiel in IL:

```
XOB       16          ; Kalt-Start
SYSWR     4014        ; Initialisiere XOB 14
          1000        ; Mit 1000 ms Interrupt-Frequenz
EXOB

COB       0
          0
                      ; Anwender-Programm
ECOB
```

```
XOB     14      ; Wird zyklisch durch Interrupt aufgerufen
COM     O 32    ; Invertiere Ausgang 32
EXOB
```

7.4.3 History Table

Die Funktion *PCD History Table* listet alle Hardware- und Software-Fehler auf, die vorher aufgetreten sind. Auch wenn die XOBs nicht programmiert sind, läuft diese Funktion ständig im Hintergrund mit.

Der Aufruf der History Table erfolgt über die *Online Configurator* Schaltfläche oder über das Menü *Tool, Online Configurator*

Online Configurator

7.4.4 Beschreibung der XOBs

XOB 0: Spannungsversorgungs-Fehler auf der Hauptplatine

Die Spannungsüberwachung im Versorgungsmodul des Haupt-Racks hat einen starken Spannungsabfall registriert. Alle Outputs werden zurückgesetzt, XOB 0 wird aufgerufen und die PCD wird angehalten.
Vom Augenblick des Aufrufs von XOB 0 bis zum Stillstand der CPU vergehen ungefähr 5 mS. Während dieser Zeit setzt der XOB 0 die Verarbeitung fort, so dass die Daten immer noch gespeichert werden können.

XOB 1: Spannungsversorgungs-Fehler im Erweiterungsgehäuse (PCD6)

Die Spannungsüberwachung im Versorgungsmodul eines Erweiterungs-Racks hat einen starken Spannungsabfall registriert. In diesem Fall werden alle Outputs des Erweiterungs-Racks innerhalb von 2 mS low gesetzt und XOB 1 aufgerufen.
Wenn die Outputs dieses „toten“ Erweiterungs-Racks weiterhin vom Nutzerprogramm verarbeitet werden (eingestellt, erneut eingestellt oder abgefragt), werden auch XOB 4 und/oder XOB 5 aufgerufen.

XOB 2: Batterie-Fehler oder verbrauchte Batterie

Die Batterie ist verbraucht, fehlerhaft oder fehlt.
Durch diesen Zustand können remanente Flags, Registerinhalte, Teile des Anwender-programms im RAM und auch der Hardware-Clock verändert werden. Ein längeres Nichtbenutzen der PCD (z.B. 2 Monate ohne anliegende Versorgungsspannung) kann gleichfalls zur Anzeige eines Batteriefehlers führen, der jedoch nicht mit einem Datenverlust einhergeht. Sogar eine neue, ungebrauchte PCD kann das gleiche Symptom zeigen.

XOB 4 : Paritäts-Fehler auf dem Adress-Bus (PCD6)Dieser XOB 4 kann nur bei PCD6-Anlagen mit Erweiterungsgehäusen aufgerufen werden. Die Überwachungsschaltung des Adressbus hat einen Paritätsfehler festgestellt.Dieser kann entweder von einem defekten Erweiterungskabel, einem defekten Erweiterungsgehäuse oder von einemBuserweiterungsmodul herrühren oder einfach deshalb, weil das adressierte Erweiterungsgehäuse nicht vorhanden oder nicht gespeist ist. Im Falle einer Störung kann ein falsches Element adressiert worden sein.

XOB 5: Keine Antwort vom I/O-Modul (PCD4/6)

Die Eingangs- und Ausgangsmodule der PCD senden der sie adressierenden CPU eine Quittung zurück. Fehlt diese Quittung, wird der XOB 5 aufgerufen. Dieser Aufruf erfolgt im allgemeinen dann, wenn das Modul gar nicht bestückt ist, kann aber auch bei defekter Adressdecodierung auf demModul erfolgen. Wird bei einem PCD4-Modul mit nur 8 Elementen eines der nicht bestückten Elemente adressiert, wird der XOB 5 nicht aufgerufen, da diese Adresse in der Adressdecodierung trotzdem verarbeitet und damit auch das Quittiersignal gesendet wird. Bei der PCD6 erfolgt diese Quittung nur bei den neuen PCD6-Modulen. (Jumper Q-I/O) gesteckt). Die bis heute verwendeten PCA-Module erwirken keinen XOB 5 Aufruf, auch wenn diese nicht bestückt sein sollten. Defekte Ausgangstransistoren können mit dem XOB 5 nicht detektiert werden.
Bei den PCDs 1/2/3 ist dieser Mechanismus nicht implementiert.

XOB 7: System-Überlastung

Der Priorisierungs-Mechanismus für XOBs mit Prioritätslevel 2 und 3 ist überlastet. XOBs mit niedriger Priorität werden solange in der Warteschlange gehalten, bis die mit der höheren Priorität abgearbeitet sind. Beim Überlauf der Warteschlange erfolgt der Aufruf von XOB 7.

XOB 8: Ungültige Instruktion

Die CPU hat eine ungültige Instruktion entdeckt. Werden editierte Anwenderprogramme oder -programmteile assembliert, gelinkt und in die PCD geladen, sind falsche Operationscode praktisch ausgeschlossen, da schon der Editor (SEDIT) oder dann entweder der Assembler oder der Linker das Programm sehr streng prüft.Wird jedoch nachträglich, direkt im Debugger oder mit dem Servicegerät PCD8.P100 das Anwenderprogramm verändert, können Fehler praktisch nach Belieben eingebaut werden, die dann dazuführen, dass der XOB 8 aufgerufen wird. Fehler, die auf diese Weise oft eingebaut werden sind der Aufruf von Blocks, die nicht existieren, Vergessen eines Ende-Block Befehls, Programmsprünge auf z.B. die 2. Zeile von mehrzeiligen Befehlen, Sprünge aus einem Block direkt in einen andern usw.

XOB 9: Zu viele aktive GRAFTEC-Verzweigungen

Mehr als 32 Graftec Verzweigungen sind gleichzeitig in einem Sequential Block (SB) aktiv. Obwohl mehr als 32 parallele Verzweigungen in einem einzelnen SB programmiert sein dürfen, ist darauf zu achten, dass nicht mehr als 32 gleichzeitig ausgeführt werden.

XOB 10: Mehr als 7 verschachtelte PB/FB-Aufrufe

PBs und FBs können bis zu einer Tiefe von 7 Ebenen verschachtelt werden. Der Aufruf einer tieferen 8. Ebene führt zur Ausführung des XOB 10, wobei der Aufruf selbst nicht bearbeitet wird.

XOB 11: COB-Monitor-Zeit überschritten

Die COB-Monitor-Zeit wurde überschritten
Dieser Wert beschreibt die maximal zulässige Ausführungszeit zwischen einer COB- und EOCB-Instruktion und wird - in 1/100 sec skaliert – in der zweiten Zeile der COB-Instruktion festgelegt. Wird die Zeit überschritten, so erfolgt der Aufruf des XOB 11.
Somit entspricht die Funktion einem „Software-Watchdog“, der ursprünglich zur Aufdeckung und Vermeidung von Blockierungen bzw. starken Verzögerungen innerhalb des Anwenderprogrammes durch ungeschickte Programmierung diente. Hierzu zählen z.B. Warte-Schleifen und überlange Zählschleifen, deren Vermeidung generell zur zeitlichen Stabilität des Programmes beitragen.
Jedoch auch in gut strukturierten Programmen kann es vorkommen, dass der eine oder andere COB lange mathematische Funktionen beinhaltet, die, aufgrund ihrer langen Ausführungszeit, andere zeitkritische COBs hinsichtlich ihrer Zykluszeit zu stark beschränken. In diesem Fall ist es dann ratsam, einen „langen“ COB mit einer kurzen Monitor-Zeit zu versehen, da nach Ablauf der Zeit die Ausprungadresse gespeichert und mit der Abarbeitung des nächsten COBs begonnen wird. Ist der Programm-Zyklus durchlaufen, so erfolgt die Fortsetzung des “langen” COB ab der gespeicherten Adresse, was einer zeitlichen Segmentierung des COBs in Einheiten der definierten Monitor-Zeit gleichkommt.
Bei dieser Technik ist natürlich die Programmierung des XOB 11 zu vermeiden, da die Überschreitung der COB-Monitor-Zeit in diesem Fall gewollt und nicht ein Fehler ist. Weitere Programmiertechniken, wie z.B. „Timeslice“ werden im Kapitel <Andere Programmiertechniken> beschrieben.

XOB 12: Index-Register-Überlauf

Die Grösse der Indexregister ist 13 bit (0 bis 8191). Dies reicht aus, um alle Element-Adressierungen auszuführen. Gerät in einem Programm ein indexiertes Element ausserhalb seines Bereiches, es wird z.B. derMerker 8000 indexiert gesetzt und das Indexregister stehe auf 500, so würde der Merker 8500 gesetzt. Dieser liegt aber ausserhalb seines Bereiches (0 - 8191). Hier wird der XOB 12 aufgerufen.

XOB 13: ERROR-Flag gesetzt

Viele Instruktionen des PCD-Befehlssatzes können das ERROR-Flag setzen. Vergleiche Befehlssatz. Tritt ein Error auf, wird neben dem Setzen des ERROR-Flags auch der XOB 13 aufgerufen, wo dann irgendwelche allgemeine Vorkehrungen (Alarm, Fehlermendung auf einen Drucker usw.) getroffen werden können. Dieser XOB 13 wird also immer aufgerufen, wenn das ERROR-Flag gesetzt wird, egal ob dieUrsache ein Rechenfehler, ein Datentransferfehler oder ein Kommunikationsfehler ist. Soll dieDiagnose bezüglich desERROR-Flags differenzierter erfolgen, so kann nach jedem Befehl der einen Error erzeugen kann ein PB (oder FB) bedingt aufgerufen werden, wobei die Bedingung eben das ERROR- Flag ist.

Beispiel:

```
...
...
DIV     R 500          ; Wert 1
        R 520          ; Wert 2

        R 550          ; Resultat
        R 551          ; Rest
CPB     E 73           ; Bei ERROR —> PB 73
...
...
PB      73
SET     O 99           ; Div : 0
INC     C 1591
EPB
```

Der PB 73 wird nach einer Division durch Null aufgerufen und schaltet den Ausgang 99 ein, der die Division durch Null anzeigt. Ein Zähler C 1591 z.B. zählt, wie oft dieses Ereignis aufgetreten ist.

Ein Überlauf bei einer Multiplikation könnte dann z.B. den Ausgang 98 aktivieren und ein Zähler C 1590 würde dieses Ereignis aufsummieren. Der XOB 13 soll gleichwohl programmiert werden, kann jedoch leer sein. Ist dieser nicht programmiert, wird beim Setzen des ERROR-Flags die ERROR Lampe auf der Frontplatte der CPU aktiviert, was unschön ist.

Achtung:
Das ERROR-Flag und auch die anderen Status-Flags (Positiv, Negativ, Zero) werden bei einem bestimmten Ereignis bzw. Zustand gesetzt undmüssen, falls von Interesse, SOFORT ausgewertet werden, da sich diese Status-Flags jeweils auf die zuletzt ausgeführte Instruktion, die diese Flags beeinflussen kann, beziehen. Würde also nach der oben erwähnten Division durch Null (ERROR-Flag gesetzt) z.B. eine korrekte Addition folgen, würde das ERROR-Flag wieder zurückgesetzt!

XOB 14, 15: Zyklischer Interrupt

Die XOB 14 und 15 werden periodisch mit einer Periodendauer zwischen 10 ms und 1000 s aufgerufen, die über die Instruktion SYSWR einstellbar ist.

XOB 16: Kaltstart

Die Ausführung des XOB 16 erfolgt einmalig sowohl nach dem Einschalten der PCD als auch nach dem Empfang eines Kaltstart-Kommando vom Programmier-Tool. Ist der XOB beendet, so beginnt die zyklische Ausführung der COBs, womit eine erneute programmunterstützte Ausführung dann nicht mehr möglich ist. Somit dient der Block nur zur einmaligen Initialisierung von Werten beim Programmstart. Sollte die Notwendigkeit bestehen, spezielle Aktionen des XOB 16 auch in den COBs auszuführen, so ist es ratsam, diese in einen PB oder FB zu integrieren und diesen dann sowohl aus dem XOB wie auch COB heraus aufzurufen.
XOB 16 verfügt über einen eigenen Register *Index*, welcher von den Indexregistern der COBs unabhängig ist.

XOB 17, 18, 19: Interrupt via S-Bus

Die XOB 17,18 und 19 werden über S-Bus-Telegramme gestartet und verhalten sich demzufolge wie Interrupt-Service-Routinen. Hierzu dient sowohl die Instruktion SYSWR wie auch die Fupla-Funktion *Special, execute XOB.*

XOB 20, 21, 22, 23 ,25: Interrupt via INBx

Wird eine positive Flanke an den Interrupt-Eingängen INB1 bzw. INB2 erkannt, so wird der XOB 20 bzw. 25 ausgeführt. Siehe dazu auch das Saia PCD Hardware-Manual.

XOB 25, 26, 27, 28, 29: Cyclic interrupt XOBs

Die XOB 25,...,29 werden periodisch mit einer Periodendauer zwischen 10 ms und 1000 s aufgerufen, die über die Instruktion SYSWR einstellbar ist.

XOB 30: Keine Verbindung zum RIO

Sollte der Verbindungstest zwischen CPU und RIO einen Fehler ergeben, so wird der XOB 30 ausgeführt. Dieser Fall tritt z.B. auf, wenn die RIO-Station vom Netz gezogen wird.

7.5 Sequential Blocks (SB 0 , 96 [1])

Sequential blocks (SBs) beinhalten ausschliesslich STEPS und TRANSITIONS. Während STEPS die abzuarbeitenden Programmteile beinhalten, warten TRANSITIONS auf das Eintreten von Bedingungen, um danach den nachfolgenden STEP zu starten. Die daraus entstehende Programmstruktur ist besser unter dem Namen Graftec bekannt.

Graftec-Programme werden mit dem S-Graf-Editor erstellt und enthalten den Suffix *.sfc. SBs lassen sich von allen Blöcken aufrufen. Der Editor ist ein exellentes Tool zur Erstellung von sequentiellen Programmen und wird näher im nächsten Kapitel beschrieben.

7.5.1 Zusammenfassung

Service	Media	Operand	Bemerkung
Cyclic Organization Block	COB	0…31	Mindestens ein COB pro Programm
Program Block	PB	0…999	Unterprogramm, aufrufbar aus einem COB, PB,FB,SB oder XOB
Function Block	FB	0…1999	Parametrierbares Unterprogramm aufrufbar aus einem COB, PB, FB, SB oder XOB
Sequential Block	SB	0…95	Sequentielles Unterprogramm, aufrufbar aus einem COB, PB oder FB (bedingt auch SB und XOB)
Step	ST	0…5999	
Transition	TR	0…5999	

Inhalt

8 Programmierung von Graftec

Saia PG5 “Graftec” basiert auf dem französischen Standard Grafcet NF C-03-190 und IEC 848, enthält allerdings einige Unterschiede und Verbesserungen. Es ist auch unter der Bezeichnung „Sequential Funtction Chart“ (SFC) (dt: Ablaufsprache) bekannt.

Grafcet wird unabhängig von der Technologie für deren Implementierung verwendet.Die Darstellung nacheinander ablaufender Prozesse wird dabei mit nur wenigen graphischen Symbolen und einigen einfachen Regeln standardisiert. Ein Grafcet-Diagramm besteht aus wenigen *Steps*, welche Handlungen definierten und *Transitionen*, die nach Ereignissen suchen.

Eine *Sequence* besteht aus einer Folge von einander abwechselnden *Steps* und *Transitionen*. Ein *Step* wird erst dann ausgeführt, wenn die vorhergehende *Transition* dies erlaubt.

Der Graftec-Editor von Saia PG5 erstellt alle für die Erstellung eines Diagramms in einem sequentiellen Block (SB) nötigen Anweisungen.

Eine Graftec-Anwendung wird in zwei Schritten programmiert:
Die Erstellung eines Step-/Transition-Diagramms in dem der sequentielle Prozess beschrieben wird.
- Der Kodierung der Steps und Transitionen mit dem Fupla- oder dem IL-Editor, S-Fup oder S-Edit.

Nach dem *Build* und dem *Download* des Programms in die PCD kann die Funktionsfähigkeit des Programmes während des Betriebs beobachtet werden. Das ist bei Tests und Inbetriebnahme eine große Hilfe.

Graftec können größere Strukturen auch in kleinere *Pages* aufgeteilt werden. Diese Pages funktionieren wie ein Zoom, der die Darstellung des Prozesses mit dem gewünschten Detaillierungsgrad ermöglicht.

Die Ausführung eines Graftec-Programmes ist streng sequentiell und folgt den Regeln von Grafcet. Daraus ergeben sich optimal Ausführungsgeschwindigkeiten mit schneller Reaktionszeit. Es werden nur die aktiven Transitionen ausgeführt, auch wenn das Programm zahlreiche *Steps* und *Transitionen* enthält. Die Ausführungszeiten anderer zyklischer Programmbestandteile werden selbst von großen Graftec-Programmen nicht beeinflusst.

8.1 Sequentielle Blöcke **(SB 0 bis 95)**

Da die Planung eines Ereignisses unbestimmbar ist, können wir die Zykluszeit eines sequentiellen Programmes nicht abschätzen. Daher ist es wichtig, zyklische Programme von sequentiellen Programmen zu trennen.

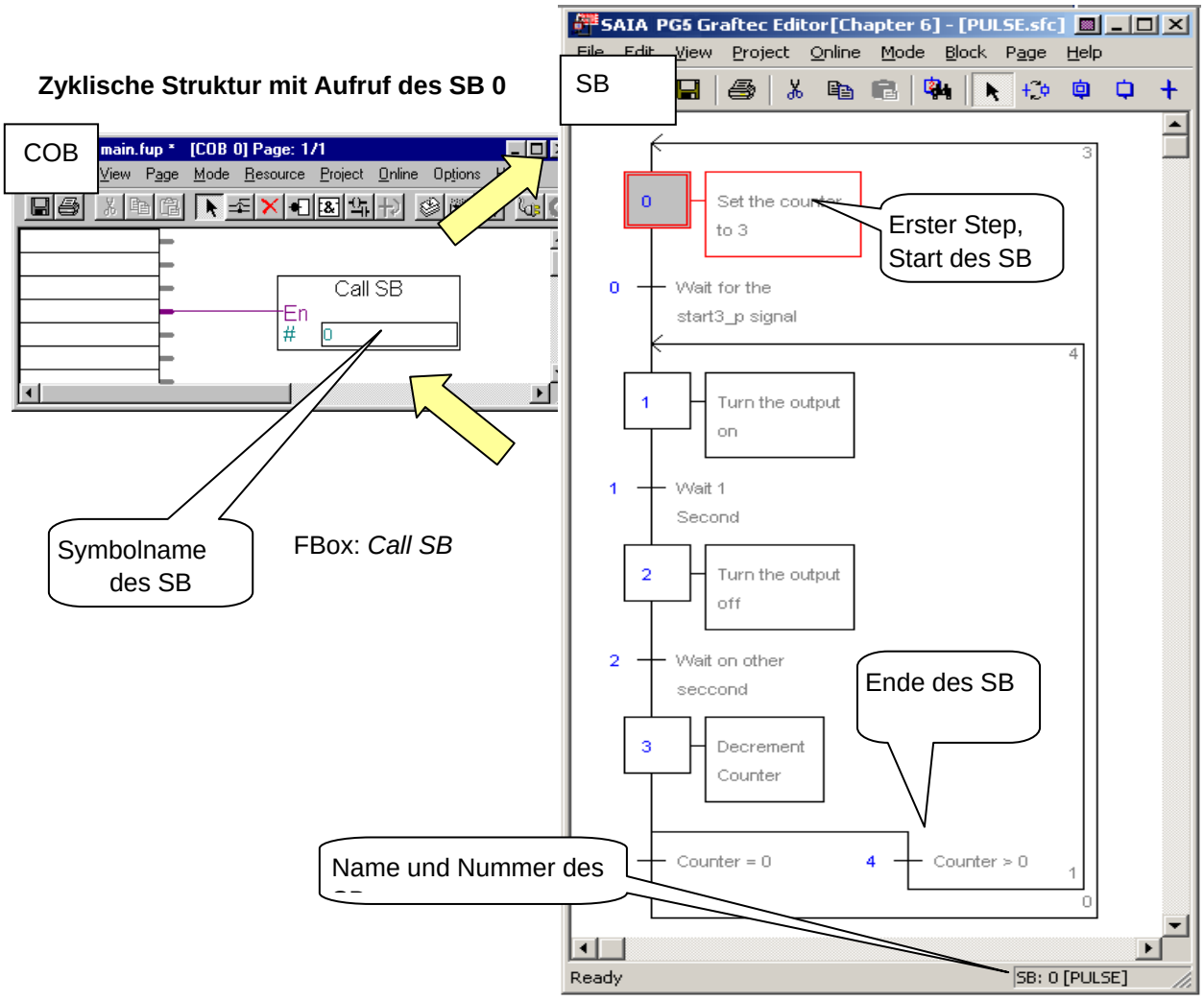
Ein zyklisches Programm wird durch das Warten auf ein sequentielles Programm nicht blockiert.Um diese Bedingung zu erfüllen, werden sequentielle Programme in einer der SB-Strukturen platziert, die bei jedem Programmzyklus aufgerufen werden.

Wartet das sequentielle Programm in einem SB auf ein Ereignis, hält die PCD die Bearbeitung des SBs an und setzt das zyklische Programm fort. Die SB wird dann im nächsten Programmzyklus wieder aufgerufen.

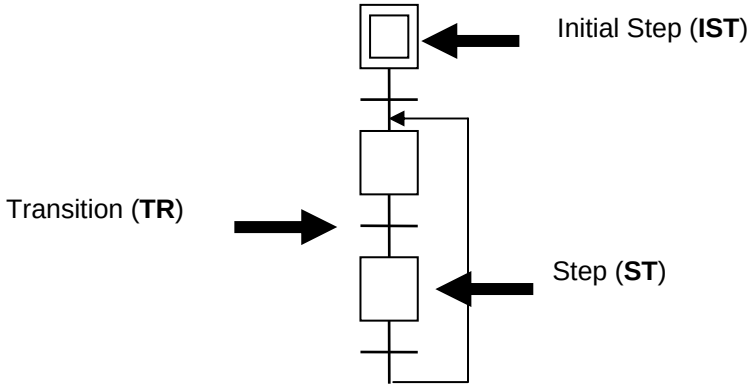
8.2

Zyklische Struktur mit Aufruf des SB 0

Sequentieller Block



Struktur eines sequentiellen Blocks (SB)



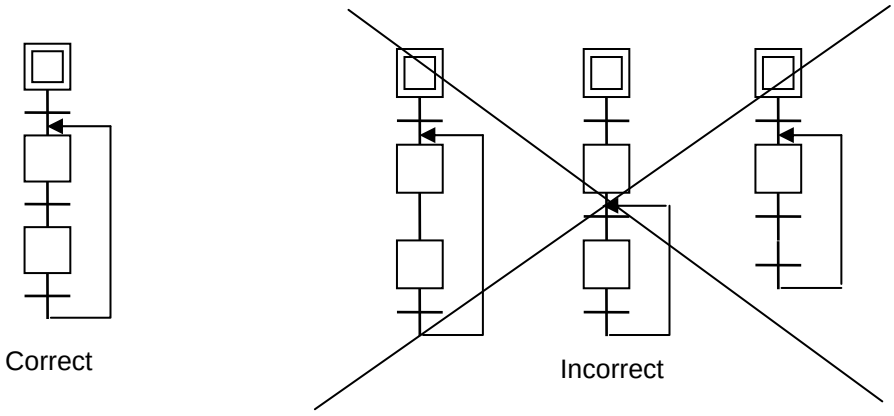
Mit dem Graftec-Editor (S-Graf) kann ein SB mit Hilfe der Steps und Transitionen, die Instruktionsliste oder graphischen Fupla-Code enthalten, erstellt werden.

Der sequentielle Block beginnt mit dem ersten Step, der durch ein Rechteck mit doppeltem Rahmen dargestellt ist. Dies ist der Anfang eines sequentiellen Prozesses, der beim ersten Aufruf des SB (Kaltstart) durchgeführt wird.

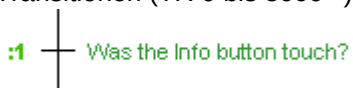
Die Struktur muss immer eine geschlossene Schleife sein.

8.2.1 Regeln für die Verbindung von Steps und Transitionen

Die Struktur eines SB hat eine einfache aber strenge Syntax.Wie Sie bereits wissen, beginnt der Block mit dem ersten Step und wechselt dann zwischen Transitionen und Steps. Es können nie zwei Steps oder zwei Transitionen unmittelbar aufeinander folgen.



Transitionen (TR 0 bis 5999 [1])



Der Graftec-Prozess wird durch den Code in den Transitions gesteuert. Die Transitionen sind so lange aktiv, bis ein Ereignis, wie beispielsweise ein Statuswechsel eines Inputs, Outputs, Indikators oder die Bewertung eines logischen Ausdrucks, entdeckt wird.

Ist das Programm in IL geschrieben, führt die Transition den nächsten Step nur dann aus, wenn der *Accumulator (ACCU)* am Ende der Transition high (1) ist.

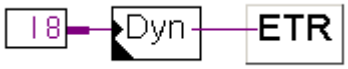
Wenn das Programm in Fupla geschrieben wurde, führt die Transition den nächsten Step nur aus, wenn der Wert der FBox *ETR* 1 beträgt.

Tritt dies am Ende einer Transition nicht ein, bleibt sie aktiv und wird so lange neu bewertet, bis der erwartete Wert erreicht ist.

Beispiel: Erkennen der Anstiegskante eines Inputsignals

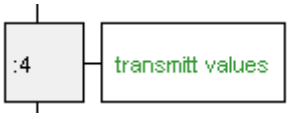
IL-Programm Fupla-Programm

```
STH     I 8
DYN     F 80
```



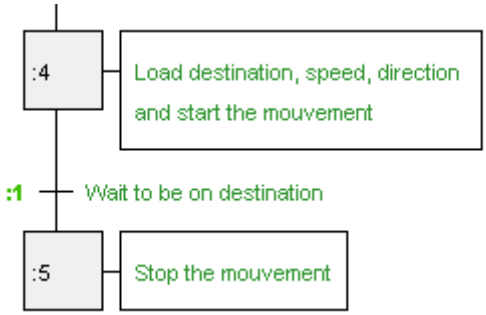
Anmerkung:
Bei Transitionen in Instruktionsliste ist der ACCU zu Beginn einer Transition oder eines Steps immer high (1). Daher werden alle vom ACCU abhängigen Instruktionen immer ausgeführt und leere Transitionen sind immer erfüllt.

8.2.2 Steps (ST 0 bis 5999)



Steps enthalten die Programme, die Ereignisse des Prozesses sind:Ein-/Ausschalten von Outputs, Flags, Berechnungen, Laden von Zählwerten usw.

Beispiel: Steuerung der Achse einer Maschine mit einem Motor.

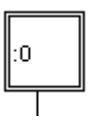


Wir definieren die Zielposition sowie die Geschwindigkeit und die Richtung der Bewegung. Danach wird die Bewegung gestartet. Da diese Aufgaben einmalig ausgeführt werden, werden sie in einem Step platziert.

Als nächstes überwachen wird die Bewegung und warten auf das Eintreffen in der Zielposition. Die tatsächliche Position wird mit der Zielposition verglichen. Diese Aufgabe muss mehrfach ausgeführt werden und wird deshalb in einer Transition platziert. Wenn das Ziel erreicht ist, endet die Transition, der ACCU oder die FBox ETR sind high. Die Transition ist erfüllt und der nächste Step wird ausgeführt.

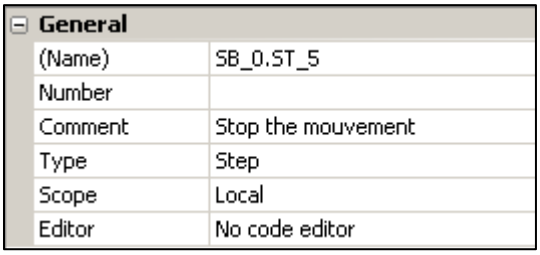
Der nächste Step hält den Motor in der Zielposition an. Da diese Aufgabe einmalig ausgeführt wird, wird sie in einem Step platziert.

Anmerkung:
Ein Step ohne Programm übergibt die Steuerung direkt an die nächste Transition. Ein Step wird nur einmal ausgeführt, er wird, im Gegensatz zu Transitionen, nicht periodisch ausgeführt.
Die gesamte Graftec-Struktur beginnt mit dem ersten Step, der durch ein Rechteck mit doppeltem Rahmen dargestellt ist. Dies ist der Anfang eines sequentiellen Prozesses, der beim ersten Aufruf des SB durchgeführt wird, also nach einem Kaltstart oder dem Auffahren.



8.2.3 Eigenschaften der Steps und Transitionen

Wird ein Step oder eine Transition in der Graftec-Struktur per Mausklick ausgewählt, werden die folgenden Informationen in einem Fenster, welches mit dem Befehl *Block, Properties* geöffnet werden kann, angezeigt.

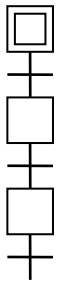


Name:	Symbolname des Steps oder der Transition.
Nummer:	Nummer des Steps oder der Transition. Diese Nummer ist standardmässig leer. Das bedeutet, dass sie dynamisch ist und vom *Build* zugewiesen wird. Bei Bedarf kann eine Nummer definiert werden. Je nach PCD-Typ stehen entweder 2000 oder 6000 Steps und Transitionen zur Verfügung.
Anmerkung:	Freitextfeld rechts vom Step der Transition.
Typ:	Step oder Transition.
Umfang:	Umfang eines Symbols (lokal oder public). Ist das Symbol public, kann von anderen Dateien aus auf den Symbolnamen zugegriffen werden. Das ist bei Steps oder Transitionen normalerweise nicht notwendig.
Editor:	IL Instruktionsliste oder Funktionsblockdiagramm.

8.2.4 Typische Graftec-Sequenzen

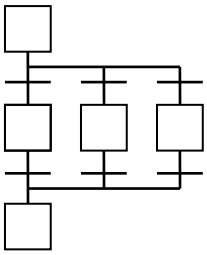
Einfache Sequenz

Abwechslung von Steps und Transitionen.
Beachten Sie, dass keine zwei Steps oder Transitionen aufeinander folgen können.



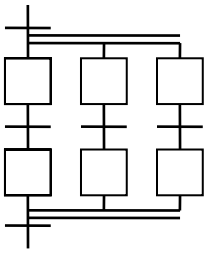
Abwechselnde Verzweigung (OU)

Ein Zweig mit einer Auswahl von Sequenzen. Die Transitionen werden von links nach rechts bewertet. Die erste aktive Transition bestimmt, welche Sequenz ausgeführt wird. Ein alternativer Zweig beginnt immer mit einem Step, der mit mehreren Transitionen verbunden ist und wird mit Transitionen beendet, die in einen einzigen Step münden. Mit dem Graftec-Editor können bis zu 32 Zweige dargestellt werden. Gibt es mehr als 32 Zweige, wird XOB 9 aufgerufen.



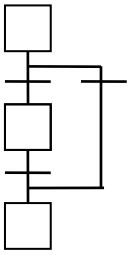
Simultane Verzweigung (ET)

Ein simultaner Zweig enthält mehrere Sequenzen, die gleichzeitig parallel ausgeführt werden. Der simultane Zweig beginnt immer mit einer Transition, die mit mehreren Steps verbunden ist und endet mit einer einzigen, synchronisierenden Transition. Mit dem Graftec-Editor können bis zu 32 parallele Zweige dargestellt werden. Gibt es mehr als 32 Zweige, wird XOB 9 aufgerufen.



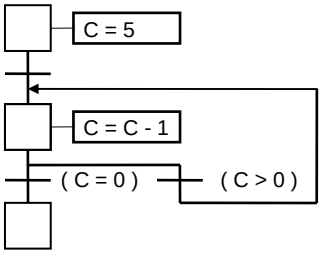
Sprungsequenz

Die Sprungsequenz ist ein abwechselnder Zweig, mit dem die Verarbeitung einer Sequenz an bestimmte Bedingungen geknüpft werden kann.



Wiederholungssequenz

Die Wiederholungssequenz ist ebenfalls ein abwechselnder Zweig, der allerdings mit einem vorhergehenden Step verbunden ist. In diesem Beispiel wird ein Zähler mit einer Anzahl von Schleifen gestartet. Es folgen einfache Sequenzen beliebiger Länge. Zum Zeitpunkt des letzten Steps sollte der Zähler bei null angekommen sein. Die Schleife wird wiederholt, wenn der Zählerstand nicht null beträgt.



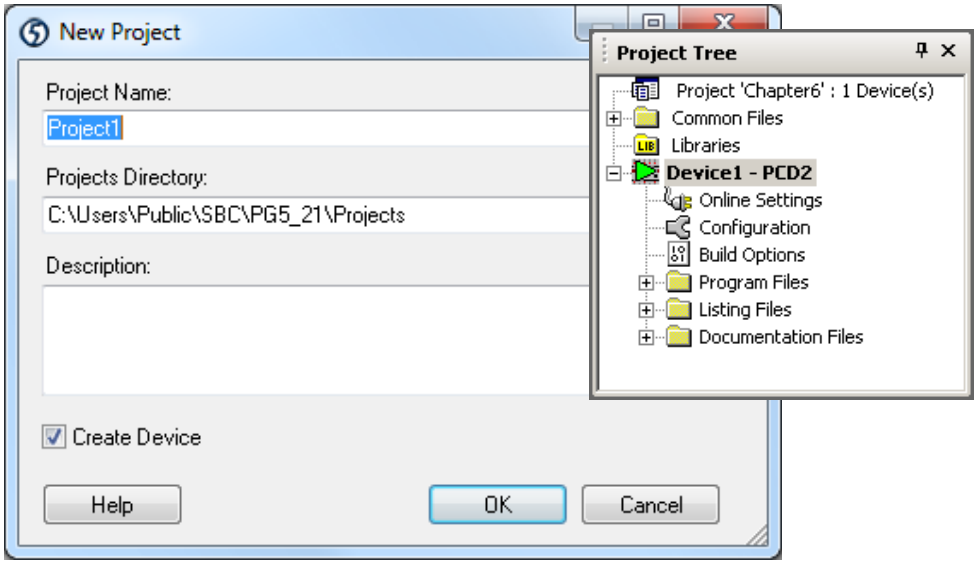
8.3 Ein Graftec-Projekt erstellen

In diesem Beispiel werden wir ein neues Projekt erstellen, welches die Dateien für das Graftec-Programm enthält.

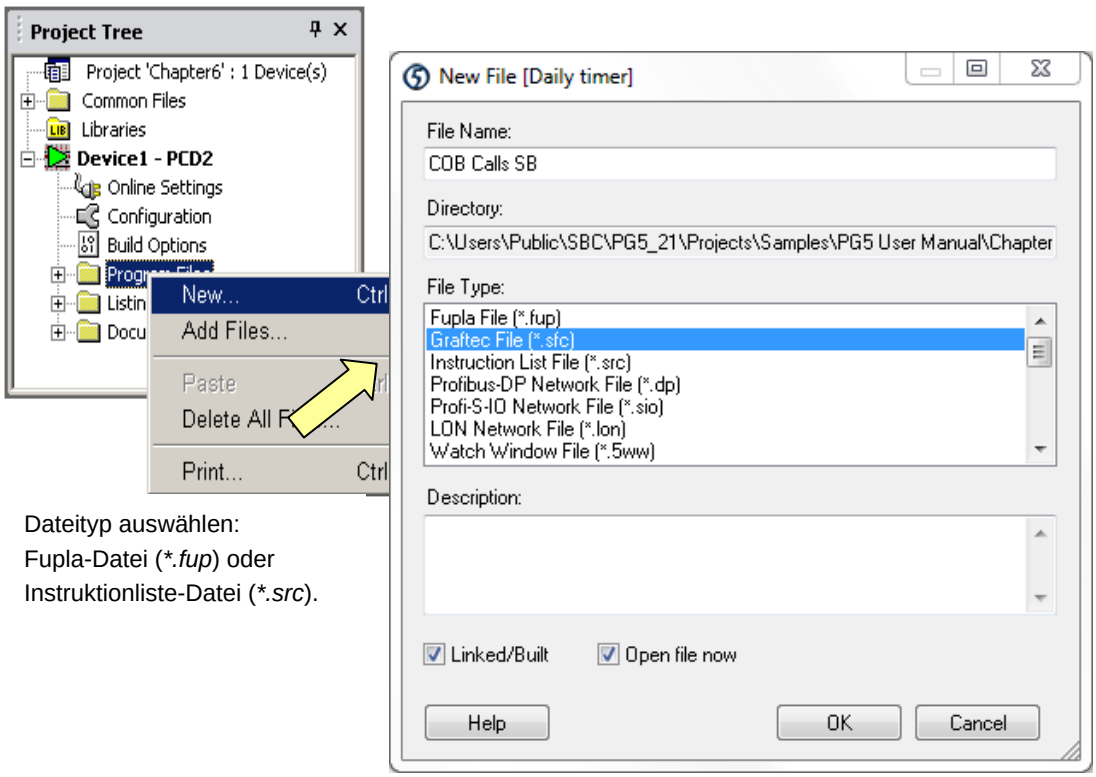
- Bereiten Sie bei der graphischen Programmierung eine Graftec-Datei und eine Fupla-Datei vor.
- Bereiten Sie bei der Verwendung von IL eine Graftec-Datei und eine „src.“-Datei von IL vor.

8.3.1 Neues Projekt anlegen

Verwenden Sie den Befehl *Project, New* aus dem *Project Manager*, um ein neues Projekt zu erstellen.



8.3.2 Fupla- oder IL-Datei hinzufügen



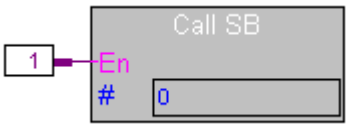
Dateityp auswählen:
Fupla-Datei (*.*fup*) oder
Instruktionliste-Datei (*.*src*).

8.3.3 SB von einem COB aus aufrufen

Rufen Sie je nachdem welches Programm Sie gewählt haben, den SB entweder mit einer CSB-Anweisung oder einer FBox *Call SB* auf. Öffnen Sie die neue Datei und schreiben Sie das Programm wie unten angezeigt.

IL-Programm: Fupla-Programm:

```
COB    1  ;start of COB
          0

CSB    0  ;call SB 0

ECOB      ;end of COB
```

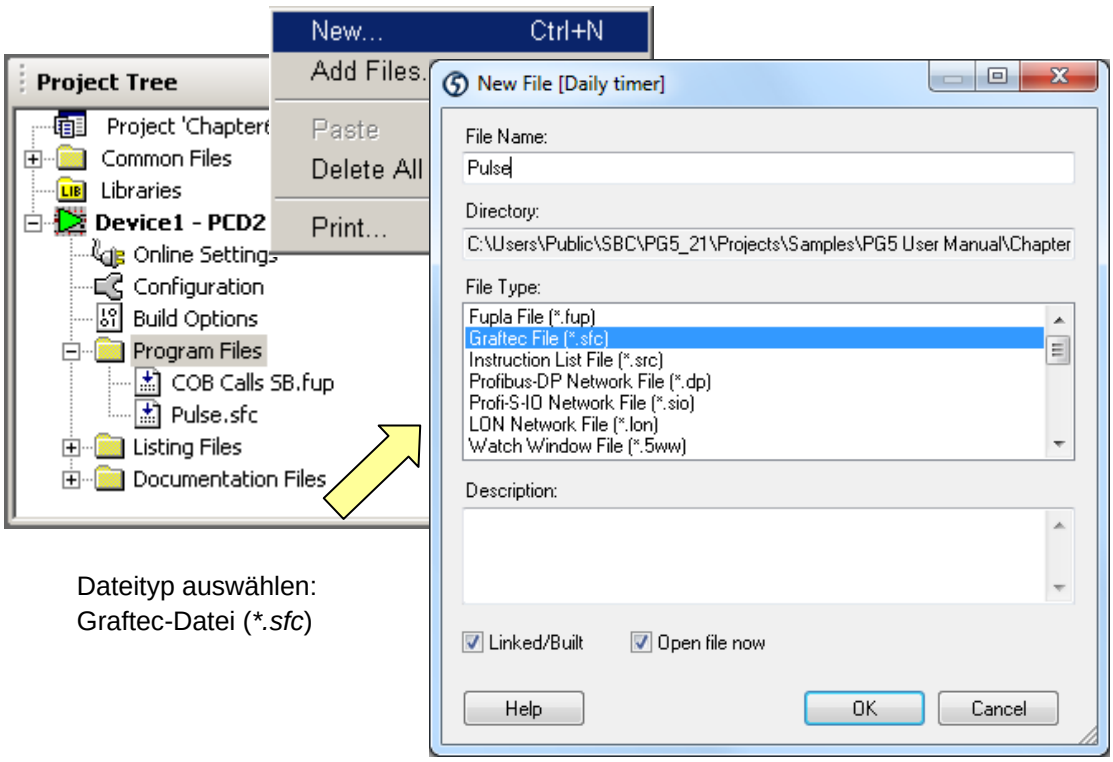


FBox: *Block Calls, Call SB*

Der SB kann wie im oben beschriebenen Beispiel aufgerufen werden. Aber es geht sogar noch einfacher. PG5 kann automatisch einen COB erstellen, der die CSB-Anweisung zum Aufruf des SB enthält. Um das zu ermöglichen, setzen Sie *Generate SB calls* in der Gruppe *Advanced* des Dialogfensters *Build Options* auf *Yes*. Standardmässig ist diese Option auf *Yes* gestellt.)

Sie können zuerst wie unten beschrieben den SB erstellen und den Aufruf nachträglich hinzufügen, da Sie so den Symbolnamen des SB anstelle der Nummer verwenden können.

8.3.4 Graftec-Datei hinzufügen



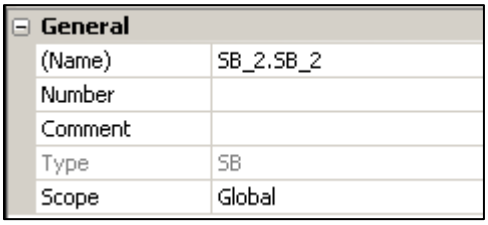
Dateityp auswählen:
Graftec-Datei (*.*sfc*)

8.3.5 Page Navigator, SB hinzufügen

Wenn eine neue Graftec-Datei hinzugefügt wird, erstellt der Editor automatisch einen sequentiellen Block, der den ersten Step enthält.

Innerhalb einer einzigen Graftec-Datei können mehrere SBs erstellt werden. Die Fenster *Page Navigator* und *Block Symbols* zeigen eine Liste der in der Datei vorhandenen SBs an.

Bei Bedarf kann mit dem Menübefehl *Block, New* ein weiterer SB zur Datei hinzugefügt werden. Die Details dieses Blockes können im Fenster *Properties* bearbeitet werden. In diesem Fenster werden auch die Eigenschaften des SBs verändert, der beim Erstellen einer neuen Datei erstellt wird.



Name:	Der Symbolname des Blocks. Die Vergabe von sinnvollen Blocknamen erleichtern Verständnis und Wartung des Programmes.
Anmerkung:	Freitextfeld, in dem Details zum Block festgehalten werden können.
Nummer:	Blocknummer.Die Blocknummer ist standardmässig leer und wird deshalb vom *Build* dynamisch zugeordnet. Bei Bedarf können Sie selber eine Nummer zuweisen.
Umfang:	Umfang des Symbolnamens des SBs (lokal oder public). Verwenden Sie public, wenn das Symbol auch von anderen Dateien aus zugänglich sein soll. Wenn sie beispielsweise von einem, in einer anderen Datei definierten, COB aufgerufen wird.

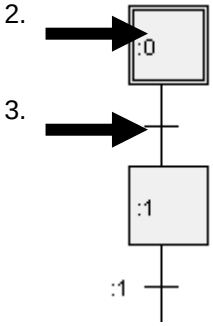
Um die Graftec-Struktur eines SBs darzustellen, klicken Sie mit der rechten Maustaste auf das Fenster *Page Navigator* und wählen Sie den Befehl *Open Block* aus dem Kontextmenü.

Jetzt strukturieren wir den SB mit Steps und Transitionen.

8.4 Graftec-Struktur bearbeiten

Eine neue Graftec-Datei enthält immer den ersten Step, der beim ersten Lauf des SBs ausgeführt wird. Weitere Steps und Transitionen werden entweder mit der Tastatur oder den Kommandos aus der Werkzeugleiste hinzugefügt.

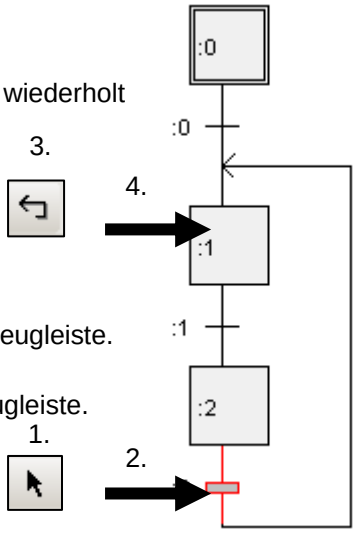
8.4.1 Einfache Sequenz bearbeiten



1. Klicken Sie auf die Schaltfläche *Transition Mode* in der Werkzeugleiste.
2. Platzieren Sie den Mauszeiger auf dem ersten Step und drücken Sie die linke Maustaste.
3. Klicken Sie auf die Schaltfläche *Step Mode* in der Werkzeugleiste.
4. Bewegen Sie die Maus auf die neue Transition und klicken Sie erneut.
5. Setzten Sie diese Abfolge fort.

8.4.2 Eine Schleife erstellen

Ist die Sequenz beendet, endet auch der SB. Soll die Sequenz wiederholt werden, muss eine Schleife hinzugefügt werden.
Bedenken Sie, dass es nicht möglich ist, zwei Steps oder zwei Transitionen direkt miteinander zu verbinden. Eine Schleife startet immer nach einer Transition. Der Verbindungspunkt ist immer über einem Step.

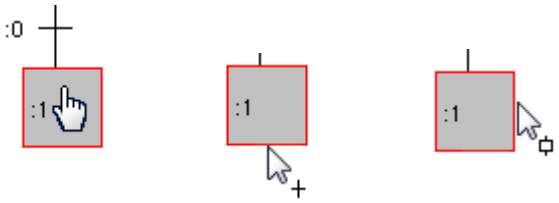


1. Klicken Sie auf die Schaltfläche *Select Mode* in der Werkzeugleiste.
2. Klicken Sie auf die Transition vor dem Sprung.
3. Klicken Sie auf die Schaltfläche *Link Mode* in der Werkzeugleiste.
4. Klicken Sie auf den Step, der den Zielpunkt des Sprunges markiert.

8.4.3 Optionen Smart cursor

Sequenzen können auch im Modus *Smart cursor* bearbeitet werden. Dieser Modus verändert den Cursormodus automatisch in Abhängigkeit vom Ort des Mauszeigers.

Smart Mode



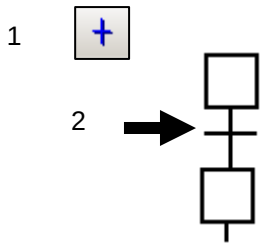
Befindet sich der Mauszeiger auf einem Schritt oder einer Transition, wird er automatisch zu einer Hand, wenn *Select Mode* aktiviert wird. Mit einem Doppelklick wird der Programmeditor zur Bearbeitung des Codes dieses Elementes geöffnet.

Wird der Mauszeiger an das untere Ende eines Steps oder einer Transition bewegt, wechselt er sein Aussehen zu der Transition oder dem Step, der danach eingefügt würde, wenn die linke Maustaste gedrückt wird.

Wird der Mauszeiger an die rechte Seite eines Steps oder einer Transition bewegt, wechselt er sein Aussehen zu einem abwechselnden oder simultanen Zweig, der erstellt würde, wenn die linke Maustaste gedrückt wird.

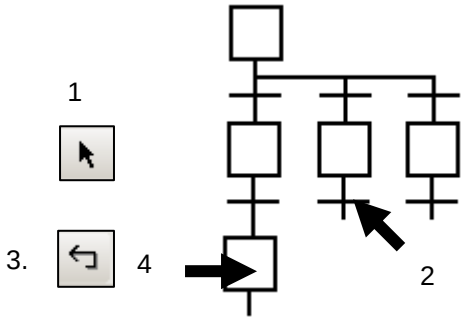
8.4.4 Einen abwechselnden Zweig erstellen (OU)

1. Aktivieren Sie den *Transition Mode*.
2. Wählen Sie die Transition, hinter der bereits ein Status angegeben ist.
3. Mit jedem Klick der linken Maustaste wird auf der rechten Seite eine neue Transition eingefügt.



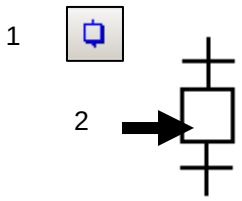
8.4.5 Abwechselndee Zweige verbinden

1. Aktivieren Sie den *Select Mode*.
2. Wählen Sie die Transition, die verbunden werden soll.
3. Klicken Sie auf die Schaltfläche *Link Mode*.
4. Klicken Sie auf einen Step. Die Transition wird nach dem Step verbunden.



8.4.6 Simultane Zweige erstellen (ET)

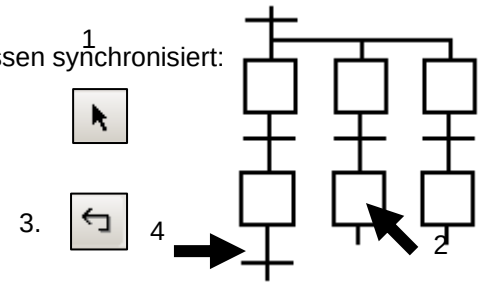
1. Aktivieren Sie den *Step Mode*.
2. Klicken Sie auf den ersten Step, um rechts einen neuen Step anzufügen.
3. Mit jedem Mausklick auf einen Step, wird rechts ein neuer Step verbunden.



8.4.7 Simultane Zweige verbinden

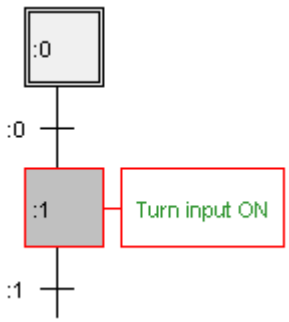
Simultane Zweige werden folgendermassen synchronisiert:

1. Aktivieren Sie den *Select Mode*.
2. Wählen Sie den Step aus, der verbunden werden soll.
3. Aktivieren Sie den *Link Mode*.
4. Klicken Sie auf die Zieltransition.



8.4.8 Kommentar hinzufügen

1. Aktivieren Sie den *Select Mode*.
2. Klicken Sie mit der rechten Maustaste auf den Step oder die Transition, um das Fenster *Properties* aufzurufen
3. Geben Sie den Kommentar in das Feld *Comment* des Fensters *Properties* ein.

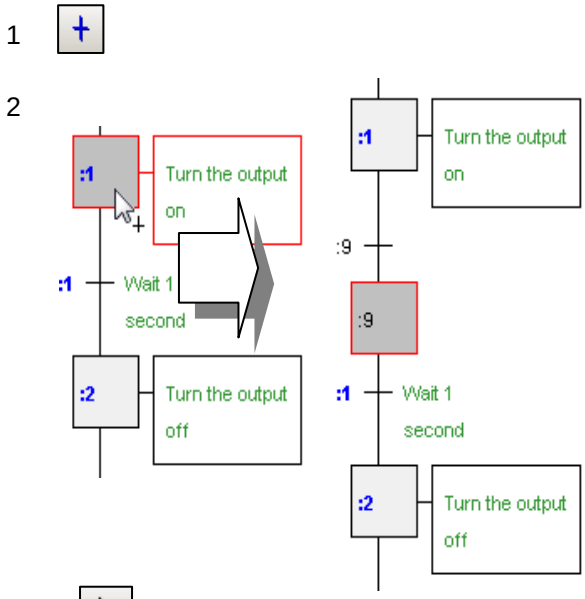


General	
(Name)	SB_0.ST_1
Number	
Comment	**Turn input ON**
Type	Step
Scope	Local
Editor	No code editor

Tipp: Um einen zweizeiligen Kommentar zu erstellen, geben Sie '\n' ein, beispielsweise:
Zeile 1\nZeile 2

8.4.9 Eine Sequenz einfügen

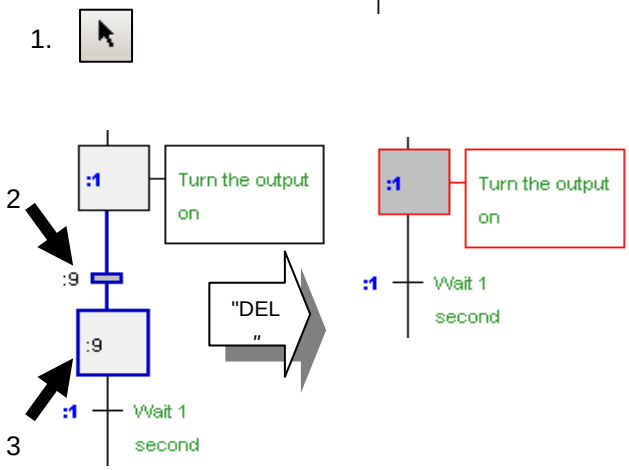
1. Aktivieren Sie den *Transition Mode*.
2. Klicken Sie auf einen Step, zu dem bereits eine Transition existiert.
3. Es werden ein neuer Step und eine neue Transition eingefügt.



8.4.10 Eine Sequenz löschen

1. Aktivieren Sie den *Select mode*.
2. Klicken Sie auf den ersten zu löschenden Step oder die erste zu löschende Transition.
3. Halten Sie die Taste *Shift* gedrückt und klicken Sie auf die letzte Transition oder den letzten Step. Die markierte Sequenz wird hervorgehoben.
4. Drücken Sie die Taste *Del*.

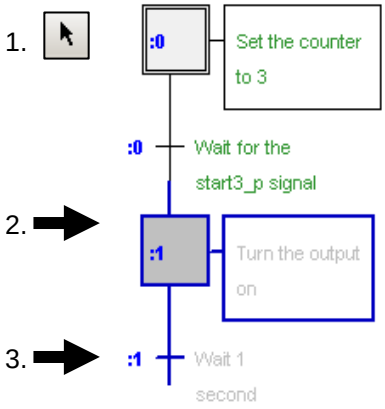
Die markierte Sequenz kann nicht gelöscht werden, wenn dadurch eine ungültige Struktur entstehen würde.



8.4.11 Eine Sequenz kopieren/einfügen

Eine Sequenz in die Zwischenablage kopieren:

1. Aktivieren sie den *Select Mode*.
2. Klicken Sie auf das erste Element der Sequenz.
3. Halten sie die Taste *Shift* gedrückt und klicken Sie auf das letzte Element der Sequenz.
4. Kopieren Sie die Sequenz mit den Menübefehlen *Edit*, *Copy* oder durch drücken von *Ctrl+C* in die Zwischenablage.

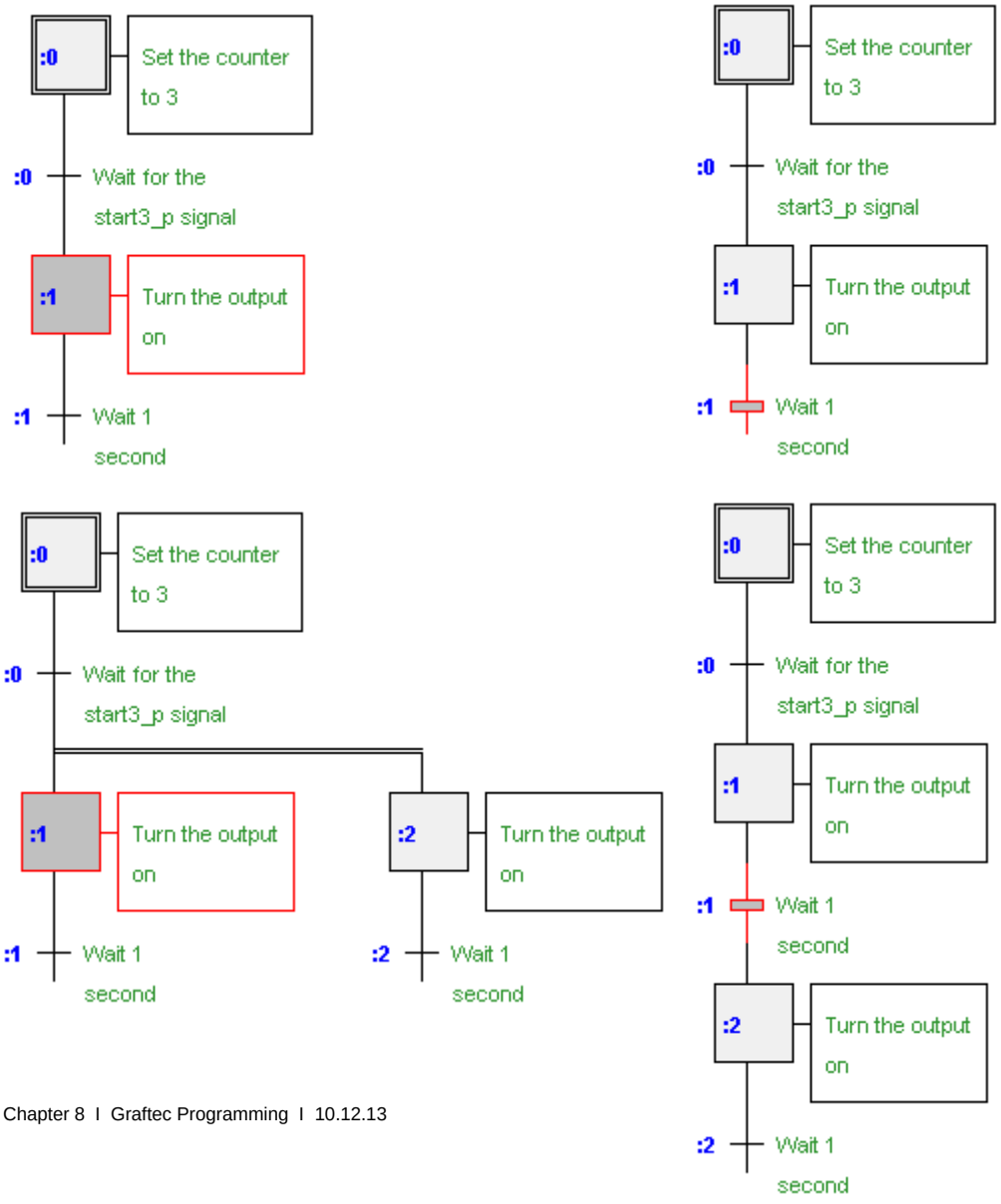


Eine Sequenz aus der Zwischenablage einfügen:

5. Aktivieren Sie den *Select Mode*.
6. Klicken Sie auf das Element vor der einzufügenden Sequenz.

Anmerkung:
Je nach Typ des Zielelementes (Step oder Transition) und den einzufügenden Elementen, wird die Sequenz unter (einfache Sequenz) oder links von (abwechselnde Sequenz) dem ausgewählten Element eingefügt.

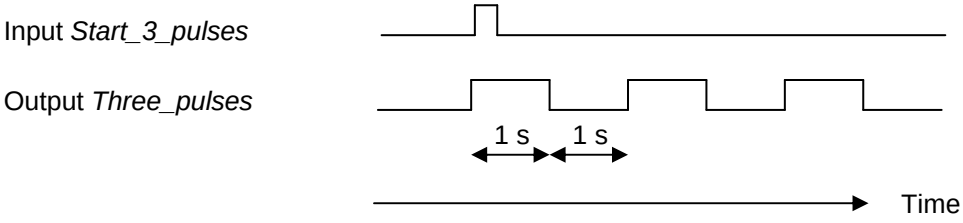
Einfügen auf einen Step, Einfügen auf eine Transition



8.5 Den ersten sequentiellen Block schreiben

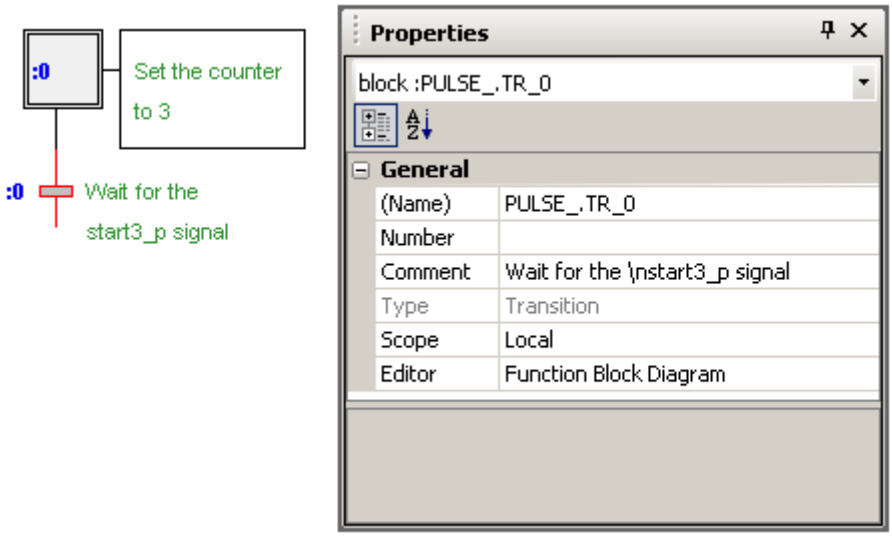
Ziel:
Schreiben Sie ein Programm, das `O 33` (*Three_pulses*) Output blinkt, wenn der Input `I 2` (*Start_3_pulses*) aktiviert ist.

Timing-Diagramm

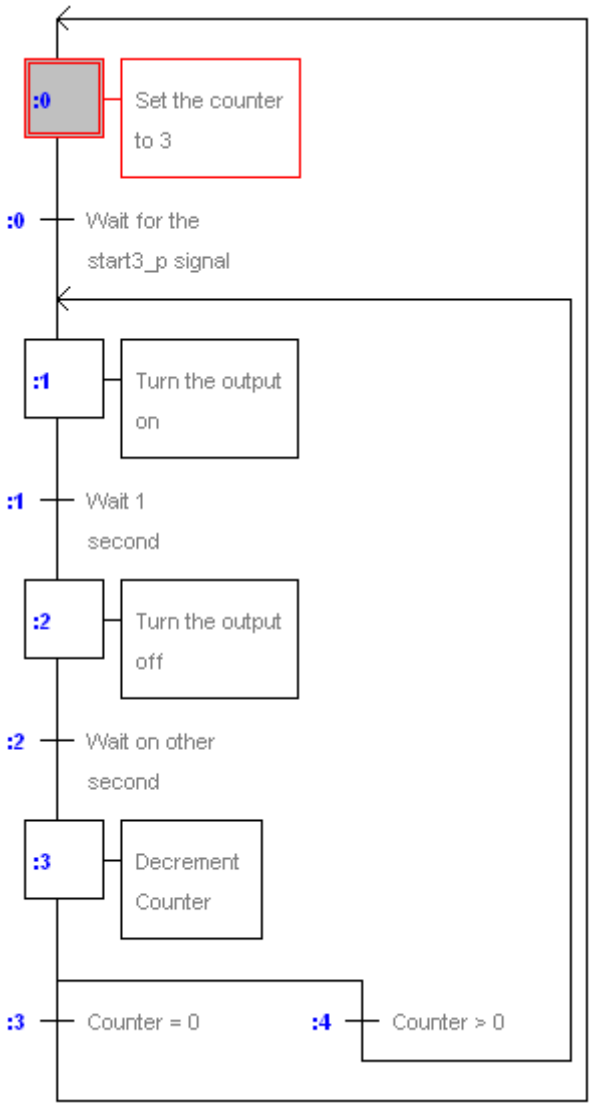


8.5.1 Graftec-Struktur erstellen

Es wird immer mit dem ersten Step begonnen, welcher auch der Ausgangspunkt ist. Dadurch wird der Zähler gestartet. (Der Zähler kann auch ohne Code oder Kommentar bleiben, falls er nicht verwendet wird.) Als nächstes warten wir auf das Signal *Start_3_Pulses*. Bearbeiten Sie die Transition wie unten beschrieben und geben Sie einen Kommentar in das Fenster *Properties* ein.



Setzen Sie den Output *Three_pulses* zu Beginn der Sequenz für eine Sekunde auf high und dann für eine Sekunde auf low. Wiederholen Sie diesen Vorgang drei Mal, wiederholen Sie dann die gesamte Sequenz einschließlich des ersten Steps.

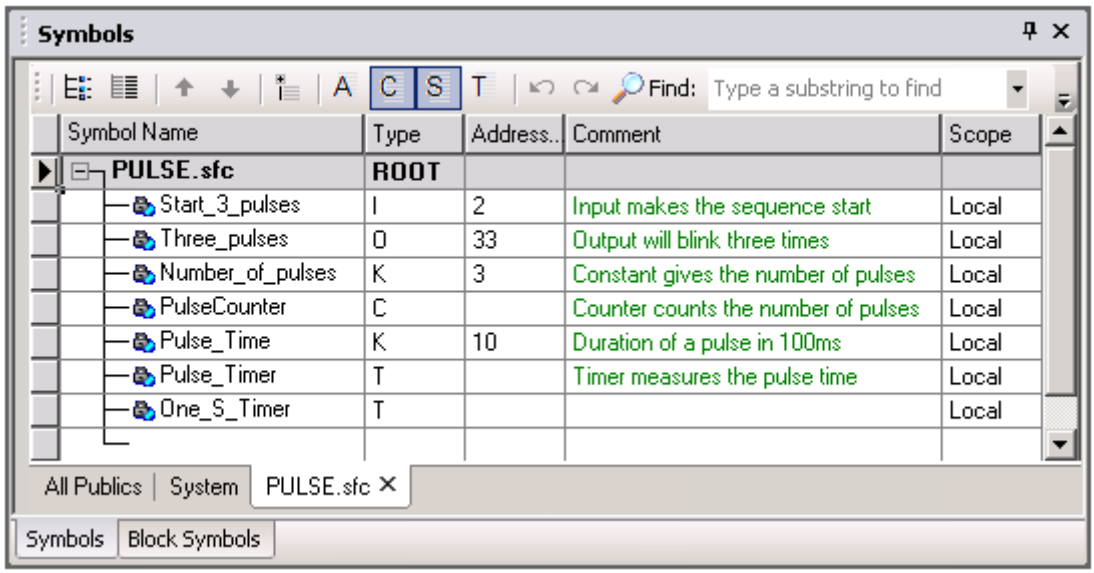


8.5.2 Editor schließen: IL oder Fupla (S-Edit oder S-Fup)

Nachdem die Graftec-Struktur erstellt wurde, muss noch der Code für jeden Step und jede Transition geschrieben werden. Dafür wird entweder der *Instruction List*-Editor (S-Edit) oder der *Function Block Diagram*-Editor (S-Fup) verwendet. Die Wahl des Standardeditors erfolgt über das Dialogfenster *Options*, *Code Editor, Default editor.*

8.5.3 Symbole bearbeiten

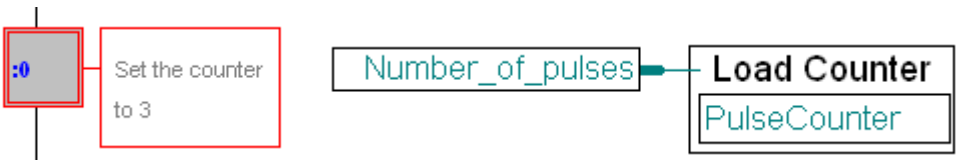
Geben Sie zuerst alle Daten mit Hilfe des Symboleditors an, indem Sie wie unten beschrieben Symbolnamen, Typen, Adressen und Kommentare zuweisen.



8.5.4 Ersten Step programmieren, Zähler laden

Öffnen Sie den ersten Step, um den Code hinzuzufügen, der den Zähler *PulseCounter* mit der konstanten *Number_of_pulses* (3) läd. Doppelklicken Sie auf das zu bearbeitende Element und wählen Sie den Editor *IL Instruction List* oder *Function Block Diagram* (Fupla).

Fupla-Programm:
Verwenden Sie die FBox:*Graftec*, *Load Counter*
Wichtig: Verwenden Sie nicht die Timer oder Zähler aus den Familien, die für zyklische Programme bestimmt sind.



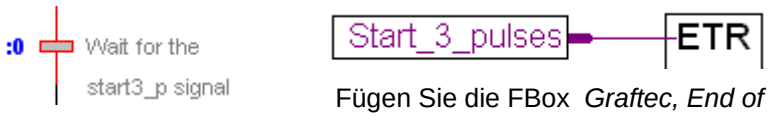
IL-Programm:

```
LD    PulseCounter        ;Initialisierung des Counters
      Number_of_pulses
```

8.5.5 Eine Transition programmieren, auf das Startsignal warten

Eine Transition wird immer wieder bearbeitet bis am Ende der Transition der Input der FBox ETR high ist (Fupla) oder der ACCU high ist (IL). Die Transition 0 wartet darauf, dass der Input *Start_3_pulses* hoch ist.

Fupla-Programm:



Fügen Sie die FBox *Graftec, End of*

IL-Programm:

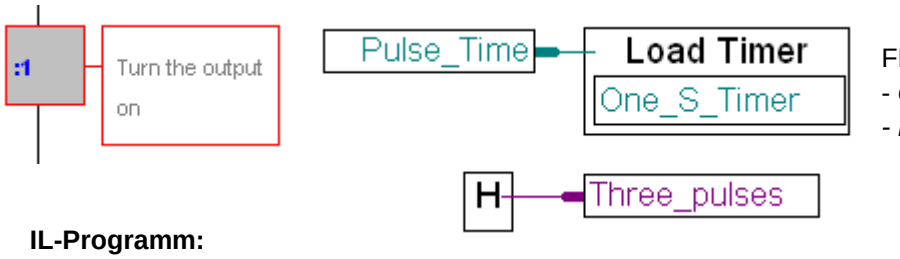
```
;Start  High  –  setzt  den  ACCU  auf  den  Status  von  Input
Start_3_pulses
STH    Start_3_pulses
```

8.5.6 Einen Step programmieren, einen Output einschalten und Timer starten

Dieser Step schaltet den Output ein und lädt den Timer. Der Step geht dann zur nächsten Transition über und wartet, dass der Timer 0 erreicht.

Fupla-Programm:

Die Timer und Zähler in der Standardbibilothek von Fupla wurden nicht für SBs sondern für COBs, die ausschließlich zyklisch durchgeführt werden, entwickelt. Stattdessen müssen Sie die Funktionen der *Graftec*-Familie verwenden, die speziell für SBs entwickelt wurden.Diese können in einen Step geladen und später während einer Transition abgefragt werden.



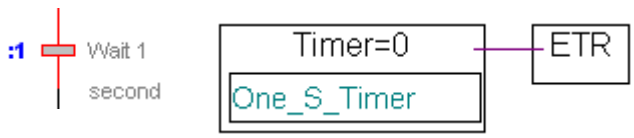
FBox:
- *Graftec: Load Timer*
- *Binary: High*

IL-Programm:

```
SET    Three_pulses      ;Output einschalten
LD     One_S_Timer       ;Timer starten
       Pulse_Tim
```

8.5.7 Auf einen Timer warten

Fupla-Programm:



Fbox:
- *Graftec, Tempo écoulé*
- *Graftec, Fin TR*

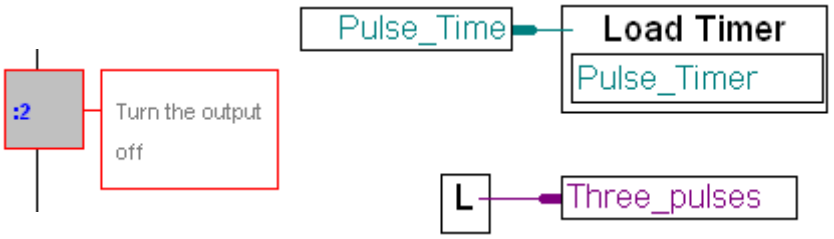
IL-Programm:

```
; den ACCU auf high setzen, wenn der Timer 0 erreicht hat
STL    One_S_Timer
```

8.5.8 Einen Output ausschalten, wenn ein Timer 0 erreicht hat

Schritt und Transition 2 sind Step und Transition 1 ähnlich. Es wird ledigich der Output *Three_pulses* low gesetzt und ein anderer Timer gestartet.

Fupla-Programm:



FBox:
- *Graftec: Load timer*
- *Binary: Low*

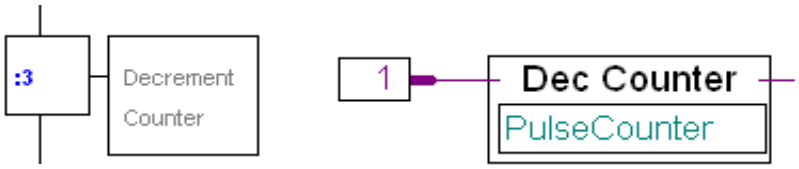
IL-Programm:

```
Three_pulses     ;Output einschalten
LD    Pulse_Timer      ;Timer laden
      Pulse_Time
```

Anmerkung: Für die Steps und Transitionen 1 uns 2 haben wir zwei verschiedenen Timer verwendet (*One_S_Timer* und *Pulse_Timer*). Um allerdings die Anzahl verwendeter Timer möglichst gering zu halten, hätten wir auch nur einen verwenden können, indem wir denselben Timer zwei Mal verwendet hätten, da die beiden nicht gleichzeitig genutzt werden.

8.5.9 Einen Zähler einstellen

Fupla-Programm:
Der Zähler wird bei jeder Ausführung dieses Steps eingestellt , da der Input der FBox high (1) ist.



FBox: *Graftec, Decrement counter*

IL-Programm:

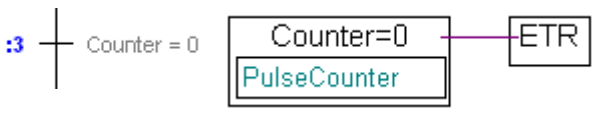
```
; ist der ACCU high, Counter dekrementieren
DEC   PulseCounter
```

Bedenken Sie, dass der zu Beginn einer ST oder TR ACCU immer high (1) ist, so dass ACCU-abhängige Instruktionen immer ausgeführt werden.

8.5.10 Abwechselnde Verzweigung

Die beiden folgenden Transitionen sind auswählbar.

Fupla-Programm:IL-Programm:




```
STL PulseCounter
```

FBox: *Graftec, Counter is zero*

Transition 3: der Input in die FBox ETR ist 1, wenn der Zähler auf 0 steht.
Transition 4: der Input in die FBox ETR ist 1, wenn der Zähler nicht auf 0 steht.

Platzieren Sie mit der Schaltfläche *Invert Binary Connector* aus der Werkzeugleiste einen Inverter auf dem Input der FBox ETR.

8.6 Erstellen und Debuggen des Programms

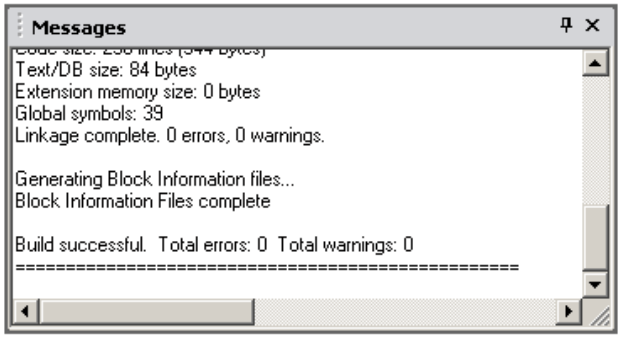
Build All

Sobald die Programmierung fertig ist, klicken Sie auf die Schaltfläche *Build* in der Werkzeugleiste, um das gesamte Programm zusammenzutragen, zusammenzufügen und zu verbinden.

8.6.1 Nachrichtenfenster

Das Fenster *Messages* des Project Managers zeigt die Ergebnisse des Erstellens an. Wenn das Programm fehlerfrei erstellt wurde, wird in der letzten Zeile des Fensters folgendes angezeigt:

Build successful. Total errors: 0 Total warnings: 0



Fehlermeldungen werden rot angezeigt. Durch einen Doppelklick auf die Fehlermeldung, gelangen Sie normalerweise direkt an die Stelle im Programm, die einen Fehler enthält.

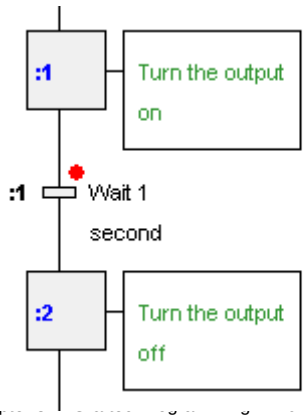
Der Project Manager hat außerdem ein Fenster *Error List*, in dem nur Fehlermeldungen und Warnungen angezeigt werden.

8.6.2 Online-Werkzeuge

Download Program

Nun brauchen Sie das Programm noch herunterladen und es in den Online  dus setzen.

Mit dem Graftec-Editor kann die Ausführung eines sequentiellen Blocks visualisiert werden, während man online ist. Die aktive Transition wird durch einen roten Punkt angezeigt, so dass der sequentielle Ablauf beobachtet werden kann.



Das Programm kann durch einen Klick auf die Schaltfläche *Stop* jederzeit unterbrochen werden und dann schrittweise fortgesetzt werden.

Bei jedem Klick auf die Schaltfläche *Step By Step* wird ein Step oder ein Übergang ausgeführt.

Die Schaltfläche *Step In* öffnet den Step oder die Transition mit dem Fupla- oder IL-Editor, so dass man den Code in dem Step oder der Transition abschreiten kann.

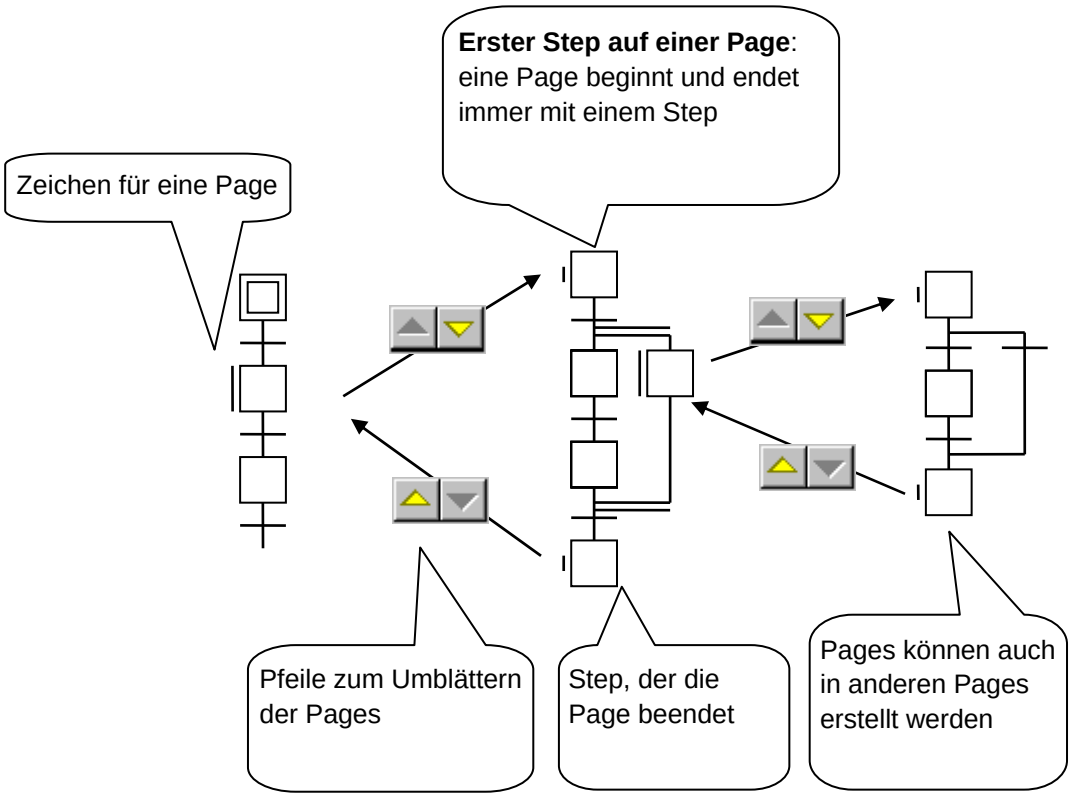
Die Schaltfläche *Run to Element* bearbeitet alle Graftec-Schritte und –Transitions und hält an, wenn das ausgewählte Element erreicht wurde. Läuft das Programm bereits, hält es an dem ausgewählten Element an.

8.7 Ein Graftec-Programm in Pages gruppieren

Mit Graftec kann eine Sequenz von Steps und Transitionen zu einem neuen Element namens *Page* gruppiert werden. Das Element sieht wie ein Step aus und kann einen eigenen Kommentar haben. Zur Unterscheidung hat es eine zusätzliche vertikale Linie auf der linken Seite.

Mit Pages machen es der Graftec-Struktur möglich den Prozess auf einem höheren Level darzustellen und jede Page kann geöffnet werden, um den nächsten Detaillierungsgrad anzuzeigen. Dieser besteht aus weiteren Pages, Steps und Transitionen.

Pages können ebenfalls wieder Pages enthalten. Es gibt keine Begrenzung für die Einbettungstiefe. Die Einbettung von Pages ist mit der Zoomfunktion vergleichbar, welche die Darstellung eines Programms auf mehreren Detaillierungsebenen ermöglicht.



8.7.1 Regeln bei der Bearbeitung von Pages

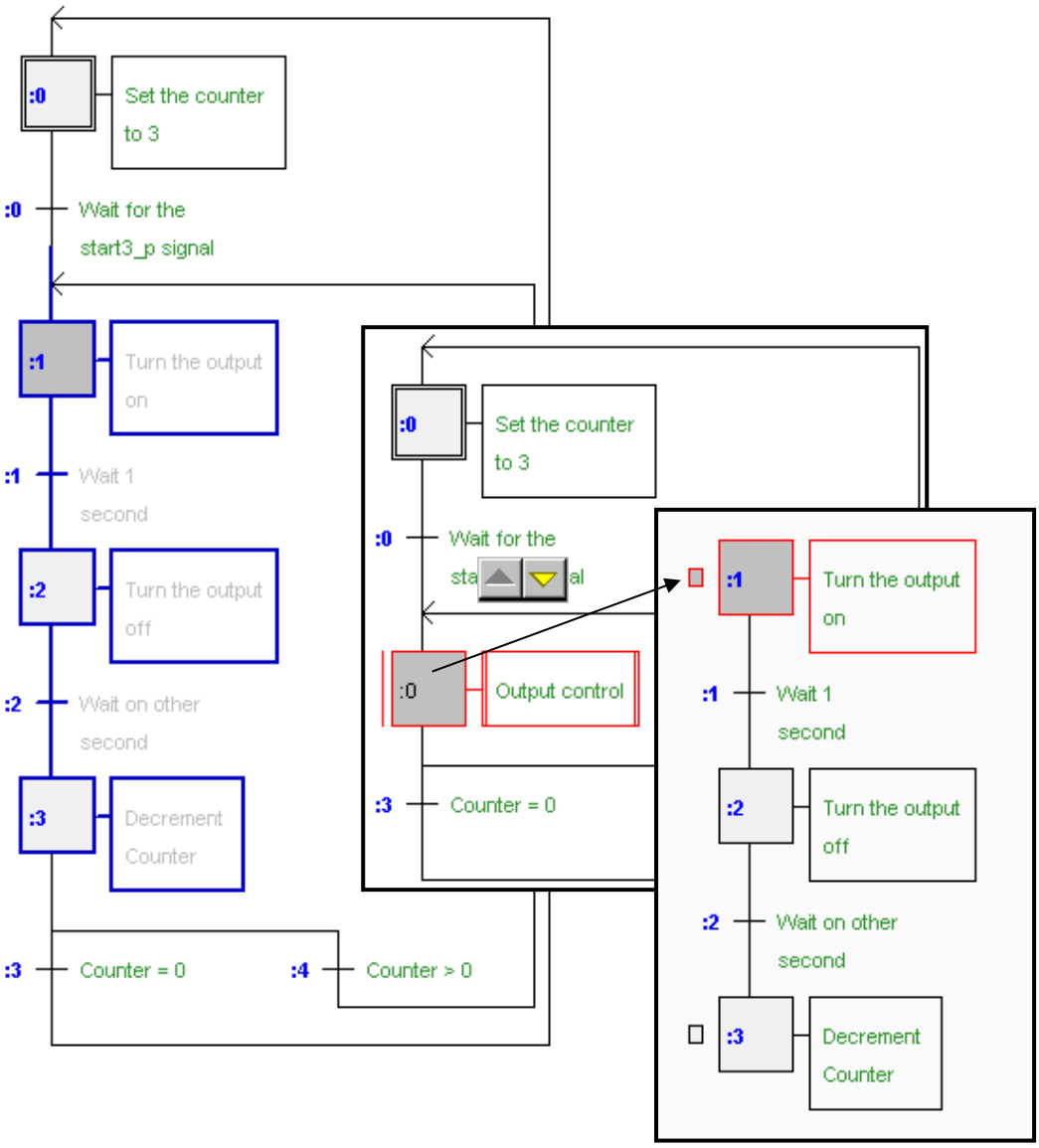
Graftec-Sequenzen, die in Pages umgewandelt werden sollen, müssen nach folgenden Regeln erstellt werden:

- Eine Page muss immer mit einem Step beginnen und enden.
- Die Sequenz darf nicht nur aus einem Step bestehen.
- Die Eingangs- und Ausgangssteps dürfen nicht gelöscht werden.

8.7.2 Eine neue Page erstellen

Gehen Sie wie folgt vor, um eine neue Page zu erstellen:

- Aktivieren Sie den *Select Mode*.
- Wählen Sie den ersten Step für die Page.
- Halten Sie die Taste *Shift* gedrückt und wählen Sie die letzte Page der Sequenz.
- Verwenden Sie den Menübefehl *Page, Create*.



8.7.3 Pages öffnen

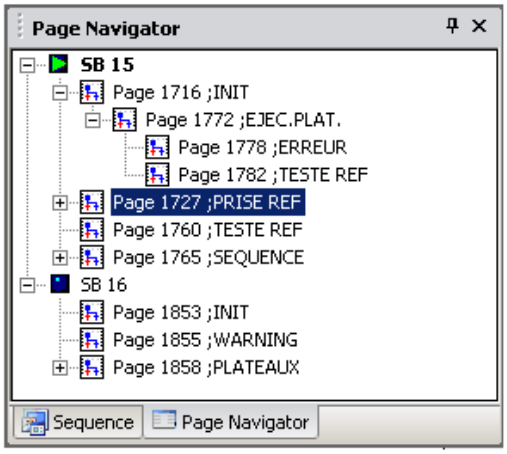
Öffnen Sie die Page und wählen Sie den Menübefehl *Page, Subpage* oder die Werkzeugleiste, um den Inhalt einer Page anzusehen.

Der Befehl *Page, Calling* schliesst die Page, so dass das nächsthöhere Level angezeigt wird. Mit dem Befehl *Go To Main* wird das oberste Level angezeigt.

8.7.4 Eine Page erweitern

Möchten Sie eine Page durch ihre Originalsequenz ersetzen, wählen Sie diese Seite aus und nehmen Sie den Befehl *Page, Expand*.

8.7.5 Block Navigator



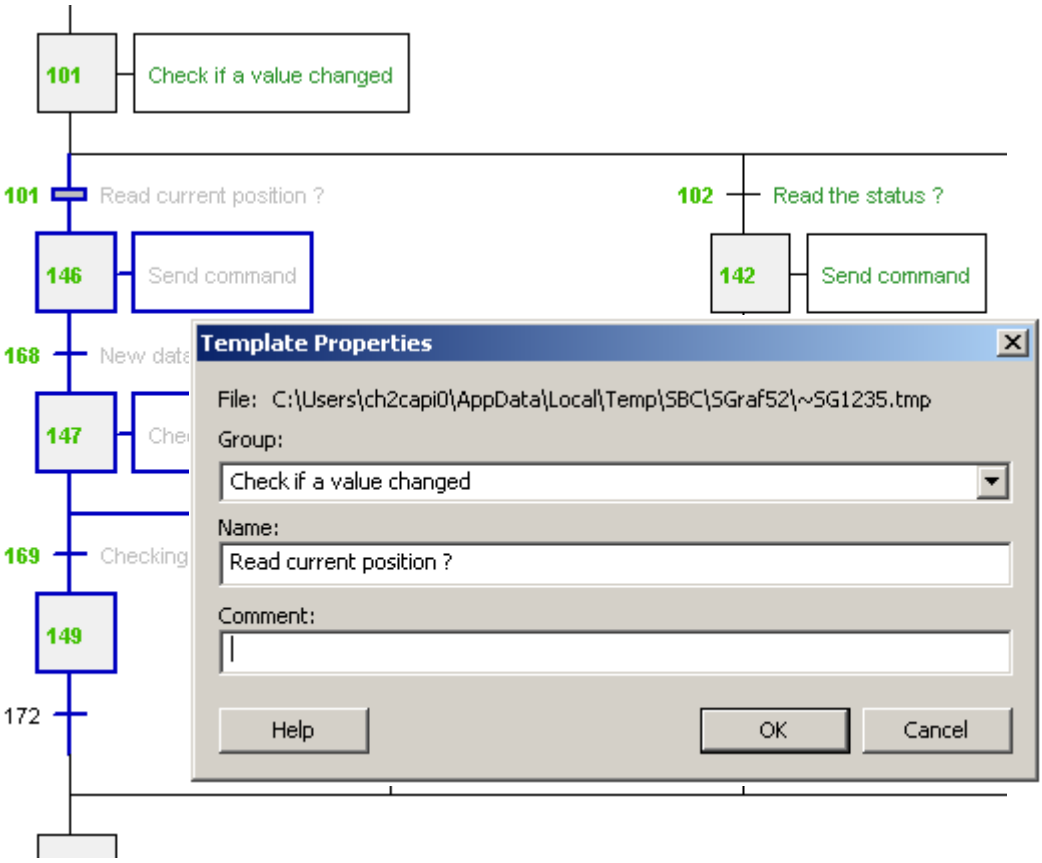
Wir empfehlen ausdrücklich, die Größe einer großen Graftec-Struktur mit Hilfe von *Pages* zu reduzieren. Dadurch kann das Programm leichter gelesen werden und das Navigieren zwischen den Funktionalitäten der höheren Level, die von den Pages dargestellt werden, wird vereinfacht.

Der *Block Navigator* bietet einen globalen Überblick über alle SBs und Pages in der Datei. Wird in dieser Ansicht ein Block oder eine Page ausgewählt, wird der entsprechende Block oder die Seite angezeigt, ohne dass in der Graftec-Struktur danach gesucht werden muss.

8.8 Graftec-*Templates*

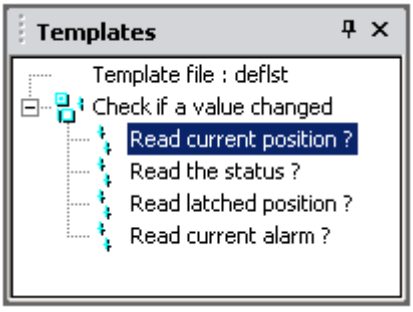
Eine Abfolge von Steps und Transitionen kann in ein *Template* zusammegefasst werden, das in anderen Programmen wie eine Sequenz-Bibliothek verwendet werden kann.

8.8.1 Ein *Template* erstellen



Ein *Template* kann ganz einfach erstellt werden. Wählen Sie eine Sequenz von Steps und Transitionen aus und verwenden Sie den Befehl *Edit, Add to templates*. Der Befehl steht auch im Kontextmenü. Ein Dialogfenster öffnet sich, in dem ein Gruppenname, ein Name für das Template und ein Kommentar eingegeben werden müssen.

Templates sind in *Groups* organisiert, die mit den *Families* der FBoxen vergleichbar sind. In den *Groups* werden die Templates entsprechend der vom Autor definierten Kriterien klassifiziert. Templates werden unter dem Namen der Group im Fenster *Templates* aufgelistet.

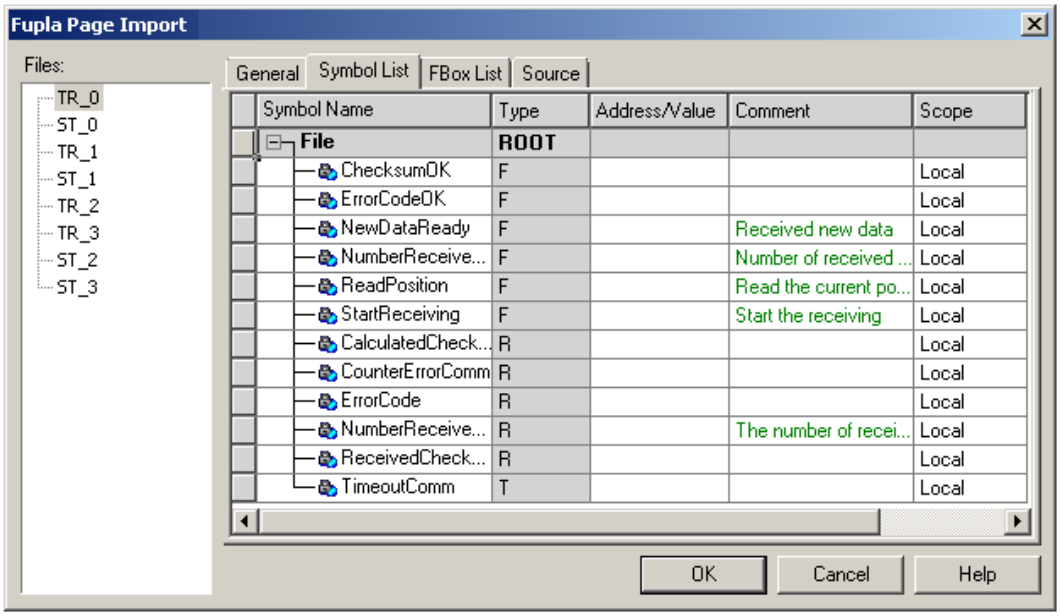


Die Icons zeigen an, wie das *Template* beginnt und endet. Jede beliebige Sequenz kann in ein Template umgewandelt werden, sie können Pages, Zweige usw. enthalten und in Fupla oder IL kodiert sein.

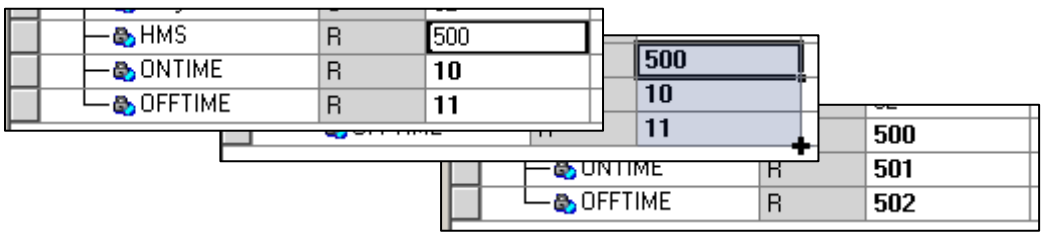
8.8.2 *Templates* importieren

Die Templates können in allen Projekten verwendet werden. Öffnen Sie das Fenster *Templates* mit *View, Templates* und ziehen Sie das Template mit Drag&Drop in die Graftec-Struktur. Alle Steps, Transitionen, Zweige, Symbole, Kommentare und der dazugehörige Fupla- oder IL-Code werden eingefügt.

Ein Dialogfenster wird angezeigt, in dem die Namen, Adressen, Kommentare und Umfang der von dem *Template* importieren Symbole sowie verschiedene andere Daten verändert werden können. Diese Funktion kann mit einem *Macro* oder einer *Function Box* mit Parametern verglichen werden.



Die *Symbol List* zeigt alle im *Template* enthaltenen Symbole an. Am schnellsten lassen sich die Symbole umbenennen und damit Duplikate von Symbolnamen vermeiden, wenn alle Symbole in einer Symbolgruppe zusammengefasst werden. Mit dem Kontextmenübefehl *Insert Pre-group* werden die unten angezeigten Symbole in die Gruppe mit einem frei wählbaren Namen verschoben.



Um die Adressen der Symbole zu aktualisieren, können sie durch einen Klick auf die Schaltfläche *Type* am Kopf der Spalte nach Typen sortiert werden. Bearbeiten Sie dann die Adresse des ersten Elementes und ziehen Sie das kleine Quadrat am unteren rechten Rand der Zelle nach unten, um alle neu zu nummerierenden Adressen auszuwählen.

Beachten Sie die Parameter auf dem Reiter *General*, wenn Sie dasselbe *Template* mehrfach importieren wollen. Damit kann durch Drücken der Taste # ein Index in die Symbolnamen oder *Groups* eingefügt werden. Dieses Zeichen wird automatisch durch die

Indexzahl ersetzt, die sich bei jeder Kopie des *Templates* um Eins erhöht. Dazu kann auch der Kontextbefehl *Indexing* verwendet werden.

Inhalt

9 IL-Programmierung (Instruction List)

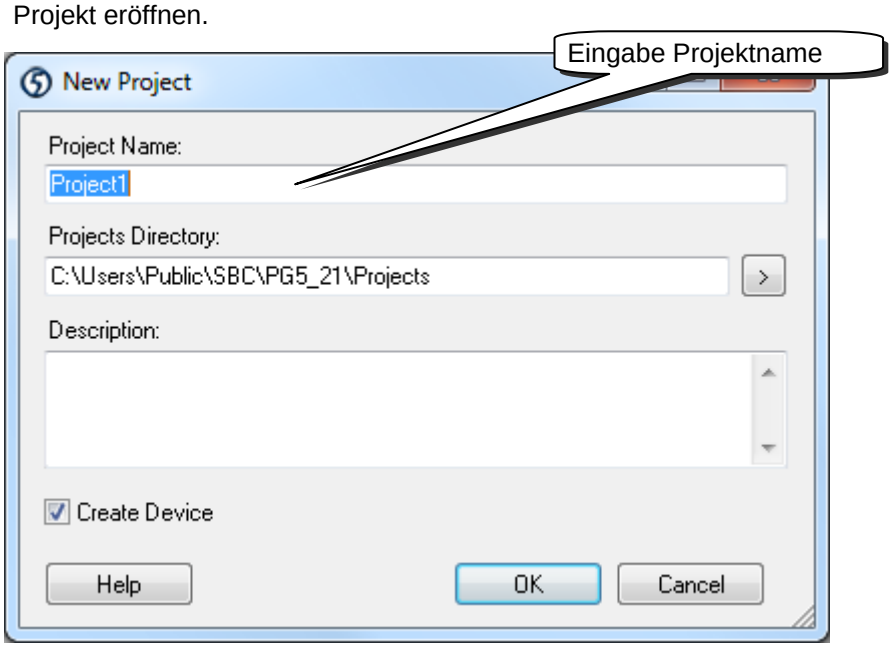
Der Saia PG5 IL Editor ist das flexibelste und mächtigste Werkzeug zum Programmieren von Saia PCD Steuerungen. IL steht für Instruction List: eine nicht-graphische Programmierumgebung zum Schreiben von Programmen mit Hilfe des mächtigen PCD Befehlssatzes. Alle PCD Steuerungen benutzen diesen Befehlssatz, dies garantiert die Portabilitität eines Programms von einer PCD zur anderen. Der IL Editor ist mehr als eine wertvolle Programmierhilfe, sondern ausserdem ein leistungsfähiges Diagnose- und On-line Test-Werkzeug.

9.1 Vorbereitung eines IL Projektes

Vor dem Schreiben eines Beispiels, empfehlen wir ein neues Projekt und eine neue Programm-Datei anzulegen.

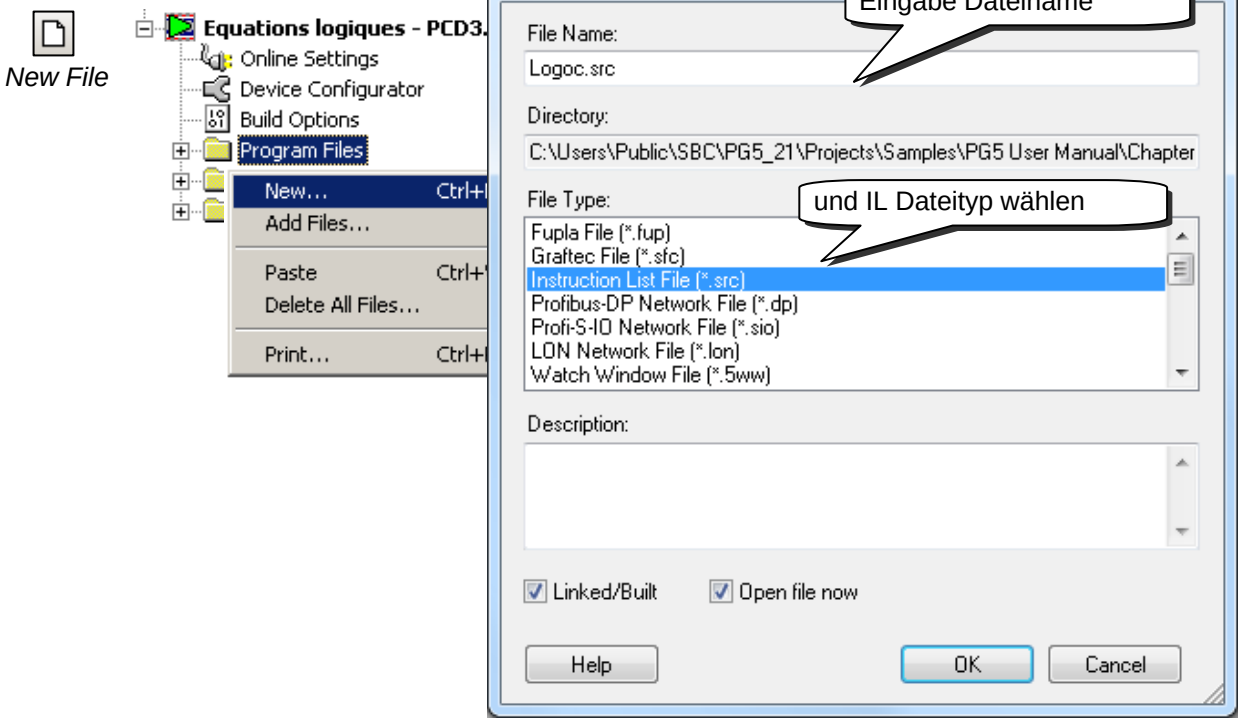
9.1.1 Eröffnen eines neuen Projekts

Im *Saia PG5 Project Manager* Fenster, Menü *Project, New…*auswählen und ein neues Projekt eröffnen.

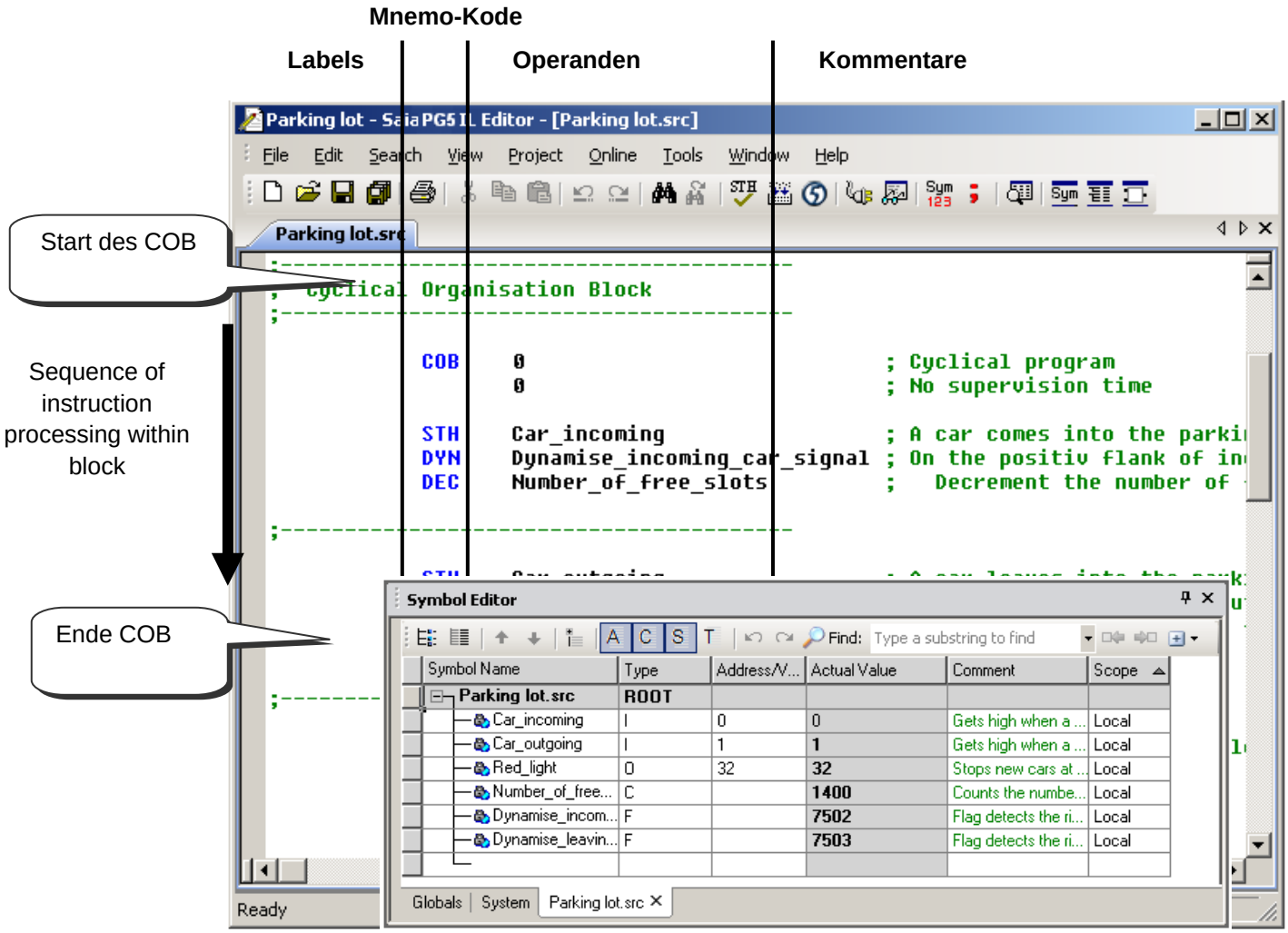


9.1.2 Eröffnen einer neuen IL Programm-Datei

Dem Projekt eine neue Programm-Datei hinzufügen, indem der Ordner *Program Files* ausgewählt wird, dann rechter Maus-Klick und Menü *New…* auswählen (oder den *New File* Knopf in der Werkzeugleiste drücken):



9.2 Aufbau des IL Editor-Fensters



Der IL Editor ähnelt einem gewöhnlichen Text Editor. Funktionen, wie *Copy/Paste* oder *Suchen/Ersetzen* sind auch hier zu finden, darüber hinaus bietet der IL Editor einiges mehr, wie:

- Seiten-Layout dem Schreiben von PCD Programmen angepasst
- Unterscheidung der Informationstypen durch farbige Schriften
- Im Programm verwendete Symbole werden im *Symbols* Fenster aufgelistet
- Das Programm wird On-line angezeigt und kann Schritt für Schritt ausgetestet werden

9.2.1 Schreiben von Programm-Zeilen

```
Label      Mnemo. Operand          Kommentar

;Increment a register
           STH     Flag            ;Copy the Flag state into the accu
           DYN     DFlag           ;On a positiv flank of the Flag , set the accu eigh
           JR      L Next          ;If the accu is low, jump to the label Next
           INC     Register        ;  Increment the register
Next:      NOP                     ;No instruction
```

IL Programm-Zeilen sind in 4 Kolonnen formatiert:

Label
Rote Schrift, das Label ist der Symbolname für eine Programm-Zeile. Dies ist hilfreich für Programmsprünge. (JR L Next)

Mnemo-Kode
Blaue Schrift, der Mnemo-Kode - oder Befehl – bestimmt welchen Zustand der Operand (Eingang, Ausgang, Flag, Register…) einnehmen soll.

Operand
Schwarze Schrift, beschreibt den Operanden-Typ: Eingang, Ausgang, Flag, Register … und die Adresse.

View Symbols or Values

Der *View Symbols or Values* Knopf zeigt entweder die Operanden-Adresse oder das entsprechend Symbol.

```
STH     Flag
```
⇒
```
STH     F 0
```

Kommentar
Anwender-Kommentare sind in grüner Schrift und beginnen mit einem Strichpunkt. Sie stehen rechts neben dem Operanden, sie können aber auch eine ganze Zeile einnehmen.

```
$SKIP
Author:     Dupont Fred
Date:       28.10.2003
File:       Logic.src
$ENDSKIP
```

Nimmt der Kommentar mehrere Zeilen ein, muss nicht jedes Mal am Zeilenanfang ein Strichpunkt gesetzt werden, stattdessen wird der Kommentar zwischen die beiden Klammer-Befehle $skip und $endskip geschrieben. Dies sagt dem Assembler, dass der ganze Text zwischen diesen beiden Befehlen als Kommentar anzusehen ist.


View User or Auto Comment

Der *View User or Auto Comment* Knopf zeigt entweder die Anwender-Kommentare oder die den Operanden automatisch angefügte Kommentare.

```
STH     Flag                ;Copy the Flag state into the accu
```
⇒
```
STH     Flag                ;Control the incrementation
```

9.2.2 Format der Programm-Zeilen

Ist die *Auto Format while Typing* Option ausgewählt, bewirkt die *Enter* Taste auf der Tastatur eine automatische Formatierung der Programm-Zeilen. Die Option ist zu finden im Menü *Tools, Options* des IL Editors. Dort kann auch die Zeilenbreite eingestellt werden.

Ist eine andere Formatierung erwünscht, kann durch markieren einiger oder aller Zeilen mit *Tools, Auto Format* neu formatiert werden.

9.2.3 Aufbau eines Organisationsblocks

IL-Datei eines kleinen Programms

Abfolge der Instruktionen innerhalb eines Blocks

```
COB   0     ;Beginn von COB 0
      0     ;Überwachungszeit
            ;  ausgeschaltet
STH   I 1   ;Beispiel für eine logische
            ;  Gleichung
AND         I 2
OUT         O 32
ECOB        ; Ende von COB 0
```

Die Saia PCD Programmier-Sprache stellt so genannte Organisationsblöcke bereit, in die der Anwender seine Programme schreibt.

Es gibt mehrere Arten von Blöcken: Der Zyklische Organisationsblock (COB) ist für immer wieder abzuarbeitende Programmteile; Sequentielle Blöcke (SB) zum programmieren von Ereignissen, die genau definiert nacheinander auftreten, Programm Blöcke (PB) für Subroutinen; Funktionsblöcke (FB) für Subroutinen mit Parametern; Ausnahme Organisationsblöcke (XOB) für spezielle Ereignisse.

Blöcke beginnen immer mit einem Start Befehl und enden mit einem Ende Befehl. z. B., der Befehl COB kennzeichnet den Beginn eines Zyklischen Organisationsblocks. Er endet mit dem gleichen Befehl, dem aber der Buchstabe E für "Ende" (ECOB) vorangestellt ist. Alle Programm-Kodes, die zu diesem Block gehören, müssen zwischen diesen beiden Befehlen COB und ECOB stehen, niemals ausserhalb des Blocks.

Selbst das kleinste PCD Programm besitzt immer einen COB. Weitere Blöcke können nach Bedarf hinzugefügt werden.

9.2.4 Abarbeiten der Befehle und Blöcke

Innerhalb eines Blocks arbeitet die PCD die Programm-Instruktionen Zeile für Zeile ab, beginnend mit dem Start-Befehl bis zum Ende-Befehl.

Die Reihenfolge, wie Instruktionszeilen innerhalb eines Organisationsblocks geschrieben werden müssen, ist wichtig. Die Reihenfolge jedoch, in welcher die Organisationsblöcke selbst geschrieben sind ist unwichtig. Verschiedene Regeln legen das Abarbeiten der Blöcke fest:

Bei einem PCD Kaltstart schaut die Steuerung immer zuerst nach dem XOB 16, dem Kaltstart-Block. Ist dieser programmiert, wird er immer zuerst ausgeführt, gleichgültig, ob er am Beginn oder am Ende einer Datei programmiert ist.

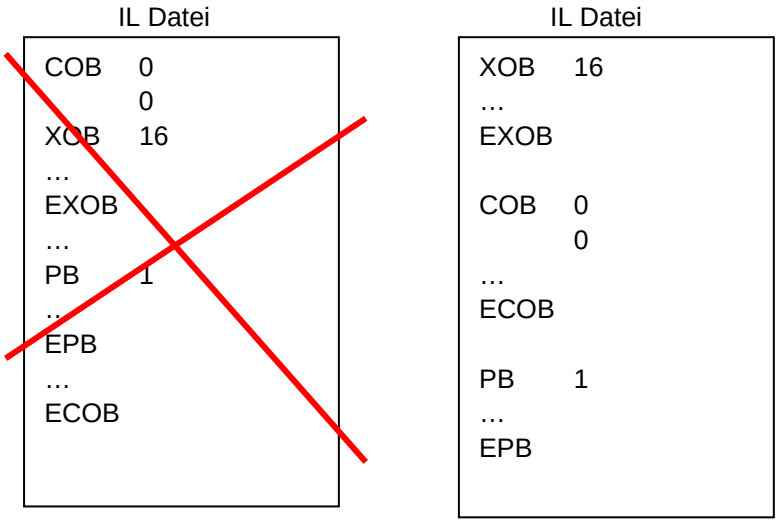
Dann folgen die COBs im Programm. Diese werden in numerischer Reihenfolge abgearbeitet: COB 0, COB 1, … COB 15, gleichgültig in welcher Reihenfolge sie in der Datei auftreten. Nach dem letzten COB beginnt das Programm wieder mit COB 0.

Alle Sequentiellen Blöcke (SB), Programm-Blöcke (PB) und Funktionsblöcke (FB) werden im Anwender-Programm mit den Befehlen CSB (Call SB), CPB (Call PB) und CFB (Call FB) aufgerufen. Das Anwender-Programm bestimmt also, wann und in welcher Reihenfolge SBs, PBs und FBs ausgeführt werden.

Alle Ausnahme Organisationsblöcke werden automatisch, wenn das betreffende Ereignis eintritt, aufgerufen. Diese Ereignisse sind unvorhersehbar und können jederzeit auftreten. Die Reihenfolge der Abarbeitung kann nicht festgelegt werden. Jedes Hardware- oder Software-Ereignis ist mit einem bestimmten XOB verbunden. Die Ereignisse können durch den Anwender nicht verändert werden. Der Anwender kann jedoch die Aktion, die innerhalb des XOBs ausgeführt werden soll, frei programmieren.

9.2.5 Regeln beim Schreiben von Blöcken

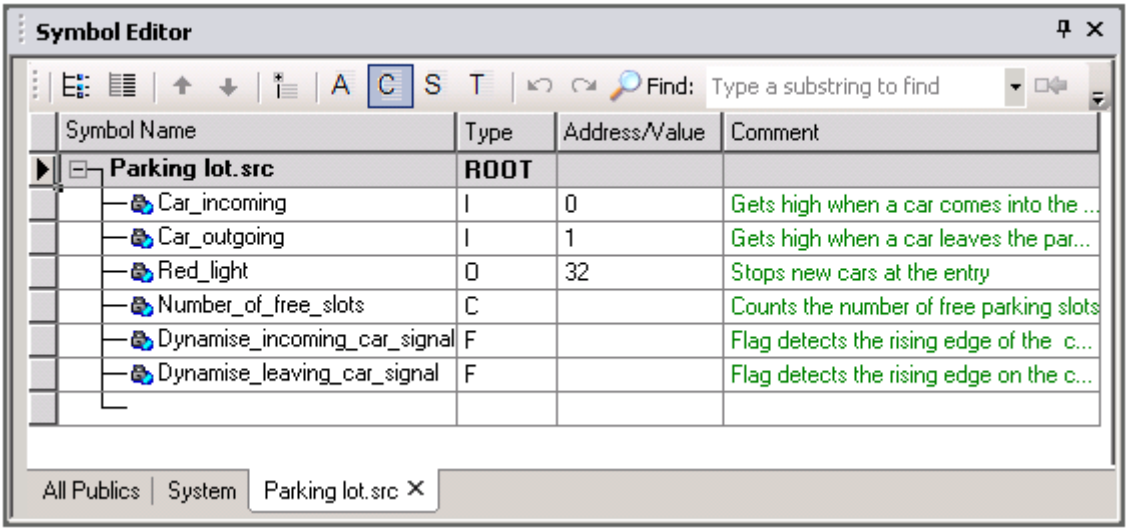
Wenn auch Blöcke in jeglicher Reihenfolge geschrieben werden können, sind doch folgende Regeln einzuhalten:



Blöcke dürfen nicht in andere Blöcke geschrieben werden. Es muss immer einer dem anderen folgen.
Keine Programm-Instruktion darf ausserhalb eines Blocks stehen, mit Ausnahme von Symbol-Definitionen, Texten und Datenblöcken.

9.3 *Symbols Editor*-Fenster

Show Hide Symbols Editor



Das *Symbols Editor* Fenster enthält eine Liste aller Operanden eines Programms. Sie kann mit dem *Show/Hide Symbol Editor* Knopf oder mit dem Menü-Befehl *View/Symbol Editor* aufgerufen werden. Jede Zeile zeigt alle Informationen die zu einem bestimmten Operanden gehören und bildet gleichzeitig ein Symbol:

Symbol Name
Ein Symbol ist ein Name der die Adresse eines Eingangs, Ausgangs, Flags, Registers…bezeichnet Es ist ratsam Symbol-Namen innerhalb eines Programms zu verwenden und nicht die absolute Adresse eines Flags oder Registers. So kann eine Adresse oder ein Datentyp im *Symbol Editor* enster leicht abgeändert werden. Anstatt die Änderung in jeder Programm-Zeile durchzuführen, muss nur im *Symbol Editor* geändert werden. Das Risiko, eine Programm-Zeile zu vergessen oder einen schwer auffindbaren Fehler zu produzieren, entfällt.

Syntax für Symbol-Namen
Das erste Zeichen ist immer ein Buchstabe, gefolgt von weiteren Buchstaben, Ziffern, oder dem Underscore-Zeichen. Sonderzeichen (ö,è,ç,…) sind zu vermeiden. Gross/Kleinschreibung hat keinen Einfluss: MotorOn und MOTORON ergeben die gleichen Symbole.

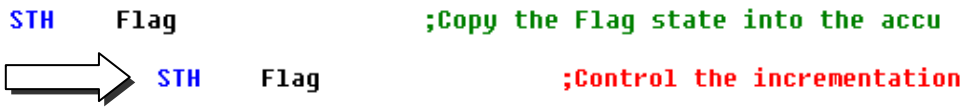
Type
Bestimmt den Typ des Operanden: Input(I), Output(O), Register(R), Counter(C), Timer(T), text(X), DB, …

Address
Jeder Operandentyp besitzt einen bestimmten Bereich verfügbarer Adressen:

Ein-/Ausgänge:	abhängig von den in der PCD gesteckten Modulen
Flags:	F 0…F 16383
Register:	R 0, …, R 16383
Timer/Counter:	T/C 0…T/C 1599

Comment
Der Kommentar ist mit dem Symbol verbunden. Er wird im *Symbols*-Fenster angezeigt und muss nicht in den Programm-Zeilen gelesen werden.
Hin- und Herschalten mit dem Knopf *View User* or *Auto Comment.*

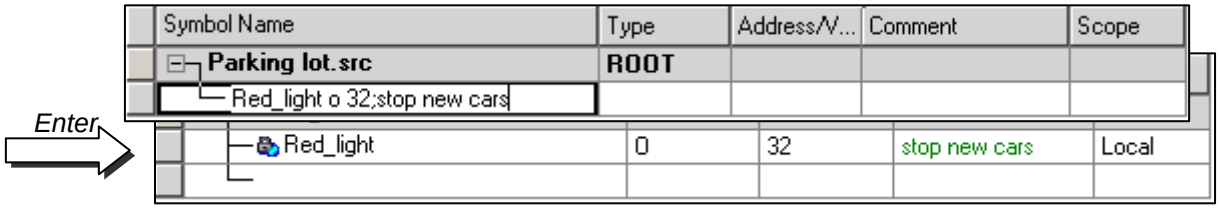
View User or Auto Comment



9.3.1 Neue Symbole der *Symbols Editor* Liste hinzufügen

Einfache Methode:
Zum hinzufügen von neuen Symbolen zur Liste das *Symbols Editor* Fenster öffnen, die Maus in der Mitte des Fensters positionieren und mit rechtem Maus-Klick das Kontext-Menü *Insert Symbol* auswählen. Dann die Felder *Symbol Name*, *Type*, *Address/Value*, *Comment* and *Scope* ausfüllen.

Schnelle Methode 1:

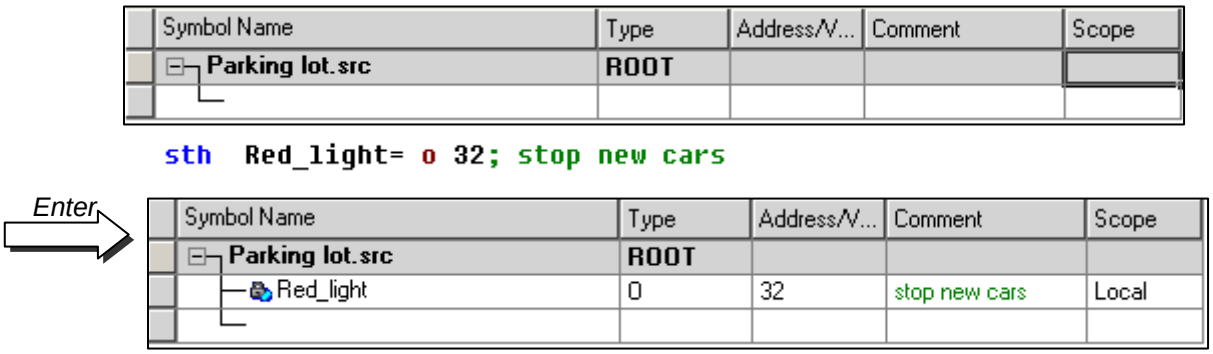


Eine weitere Möglichkeit ist die Eingabe von Variablen für die verschiedenen Informations-Felder vom *Symbol Name* Feld. Das ist praktischer und schneller. Siehe Beispiel unten.

Folgende Syntax einhalten:
symbol_name, type, address; comment

Wurde das neue Symbol nach obiger Syntax festgelegt, *Enter* Taste auf der Tastatur betätigen und die Informationen werden automatisch in die richtigen Felder geschrieben.

Schnelle Methode 2:



Neue Symbole könne auch während des Programmierens hinzugefügt werden. Dazu eine Programm-Zeile mit Mnemo-Kode und Operand editieren. Für den Operanden Symbol-Name und Definition nach folgender Syntax eingeben:
symbol_name = type, address; comment

Enter Taste auf der Tastatur betätigen und das neue Symbol wird automatisch der *Symbols Editor* hinzugefügt, jedoch nur bei richtiger Symbol-Definition und nur, wenn die *Automatically add entered type/value to the Symbol Table* Option ausgewählt war (Menü *Tools, Options* im IL Editor).

9.3.2 Operanden Adressierung

Eine Symbol-Definition muss nicht unbedingt all die Information wie unten enthalten. Drei Typen der Adressierung werden unterschieden:

Absolute Adresse

Group/Symbol	Type	Address/Value	Comment
	Output	32	Stops new cars

Es sind nur der Typ und die Adresse (z.B. 32) definiert und optional ein Kommentar. Wird direkt im Programm die absolute Adresse benutzt, ist dies beim ändern von Typ oder Adresse von Nachteil. Das Anwenderprogramm übernimmt die in der Symbol-Liste durchgeführten Änderungen nicht. Änderungen müssen manuell in jeder betroffenen Programm-Zeile durchgeführt werden. Deshalb sind Symbol-Namen vorzuziehen, optional mit dynamischer Adressierung.

Symbol-Namen

Group/Symbol	Type	Address/Value	Comment
red_light	Output	32	Stops new cars

Die Daten sind durch Symbol-Name, Typ, Adresse und optionalem Kommentar definiert. Änderung des Symbols, Typs oder der Adresse wird von der Symbol-Liste unterstützt und jede betroffene Programm-Zeile des Anwender-Programms wird automatisch geändert.

Dynamische Adressierung

Group/Symbol	Type	Address/Value	Comment
red_light	F		Stops new cars

Dies ist eine Form von symbolischer Adressierung bei der die Adresse nicht definiert wird. Die Adresse wird automatisch beim Verarbeiten (build) des Programms hinzugefügt. Die Adresse wird von einem Adressbereich genommen, der in den *Build Options* definiert ist. (Siehe Project Manager.)

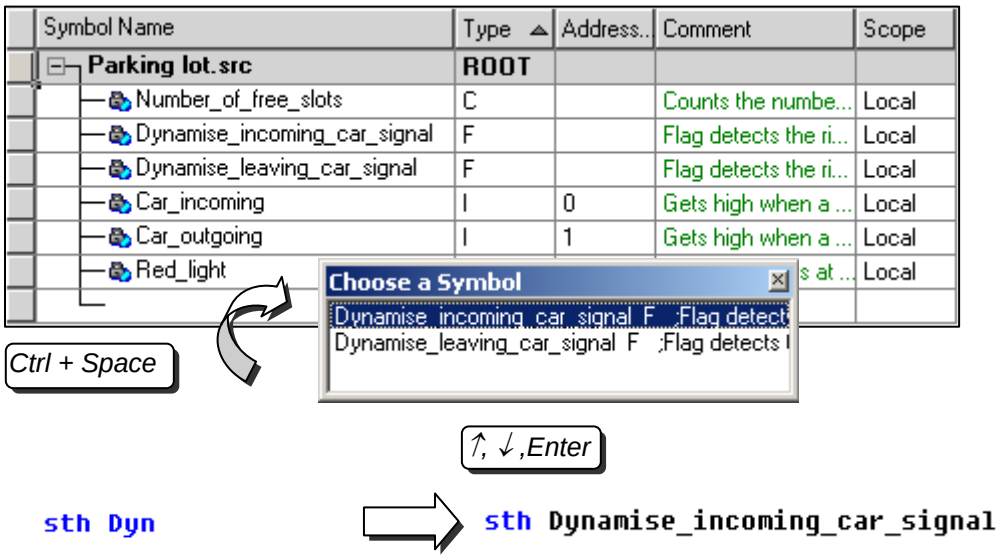
Bemerkung: Dynamische Adressierung kann für Flags, Counter, Timer, Register, Texte, DBs, COBs, PBs, FBs und SBs angewendet werden. Eingänge, Ausgänge und XOBs müssen jedoch absolut adressiert werden.

9.3.3 Benutzen der Symbole aus der *Symbols Editor* im IL Programm

Nach erfolgter Programmierung können die im *Symbols Editor* festgelegten Symbole auf verschiedene Arten benutzt werden:

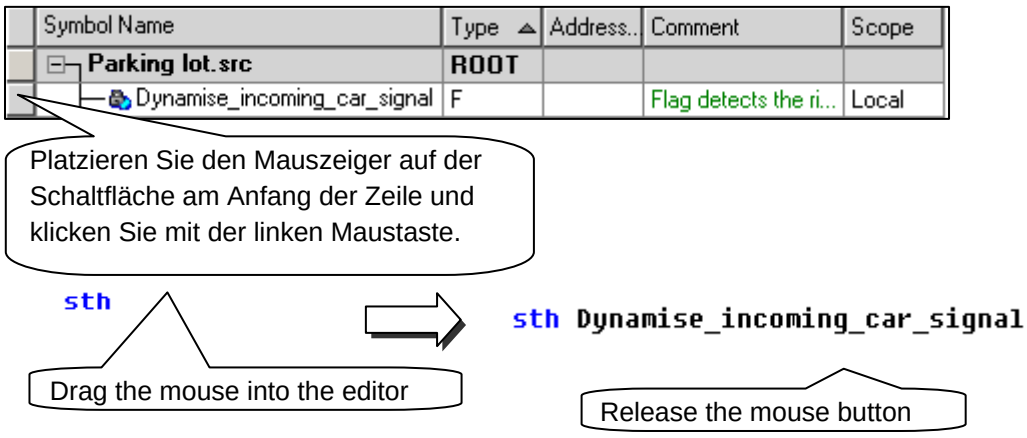
Symbol eingeben über die Tastatur
Der Symbol-Name wird für jede Instruktion, die diesen benutzt, vollständig über die Tastatur eingegeben. Diese Methode birgt die Gefahr von Schreibfehlern, die erst bei der Verarbeitung des Programms bemerkt werden.

Symbol eingeben durch selektive Suche



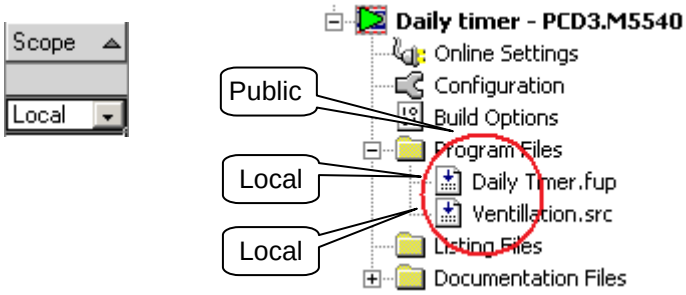
Nur einige wenige Zeichen des Symbol-Namens auf der Tastatur eingeben und dann die *Ctrl+Space* Tasten drücken hat zur Folge, dass ein Fenster mit einer Liste all der Symbole erscheint, die mit der eingegebenen Buchstabenkombination beginnen. Das benötigte Symbol kann mit der Maus oder den Pfeiltasten auf der Tastatur ausgewählt (↑ ↓) und mit *Enter* übernommen werden.

Symbol eingeben mit Drag-and-Drop



Diese Methode der Symbol-Eingabe schliesst alle Schreibfehler aus. Im *Symbols Editor* den Maus-Cursor über der Schaltfläche am Anfang der Symbolzeile platzieren, linke Maustaste drücken und gedrückt halten. Maus-Cursor in den IL-Editor ziehen und Maustaste loslassen. Das ausgewählte Symbol wird automatisch an den vom Maus-Cursor angezeigten Platz eingefügt.

Local, Public und External Symbole



Die Zugänglichkeit eines Symbols wird über dessen Reichweite definiert:

Lokale Symbole sind nur innerhalb der Datei zugänglich, in der die Definition des Symbols enthalten ist. So sind beispielsweise *Public* Symbole nur in *Daily timer.fup* aus allen Dateien im *device* zugänglich.

Public Symbole werden von den beiden Dateien *Parking lot.fup* und *Ventilation.src* im *Device Daily timer* genutzt, unabhängig davon, welche Datei das Symbol definiert.

Normalerweise werden im Symbol Editor drei Reiter angezeigt: ein Reiter mit dem Namen der geöffneten Fupla-Datei, ein Reiter namens *All Publics* und ein Reiter namens *System*.

In dem Reiter mit dem Namen der geöffneten Datei können sämtliche in der Datei verwendeten Symbole bearbeitet werden. Die Definitionen der public und lokalen Symbole sowie Referenzen auf externe Symbole werden ebenfalls auf dieser Seite vorgenommen und in einer Datei, beispielsweise *Daily Timer.fup*, gespeichert. Das Feld *Scope* definiert, ob ein Symbol *Local*, *Public* oder *External* ist:

In Fupla sind neue, mit dem Symbol Editor oder dem Fupla Editor erstellte Symbole standardmässig entweder *Local* oder *Public*. Das ist davon abhängig, welche Option unter *View, Options, Symbols, Add symbols with Public scope* in Fupla definiert wurde.

Der Reiter *All Publics* zeigt alle public Symbole im Device an. Der Reiter *System* zeigt alle Systemsymbole an. Die Symbole auf den Seiten *All Publics* und *System* werden aktualisiert, wenn eine Datei gespeichert wird oder ein *Build* beendet wurde. Diese Seiten nutzen die Ergebnisse des Build, um alle public und Systemsymbole aus allen Dateien im Programm des Device zu sammeln und sie alle in einer Tabelle anzuzeigen.

Soll ein public oder ein Systemsymbol im Programm platziert werden, wählen Sie das Symbol auf der Seite aus und ziehen Sie es mit der Drag-and-Drop-Funktion auf die Seite. Der Verweis auf das public Symbol wird auf der Symbolseite der Datei mit der Reichweite *External* angegeben. Damit wird angezeigt, dass das Symbol in einer anderen Datei definiert ist.

Die Symbole auf der Seite *All Publics* können nicht bearbeitet werden. Die Definitionen public Symbole können nur innerhalb der Datei bearbeitet werden, in der sie definiert sind. Mit dem Befehl *Goto Definition* im Kontextmenü können Sie die Datei öffnen, die Symbol definiert. Die Spalte *File* zeigt den Namen der Datei an, in der das Symbol definiert wird. Diese Datei muss zur Veränderung des Symbols geöffnet werden.

9.4 Einführung in den PCD-Befehlssatz

Dieses Kapitel gibt eine Übersicht in den PCD-Befehlssatz. Tiefer gehende Informationen zu jedem Befehl sind im “Handbuch Befehlssatz für die PCD Familie 26/733“ oder in der PG5 Hilfe zu finden. Um spezielle Hilfe zu einem Befehl aus dem IL Editor zu erhalten, den Befehl schreiben, den Cursor darauf positionieren und F1-Taste drücken. Generelle Hilfe gibt es auch im Menü *Help, Instruction List Help.*

9.4.1 Der Akkumulator

Der Akkumulator ist ein binärer Wert, der durch einen Binärbefehl und einige ganzzahlige Befehle gesetzt wird. Jede PCD hat nur einen Akkumulator, der als spezielles Flag angesehen werden kann. Der Status des Akkumulators kann mit dem ACC Befehl beeinflusst werden. Mit dem ACC Befehl kann der Akkumulator auch mit dem Wert eines Status Flags beeinflusst werden (siehe Beschreibung der Status Flags).

Beispiele:
ACC H
Setzt den Akkumulator auf high

ACC L
Setzt den Akkumulator auf low

ACC C
Invertiert (Komplement) den Akkumulator Status

9.4.2 Binäre Befehle

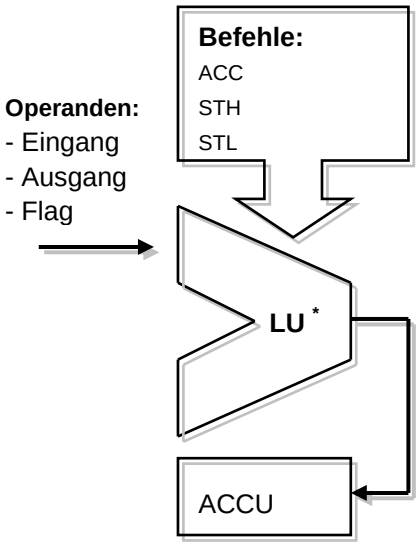
Binäre Befehle nehmen nur zwei Zustände ein: 0 oder 1 (low oder high). Diese Befehle werden für binäre Gleichungen mit den Zuständen der PCD Eingänge, Ausgänge, Flags, Counter und Timer benutzt.

Binäre Befehle betreffen immer den Akkumulator. Einige binäre Befehle wirken auf den Status des Akkumulators:

Beispiele:
ACC H
Setzt den Akkumulator auf high

ACC L
Setzt den Akkumulator auf low

STH I 4
Kopiert Status an Eingang 4 in Akkumulator
Akkumulator Status ist high, wenn 24 V am Eingang 4 liegen.
Akkumulator Status ist low, wenn 0 V am Eingang 4 liegen..

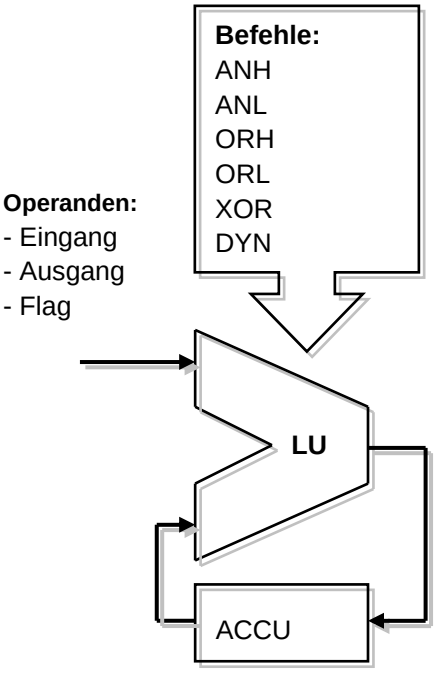


Weitere Befehle lesen den Status des Akkumulators, um binäre Funktionen auszuführen und das Ergebnis an den Akkumulator zurück zu schicken:

Beispiele:
ANH I 5
Liest Akkumulator Status und führt eine logische UND Funktion mit dem Status des Eingangs 5 aus. Der Akkumulator wird auf den Wert des Ergebnisses gesetzt.

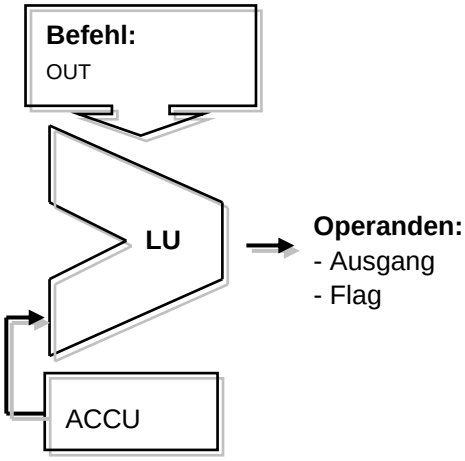
ORH F 100
Liest Akkumulator Status und führt eine logische ODER Funktion mit dem Status des Flags 100 aus. Der Akkumulator wird auf den Wert des Ergebnisses gesetzt.

XOR T 3
Liest Akkumulator Status und führt eine logische XODER Funktion mit dem Status des Timers 3 aus. Der Akkumulator wird auf den Wert des Ergebnisses gesetzt.



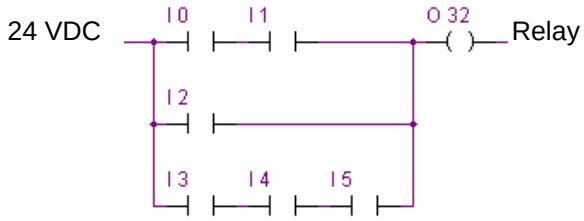
Das Ergebnis aus einer binären Gleichung wird immer im Akkumulator gespeichert. Der *OUT* Befehl kopiert den Inhalt des Akkumulators an einen Ausgang oder ein Flag:

Beispiel:
OUT O 32
Kopiert Akkumulator Status an Ausgang 32.
Ist der Akkumulator Status high, stehen 24 V am Ausgang 32.
Ist der Akkumulator Status low, stehen 0 V am Ausgang 32.



Beispiel: Programmierung einer einfachen binären Gleichung

Dieses Programm-Beispiel verarbeitet die binäre Gleichung: O32 = I0*I1+I2+I3*I4*I5
Die Gleichung wird auch durch diese Schaltung veranschaulicht:

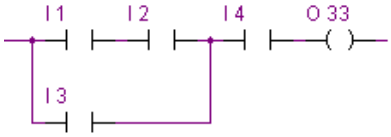


Eine binäre Gleichung beginnt immer mit einem STH oder STL Befehl, dem dann die notwendigen ANH (*), ORH (+), XOR Funktionen folgen.
Beachten, dass der ORH Befehl Priorität vor dem ANH Befehl hat. Jeder ORH Befehl kennzeichnet den Beginn einer neuen Kontaktreihe in der obigen Schaltung. Das Teil- oder Endergebnis aus einer binären Gleichung wird immer in den Akkumulator geschrieben. Mit dem OUT Befehl kann das Akkumulator Ergebnis zum Ändern des Status eines Ausgangs oder Flags benutzt werden.

```
COB  0      ;Start des zyklischen Programms
     0
STH  I 0    ;Status von Eingang I 0 wird an Akkumulator kopiert:
            ;     Accu = I0
ANH  I 1    ;AND-Funktion zwischen Akkustatus und Status von
            ;     Eingang 1:Accu = I0*I1
ORH  I 2    ;OR-Funktion zwischen Akkustatus und Status von
            ;     Eingang 2:Accu= I0*I1+I2
ORH  I 3    ; Accu = I0*I1+I2+I3
ANH  I 4    ; Accu = I0*I1+I2+I3*I4
ANH  I 5    ; Accu = I0*I1+I2+I3*I4*I5
OUT  O 32   ;Ergebnis der Gleichung im Akkumulator wird an
            ;     einen Ausgang kopiert
ECOB        ;Ende des zyklischen Programms
```

Beispiel: Programmierung einer binären Gleichung mit anders bewerteter Reihenfolge

Dieses Programm-Beispiel verarbeitet die binäre Gleichung: O33 = (I1*I2+I4)*I3
Die Gleichung wird auch durch diese Schaltung veranschaulicht:



Manchmal ist es notwendig die Reihenfolge der Prioritäten von binären Gleichungen zu ändern. Dazu werden Klammern in die Gleichung eingefügt. Der PCD Befehlssatz enthält jedoch keine Klammern. Die Gleichung muss daher in zwei kleinere Gleichungen aufgeteilt werden. Die erste Gleichung berechnet den Teil in den Klammern und speichert den Wert vorübergehend in ein Flag. Die zweite Gleichung nimmt das Zwischenresultat vom Flag und berechnet das Endergebnis.

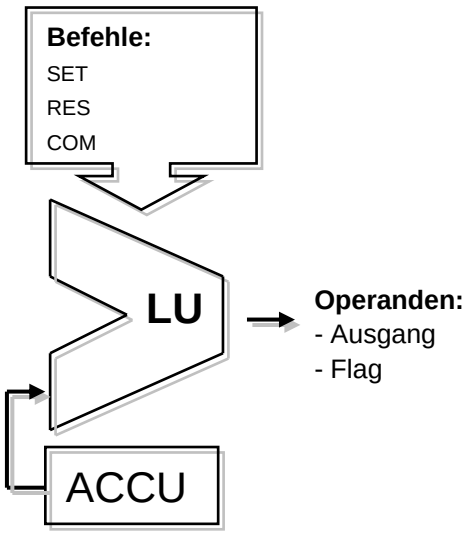
```
COB  0
     0
STH  I 1   ;Erste Gleichung
ANH  I 2
ORH  I 4
OUT  F 0   ;Ergebnis der Funktion in Klammern: F0 =(I1*I2+I4)

STH  F 0   ;Zweite Gleichung
ANH  I 3
OUT  O 33  ;Endergebnis: O 33 = F0*I3
ECOB
```

Mit weiteren binären Befehlen kann der Akkumulator dazu benutzt werden, den Zustand von Ausgängen oder Flags zu ändern. Jeder Befehl unterstützt eine andere Funktion.

Beispiele:
SET O 32
Ist der Akkumulator Status high, geht Ausgang 32 auf high. Andernfalls verharrt der Ausgang im gegenwärtigen Zustand.

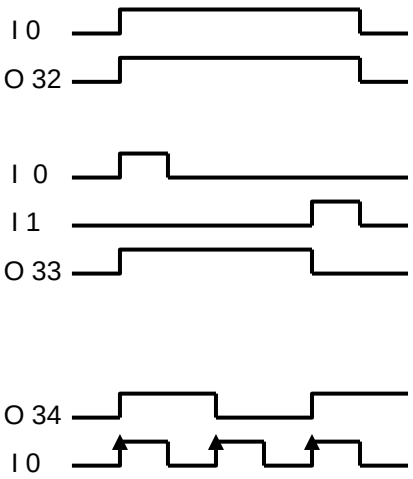
RES O 32
Ist der Akkumulator Status high, geht Ausgang 32 auf low. Andernfalls verharrt der Ausgang im gegenwärtigen Zustand.

COM O 33
Ist der Akkumulator Status high, wird Ausgang 33 auf high invertiert. Andernfalls verharrt der Ausgang im gegenwärtigen Zustand.



Beispiel:
Dieses Beispiel zeigt die Unterschiede der Befehle OUT, SET, RES, und COM

```
COB  0
     0
STH  I 0
OUT  O 32  ;I 0 wird an O 32 kopiert

STH  I 0
SET  O 33  ;Hohen Status (high) in
           ;     Ausgang 33 speichern

STH  I 1
RES  O 33  ;Niedrigen Status (low) in
            ;    Ausgang 33 speichern

STH  I 0   ;Auf steigender Flanke von I 0
DYN  F 1
COM  O 34  ;Invertierter Status von
           ;     Ausgang 34
ECOB
```



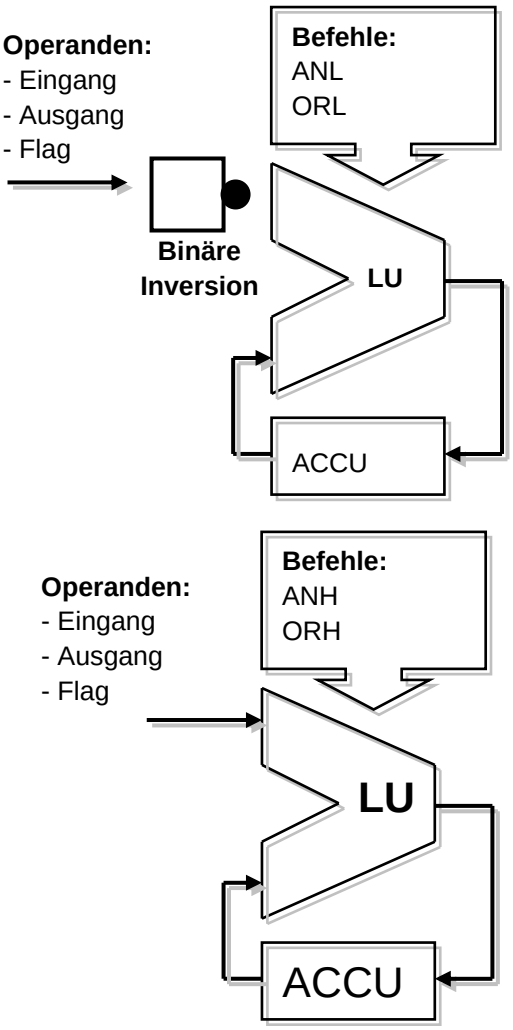
Einige binäre Befehle enden mit dem Buchstaben H oder L. Befehle, die mit L enden invertieren den Zustand einer Information bevor sie ihre Funktion ausführen.

Beispiele:
STH I 4
Kopiert Status von Eingang 4 in den Akkumulator. Akkumulator Status ist high, wenn 24 V am Eingang 4 liegen.

STL I 4
Kopiert den invertierten Status von Eingang 4 in den Akkumulator. Akkumulator Status ist low, wenn 24 V am Eingang 4 liegen.

ANH I 5
Führt eine logische UND Funktion mit dem Akkumulator Status und dem Status von Eingang 5 aus.

ANL I 5
Führt eine logische UND Funktion mit dem Akkumulator Status und dem invertierten Status von Eingang 5 aus.



9.4.3 Dynamisierung

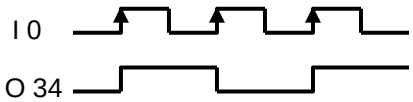
Binäre Befehle benutzen in der Regel den binären low oder high Zustand zur Ausführung einer binären Funktion oder Statusänderung eines Ausgangs oder Flags.

Manchmal interessiert aber nicht der binäre low oder high Zustand, sondern der Übergang von low zu high (z. B. um einen Counter hochzuzählen).

Zum Erkennen einer steigenden Flanke, ist folgendermassen vorzugehen: Das Ergebnis einer binären Gleichung in den Akkumulator stellen und mit dem *DYN* Befehl den positiven Wechsel feststellen. Nach dem *DYN* Befehl wird der Akkumulator Status, beim Erkennen eines positiven Wechsels high, andernfalls wird er low. Das Flag für den *DYN* Befehl darf nur für einen einzigen Dynamisierungsbefehl eingesetzt werden, weil der Status des Flags für den nächsten Programm-Zyklus erhalten bleibt.

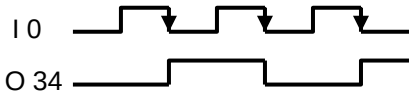
Beispiel: Erkennen einer steigenden Flanke

```
STH  I 0
DYN  F 3
COM  O 34
```



Beispiel: Erkennen einer fallenden Flanke

```
STL  I 0
DYN  F 3
COM  O 34
```



Zum besseren Verständnis des *DYN* Befehls im Programm oben, schlagen wir vor, den *DYN* Befehl zu entfernen und zu beobachten wie sich das Programm dann verhält.

9.4.4 Status Flags

Im Gegensatz zu binären Befehlen benutzen Wort-Befehle den Akkumulator selten, aber fast immer verändern sie die speziellen Status Flags.

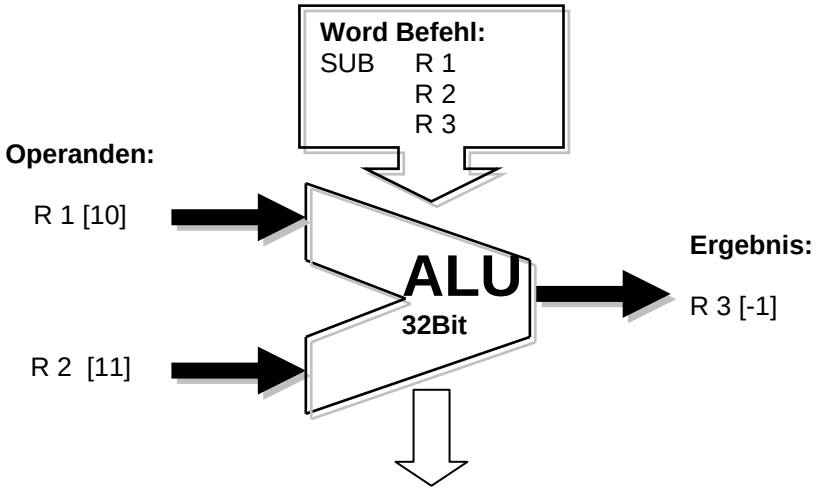
Die speziellen Status Flags werden mit Wort-Befehlen geändert und informieren über das Ergebnis:

Flag positiv	P	Gesetzt, bei positivem Ergebnis
Flag negativ	N	Gesetzt, bei negativem Ergebnis
Flag Null	Z	Gesetzt, bei Ergebnis Null
Flag Error	E	Gesetzt, im Fehlerfall

Das Error Flag kann, um den Ausnahme Block XOB 13 aufzurufen, aus mehreren Gründen gesetzt werden:
Überlauf, verursacht durch die Multiplikation zweier grosser Zahlen
Division durch Null
Quadratwurzel aus einer negativen Zahl
Fehler an der Kommunikations- Schnittstelle (SASI Befehl)

Beispiel: Status Flags nach einer Subtraktion

Status Flags werden gesetzt, abhängig vom Ergebnis einer Subtraktion (R 3 = R 1 – R 2). Register Werte stehen in eckigen Klammern []. Das Ergebnis der Subtraktion ist negativ: einzig Flag N ist gesetzt.

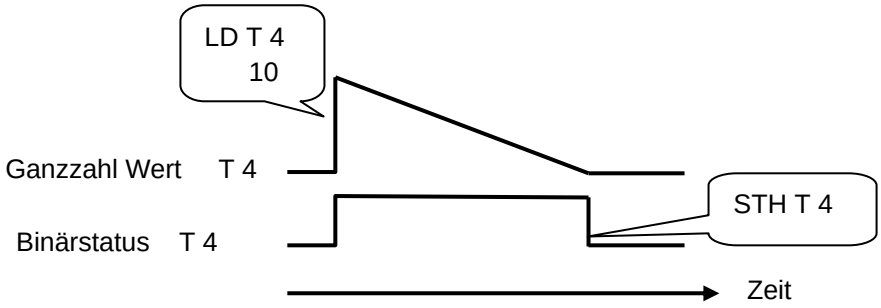


E	Z	N	P
0	0	1	0

Status Flags können auch in den Akkumulator kopiert werden: für binäre Befehle, Programm Sprungbefehle, oder den Aufruf von PBs, FBs oder SBs:

ACC P	Status Flag P in den Akkumulator kopieren
ACC N	Status Flag N in den Akkumulator kopieren
ACC Z	Status Flag Z in den Akkumulator kopieren
ACC E	Status Flag E in den Akkumulator kopieren

9.4.5 Wort-Befehle für Timer



Timer enthalten zwei Werte: den Wert für die feste Verzugszeit und den Binärstatus.
Zum Einfügen einer Verzugszeit, die Zeit als positiven ganzzahligen Wert laden entsprechend der Länge der Verzugszeit in Zehntelsekunden. Die Steuerung dekrementiert diesen Wert automatisch, bis Null erreicht ist. Der Binärstatus des Timers während des Dekrementierens ist high und geht auf low, wenn die Zeit den Wert Null erreicht.

Laden einer Verzugszeit
LD T 4

Ist der Akkumulator Status high, wird Timer T 4 mit einer Konstante 10 geladen. Andernfalls behält der Timer den gegenwärtigen Wert bei.

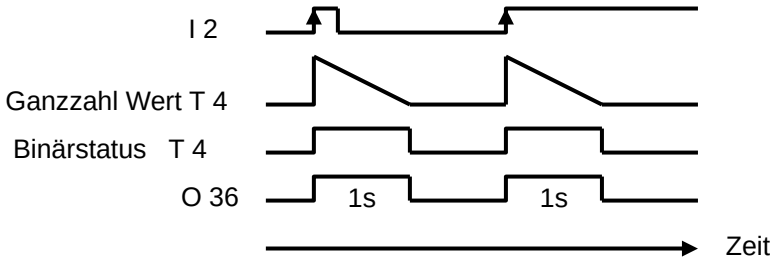
Lesen des Timer-Zustands

Use a binary instruction, such as:

STH T 4 , ANH T 4, ORH T 4, …

Beispiel:
Sende bei jeder aufsteigenden Flanke an Eingang 2 einen Ein-Sekunden-Impuls an Ausgang 36.

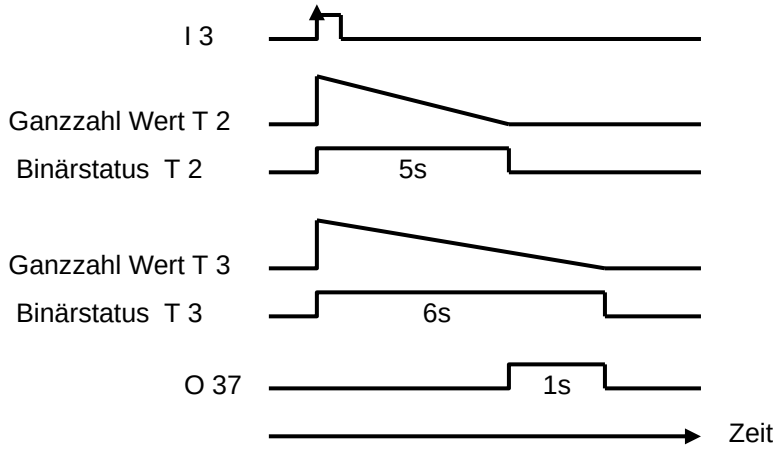
Statusdiagramm:



Programmierung:

```
COB  0
     0
STH  I 2   ;Erkennen einer steigenden Flanke an Eingang 2 ...
DYN  F 2   ;setzt Status des Akkus auf ‚high'
LD   T 4   ;Wenn Akku high ist, wird Verzugszeit
     10    ;für 10 Zeiteinheiten geladen

STH  T 4   ;Logischer Status des Zeitverzugs wird an
           ;Ausgang 36 kopiert
OUT  O 36
ECOB
```

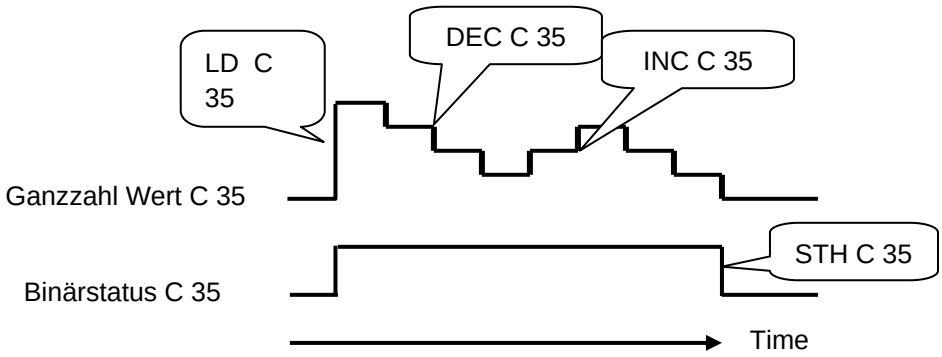
Beispiel:
Sende, mit einer Verzögerung von 5 Sekunden, bei jeder aufsteigenden Flanke an Eingang 3 einen Ein-Sekunden-Impuls an Ausgang 37.
Status-Diagram:



Programmierung:

```
COB  0
     0
STH  I 3
DYN  F 3
LD   T 2
     50
LD   T 3
     60
STH  T 2
XOR  T 3
OUT  O 37
ECOB
```

9.4.6 Befehle für Counter



Wie die Timer haben auch die Counter zwei Werte: den Ganzzahlwert und den Binärstatus des Counters.
Zum Ausführen des Zählens, den Counter mit einem ganzzahligen positiven Wert laden.
Im Gegensatz zu den Timern, werden Counter nur durch Befehle im Anwender-Programm inkrementiert oder dekrementiert. Der Binärstatus des Counters ist high, solange der Zählwert grösser als Null ist und geht auf low, wenn der Wert Null erreicht ist.

Laden eines Counters
LD C 35
10
Ist der Akkumulator Status high, wird Counter 35 mit der Konstante 10 geladen. Andernfalls behält der Counter den gegenwärtigen Wert bei.

Lesen des Counter-Zustands

Binär-Befehle einsetzen, wie:

STH C 35, ANH C 35, ORH C 35, …

Counter inkrementieren
INC C 35
Ist der Akkumulator Status high, wird Counter 35 um eine Einheit inkrementiert. Andernfalls behält er den gegenwärtigen Wert bei.

Counter dekrementieren
DEC C 35
Ist der Akkumulator Status high, wird Counter 35 um eine Einheit dekrementiert. Andernfalls behält er den gegenwärtigen Wert bei.

Status Flags
Die Befehle Counter inkrementieren und Counter dekrementieren ändern, abhängig vom Ergebnis der Operation die Status Flags an (**P**ositiv, **N**egativ, **Z**ero, **E**rror).

Beispiel: Zählimpulse von einem binären Eingang mit einem Counter.

```
COB  0
     0
STH  I 2   ;Eingangsstatus wird an Akkumulator kopiert
DYN  F 3   ;Akkustatus wird an der positiven Flanke von I 2
           ;  auf ‚high' gesetzt
INC  C 35  ; Wenn Akkustatus high ist, Counter inkrementieren
ECOB
```

Die Befehle *STH* und *DYN* lesen die Information von Eingang 2 und setzen bei jeder aufsteigenden Flanke den Akkumulator Status high oder auf low beim Ausbleiben einer Flanke. Abhängig vom Akkumulator Status, inkrementiert der *INC* Befehl den Counter 35.

9.4.7 Akkumulator-abhängige Befehle

Binäre Befehle benutzen den Akkumulator sehr häufig und manche Wort-Befehle ebenso.

Aber nicht alle Befehle benutzen den Akkumulator auf die gleiche Weise. Es gibt 7 Befehle, die ihn in spezieller Weise nutzen, die Akkumulator-abhängigen Befehle. Sie werden nur erzeugt, wenn der Akkumulator vorher auf high gesetzt war. Der Akkumulator Status ist daher eine entscheidende Bedingung.

Die 7 Akkumulator-abhängigen Befehle:

SET	
RES	
COM	
LD	Nur für Timer und Counter
LDL	Nur für Timer und Counter
INC	Nur für Timer und Counter
DEC	Nur für Timer und Counter

Beispiel:
Entwerfe eine Zeitbasis, die einen Ausgang jede Sekunde invertiert.
Dieses Beispiel benutzt drei Befehle. Der erste (*STL*) setzt den Akkumulator in den invertierten Zustand des Timers. Die beiden folgenden (*LD* und *COM*) hängen vom Akkumulator ab. Sie laden die Zeitbasis und invertieren den Ausgang nur, wenn der Akkumulator vorher durch den Befehl *STL* auf high gesetzt wurde.

```
COB  0
     0
STL  T 1    ;Wenn der Status das Timers low ist, ist der
            ;Akkustatus high
LD   T 1    ;Verzugszeit mit 10 Zeiteinheiten wird geladen
     10
COM   O 38 ;Invertierter Ausgangsstatus
ECOB
```

9.4.8 Wort-Befehle für ganzzahlige Arithmetik

Diese Befehle werden für die Berechnung von arithmetischen Gleichungen eingesetzt, die ganzzahliges Format, Register und Konstanten benutzen. Jede arithmetische Instruktion hat mehrere Zeilen und verwendet Operanden wie Register oder Konstanten, das Ergebnis jedoch wird immer in einem Register abgelegt.

Addition	**Subtraktion**	**Quadratwurzel**
ADD R 0 R 1 R 3 ;R3=R0+R1	SUB R 0 K 18 R 3 ;R3=R0- 18	SQR R 100 R 101
Multiplikation	**Division**	**Vergleich**
MUL K 5 R 1 R 3 ;R3=5*R1	DIV R 0 R 1 R 3 ;R3=R0/R1 R 4 ;Reste	CMP R 0 R 1
Inkrementieren	**Dekrementieren**	**Register initialisieren**
INC R 0 ;R0= R0+1	DEC R 0 ;R0= R0-1	LD R 0 K 19 ; R 0 = 19

Status Flags
Alle arithmetischen Befehle verändern Status Flags, abhängig vom Ergebnis der Operation (**P**ositive, **N**egativ, **Z**ero, **Error**), mit Ausnahme des Befehls zum Laden eines Registers mit einer Konstante (LD).

Unterschiede zwischen Registern und Timern/Countern
Im Gegensatz zu Timern, sind die Befehle zum Laden einer Konstante in ein Register, inkrementieren oder dekrementieren eines Registers nicht abhängig vom Akkumulator Status.
Der Wert im Register, der inkrementiert oder dekrementiert werden soll, kann ganzzahlig positiv oder negativ sein.

Beispiel:
Vergleiche den Inhalt zweier Register und schalte drei Ausgänge, nach folgenden Bedingungen:

Register	**O 32**	**O 33**	**O 34**
R 0 > R 1	High	Low	Low
R 0 = R 1	Low	High	Low
R 0 < R 1	Low	Low	High

Der Vergleichen-Befehl führt eine Subtraktion R 0 – R 1 aus und setzt das Status Flag entsprechend dem Ergebnis:

Registers	**P**	**N**	**Z**	**E**
R 0 > R 1	1	0	0	0
R 0 = R 1	1	0	1	0
R 0 < R 1	0	1	0	0

```
CMP  R 0   ; Durchführung von Subtraktion R 0 – R 1,
           ;  Status Flags sind
     R 1   ; geändert entsprechend dem Ergebnis der Subtraktion
ACC  P
OUT  O 32  ; R 0 > R 1
ACC  Z
OUT  0 33  ; R 0 = R 1
ACC  N
OUT  O 34  ;R 0 < R 1
```

9.4.9 Wort-Befehle für Fliesskomma Arithmetik

Diese Befehle werden für die Berechnung von arithmetischen Gleichungen eingesetzt, die Fliesskomma Format Registers und Konstante verwenden. Jede arithmetische Instruktion beginnt mit dem Buchstaben F zur Kennzeichnung eines Fliesskomma Befehls. Die Operanden dieses Befehls sind immer Register, nie Konstanten. Werden Konstante benötigt, müssen diese in ein Register geladen werden, dann kann das Register in dem Fliesskomma Befehl benutzt werden.

Addition	**Subtraktion**	**Quadratwurzel**
FADD R 0 R 1 R 3 ;R3=R0+R1	FSUB R 0 R 1 R 3 ;R3=R0-R1	FSQR R 100 R 101 ;result
Multiplikation	**Division**	**Vergleich**
FMUL R 0 R 1 R 3 ;R3=R0*R1	FDIV R 0 R 1 R 3 ;R3=R0/R1	FCMP R 0 R 1
Sinus	**Cosinus**	**Arc Tangens**
FSIN R 10 R 11 ;result	FCOS R 10 R 11 ;result	FATAN R 10 R 11 ;result
Exponent	**Natürlicher Logarithm.**	**Absolutwert**
FEXP R 20 R 21 ;result	FLN R 20 R 21 ;result	FABS R 30 R 31 ;result

Status Flags
Alle Befehle oben verändern das Status Flag, mit Ausnahme des *LD* Befehls zum Laden einer Fliesskomm-Format Konstante.

Register initialisieren
LD R 0 3.1415E0 ; R 0 = PI

9.4.10 Umwandlung von Ganzzahl- und Fliesskomma Registern

Die PCD hat unterschiedliche Befehle für arithmetische Operationen mit Ganz- oder Fliesskomma Zahlen. Wenn eine Anwendung zwei Register multiplizieren soll, in dem das eine Ganze Zahlen und das andere Fliesskomma Zahlen enthält, muss zuerst ein Register gewandelt werden entweder in Ganzzahl oder Fliesskomma, bevor die arithmetische Operation durchgeführt werden kann.

Wandeln integer-fltg point	**Wandeln fltg point-integer**
IFP R 0 ; integer -> float 0 ; exponent	FPI R 0 ;float ->integer 0 ; exponent

9.4.11 Index Register

Jeder COB hat ein ganz spezielles Register: das Index Register. Der Inhalt des Index Registers kann mit folgenden Befehlen überprüft werden:

SEI K 10	**SE**t **I**ndex register	Lädt das Index Register mit der Konstante 10
INI K 99	**IN**crement **I**ndex register	Inkrementiert das Index Register und setzt Akkumulator Status high, so lang das Index Register <= K 99 ist
DEI K 5	**DE**crement **I**ndex register	Dekrementiert das Index Register und setzt Akkumulator Status high, so lang das Index Register >= K 5
STI R 0	**ST**ore **I**ndex register	Kopiert Index Register nach Register 0
RSI R 0	**R**e**S**tore **I**ndex register	Kopiert Register 0 nach Index Register

Viele PCD Befehle unterstützen das Anwenden des Index Registers. Dieses Register lässt durch Befehle im Programm die indirekte Adressierung von Registern, Flags, Eingängen, Ausgängen, Timern etc. zu. Diese Befehle sind die gleichen, wie die sonst üblichen, ergänzt um den Buchstaben X.

Beispiel:

Register sind nicht-flüchtige Speicher. Das heisst sie behalten ihre Information auch bei Spannungsausfall oder bei einem Kaltstart. Wenn wir nun einen Bereich von 100 flüchtigen Registern wollen, müssen diese 100 Register mit dem Wert Null während eines Kaltstarts initialisiert werden. Für diese Initialisierung folgende Befehle benutzen:

```
LD   R 10
     K 0
```

Für 100 Register (R 10 bis 109) wäre diese Instruktion 100-mal zu schreiben, jeweils mit geänderter Register Adresse. Das wäre ziemlich langweilig.
Eine andere Lösung ist, das Index Register mit einem Index Null zu initialisieren und eine Programm-Schleife zum Laden des ersten Registers mit Null einfügen, mit anschliessender Erhöhung des Indexes um 1. Das heisst, bei jeder Schleife wird Null in ein anderes Register (R 10, R 11…R 109) geladen. Mit der 100. Schleife erreicht der Indexzähler den maximalen Indexwert (K 99) und zieht den Akkumulator Status auf low. Die Schleife wird verlassen und das verbleibende Programm ausgeführt.

```
          XOB   16    ; Kaltstart-Block
          SEI   K 0   ;Index = 0
LOOP:     LDX   R 10  ;Laden der Registeradresse = 10 +  Index
          0           ;mit Null
          INI   K 99  ;Index wird inkrementiert und
                      ;  Akkustatus wird geändert
    JR    H LOOP      ;Wenn Akku high ist, springt das
                      ;  Programm zu Label LOOP
    EXOB
    COB   0           ;Zyklischer Organisationsblock
          0
    …
    ECOB
```

9.4.12 Programmsprünge

Der IL Befehlssatz kennt drei Programmsprung-Befehle. Mit diesen kann eine Befehlsfolge abhängig von einer binären Bedingung abgearbeitet werden, oder es können Programm-Schleifen für wiederkehrende Aufgaben eingefügt werden (Indexierung).

Sprungbefehle		
JR	Jump relative	Sprünge einige Zeilen vor oder zurück, ab der Zeile, die den Sprungbefehl JR enthält.
JPD	Jump direct	Springt zu einer Zeilennummer, gezählt ab dem Start des Blocks (COB,PB…).
JPI	Jump indirect	Wie JPD, aber die Zeilennummer ist in einem Register enthalten.

Das Sprungziel ist in der Regel als Programmzeile in einem Label enthalten. Bei einem relativen Sprung muss die Anzahl Zeilen angegeben werden, die vorwärts oder rückwärts gesprungen werden soll.

Sprung mit Zeilen-Label:

```
    JR    L  Next
    INC   R 10
Next:NOP
```

Sprung mit Anzahl Zeilen:

```
JR    L  +1
INC   R 10
NOP
```

Ein Sprung muss sich immer innerhalb eines Blocks ereignen(COB, PB,…) nie ausserhalb.

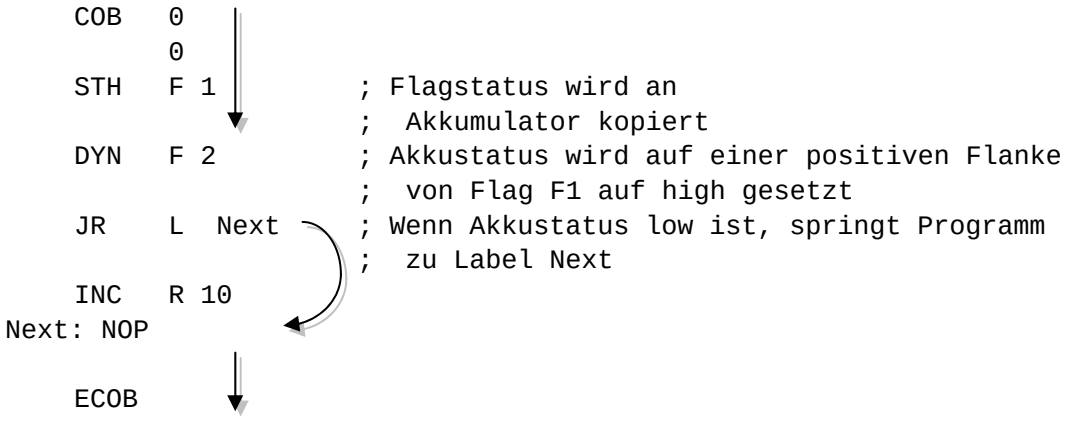
Wenn notwendig, kann ein Sprung immer ausgeführt werden, oder nur unter vorbestimmten binären Bedingungen, wie z.B. der Akkumulator Status oder ein Status Flag.

Syntax für unbedingten Sprungbefehl		
Mnemo-Kode	**Label**	**Beschreibung**
JR		Sprung immer auf Zeile in Übereinstimmung zum Label
JPD		
JPI		

Syntax für einen bedingten Sprungbefehl			
Mnemo Kodec	**Bedingung**	**Label**	**Beschreibung**
JR	H		Wenn Akkumulator Status high ist
JPD	L		Wenn Akkumulator Status low ist
JPI	Z		Wenn Status Flag Z high ist
	P		Wenn Status Flag P high ist
	N		Wenn Status Flag N high ist
	E		Wenn Status Flag E high ist

Beispiel: Impulse eines binären Eingangs binär zählen mit einem Register (relativer Sprung)

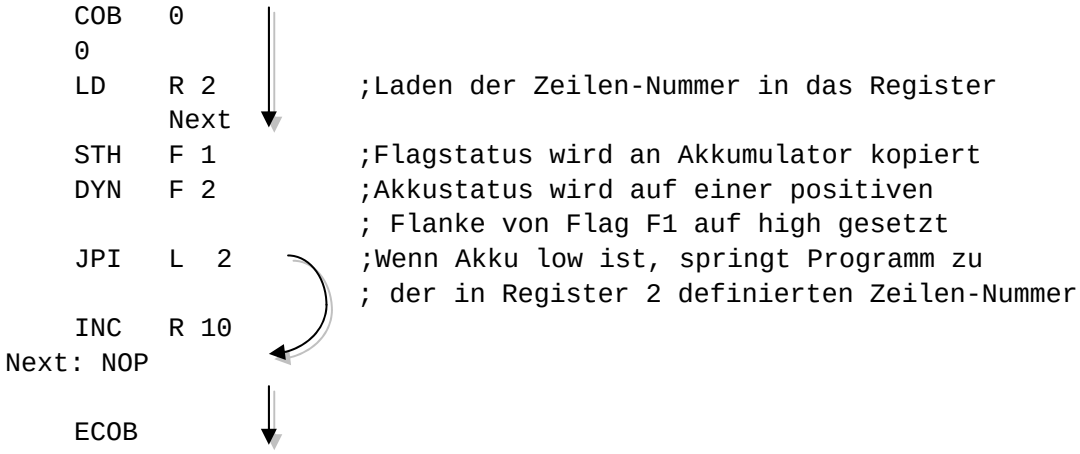
Im Gegensatz zu Countern, hängt der Befehl zum inkrementieren eines Registers nicht vom Akkumulator Status ab. Daher ist es praktisch einen Sprungbefehl zum inkrementieren eines Register einzusetzen.



```
    COB   0
          0
    STH   F 1          ; Flagstatus wird an
                       ;  Akkumulator kopiert
    DYN   F 2          ; Akkustatus wird auf einer positiven Flanke
                       ;  von Flag F1 auf high gesetzt
    JR    L  Next      ; Wenn Akkustatus low ist, springt Programm
                       ;  zu Label Next
    INC   R 10
Next: NOP

    ECOB
```

Die Befehle *STH* und *DYN* lesen Informationen vom Flag F 1 und setzen den Akkumulator Status high bei einer aufsteigenden Flanke oder low bei einer abfallenden. Abhängig vom Akkumulator Status, veranlasst der Befehl *JR* entweder einen Sprung zur Zeile übereinstimmend zum Label *Next:* oder inkrementiert das Register mit der Instruktion *INC*. Der Buchstabe *L* bezeichnet die Bedingung, unter der gesprungen werden soll (in diesem Beispiel erfolgt der Sprung nur, wenn der Akkumulator Status low ist).

Beispiel: Lösung mit indirektem Sprung

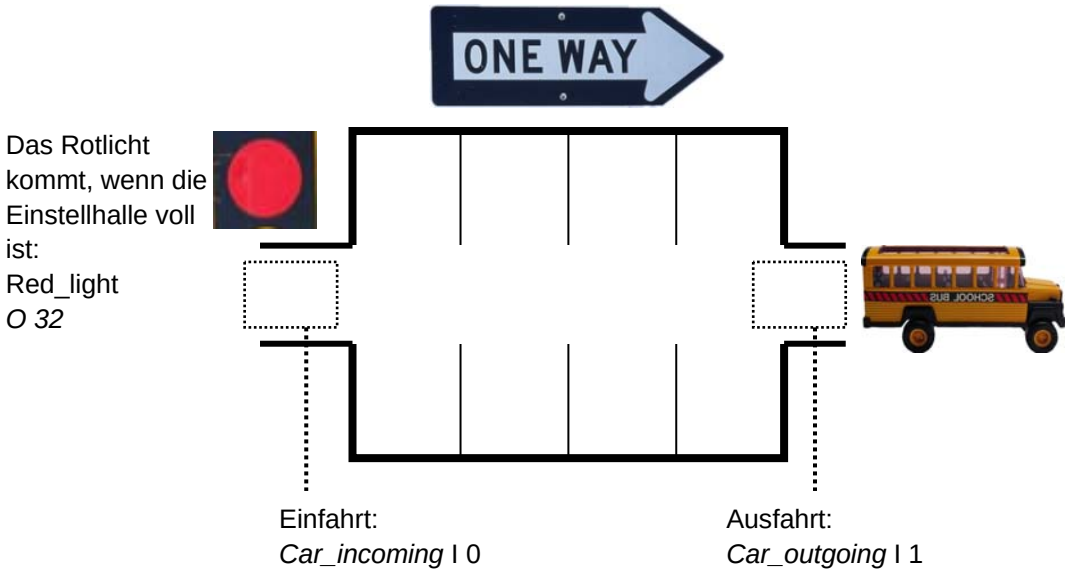


```
    COB   0
    0
    LD    R 2          ;Laden der Zeilen-Nummer in das Register
          Next
    STH   F 1          ;Flagstatus wird an Akkumulator kopiert
    DYN   F 2          ;Akkustatus wird auf einer positiven
                       ; Flanke von Flag F1 auf high gesetzt
    JPI   L  2         ;Wenn Akku low ist, springt Programm zu
                       ; der in Register 2 definierten Zeilen-Nummer
    INC   R 10
Next: NOP

    ECOB
```

Der indirekte Sprung ist sehr flexibel. Das Programm passt die Zeilen-Nummer, zu der gesprungen werden soll, selbständig an.

9.5 Editieren eines ersten Anwender Programms

Zählen der verbleibenden Parkplätze in einer Einstellhalle mit 8-Parkplätzen und einschalten eines Rotlichts, wenn die Einstellhalle voll ist.

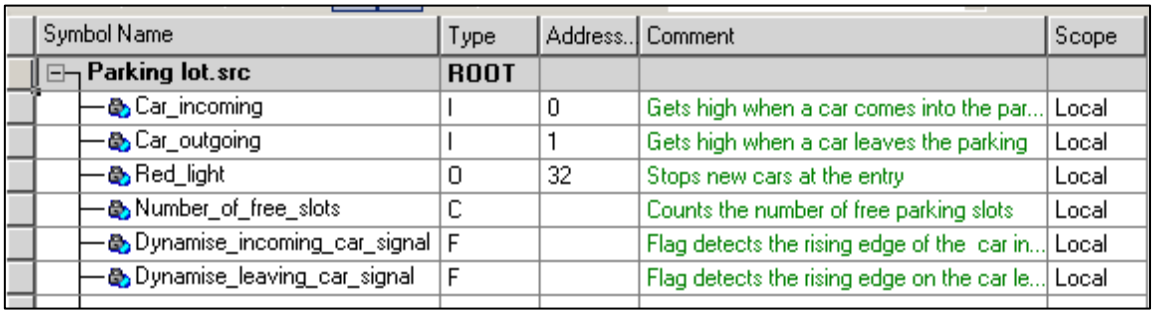


Beim Einschalten der Spannungsversorgung wird angenommen, dass alle Parkplätze frei sind. Deswegen wird beim initialisieren der Zähler für die freien Parkplätze mit dem Wert 8 geladen. Dies wird nur einmal beim Aufstarten der PCD durchgeführt und daher im Kaltstart-Block XOB 16 programmiert. Die verbleibenden Programm-Funktionen werden in einem Zyklischen Organisationsblock (COB) ausgeführt.

Der Sensor bei der Einfahrt *Car_incoming* sendet jeweils einen Zählimpuls, wenn ein neues Auto einfährt. Die aufsteigende Flanke dieses Signals dekrementiert den Zähler für die freien Parkplätze.

Der Sensor bei der Ausfahrt *Car_outgoing* sendet jeweils einen Zählimpuls, wenn ein neues Auto ausfährt. Die aufsteigende Flanke dieses Signals inkrementiert den Zähler für die freien Parkplätze.

Ist die Einstellhalle voll, zeigt der Zählerwert Null freie Parkplätze. Der Zähler-Status geht auf low. Das Rotlicht an der Einfahrt zur Einstellhalle leuchtet auf.



Symbol Name	Type	Address..	Comment	Scope
Parking lot.src	**ROOT**			
Car_incoming	I	0	Gets high when a car comes into the par...	Local
Car_outgoing	I	1	Gets high when a car leaves the parking	Local
Red_light	O	32	Stops new cars at the entry	Local
Number_of_free_slots	C		Counts the number of free parking slots	Local
Dynamise_incoming_car_signal	F		Flag detects the rising edge of the car in...	Local
Dynamise_leaving_car_signal	F		Flag detects the rising edge on the car le...	Local

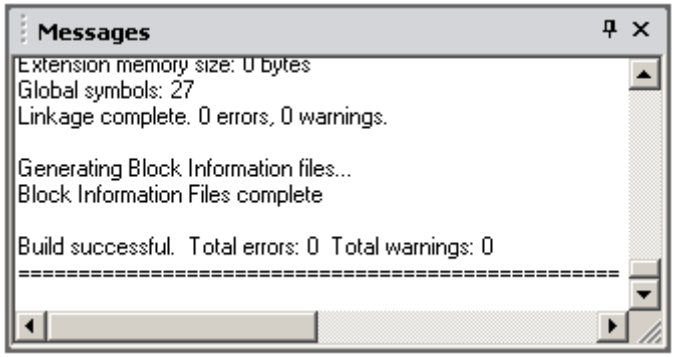
```
; Kaltstart Organisationsblock
;-----------------------------

    XOB     16                  ; Kaltstart-Block
    ACC     H
    LD      Number_of_free_slots      ; Initialisierung des freie
                                      ;   Steckplätze Zählers
            8                   ; mit dem Wert 8 (vorbehaltlos)
    EXOB                        ; Ende der start-up Programm

; Zyklischer Organisationsblock
;-----------------------------

    COB   0                     ; Zyklischer Organisationsblock
          0                     ; Ohne Überwachungzeit

    STH   Car_incoming          ; Das Auto kommt auf den Parkplatz
    DYN   Dynamise_incoming_car_signal      ; Auf die steigende Flanke
    DEC   Number_of_free_slots  ; Dekrementieren der Anzahl von
                                ;   freien Stellplätzen

;---------------------------------------

    STH   Car_outgoing          ; das Auto verlässt den Parkplatz::
    DYN   Dynamise_leaving_car_signal  ; Auf die steigende Flanke,
    INC   Number_of_free_slots  ; Inkrementieren der Anzahl der
                                ;   freien Stellplätzen

;---------------------------------------

    STL   Number_of_free_slots  ; keine freien stellplätzen
                                ; (Zählerstand = Low)
    OUT   Red_light             ; Das rote Licht einschalten

    ECOB                        ; Ende des zyklischen Programms
```

9.5.1 Verarbeiten (build) des Programms

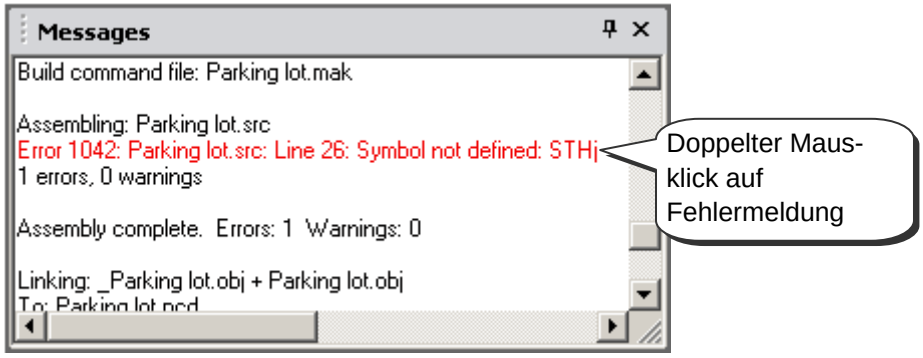
Rebuild All Files

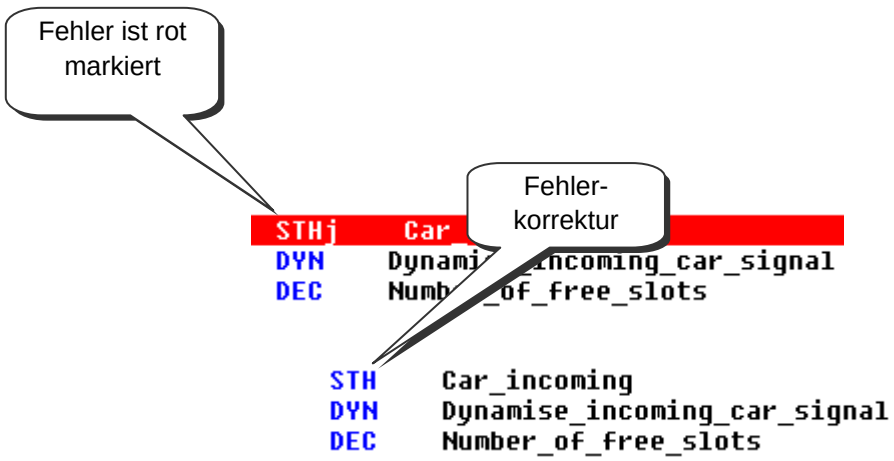
Das Anwender-Programm ist komplett editiert, aber für die PCD noch nicht brauchbar. Es muss in eine Binärdatei übersetzt werden. Dies führt das Programmier-Werkzeug aus, wenn der Anwender das *Device*, *Rebuild All Files* Menü aktiviert oder den *Rebuild All Files* Knopf im Projekt Manager oder IL Editor betätigt.

Das Messages Fenster zeigt uns den Fortschritt der Verarbeitung. Es zeigt die *Assembly* und *Linkage* Phasen. Ist das Programm in Ordnung, endet die Verarbeitung mit der Nachricht *Build sucessful. Total errors 0 Total warnings: 0*



Eventuelle Fehler werden in roter Schrift angezeigt. Ein doppelter Mausklick auf die Fehlermeldung lokalisiert den Fehler im Anwender-Programm.





9.6 Laden des Programms in die PCD

Download Program

Das Anwender-Programm ist fertig und muss nun vom PC in die PCD übertragen werden. Dies geht entweder mit dem Menü *Online, Download Program,* oder mit dem *Download Program* Knopf im Projekt Manager Fenster.

Wenn Kommunikationsprobleme auftreten, die Konfiguration nochmals in *Settings Online* sowie die Verbindungskabel zwischen PC und der PCD (PCD8.K111, USB) überprüfen.

9.7 Debuggen in einem Programm

Die erste Version von Programmen ist nicht immer perfekt. Ein sorgfältiger Test kann hilfreich sein. Programmtests werden mit demselben Editor durchgeführt, der auch für die Bearbeitung verwendet wird.

Die weissen Zeilen enthalten den ursprünglichen Quellcode mit Symbolen und Kommentaren.

Die grauen Zeilen enthalten den vom Build erstellten Code, einschliesslich Adressen der Operanden und Zeilennummern im Programm.

```
;----------------------------------------
;  Cyclical Organisation Block
;----------------------------------------

            COB     0                           ; Cyclical program
                    0                           ; No supervision time
    000007  COB     0
    000008          0
    000010  NOP

            STH     Car_incoming                ; A car comes into the park
    000011  STH     I|O 0               [0]
            DYN     Dynamise_incoming_car_signal ; On the positiv flank of i
    000012  DYN     F 7502              [0]
            DEC     Number_of_free_slots        ;   Decrement the number of
    000013  DEC     C 1400              [8]
```

9.7.1 On-/Off-line, Run und Stop

Im On-line Modus kann der korrekte Betrieb der PCD überprüft werden (Run, Stop, Step-by-step). Jegliche Information, die zur Prüfung des Programms benötigt wird, kann angezeigt werden.

Go On /Offline Knopf drücken

Run Knopf setzt die Steuerung in den Run Modus

An der Front der PCD ist die *Run*-LED zu beobachten. Diese sollte bei Start einschalten. Die PCD führt das Anwender-Programm aus.

Wird der *Stop* Knopf gedrückt, geht die *Run*-LED aus. Die PCD stoppt das Anwender-Programm.

Bei einem *Stop* zeigt die Zeile mit roter Schrift den Befehl, wo das Programm angehalten hat. Die Zahl in eckiger Klammer zeigt den ganzzahligen Wert von Counter 1400. Weiter rechts sind die Zustände von Akkumulator, Status Flags und Index Register zu sehen.

```
          STH    Car_incoming                    ; A car comes into the par
000011    STH    I|O 0                  [0]
          DYN    Dynamise_incoming_car_signal ; On the positiv flank of
000012    DYN    F 7502                 [0]
          DEC    Number_of_free_slots            ;   Decrement the number o
000013    DEC    C 1400                 [8]        A0 Z0 N0 P1 E0 IX0000
```

9.7.2 Step-by-step Modus

Run to Cursor

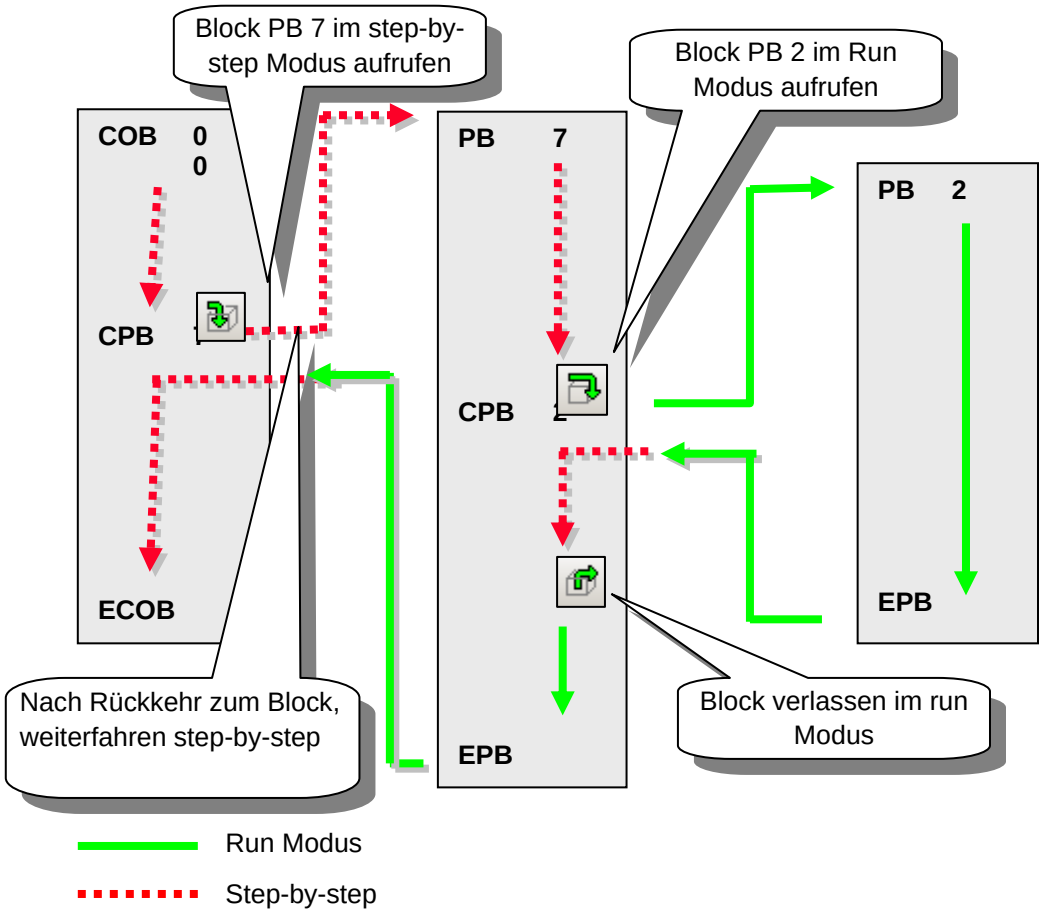
Ist die PCD im *Run*-Modus, die erste Zeile, die im step-by-step Modus näher betrachtet werden soll markieren und den *Run to Cursor* Knopf betätigen.
Die PCD stoppt, wenn die Zeile mit dem Cursor erreicht ist. Mit der Taste F11 auf der Tastatur wird das step-by-step Programm ausgeführt, oder man benützt die Knöpfe unten.

Enthält das Programm PBs, FBs oder SBs, muss nicht unbedingt im step-by-step Modus durchgegangen werden. Drei Optionen sind verfügbar:

In den Block eintreten und durchgehen

Den aufgerufenen Block in den Run Modus setzen, nach der Rückkehr zum Block der den Aufruf veranlasste, weiterfahren im step-by-step Modus.

Ist das Programm in einem Block, der nicht interessiert, kann mit diesem Knopf der Block verlassen werden. Nach der Rückkehr zum Block der den Aufruf veranlasste, weiterfahren im step-by-step Modus.




```
          STH    Car_outgoing                    ; A car leaves into the p
000014    STH    I|O 1                 [0]         A0 Z0 N0 P1 E0 IX0000
          DYN    Dynamise_leaving_car_signal     ; On the positiv flank of
000015    DYN    F 7503                [0]
```

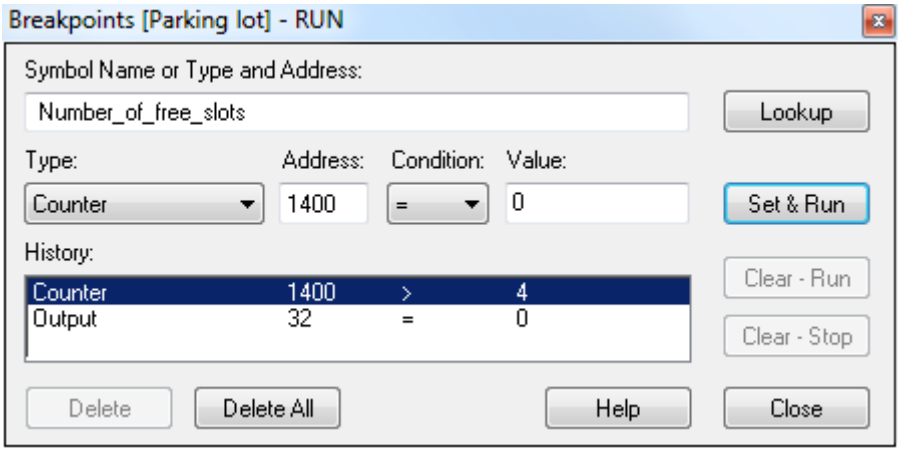
Bei jedem Programmschritt die Zeile mit der roten Schrift beachten. Die Zahl in eckiger Klammer zeigt den logischen Status von Eingang I 1. Weiter rechts sind die Zustände von Akkumulator, Status Flags und Index Register zu sehen.

9.7.3 Anhalte-Punkte (Breakpoints)

Set/Clear Breakpoints

Anhalte-Punkte ermöglichen ein Anhalten des Programms bei bestimmten Ereignissen, die sich auf eine Programmzeile oder ein Symbol beziehen können:
Zustand eines Ein-/Ausgangs, Flag, Status Flag
Gegenwärtiger Wert in einem Register oder Counter

Anhalte-Punkt auf einem Symbol
Die Breakpoint Bedingung wird mit dem *Online Breakpoints* Menü oder mit dem *Set/Clear Breakpoint* Knopf definiert.



Im Fenster oben den Symboltyp und Adresse festlegen, dann die Bedingung und den Status/Wert des Anhalte-Punkts setzen.

Beim Drücken des *Set&Run* Knopfes geht die PCD in den *conditional run* Modus. Die PCD *Run* LED blinkt und das PCD *Run* Icon wechselt zwischen grün und rot.

Die PCD geht automatisch in den Stop Modus, wenn die Breakpoint Bedingung erfüllt ist. Beispiel: Ein Befehl ändert den Wert von Counter 1400 in einen Wert grösser als 4. Die Zeile nach dem letzten ausgeführten Befehl wird rot markiert. Es ist dann möglich im step-by-step Modus oder mit einer anderen Breakpoint Bedingung im Programm weiterzufahren.

Falls nötig, kann der Conditional Run Modus folgendermassen unterbrochen werden:
Der *Clear-Run* Knopf bringt die PCD in den Run Modus. Die *Run*-LED kommt und das *Run* Icon wird grün.
Der *Clear-Stop* Knopf bringt die PCD in den Stop Modus. Die *Run* LED erlischt und das *Run* Icon wird rot.

Sind mehrere bedingte Breakpoints gesetzt, sind diese alle im *History*-Feld gespeichert. Sie werden mit der Maus ausgewählt und mit dem *Set&Run* Knopf aktiviert.

Run to Cursor

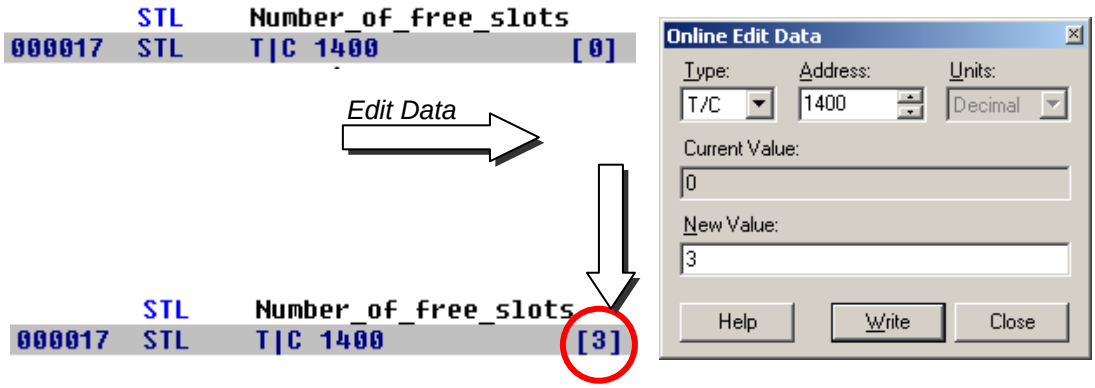
Anhalte-Punkt auf einer Programmzeile
Wählen Sie eine Programmzeile aus und führen Sie das Menü oder Taste *Online, Run to, Cursor* aus, um das Programm an der gewählten Zeile anzuhalten. Setzen Sie Ihre Arbeit im Step-by-Step-Modus fort.

9.7.4 On-line Änderung am Programm

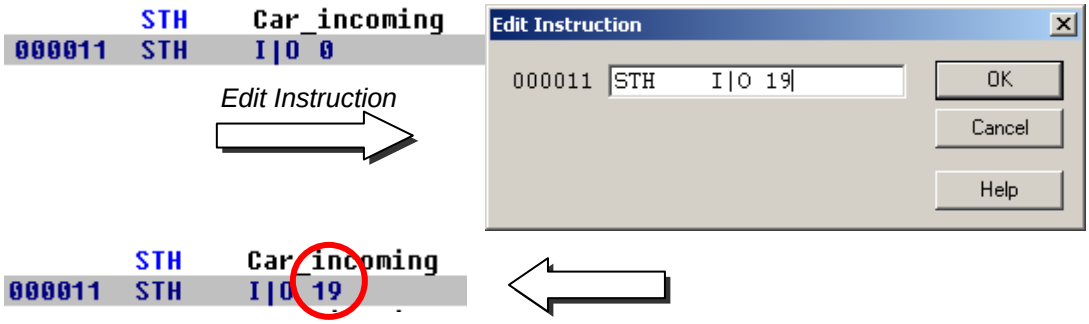
Beim Testen eines Programm step-by-step, ist es oft hilfreich Zustände/Werte bestimmter Operanden/Symbole zu verändern, um damit das Programmverhalten unter verschiedenen Bedingungen zu prüfen.

Eine der aktiven Zeilen (grau) mit der Maus auswählen und mit rechtem Mausklick das Kontext-Menü anzeigen.

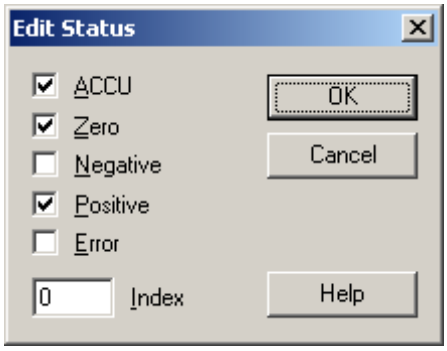
Im *Edit Data* Kontext-Menü kann der Zustand/Wert des ausgewählten Operanden geändert werden.



Im *Edit Instruction* Kontext-Menü kann der Mnemo-Kode und die Adresse eines ausgewählten Operanden geändert werden.

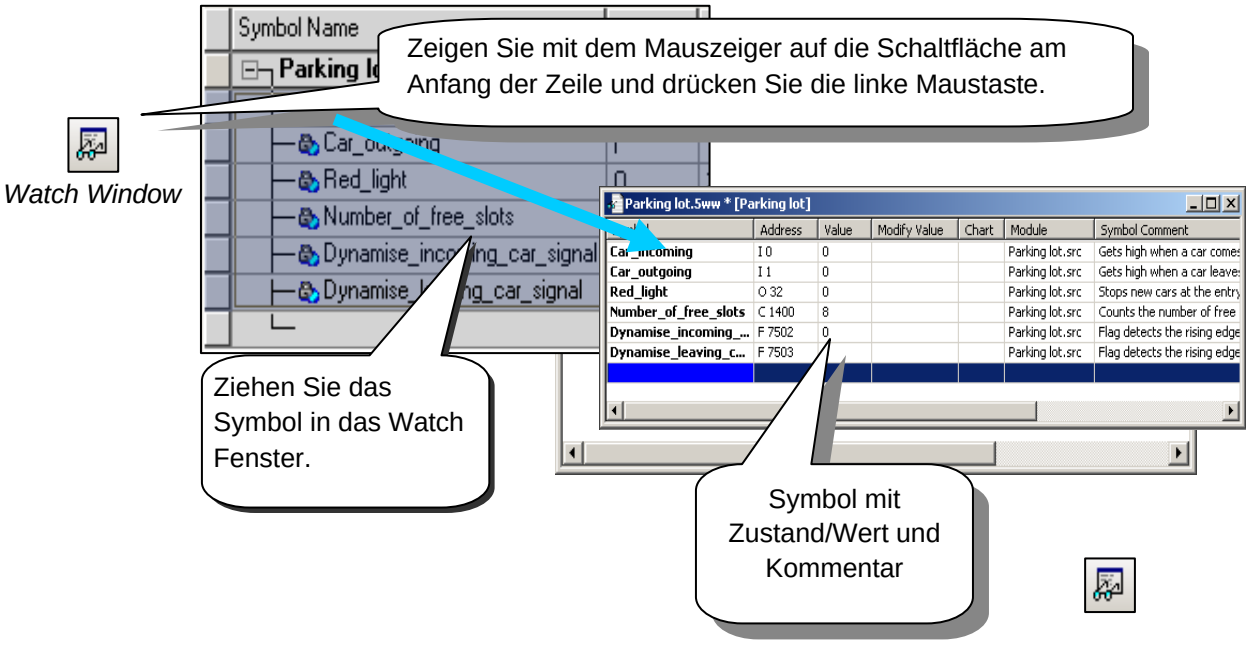


Status Flags können im *Edit Status* Kontext Menü geändert werden.

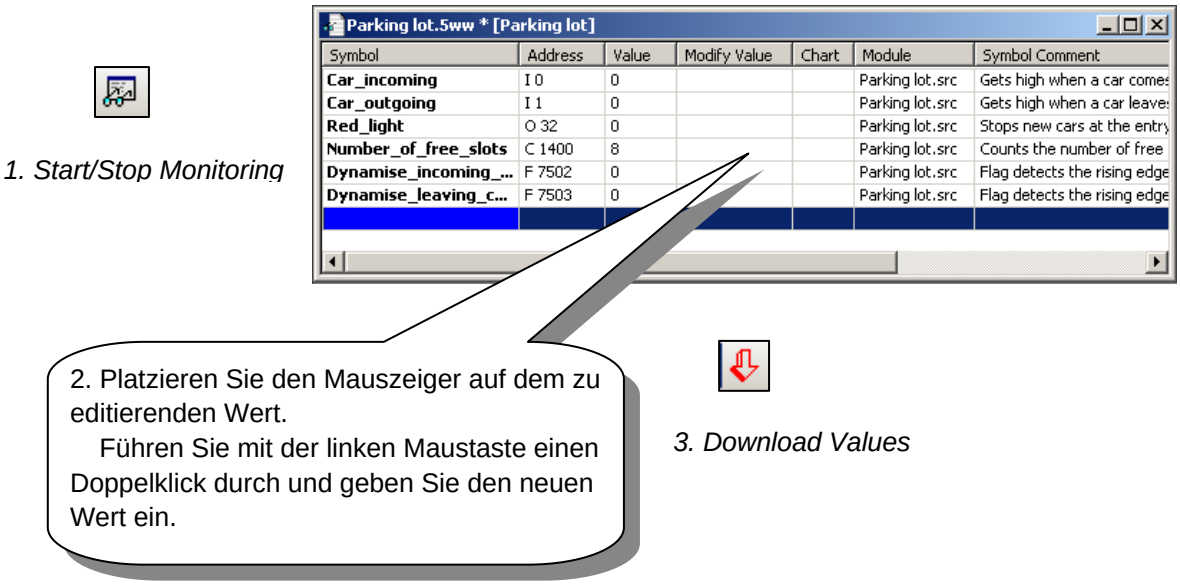


9.7.5 Betrachten und Ändern von Symbol-Zuständen mit dem *Watch Window*

Mit dem *Watch Window* wird ein weiteres hilfreiches Werkzeug zum testen und betrachten von Symbol-Zuständen zur Verfügung gestellt. Den *Watch Window* Knopf im Projekt Manager drücken und Symbole vom Symbol Editor ins *Watch Window* ziehen.



Zum ändern des Zustands/Wertes eines Symbols ist folgendermassen zu verfahren:



9.8 Inbetriebnahme eines Analogmoduls

Alle bisher vorgestellten Programme haben digitale Ein- oder Ausgänge benutzt und ihre Adressen oder Symbole vor die Mnemonik gesetzt.
Beispiel: ANH I 45

Analoge E/As benötigen ein kleines Programm zum Lesen der Werte aus jedem der analogen Module, die das Multiplexen und die A/D- sowie die D/A-Umwandlung übernehmen. Diese können in IL oder mit den *Media mapping* Funktionen des *Device Configurators* programmiert werden, die in der Dokumentation des *Device Configurator*s beschrieben werden.

9.8.1 Beispiel für PCD2.W340 Analog-Eingangsmodule

Wenn der PCD mit einem PCD2.W340-Modul ausgerüstet ist, das über 8 universelle Eingangskanäle verfügt, dann kann folgende Routine verwendet werden:

```
BA   EQU   O 96   ; Modul-Basisadresse im PCD
     ACC   H      ; ACCU muss hoch sein
     LD    R 100  ; Legt den Messkanal fest ( 0…7)
           2

     MUL   R 100
           K 32   ; Berechnet
           R 100  ; Steuerbyte
     ADD   R 100  ; einschließlich
           K 264  ; Freigabebit.
           R 100

     SET   BA+15  ; Ansteuerung A/D-Umkehr

     BITO  9      ; sendet Steuerbyte
           R 100  ; einschließlich Freigabebit.
           BA+0   ; bis W3xx

     BITIR 12     ; Liest die 12 Messbits (0…4095) in R 77
           BA+0
           R 77
     RES   BA+15  ; Stop A/D-Umkehr
```

Das PCD2.W340 ist ein Universalmodul. Es unterstützt die Messung der Bereiche 0..10V, 0..2.5V, 0..20 mA und Pt/Ni 1000 Temperatursonden. Zum Festlegen der Messreihe muss eine Brücke auf dem Modul gewählt werden. Die Auflösung beträgt 12 Bit, was mit 4095 gemessenen Werten gleichzusetzen ist.

Die o. a. Routine dringt in den in Register 100 festgelegten Kanal ein und liefert eine Rohmessung an Register 77.

Bei diesem Modul mit einer Auflösung von 12 Bits, die einem gemessenen Wert zwischen 0 und 4095 entspricht, muss der Benutzer die Messungen in eine physikalische Standardeinheit umrechnen.

9.8.2 Beispiel für PCD2.W610 Analog-Ausgangsmodule

Die Ausgangsmodule arbeiten ähnlich wie die Eingangsmodule.

Wenn der PCD mit einem PCD2.W610-Modul ausgerüstet ist, das über 4 universelle analoge Ausgangskanäle verfügt, dann kann folgende Routine verwendet werden:

```
BA   EQU    O 96   ; Modul-Basisadresse im PCD
     ACC    H      ; ACCU muss hoch sein
     LD     R 100  ; Legt den Ausgangskanal fest ( 0…6)
            2
     BITOR  2      ; Überträgt Kanal auf W6x0
            R 100
            BA+0
     BITOR  2      ; Schreibt 2 Füllbits
            R 100
            BA+0
     LD     R 277  ; Legt den Digitalwert des Ausgangs fest ( 0…4095)
            3879
     BITO   R 12   ; Überträgt die 12 Bits auf den Ausgangswert zum W6x0
            R 277
            BA+0
            SET BA+12 ; Ansteuerung D/A-Umkehr
```

Zum Festlegen der Ausgangsreihe muss eine Brücke auf dem Modul gewählt werden. 0…20 mA oder 0…10 V. Auflösung beträgt 12 Bit, was mit 4095 gemessenen Sollwerten gleichzusetzen ist.

Der Ganzzahlwert im Register 12 bestimmt die Ausgangsspannung oder den Ausgangsstrom des in Register 100 festgelegten Kanals:

Eingangswert in Regist 12	Ausgangsspa nnung [V]	Ausgangsstror [mA]
0	0	0
2047	5	10
4095	10	20

Weitere Einzelheiten und Beispielversionen von IL-Programmen für Analogmodule finden Sie in Ihrem Hardwarehandbuch oder auf unserer Website unter:
http://www.sbc-support.com

Inhalt

10 Zusätzliche Werkzeuge

10.1 Zusammenfassung

PG5 stellt ihnen mehrere zusätzliche Dienstprogramme für eine Vielfalt von Anwendungen zur Verfügung.

10.2 Datenübertragungsprogramm

10.2.1 Einsatz der Datenübertragung

Mit diesem Programm können Daten von der PCD in eine ASCII Datei (*.dt5) oder von dieser Datei zurück in die PCD gespeichert werden.

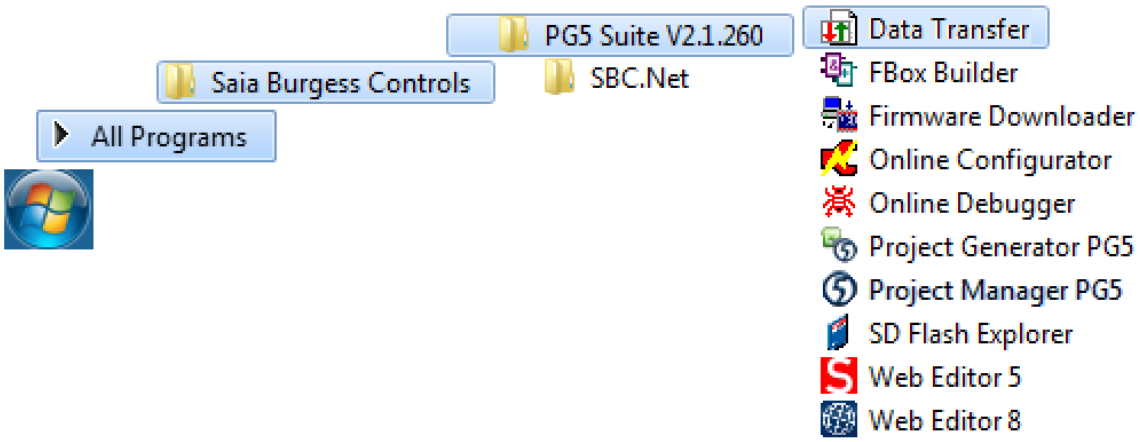
Folgende Daten werden mit diesem Werkzeug übertragen:
Eingänge, Ausgänge, Flags, Timer, Counter, Registers, Daten und Textblöcke.

Achtung! Das PCD Programm und Hardware Konfigurationen können mit dem Datenübertragungsprogramm nicht gespeichert werden. Zum speichern von Programm, Hardware Konfigurationen und Daten ist es ratsam eine Sicherungskopie anzulegen. Siehe Beschreibung des Projekt Managers.

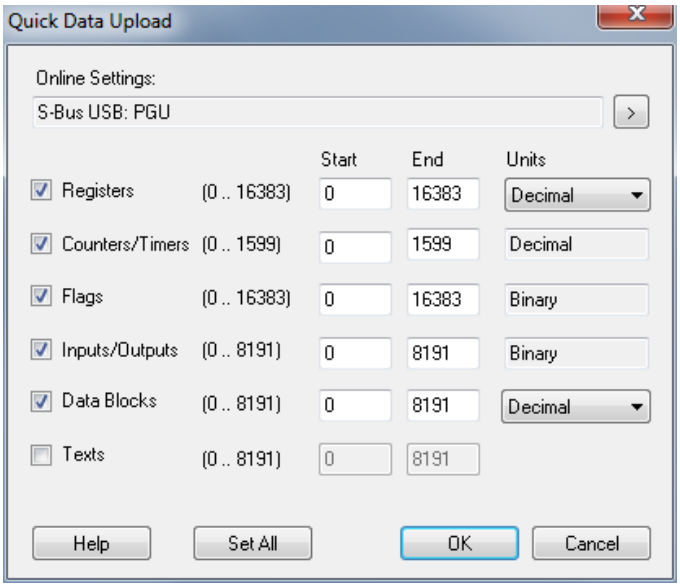
10.2.2 Datenübertragung starten

Aufstarten des Programms mit:

Start → Programs → Saia Burgess Controls → PG5 Suite 2.1 → Data Transfert



10.2.3 Datensicherung mit Quick Data Upload



Quick Data Upload

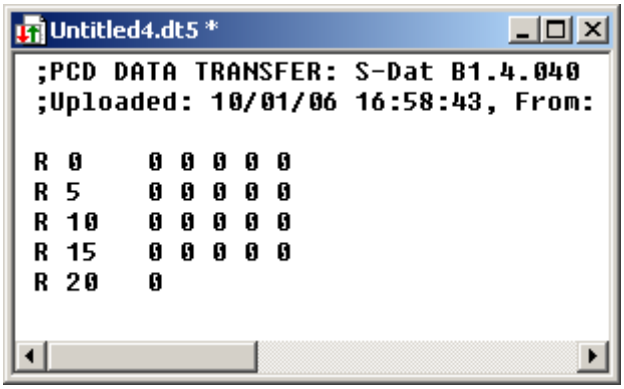
Im Menü wählen *Online, Quick Data Upload …* oder den *Quick Data Upload* Knopf drücken. Es erscheint das Fenster oben.

Datentypen auswählen, die gesichert werden sollen, deren Adressbereich und Zahlenformat eingeben.

Mit OK werden die Daten geladen.

Kommt die nebenstehende Meldung, sind die Kommunikationsparameter mit *Online, Online Settings* zu prüfen und die korrekte Kabelverbindung zwischen PC und PCD sicherzustellen.

Das Laden der Daten dauert einen Moment, dann kommt folgende Anzeige:



Die Datei kann mit neuen Werten überschrieben und dann im Menü *File, Save* oder mit dem *Save* Knopf gesichert werden.

10.2.4 Daten zurückspeichern

Open

Früher gesicherte Dateien können mit *File, Open* oder dem *Open* Knopf wieder angezeigt werden.

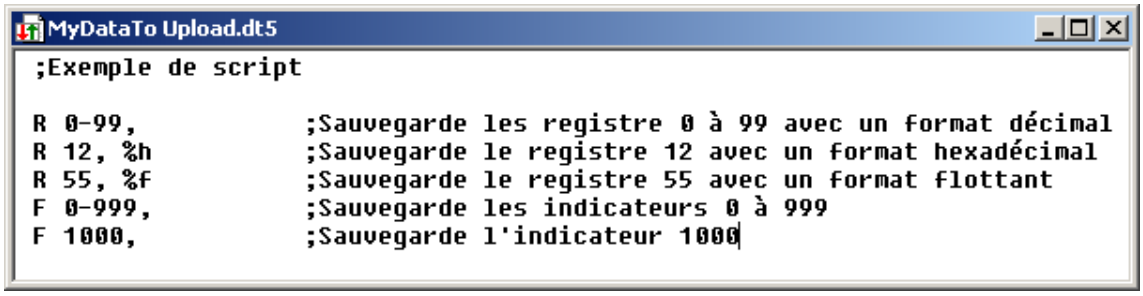
Die Werte in der Datei können erneut geändert werden.

Download To PCD

Zurückspeichern zum PCD Speicher mit *Online, Download Data to the PCD* oder dem *Download* Knopf.

10.2.5 Datensicherung mit Hilfe einer Script-Datei

Falls nötig, kann die Liste der zu sichernden Daten auch in einer Script-Datei geändert werden. Beispiel:



Upload From PCD

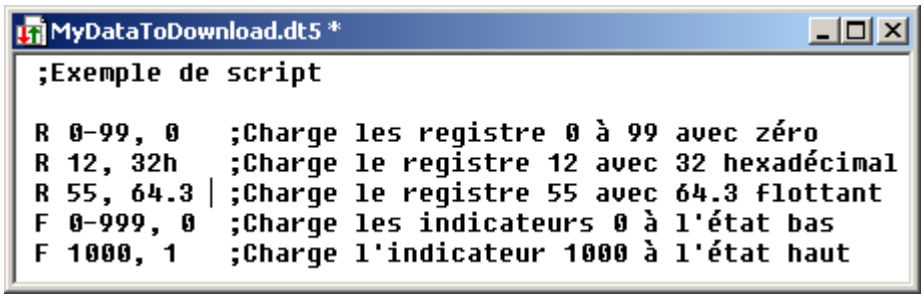
Zum Hochladen der PCD Daten in ein zweites anderes Fenster, *Online, Upload Data from PCD* im Menü auswählen, oder den *Upload* Knopf drücken.

Weitere Informationen über Script Befehle sind in der Programmhilfe zu finden: Menü *Help, Help Topics F1, General.*

10.2.6 Zurückspeichern mit Hilfe einer Script-Datei

Mit einer Script-Datei können auch Daten die bereits gespeichert waren, geändert werden. Beispiel:

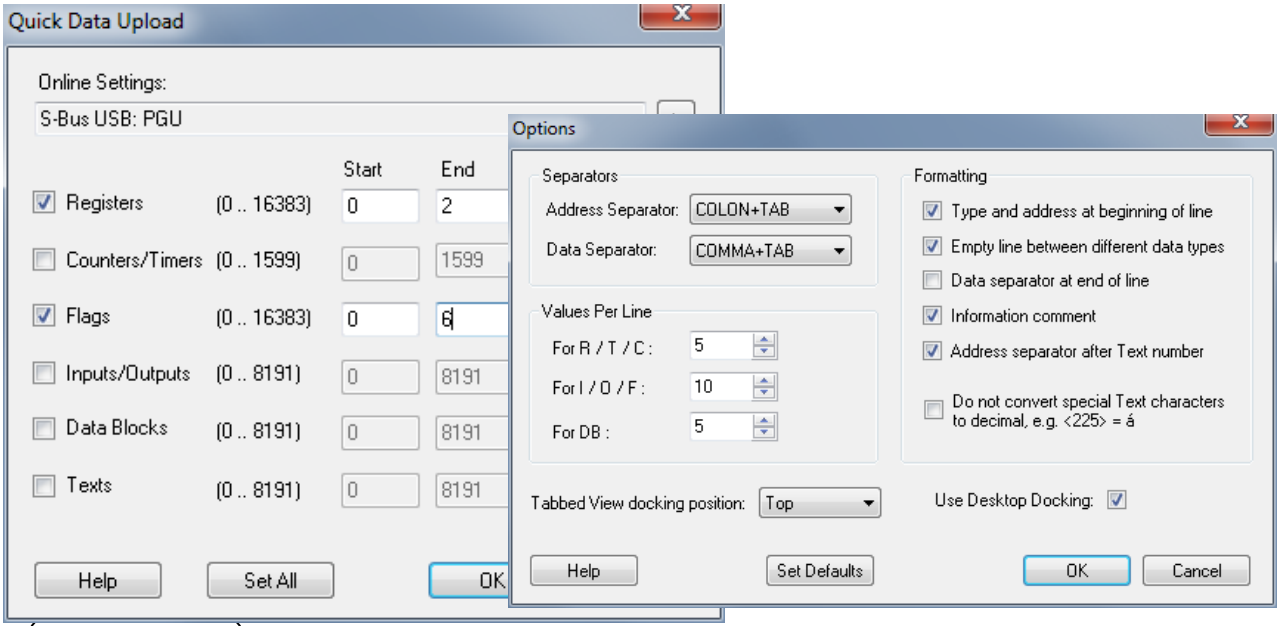
Download To PCD

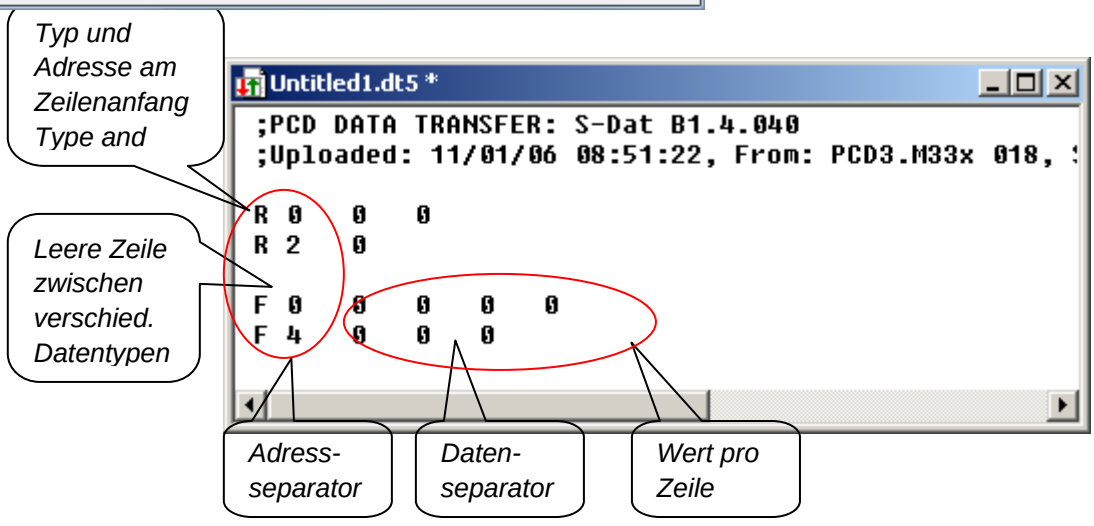


Zurückspeichern der Script-Datei in die PCD mit *Online, Download Data to PCD* im Menü oder den *Download* Knopf drücken.

10.2.7 Hochlade-Optionen

Im Fenster, das mit *Edit, Options* angezeigt wird, kann das Format der zu sichernden Daten in Datei '*.dt5' angepasst werden.
Mit der folgenden Option kann eine Datei in *Microsoft Excel* importiert werden.





10.2.8 Datensicherung mit Befehlszeilen

Die Datenübertragung kann auch mit Hilfe von DOS Befehlszeilen gesteuert werden. Damit können Batch Dateien zur regelmässigen, automatischen Sicherung von PCD Dateien geschrieben werden. Die Daten können dann mit Microsoft Excel bearbeitet werden.

Befehlszeilen Syntax:

SDAT [Name_of_file[.dt5][data…]][/R=nnn][/I0nnn][/A=nnn][/D=nnn]

Name_of_file	Name der Datei zum speichern/rückspeichern
Data …	Definition der zu speichernden Daten. Ist nichts definiert, wird die Datei in die PCD zurückgespeichert *Format : <type><start>[-<end>][units]* type R,C,O,F,DB (C= counters/timers, O = inputs/outputs) Erste Adresse start Letzte Adresse end D,H,F (Dezimal, Hexadezimal, Fliesskomma) für R,C,DB units
/R=nnn	nnn = Wert pro Zeile für R,T,C,DB (1..256, default = 5) nnn = Wert pro Zeile für I,O,F (1..256, default = 10)
/I=nnn	nnn = Adressseparator (TAB,SPACE,COMMA,COLON, default= TAB)
/A=nnn	nnn = Datenseparator (TAB,SPACE,COMMA,COLON, default= TAB)
/D=nnn	

Beispiel:
sdat5 MyDatas.dt5 R0-99 R12H R55F F0-999 F1000 /R005 /I010

10.3 Watch window

Das *Watch Window* wird ein hilfreiches Werkzeug zum testen, ändern und betrachten von Symbol-Zuständen und Anwendungen.

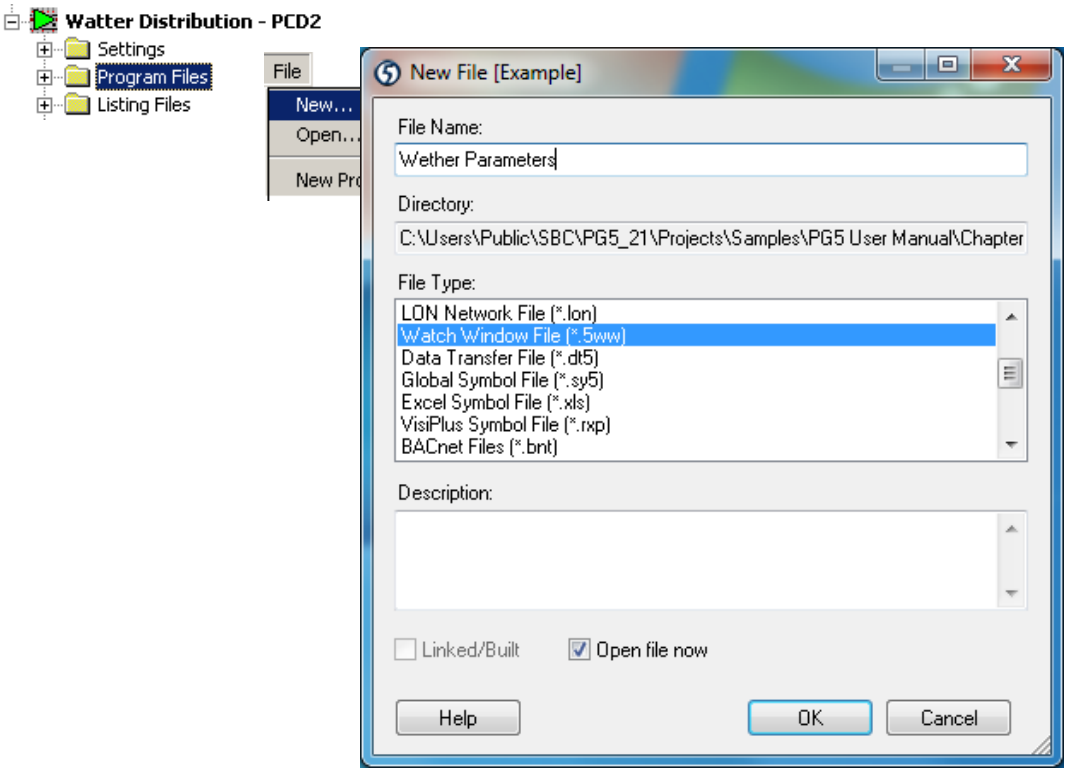
10.3.1 Öffnen des *Watch Windows*

Watch Window

Das *Watch Window* wird durch Auswahl im Menü *View, Watch Window* oder durch drücken des *Watch Window* Knopfes angezeigt.

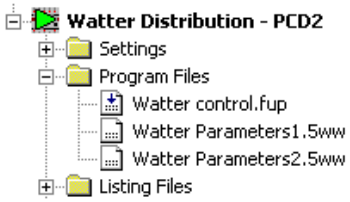
Man kann auch mehrere, verschiedene *Watch Windows* im *Program File* Verzeichnis des Projekt Managers vorbereiten. Dazu im Menü mit *File New* oder mit dem *New File* Knopf ein neues *Watch Window File* (*.5ww) hinzufügen.

New File



Hinweis: Dateien des Typs *.5ww sind nie mit einem Projekt verknüpft (kein Pfeil im Datei Icon). Ihre Informationen beziehen sich nicht auf die Programm-Verarbeitung.

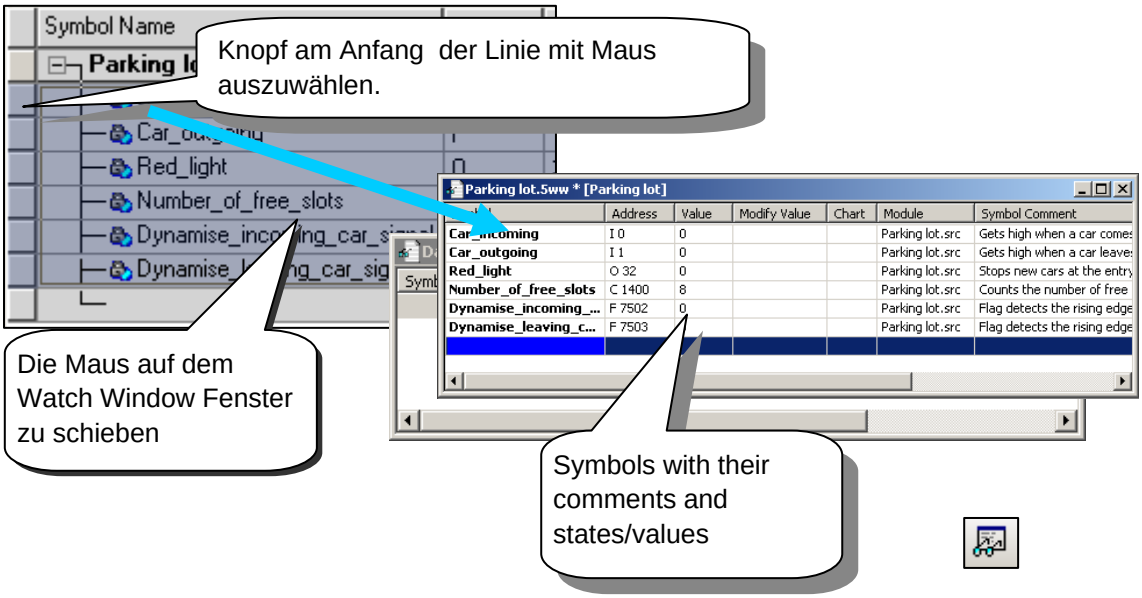
Zum öffnen einer *.5ww Datei, mit doppeltem Mausklick auswählen, oder Datei markieren und im Menü *File Open* wählen.



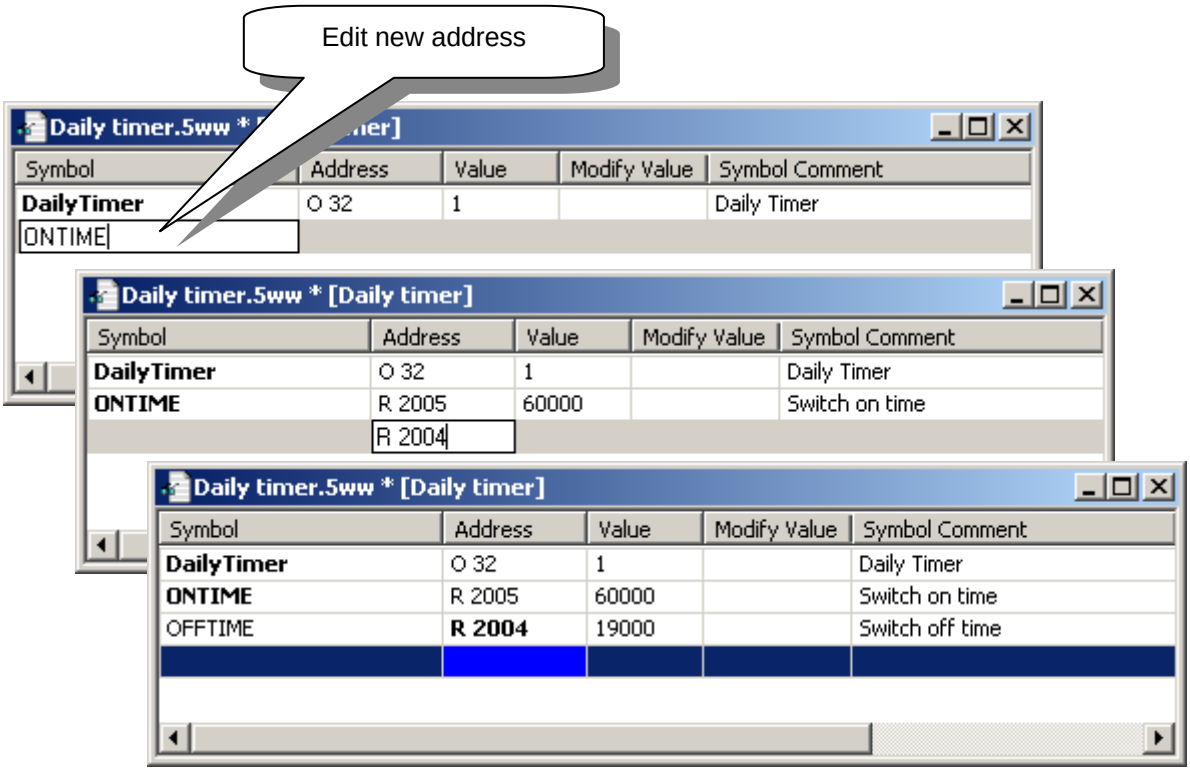
10.3.2 Daten zum *Watch Window* hinzufügen

Symbole aus dem Programm oder aus dem Symbol Editor ins *Watch Window* ziehen.

Watch Window



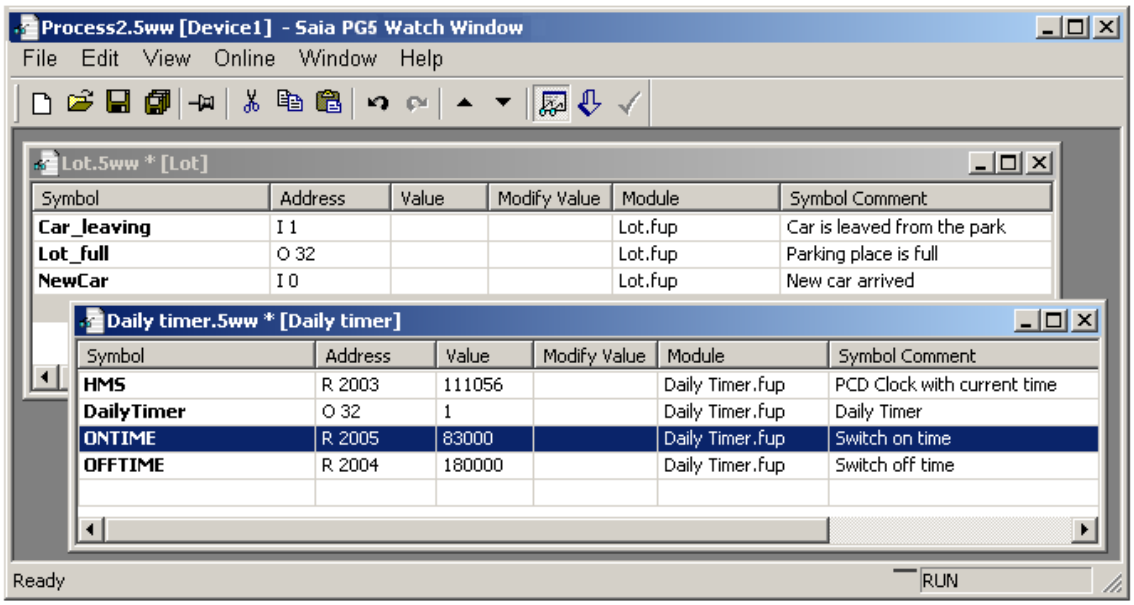
Die Symbole können direkt im Fenster editiert werden



10.3.3 On-line Anzeige der Daten

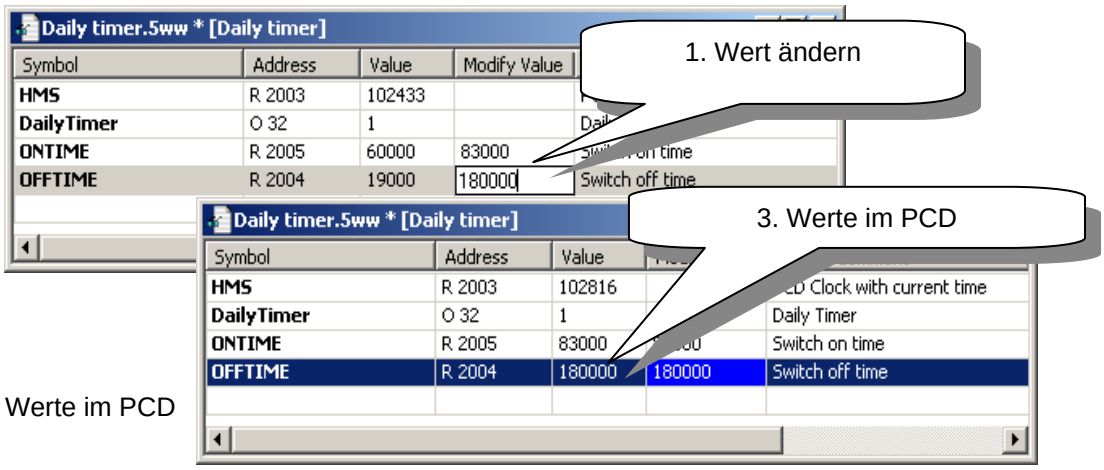
Zum betrachten der Werte/Zustände eines Symbols den *Go On/Offline* Knopf drücken.

Start/Stop Monitoring



10.3.4 On-line Änderung von Daten

Die Werte der Symbole können On-line geändert werden.

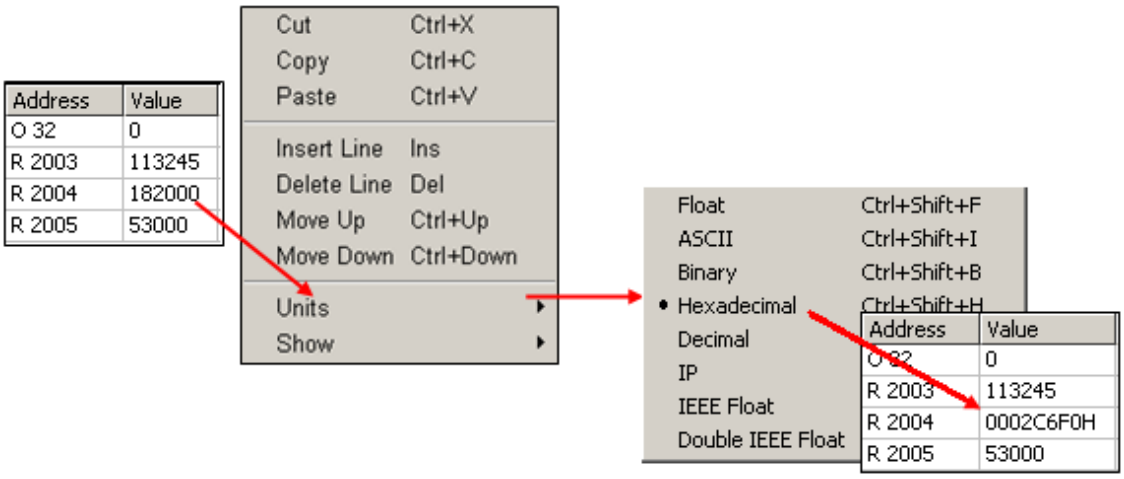


2. Die Werte im PCD

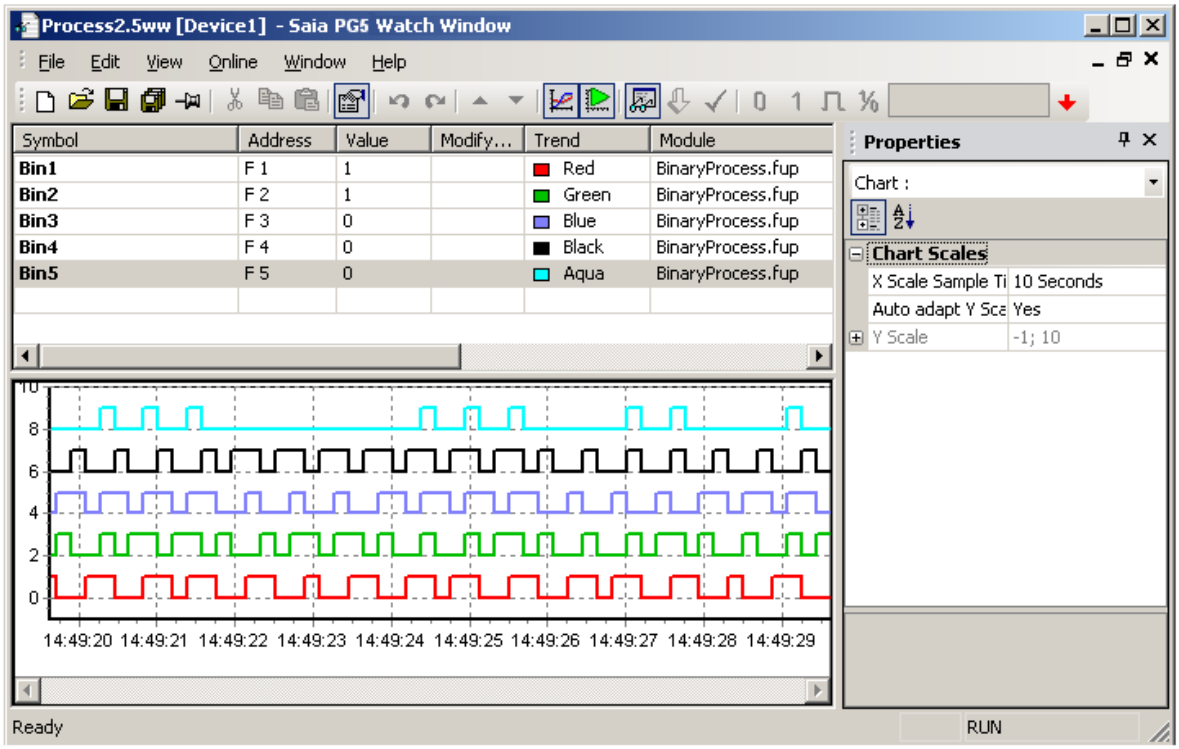
10.3.5 Anzeigeformat

Das Anzeigeformat der Werte kann den jeweiligen Erfordernissen angepasst werden.

Beispiel: Anzeige des Registers R 2004 hexadezimal



10.3.6 Trendfunktionen



Show/Hide Trend

Watch Window kann einen Kurvenverlauf von maximal acht verschiedenen Werten zeichnen.Z.B. von Register, Flags, etc.
Sollten mehr als acht Werte visualisiert werden, so kann dies in einem weiteren Window Fenster geschehen.

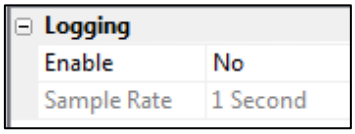
Betätigen Sie den *Show/Hide Trend* Knopf um die Trendlinien anzuzeigen bzw. zu verstecken. Setzen Sie eine Kurvenfarbe für die gewünschten Symbole in der Tabelle. Um die Trendzeichnung zu starten klicken Sie auf den *Start/Pause Trend Update Knopf* in der Toolbar.

Start/Pause Trend Update

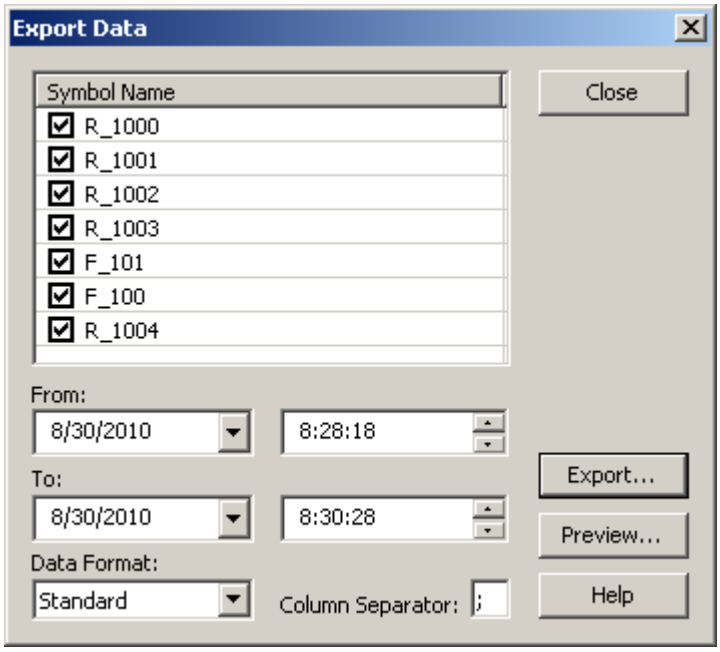
Durch fokussieren der Symboltabelle oder des Kurvenverlaufs werden Konfigurationswerte auf der Seite angezeigt. Über diese Werte lassen sich Einstellungen wie das Aufzeichnungsintervall oder die Skalierung anpassen.

10.3.7 Aufzeichnungsfunktion

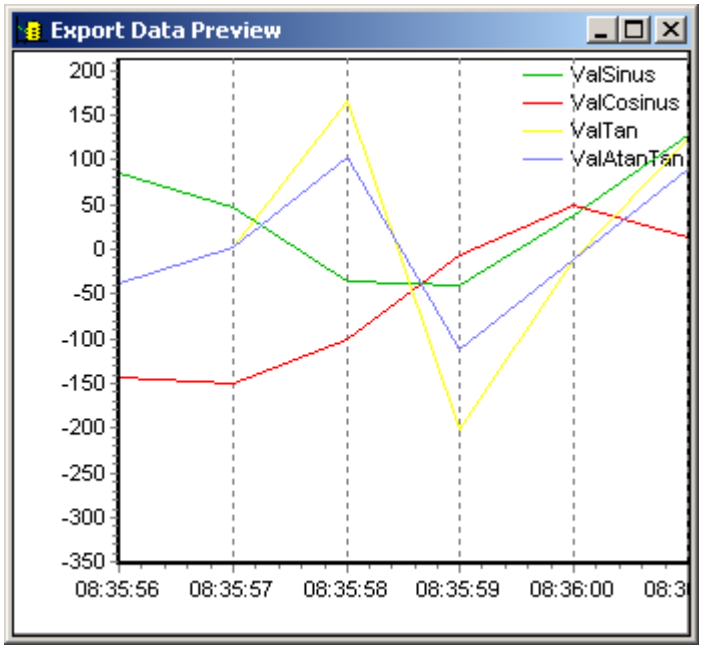
Öffnen Sie das *Properties* Fenster von Watch Windows über das Menu *View->Properties*. Wählen Sie ein oder mehrerer Symbole in der Symboltabelle und setzen Sie *Logging Enable* auf *Yes*. Nach dem Ende der Messung setzten sie *Logging Enable* wieder auf *No*.



Das Menu *Online->Export Data* erlaubt das setzen der Aufzeichnungsperiode und der Datenquelle(n).



Die Schaltfläche *Preview…* zeigt eine Vorschau der Daten an.



Die Schaltfläche *Export* im *Export Data* Dialog speichert die Trenddaten in eine Datei. Der Anwender gibt den Namen und Ort an wo die Daten gespeichert werden.

10.3.8 Symbole mit grosse und kleiner Magnitude im selben Graph

Wenn Symbol mit verschiedenen grosser Magnitude in einem Graphen dargestellt werden, kann es vorkommen, dass das Symbol mit viel Variation in der y-Achse den granzen Graph füllt, währenddessen das Symbol mit einer kleinen Magnitude nur einen Bruchteil der vertikalen Skala verwendet.

Es gibt zwei Möglichkeiten, das Anzeigen der Werte zu verbessern:

1. Ein ‘Trending scale factor’ kann im *Properties* Fensters des entsprechenden Symbols definiert werden. Dieser Faktor ermöglicht das

Verstärken oder Reduzieren der Magnitude. Wenn der Graph gelesen wird, muss dieser Faktor natürlich dann auch berücksichtig werden und der ausgelesene werde durch den Faktor dividiert werden um den korrekten Wert zu erhalten.

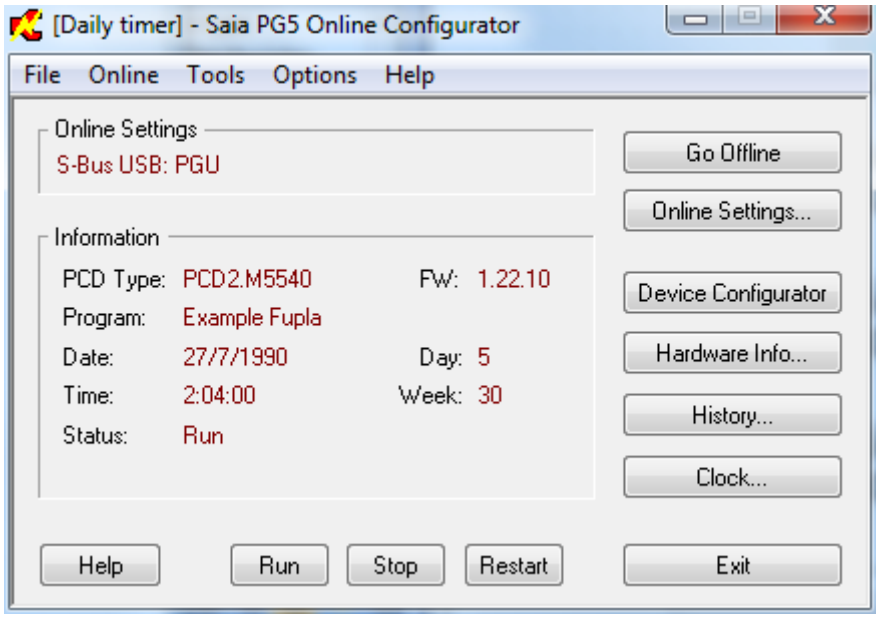
2. Es kann eine zweite Y-Achse zum Graphen hinzugefügt werden. Dieser weisst dann eine andere Skala auf als die primäre Y-Achse. Wählen Sie das Symbol aus der Symboltabelle aus, welches Sie über die 2. Y-Achse visualisieren wollen und wählen Sie *Trending, Axis* aus dessen *Properties* Fenster.

Anmerkung:
Der Trend ist besser wenn die vertikale Skalierung automatisch der Magnitude des Symbols angepasst wird. Um dies zu erreichen setzen Sie *Left/Right Axis, Auto adapt Y Scale* auf *Yes*.

10.3.9 Trends mit mehreren binären Symbolen

Wenn Kurven von mehreren Symbolen mit derselben vertikalen Magnitude dargestellt werden sollen (z.B. binäre Symbole wie Flags etc.), dann kann es hilfreich sein, einen jeweils leicht verschieden Offset für die jeweiligen Symbole zu definieren um die Lesbarkeit des Graphen zu erhöhen.
Für binäre Symbole wird dieser vertikale Offset in der Regel automatisch vorkonfiguriert, er kann jedoch über die Eigenschaft *Trending, Offset* im *Properties* Fenster manuell angepasst werden.

10.4 On-line Konfigurator



PCD type	PCD Typ Referenz-Nummer
Version	Version der PCD Firmware
Program Name	Anwender-Programmname
Date	PCD Uhrdatum (wenn keine Uhr: 1/1/92)
Time	PCD Uhrzeit
Day	Wochentag: 1 = Montag, ... 7 = Sonntag
Week	Wochennummer
Status	Betriebszustand: Run, Stop, Halt, Conditional Run
Onlinesettings	Kommunikationsverbindung: PGU, S-BUS,…

Wird die rote Information nicht, oder eine *No response* Nachricht angezeigt, kann nicht zwischen PCD und *Online Configurator* kommuniziert werden.

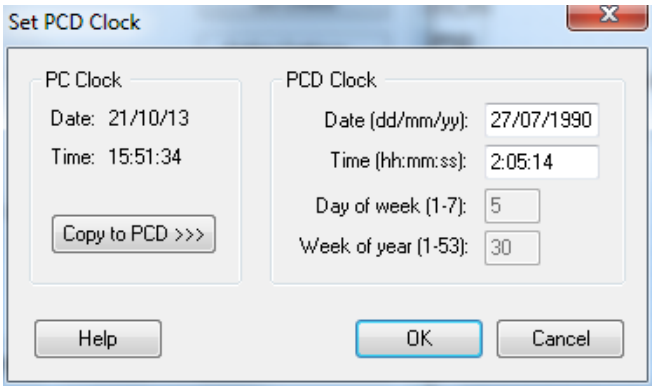
Wenn dem so ist, überprüfen ob:

- der PC korrekt mit der PCD mit dem Kabel PCD8.K111 verbunden ist?
- die Kommunikations-Parameter korrekt mit dem *Settings* Knopf ausgewählt wurden?

10.4.1 PCD-Zeit einstellen

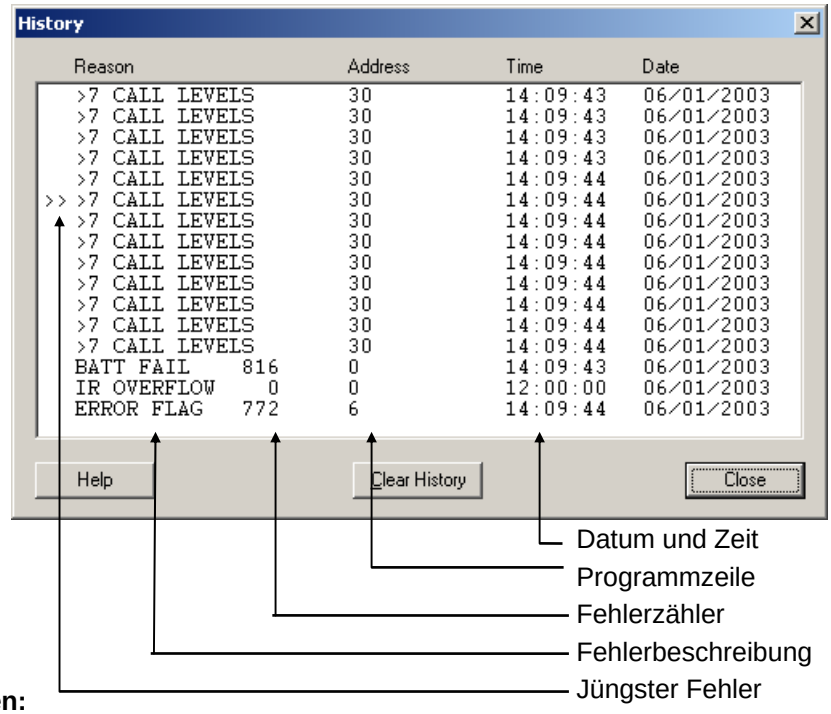
1. *Online configurator* Knopf im *Projekt Manager* Fenster drücken. Dann *Clock* Knopf drücken.

2. Mit dem *Copy to PCD* >>> Knopf PC-Zeit in die Steuerung kopieren, oder die Daten direkt in den Feldern des *PCD Clock* Fensters anpassen.



10.4.2 Das History

Das *History* zeichnet alle Hardware oder Software Fehler auf, die während des Betriebs der PCD auftauchen. Diese Tabelle wird laufend auf den neuesten Stand gebracht, auch wenn keine XOBs programmiert sind. Das History ist anzuschauen, wenn die CPU *Error* LRD kommt.



Bemerkungen:

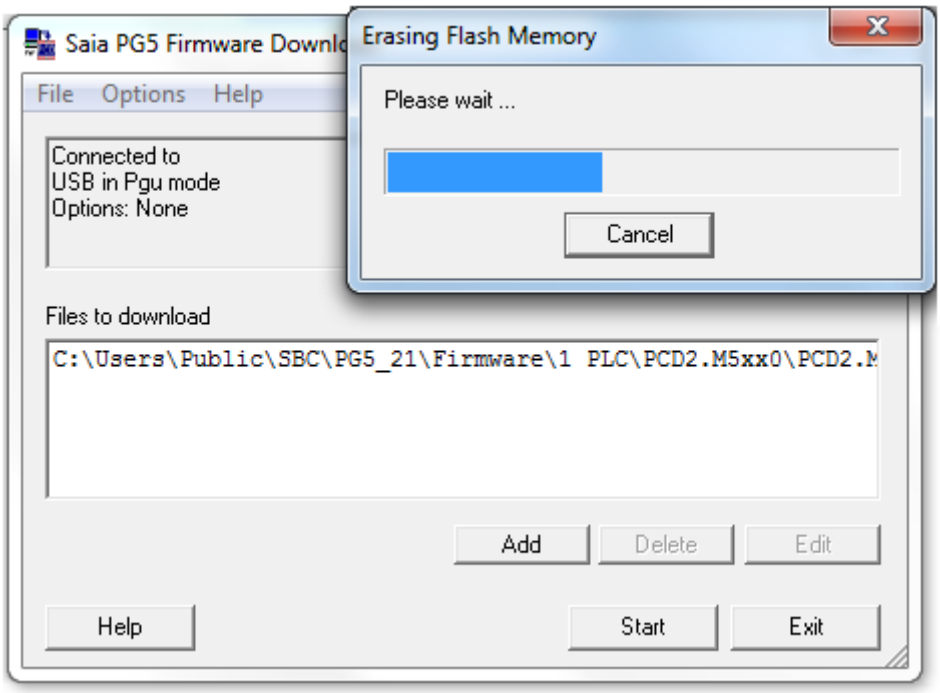
- Wenn ein Fehler bis zur entsprechenden Programm-Zeile verfolgt werden kann, wird diese gezeigt. Andernfalls wird er hexadezimal angezeigt.
- XOB 0 erscheint nur, wenn er programmiert wurde.

10.5 Aktualisieren der Firmware. *(Firmware Downloader)*

Ab und zu muss die Programm Firmware, um von den neuesten PCD Produkt-Innovationen zu profitieren, aktualisiert werden.

Bei den meisten Steuerungen Kann die Firmware durch Austausch des EPROM aktualisiert werden.

In den neuesten[1)] PCDs kann die Firmware in Flash Speicher mittels eines kleinen Dienstprogramms geladen werden. Dies ist zugänglich im Menü über *Tool, Firmware Downloader* im Projekt Manager.



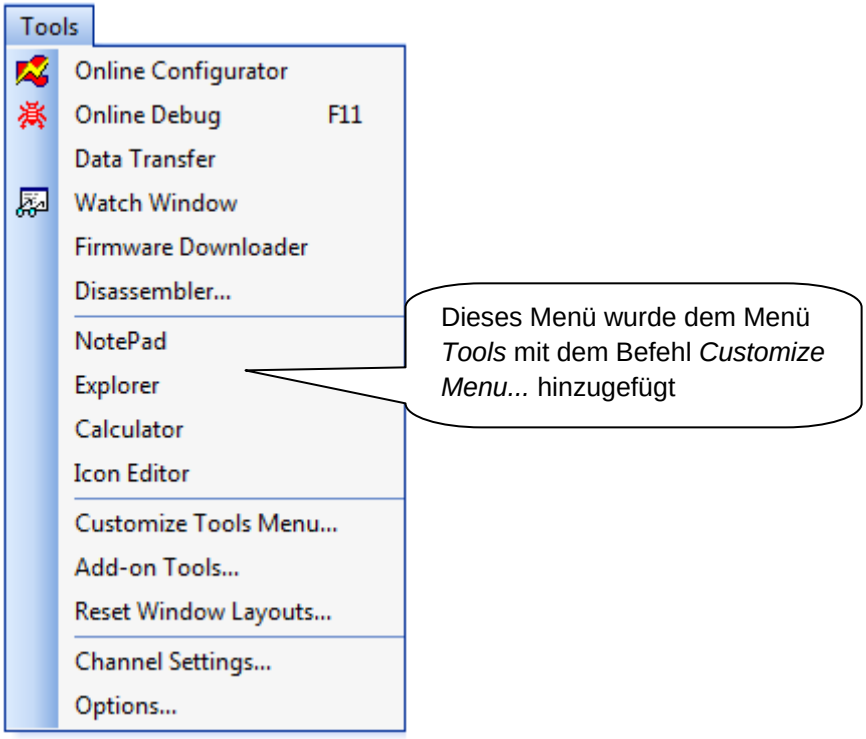
Download Instruktionen:

- Der *ADD* fügt eine neue Firmware Datei (*.blk) der Liste *Files* hinzu.
- Die neuesten Firmware Dateien sind im Verzeichnis *FW* auf der PG5 CD zu finden.
- Im Menü *File, Settings* Kommunikations-Parameter an PGU Modus anpassen (zur Zeit der einzige unterstützte Modus).
- Firmware zum download in die PCD auswählen.
- Kabel PCD8.K111 mit dem PCD PGU Port verbinden.
- Spannungsversorgung abschalten PCD, dann wieder einschalten.
- Bei der PCD2.M480 den *Run/Halt* Knopf zweimal drücken, währenddem die *Run* LED blinkt.
- Download der Firmware mit dem *Start* Knopf. Eine Dialogbox zeigt den Fortschritt der Datenübertragung.
- Ist die Datenübertragung abgeschlossen, beginnen die PCD *Run*, *Halt* und *Error* LEDs zu blinken. Die PCD reorganisiert ihre Speicherinformationen. Bitte, eine weitere Minute warten, bevor Sie die Spannung der Steuerung abschalten, oder Sie in Ihrer Arbeit fortfahren.

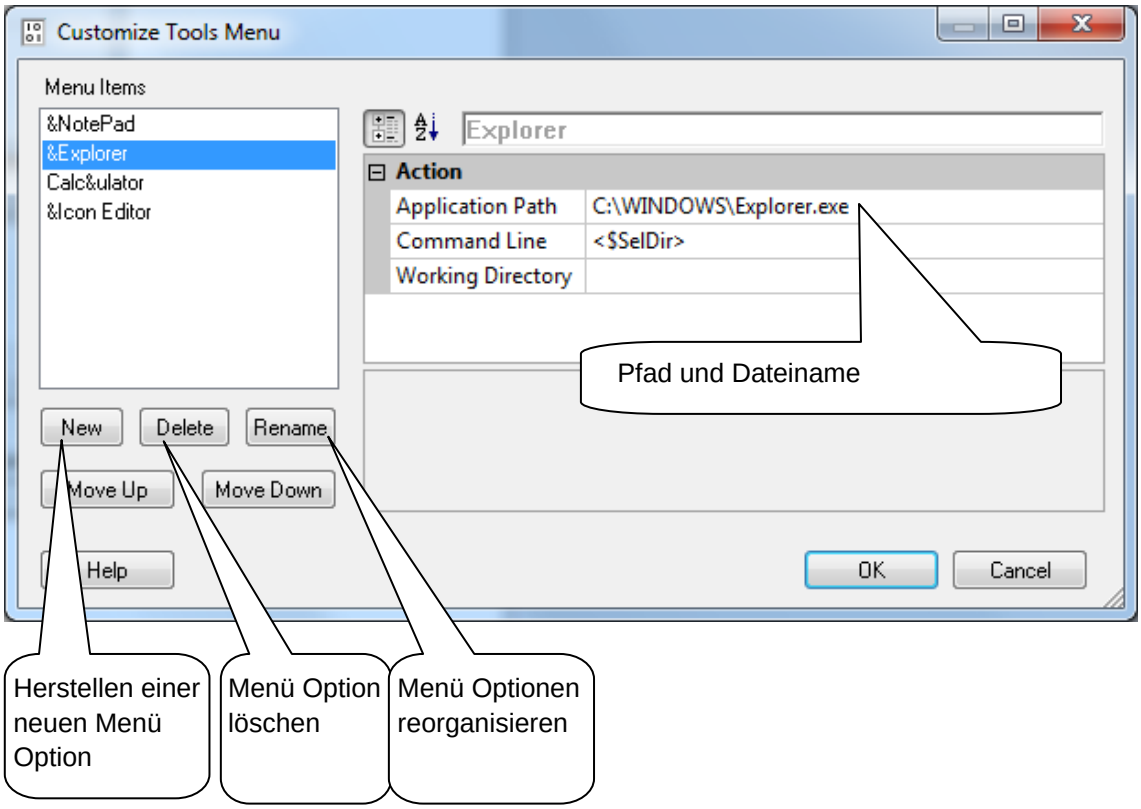
[1)] PCD2.M170, PCD4.M170, PCD2.M480

10.6 Anwender Menüs

Das Menü *Tools* im *Projekt Manager* Fenster kann mit Shortcuts zu Ihren bevorzugten Programmen erweitert werden.



Zum hinzufügen eines Shortcuts, folgendermassen verfahren:



Inhalt

11 Saia PCD-Netzwerke (S-Net)

11.1 Zusammenfassung

Automatisierungslösungen bestehen oft aus verschiedenen, dezentralisierten PCD-Controllern, Terminals und Überwachungs-Computern, die in einem Kommunikationsnetzwerk verbunden sind. Jede Station kontrolliert einen Teil des Prozesses und tauscht Daten mit den anderen Stationen im Netzwerk aus.

Um die Flexibilität eines solchen Konzepts zu garantieren, unterstützt das PCD-System verschiedene Arten von Kommunikationsnetzwerken. Jedes Netzwerk hat seine eigenen Fähigkeiten, so dass der Anwender auswählen sollte, welches Netzwerk für die Anwendung das richtige ist.

Der PG5 ist ein effizientes Werkzeug für das Umsetzen dieser Lösungen:

- *Saia Project Manager* bietet einen Überblick über die Stationen (PCDs) und ihre Konfigurationsparameter, eingeschlossen die Kommunikationsparameter des Netzwerkes.

- Der Fupla oder der IL-Editor erlauben das Programmieren des Datenaustauschs zwischen den PCD-Stationen im Netzwerk.

Die Programmierbeispiele aus den folgenden Kapiteln wurden alle mit der PG5 installiert und sind die Basis für einen Test und das Verständnis der Funktionalität des Datenaustauschs über unterschiedliche PCD-Netzwerke. Sie werden feststellen, dass einige Beispiele sehr nah an praktischen Umsetzungen sind.

11.2 Wahl des Netzwerks

Die Wahl des Netzwerks hängt von den Anforderungen der Anwendung ab. Die folgenden S-Net-Netzwerktypen sind verfügbar:

- Profi-S-Bus : Feldbus-Netzwerk basierend auf dem Standard "Profibus FDL"
- Ether-S-Bus : Informationsnetzwerk basierend auf dem Ethernet-Standard
- Serieller S-Bus : Netzwerk basierend auf der seriellen Schnittstelle RS 485/232
- S-Bus Modem : Netzwerk basierend auf einem analogen oder digitalen Telefonnetz
- Profi-S-IO : Feldbus-Netzwerk basierend auf dem Standard "Profibus DP" ("Profi-S-IO")
- Profibus DP: Feldbus-Netzwerk basierend auf dem Standard "Profibus DP"

Die verschiedenen Netzwerke werden nach den Services, der technischen Charakteristik und ihren Anwendungsgebieten unterschieden.

11.2.1 Unterstützte Services

Alle Kommunikationsnetzwerke unterstützen den Transport von PCD-Daten als Eingang, Ausgang, Flags, Register usw., einige unterstützen außerdem die Programmierung, Steuerung und Inbetriebnahme der PCD-Systeme über Netzwerke mittels den PG5-Werkzeugen.

11.2.2 Funktionen

Kommunikationsgeschwindigkeit
Die Kommunikationsgeschwindigkeit bestimmt die Reaktionszeit für den Datentransfer zwischen den Stationen. Wenn die Menge der zu übertragenden Daten groß ist, oder wenn die Reaktionszeit kurz sein soll, dann muss die Kommunikationsgeschwindigkeit hoch sein. Beachten Sie, dass die Kommunikationsgeschwindigkeit des Netzwerks einstellbar ist. Alle Stationen im Netzwerk müssen die gleiche Geschwindigkeit nutzen.

Maximale Entfernung
Die Entfernung zwischen den Stationen kann eine Begrenzung für weit auseinander liegende Stationen sein. Die maximale Entfernung kann nicht ohne Verstärkung der elektrischen Signale durch einen Repeater oder Switch / Hub überschritten werden. Im Allgemeinen hängt die maximale Entfernung auch von der Kommunikationsgeschwindigkeit ab. Je höher die Geschwindigkeit, desto kürzer die Entfernung. Das Reduzieren der Kommunikationsgeschwindigkeit ist vielfach eine Lösung für das Überbrücken großer Entfernungen.

Kommunikationsprotokoll
Ein "Protokoll" ist das Nachrichtenformat, das für den Datenaustausch zwischen zwei Stationen im Netzwerk benutzt wird. Das Protokoll ist vergleichbar mit der Sprache im Gespräch zwischen zwei Personen – sie verstehen sich nur, wenn sie dieselbe Sprache sprechen. In ähnlicher Weise können zwei Stationen nur dann Daten austauschen, wenn sie das gleiche Protokoll benutzen.

Die Protokolle einiger Kommunikationsnetzwerke sind offizielle Standards. Dies ist ein großer Vorteil, wenn Geräte von verschiedenen Herstellern miteinander kommunizieren müssen. Feldbusse und Sensoren nutzen oft das Profibus DP-Protokoll.

Bei bestimmten Kommunikationsnetzwerken wie Ethernet oder Profibus FDL ist es möglich, Daten auf demselben physischen Netzwerk mit verschiedenen Protokollen auszutauschen. Aber in jedem Fall müssen die beiden kommunizierenden Stationen das gleiche Protokoll nutzen.

Datenaustausch im Master/Slave- oder im Multimaster-Modus
Ein "Master/Slave"-Netzwerk besteht aus einer Masterstation und mehreren Slavestationen. Die Masterstation kontrolliert den Datenaustausch zwischen den Slavestationen.

Ein "Multimaster"-Netzwerk besteht aus mehreren Master- und mehreren Slavestationen. Jede Masterstation tauscht Daten mit anderen Master- und Slavestationen aus.

In beiden Fällen ist der direkte Datenaustausch zwischen den Slaves nicht erlaubt.

Einsatzgebiete
Einige Netzwerke sind für spezielle Zwecke gestaltet worden. Ein Beispiel: Profibus DP ist ein Protokoll, das sich auf die Maschinerie ausrichtet. Das Protokoll dieses Netzwerks ist gut standardisiert und es gibt eine große Anzahl an kompatiblen Geräten von verschiedenen Herstellern, die die Datenübertragung auf demselben Bus (z.B. Motorbefehle) erlauben.

Das Ether-S-Bus-Netzwerk ist eher auf Überwachungssysteme und OPC-Server ausgerichtet, oder kann für die PG5-Programmierung und Inbetriebnahme genutzt werden.

Der serielle S-Bus bietet einen einfachen Weg, PCD-Systeme zu verbinden. Es ist ein sehr ökonomisches Netzwerk, da es dieselben Services wie Ether-S-Bus über RS 485 sowie das analoge Telefonnetz und ISDN bietet (S-Bus Modem).

Das neue Profi-S-Bus-Netzwerk verbindet alle Vorteile eines Multimaster-Netzwerks und einer hohen Kommunikationsgeschwindigkeit mit einem Feldbus-Netzwerk für industrielle Automatisierungsanwendungen.

Kommunikationsnetzwerk S-Net

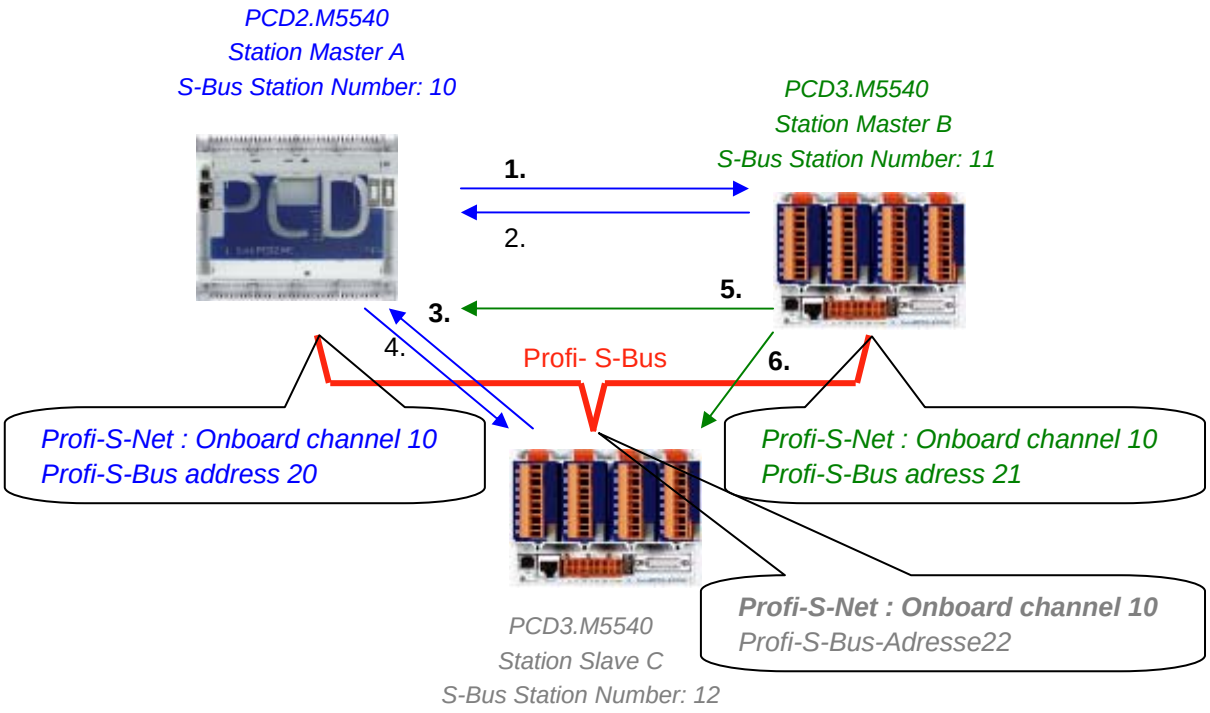
Services :	**Ether-S-Bus**	**Profi-S-Bus**	**Serieller S-Bus**	**S-Bus Modem**	**Profi-S-IO Profibus DP**
PCD-Programmierung	Ja	Ja	Ja	Ja	Nein
Datenaustausch	Ja	Ja	Ja	Ja	Ja
Charakteristik :					
Max. Übertragungs-geschwindigkeit	10 und 10 Mbd	12 Mbd	38,4 / 115,2 Kbd	38,4 / 115,2 Kbd	12 Mbd
Max. Entfernung ohne Repeater oder Switch/Hub	100 m	100 m	1200 m	-	100 m
Kabeltyp	4 x Twisted Pair	1 x Twisted Pair	1 x Twisted Pair	-	1 x Twisted Pair
Protokoll	Saia PCD	Saia PCD	Saia PCD	Saia PCD	Normalisiertes ISO
Austauschmodus	Multimaster	Multimaster	Master/Slave	Multimaster	Master/Slave
Max. Anzahl an Stationen	Unbegrenzt	126	254	Unbegrenzt	126
Einsatzgebiet	Industrie, Gebäude	Industrie, Gebäude	Industrie, Gebäude	Industrie, Gebäude	Industrie, Gebäude

Inhaltsverzeichnis

12 Profi-S-Bus

Dieses Beispiel zeigt, wie man ein paar Daten, wie beispielsweise Register und Indikatoren zwischen den am Profi-S-Bus-Netz angeschlossenen PCD austauscht.

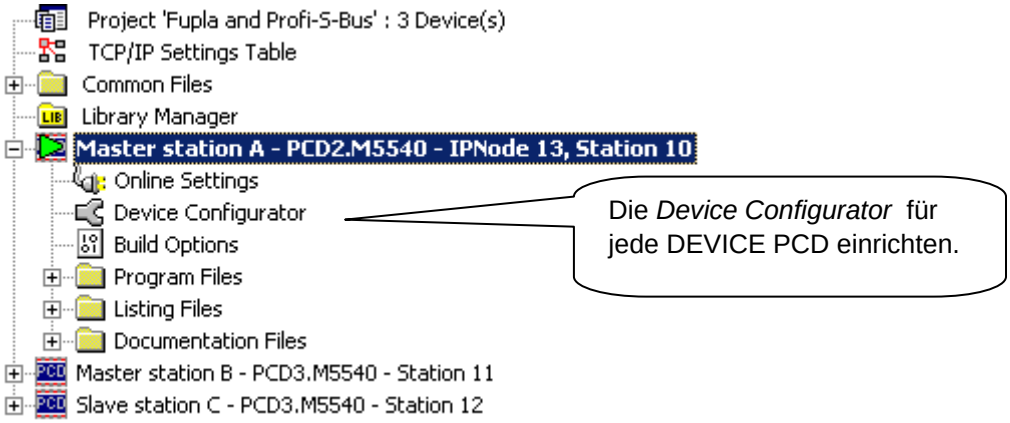
12.1 Beispiel eines Profi-S-Bus-Netzes.



12.2 Beispiele eines Profi-S-Bus-Datenaustausch.

	Master with data exchanges	*Data on the network*	*Passive master or slave*
	Master station A		*Master station B*
1	*Blinker 0 .. 7* *F 0 7*	*Write 8 flags in the Master station B*	*Station_A.Blinker 0 .. 7* *F 100 .. 107*
2	*Master_B .Value100* *R 125*	*Read 1 register in the Master station B*	*Value 100* *R 25*
			Slave station C
3	*Slave_C.Binary 0 .. 7* *F 100 .. 107*	*Read 8 flags in the slave station C*	*Binary0 .. 7* *F 20 .. 27*
4	*Value 0 .. 5* *R 0 .. 5*	*Write 6 registers in the slave station C*	*Master_A. Value 0 .. 5* *R 20 .. 25*
	Master station B		***Master station A***
5	*Temperature 1 .. 4* *Dynamic registers*	*Write the temperature measures to the slave C*	*Master_B.Temperature 1 .. 4* *R 100 .. 104*
			Slave station C
6	*Temperature 1 .. 4* *Dynamic registers*	*Write the temperature measures to the master A*	*Master_B.Temperature 1 .. 4* *R 100 .. 104*

12.3 Projekt PG5

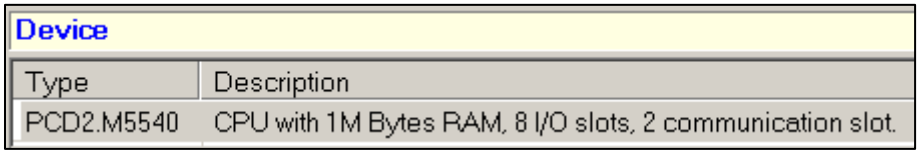


Der *Saia PG5 Project Manager* gibt einen Überblick über die PCD eines Applikationsprojektes sowie die Parameter für die Kommunikationsnetze. Beginnen wir mit dem Hinzufügen einer Device in das Projekt für jede auf dem Netzwerk verfügbare Station.

12.4 *Device Configurator*

Die Konfigurationen der *Device Configurator* einer Masterstation und der Slaves sind fast gleich.

12.4.1 Festlegen der PCD-Parameter

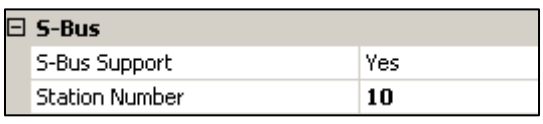


Device

Type	Description
PCD2.M5540	CPU with 1M Bytes RAM, 8 I/O slots, 2 communication slot.

PCD-Type
Festlegen des Device-Typs

12.4.2 Festlegen der S-Bus-Stationsnummer auf dem Netzwerk



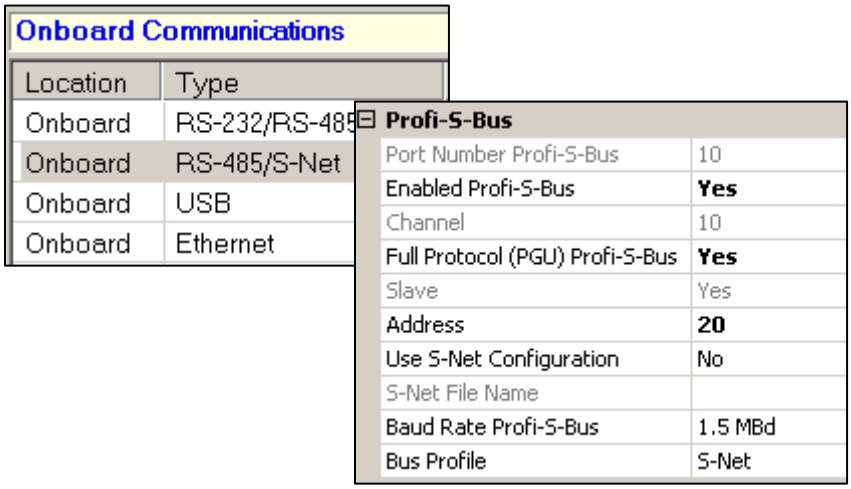
S-Bus

S-Bus Support	Yes
Station Number	**10**

Device properties:

Station Number
Die S-Bus-Stationsnummer ist für alle Kommunikationskanäle der PCD gleich.

12.4.3 Festlegen des Profi-S-Bus-Kommunikationskanals



Onboard Communication, properties:

Full Protocol (PGU) Profi-S-Bus
Festlegen des Kanals als Slave oder PGU. Diese Festlegung kann mit der Masterfunktion zusammen erfolgen, indem man eine Fbox SASI Master in das Fupla-Programm aufnimmt.

Slave + PGU
Unterstützt den Datenaustausch mit den Masterstationen, den Überwachungssystemen und Terminals. Unterstützt aber auch das Hilfssystem zur Programmierung und Inbetriebnahme von PG5.

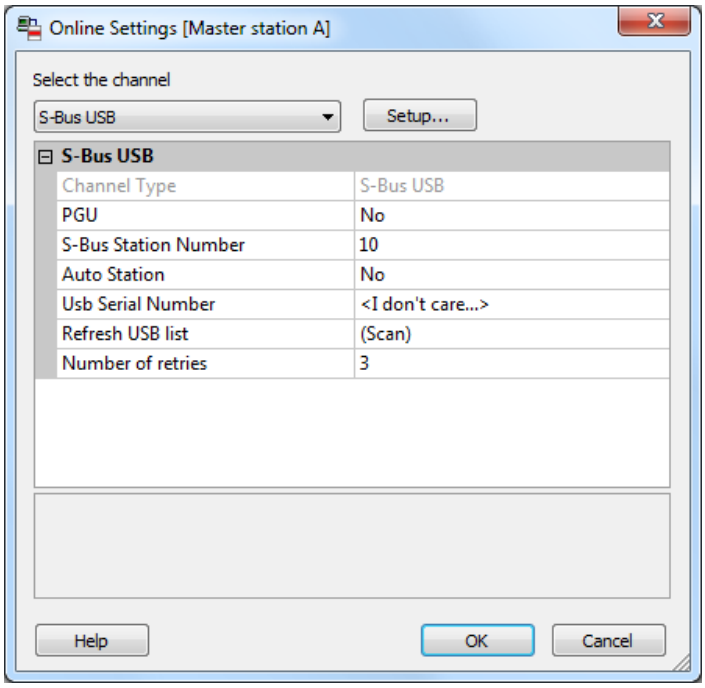
Slave
Unterstützt nur den Datenaustausch mit den Masterstationen, den Überwachungssystemen und Terminals.

Address
Sie mit dem Kanal verbundene Profi-S-Bus-Stationsnummer

Baud Rate Profi-S-Bus
Kommunikationsgeschwindigkeit, muss auf allen Stationen des Netzwerks identisch sein.

Bus Profile
Die Timings zur Übertragung werden in drei Profile eingeteilt: S-Net, DP oder Benutzer. Beim Benutzerprofil kann man über die Schaltfläche *Bus Parameter* seine eigenen *timings* festlegen. Das Profil muss auf allen Stationen des Netzwerks identisch sein. Das Profil S-Net ist bei Verwendung von RIO PCD3.T76x auf den Netzwerken erforderlich.

12.4.4 Laden der Device Configurator Parameter auf den Device

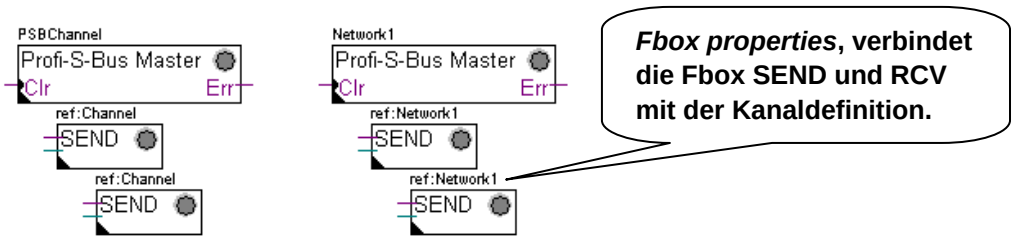


Mit den neuen Systemen PCD2 und PCD3, können die *Device Configurator Settings* über die USB-Schnittstelle heruntergeladen werden. Man muss nur prüfen, ob die *Online Settings* mit einem *Profi-S-Bus + PGU*-Kanal definiert worden sind.

Die Parameter, über die Schaltfläche *Download* im Fenster *Device Configurator*, in die PCD herunterladen.

12.5 Fupla-Programm

12.5.1 Zuweisung des Kanals mithilfe einer Fbox SASI



Die Zuweisung eines Kanals erfolgt über eine zu Beginn der Fupla-Datei platzierte FBox SASI. Jedes Kommunikationsnetz verfügt über seine eigene Fbox SASI, weil sich die Zuweisungsparameter von einem zum anderen Netz voneinander unterscheiden. Dasselbe gilt für eine Slave- oder Masterstation.

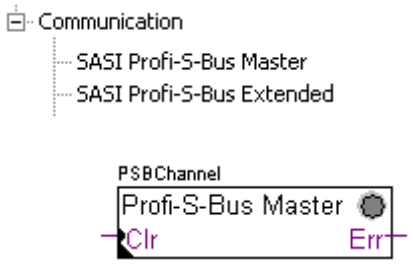
Wenn die PCD mehrere Kommunikationskanäle benutzt, sollte jeder einzelne Kanal über die entsprechende Fbox SASI definiert werden. Anschließend den Cursor mit der Maus auf die Fbox SASI setzen, das Kontextmenü markieren und für jede Fbox, je nach benutztem Kanal, den Parameter *Name* anders festlegen. Dieser Name

ermöglicht es, verschiedene Fbox zum Austauschen von SEND und RCV mit der dem benutzten Kanal entsprechenden Fbox SASI zu verbinden.

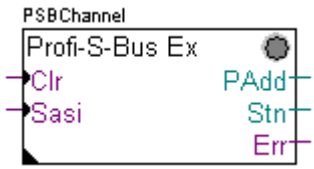
Je nach Netzwerk, können die Kommunikationsparameter teilweise im Einstellungsfenster der Fbox SASI festgelegt und in den *Device Configurator* vervollständigt werden.

Aber die Nummer des Kommunikationskanals wird immer über das Einstellungsfenster der FBox SASI festgelegt. Die Kanalnummer hängt von der PCD-Hardware und der verwendeten Kommunikationshardware ab : Slot B1, B2, serielle Schnittstelle PCD7.F, …

12.5.2 Zuweisung des Masterkanals



Masterstation



Masterstation mit Festlegung der timings

Die Zuweisung des Masterkanals erfolgt über die Vervollständigung der *Device Configurator* mit einer der obigen Fboxen.

Nur der Kommunikationskanal und die Masterkanaltimings können über die Fbox eingestellt werden. Die anderen Parameter werden in den *Device Configurator* festgelegt.

Properties Parameters des Einstellungsfensters:

Kanal
Festlegung des entsprechenden Kanals, der am Netzwerk angeschlossenen seriellen Schnittstelle. Hängt von der PCD und ihrer Hardware ab.

Timing
Das Timeout wird allgemein mit dem Standardwert (0) festgelegt und wird nur für ganz bestimmte Applikationen (Gateway) verändert.

12.5.3 Zuweisung des Slavekanals

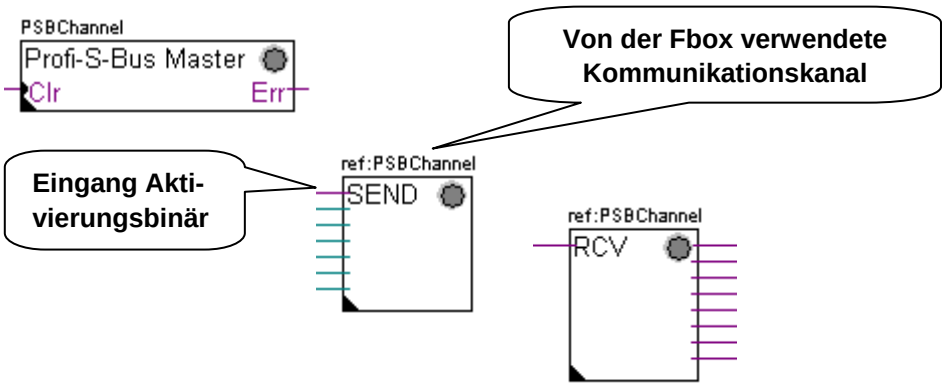
Für die Slavestation eines Profi-S-Bus-Netzes ist keine FBox SASI erforderlich. Alle erforderlichen Festlegungen wurden bereits in den *Device Configurator* gemacht.

12.5.4 Prinzip eines Datenaustauschs auf einem Multimasternetzwerk.

Das Multimaster-Kommunikationsnetzwerk setzt sich aus mehreren Master- und Slavestationen zusammen. Die Masterstationen sind die Stationen, die alleine dazu berechtigt sind, die Daten anderer Master- aber auch Slavestationen zu lesen oder zu schreiben. Der Datenaustausch zwischen den Slaves ist nicht erlaubt.

Mit einem Multimaster-Kommunikationsmodus werden die über das Netzwerk ausgeführten Daten unter mehreren Mastern aufgeteilt. Ein einziger Master verfügt jedoch an einem bestimmten Moment einen Token, der ihn dazu berechtigt, Daten mit sämtlichen Master- oder Slavestationen innerhalb des Netzwerks auszutauschen. Wenn der Master die Übertragung seiner Daten auf dem Netzwerk beendet hat, wird dieser Token einem anderen Master zugewiesen, der nun die Möglichkeit hat, Daten mit allen Master- oder Slavestationen auszutauschen. Der Token kreist automatisch durch die Masterstationen, wobei die Slaves niemals den Token erhalten und daher auch keine Daten anderer Stationen innerhalb des Netzwerks lesen oder schreiben können.

12.5.5 Datenaustausch zwischen Master- und Slavestationen.



Der Benutzer kontrolliert den Datenaustausch zwischen den Stationen mit Hilfe verschiedener Fupla-Fbox, die in den Fupla-Seiten platziert und in dem *Fbox Selector* zur Verfügung stehen. Wir finden hier Fboxen zum Schreiben (*SEND*) oder zum Lesen (*RCV*) eines Datenpaketes, aber auch zur Unterstützung verschiedener Datenformate: Binär, Ganzzahl, gleitend, Datenblocks, usw.

Die Fboxen *SEND* oder *RCV* können mit mehr oder weniger Ein- und Ausgängen erweitert werden, was ihnen ermöglicht, die Größe des mit einer anderen Station auszutauschenden Datenpakets zu definieren.

Die Adresse des von der Datenübertragungs-Fbox verwendeten Kommunikationskanals, wird über das oben links von der Fbox angezeigte Symbol definiert und verbindet sie mit der Fbox SASI desselben Namens, in der die Kanaladresse festgelegt ist. Dieses Symbol kann bearbeitet werden, indem man den Cursor auf die Fbox setzt und das Kontextmenü *properties* markiert.

Jede Fbox *SEND* und *RCV* ist mit einem binären Eingang für den Datenaustausch ausgestattet. Wenn sich dieser Eingang ständig im Zustand “high” befindet, dann wird der Datenaustausch so schnell wie möglich wiederholt. Wenn ein kurzer Impuls auf diesen Eingang gebracht wird, dann erfolgt der Datenaustausch mindestens einmal, es ist aber immer möglich, diesen über die Schaltfläche *Execute* oder beim Kaltstart des PCD mit der Option *Initialization* im Einstellungsfenster zu forcieren.

Die auf den Eingängen einer Fbox *SEND* vorhandenen Daten der Masterstation, werden zu der im Einstellungsfenster festgelegten Slavestation geschickt. Die auf den Ausgängen einer Fbox *RCV* vorhandenen Daten stammen aus der Slavestation anhand der Parameter im Einstellungsfenster: Adresse der Slavestation, Quellelemente und Basisadresse.

Nur die Masterstationen werden über die Fbox *SEND* und *RCV* programmiert! Die Slavestationen dürfen nur mit dem Kommunikationskanal zugewiesen werden.

Je nach verwendeter Fbox kann man im Einstellungsfenster festlegen, zu welchen Slavestationen die Daten der Masterstation (*SEND*) geschickt werden oder in welchen Slavestationen der Master liest. (*RCV*)

Properties Parameters des Einstellungsfensters:

Profi-S-Bus Address
Festlegung der Stationsnummer Profi-S-Bus Bus Slave

Source, destination station
Festlegung der Stationsnummer S-Bus Bus Slave

Source, destination element
Festlegung des im Slave zu schreibenden oder zu lesenden Datentyps.

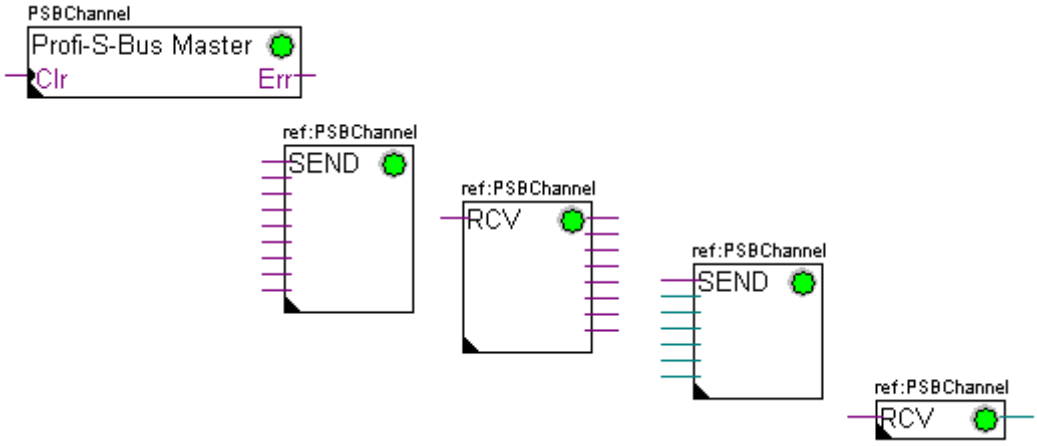
Source, destination address
Festlegung der Adresse des ersten im Slave zu schreibenden oder zu lesenden Datensatzes.
Die Anzahl der ausgetauschten Daten hängt von der Anzahl der Ein- oder Ausgänge der Fbox *SEND*, *RCV* ab.

12.5.6 Diagnosen

Wenn das Programm online ist, wird eine grüne oder rote LED oben rechts der Fbox *SASI*, *SEND* und *RCV* angezeigt. Grün bedeutet, dass die Datenübertragung OK ist, rot zeigt einen Fehler an.

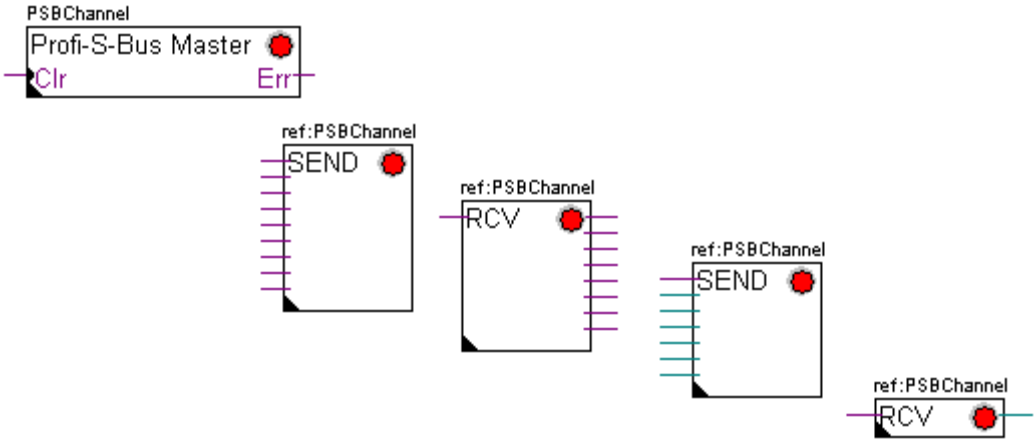
Einwandfreie Funktion

Alle Fboxen sind grün, der Datenaustausch funktioniert richtig.



Es können keine Daten über das Netzwerk ausgetauscht werden.

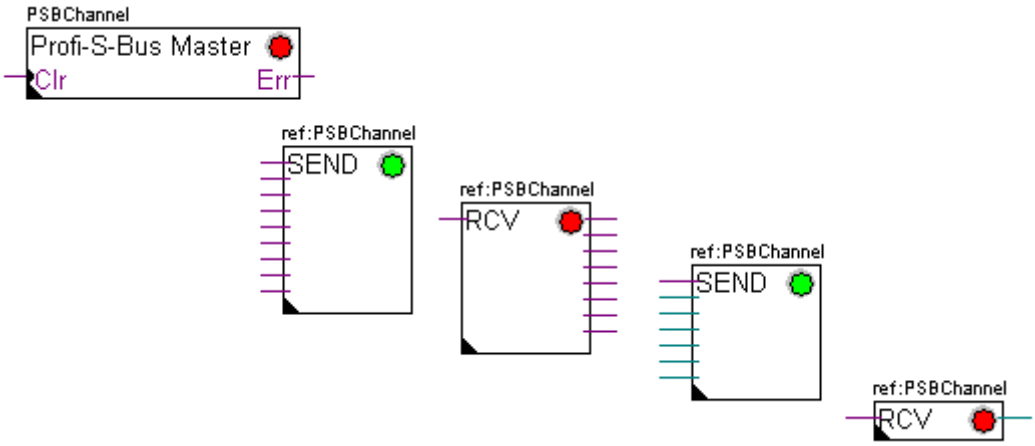
Die Fboxen *SASI* , *SEND* und *RCV* sind rot, es können keine Daten über das Netzwerk ausgetauscht werden.



- Netzwerkverkabelung prüfen
- Prüfen, ob die Firmwareversion Profi-S-Bus unterstützt.

Einige Daten können nicht über das Netzwerk ausgetauscht werden.

Die Fbox *SASI* sowie ein paar Fboxen *SEND* und *RCV* sind rot. Die grünen Fboxen tauschen die Daten einwandfrei aus.



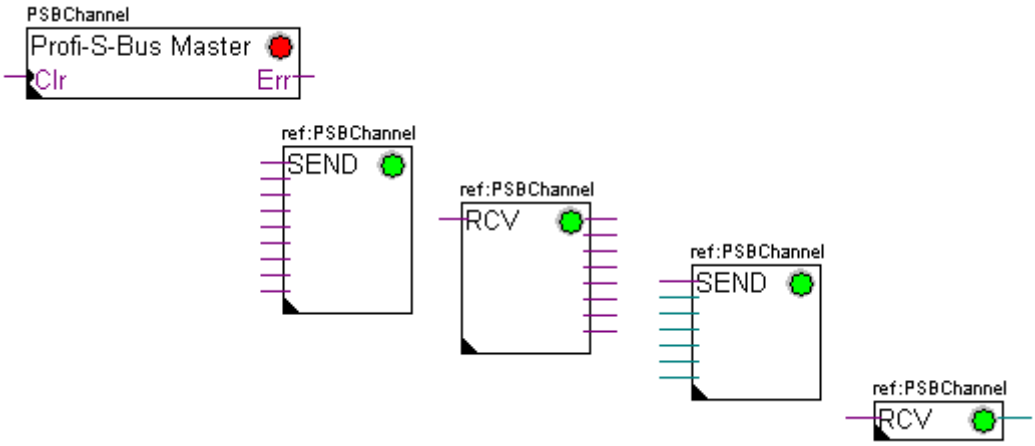
Mögliche Korrekturmaßnahmen auf der Masterstation
Parameter des Einstellfensters der Fbox *SEND* und *RCV* prüfen, deren Diagnose rot anzeigt. Prüfen, ob die Slaveadresse auch im Netzwerk vorhanden ist.

Mögliche Korrekturmaßnahmen auf der Slavestation
Für jede fehlerhafte Fbox *SEND* und *RCV* die Slavestationsnummer notieren und die entsprechenden Stationen prüfen.

- Prüfen, ob die *Device Configurator* richtig festgelegt worden sind.
- Prüfen, ob die PCD mit der erforderlichen Kommunikationshardware bestückt ist.
- Prüfen, ob die Station am Netz angeschlossen ist und unter Spannung steht.
- Netzwerkverkabelung prüfen
- Prüfen, ob die Firmwareversion Profi-S-Bus unterstützt.

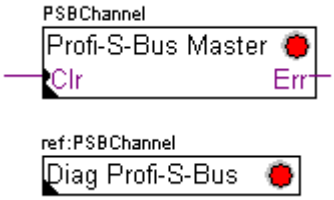
Nur die Fbox SASI leuchtet rot

Einstellungsfenster der Fbox *SASI* öffnen und den letzten Fehler über die Schaltfläche *Clear* auf Null zurückstellen.



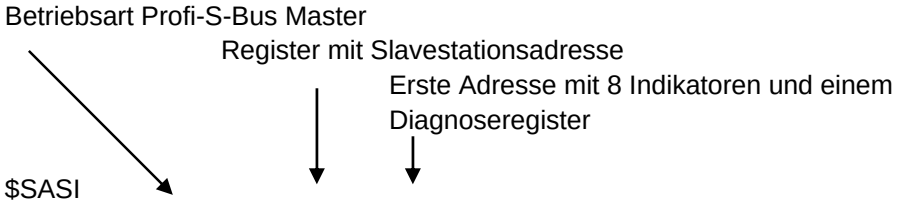
Diagnose-Fbox

Wenn die *SASI*-Anzeige rot leuchtet, dann ist es immer möglich, eine Diagnose zu erhalten, indem man sich die Funktion *SASI Diagnostic* im Einstellungsfenster ansieht. Diese Funktion muss direkt nach der *SASI*-Funktion platziert werden.



12.6 IL-Programm

12.6.1 Zuweisung des Masterkanals mithilfe einer SASI-Anweisung




```
$SASI
TEXT 10   "MODE:PSM,R0500;DIAG:F1000,R0501;"
$ENDSASI
```

```
$SASI   10        ;Adresse des Kommunikationskanals
        10        ;Adresse des Texts mit den Zuweisungsparametern.
```

Die Zuweisung eines Kanals erfolgt über eine zu Beginn des Programms platzierte SASI-Anweisung: Graftec-Initialisierungssequenz oder XOB 16-Initalisierungsblock.

Die SASI-Anweisung besteht aus zwei Parametern: Der Adresse des Kommunikationskanals und der Adresse eines Textes mit allen für den Kanal erforderlichen Parametern.

Die Parameter des Zuweisungstexts sind je nach Kommunikationsnetzwerk verschieden. Die Parameter des Zuweisungstexts sind auch für eine Slave- oder Masterstation verschieden.

Wenn die PCD mehrere Kommunikationskanäle benutzt, jeden einzelnen Kanal mithilfe einer SASI-Anweisung und einem Zuweisungstext definieren.

Je nach Netzwerk können die Kanalparameter über die *Device Configurator* vervollständigt werden.

12.6.2 Zuweisung des Slave-Kanals

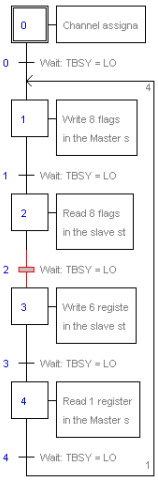
Für die Slavestation eines Profi-S-Bus-Netzes ist keine SASI-Anweisung erforderlich. Alle erforderlichen Festlegungen wurden bereits in den *Device Configurator* gemacht.

12.6.3 Prinzip eines Datenaustauschs auf einem Multimasternetzwerk.

Das Multimaster-Kommunikationsnetzwerk setzt sich aus mehreren Master- und Slavestationen zusammen. Die Masterstationen sind die Stationen, die alleine dazu berechtigt sind, die Daten anderer Master- aber auch Slavestationen zu lesen oder zu schreiben. Der Datenaustausch zwischen den Slaves ist nicht erlaubt.
Mit einem Multimaster-Kommunikationsmodus werden die über das Netzwerk ausgeführten Daten unter mehreren Mastern aufgeteilt. Ein einziger Master verfügt jedoch an einem bestimmten Moment über einen Token, der ihn dazu berechtigt, Daten mit sämtlichen Master- oder Slavestationen innerhalb des Netzwerks auszutauschen. Wenn der Master die Übertragung seiner Daten auf dem Netzwerk beendet hat, wird dieser Token einem anderen Master zugewiesen, der nun die Möglichkeit hat, Daten mit allen Master- oder Slavestationen auszutauschen. Der Token kreist automatisch durch die Masterstationen, wobei die Slaves niemals den

Token erhalten und daher auch keine Daten anderer Stationen innerhalb des Netzwerks lesen oder schreiben können.

12.6.4 Datenaustausch zwischen Master- und Slavestationen.



Anfangsetappe: Kanalzuweisung

Etappe: Datenaustausch

Übergang: Wartet auf das Ende des Datenaustauschs

Der Datenaustausch zwischen den Stationen ist ein sequenzielles Programm: Die Zuweisung des Kommunikationskanals wird nur ein einziges Mal bearbeitet, der Datenaustausch über das Netzwerk findet erst statt, wenn der vorherige Datenaustausch beendet ist. Deshalb schlagen wir vor, den Datenaustausch IL mit dem Graftec-Editor zu bearbeiten.

Die Anfangsetappe ermöglicht die Zuweisung des Kommunikationskanals bei Kaltstart des PCD.

Die anderen Etappen werden in der Schleife bearbeitet und unterstützen jede den Austausch eines Datenpakets.

Jede Etappe wird durch einen Übergang getrennt, der den TBSY-Diagnoseindikator testet und festlegt, ob der Datenaustausch beendet ist. Wir dürfen erst dann Daten über nachfolgende Etappe austauschen, wenn sich TBSY im unteren Zustand befindet.

Datenaustausch mithilfe einer Etappe

Vor dem Datenaustausch müssen wir die Adresse der Slavestation in dem hierzu über den Zuweisungstext deklarierten Register definieren.

Definition der Slavestationsadresse

```
LDL     R 500   ; Adresse des Registers mit Slavestationsadresse
        11      ; Adresse S-Bus

LDH     R 500   ; Adresse des Registers mit Slavestationsadresse
        21      ; Adresse Profi-S-Bus
```

Der Datenaustausch zwischen den Stationen wird mithilfe von zwei Anweisungen unterstützt:
STXM zum Schreiben der Daten in einer Slavestation (*SEND*)
SRXM zum Lesen der Daten einer Slavestation (*RCV*)

Jede dieser Anweisungen besteht aus vier Parametern: Kanaladresse, Anzahl der auszutauschenden Daten und Adresse der ersten Quelldaten, Bestimmungsort

Eintragen der 8 Indikatoren (F 0,..,F 7) in eine Slavestation (F 200,..,F 207)

```
STXM  10     ;Kanaladresse
      8      ;Anzahl der auszutauschenden Daten
      F 0    ;Adresse der ersten Quelldaten (lokale Station)
      F200   ;Adresse der ersten Bestimmungsdaten (lokale Station)
```

Einlesen eines Registers (R 25) einer Slavestation (R 125)

```
SRXM  10     ;Kanaladresse
      1      ;Anzahl der auszutauschenden Daten
      R 25   ;Adresse der ersten Quelldaten (lSlavestation)
      R 125  ;Adresse der ersten Bestimmungsdaten (lokale Station)
```

Anmerkung:
Nur die Masterstationen werden mit STXM und SRXM programmiert! Die Slavestationen dürfen nur mit dem Kommunikationskanal zugewiesen werden.

Warten auf das Übertragungsende mithilfe einer Transition

```
STL    F 1003  ; Prüft, ob sich TBSY im unteren Zustand befindet
```

Der Zuweisungstext legt einen Bereich mit 8 Diagnoseindikatoren zur Kommunikation fest, wobei der dritte in den oberen Datenübertragungszustand versetzt wird und in den unteren fällt, wenn der Austausch beendet ist.

12.6.5 Diagnosen

Kanalzuweisungen
Bei einem Kommunikationsproblem prüfen, ob die Kanalzuweisung richtig durchgeführt worden ist. Das Programm in der Betriebsart Step-by-Step bearbeiten und prüfen, ob die SASI-Anweisung kein Fehlerflag enthält. Ansonsten kann die Kanalzuweisung nicht richtig durchgeführt werden und die Kommunikation funktioniert nicht.

Mögliche Korrekturmaßnahmen auf der Master- oder Slavestation:

- *Device Configurator* prüfen.
- Prüfen, ob die *Device Configurator* in die PCD heruntergeladen worden sind.
- Prüfen, ob die Stationen alle dasselbe Profil verwenden. S-Net, DP
- Prüfen, ob alle Stationen mit derselben Geschwindigkeit kommunizieren.
- Prüfen, ob der mit den *Device Configurator* festgelegte Kommunikationskanal und die *SASI*-Anweisung identisch sind (dieselbe Kanalnummer)
- Prüfen, ob die PCD mit der erforderlichen Kommunikationshardware bestückt ist.
- Prüfen, ob die Stationen am Netz angeschlossen sind und unter Spannung stehen.
- Netzwerkverkabelung prüfen
- Prüfen, ob die Firmwareversion Profi-S-Bus unterstützt.

Die Daten werden nicht über das Netzwerk ausgetauscht

Der Zuweisungstext definiert einen Bereich mit 8 Diagnoseindikatoren zur Kommunikation, wobei der fünfte (*TDIA :Transmitter diagnostic*) bei einem Datenübertragungsfehler in den oberen Zustand versetzt wird. Der Step-by-Step-Test des Kommunikationsprogramms ermöglicht Ihnen festzustellen, welche der STXM- und SRXM-Anweisungen einen Fehler enthalten.

Vorsicht! Wenn sich ein Kommunikationsfehler ereignet, dann bleibt das TDIA-Diagnoseflag ganz oben, während das Diagnoseregister nicht wieder auf Null zurückgestellt wird.

Mögliche Korrekturmaßnahmen auf der Masterstation
Prüfen, ob die Parameter der STXM- und SRXM-Anweisungen fehlerhaft sind.
Prüfen, ob die Slaveadresse auch im Netzwerk vorhanden ist.

Mögliche Korrekturmaßnahmen auf der Slavestation
Für jede fehlerhafte STXM- und SRXM-Anweisung die Slavestationsnummer notieren und die entsprechenden Stationen prüfen.

- Prüfen, ob die *Device Configurator* richtig festgelegt worden sind.
- Prüfen, ob die PCD mit der erforderlichen Kommunikationshardware bestückt ist.
- Prüfen, ob die Station am Netz angeschlossen ist und unter Spannung steht.
- Netzwerkverkabelung prüfen
- Prüfen, ob die Firmwareversion Profi-S-Bus unterstützt.

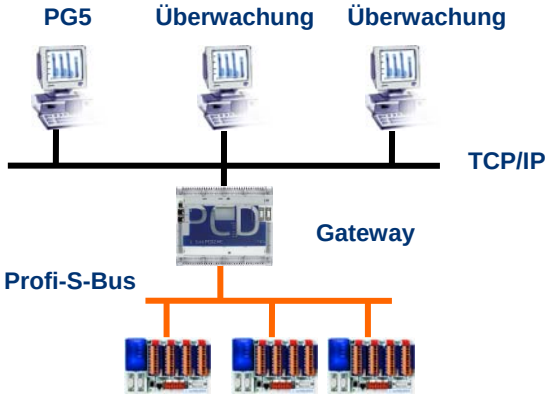
Diagnoseregister

Das Diagnoseregister kann mehr Informationen zur Art des Kommunikationsfehlers geben. Lassen Sie den Binärinhalt des Registers anzeigen und vergleichen Sie ihn mit den Beschreibungen in der Betriebsanleitung der PCD oder des Kommunikationsnetzwerks.

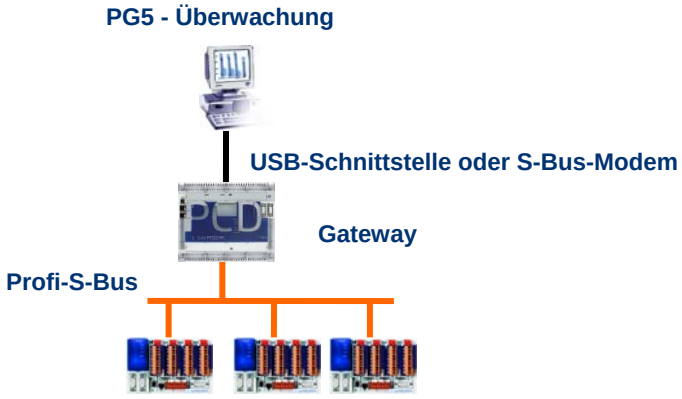
12.7 Gateway-Funktion

Die *Gateway*-Funktion wird häufig verwendet, damit zwei verschiedene Kommunikationssysteme zusammen kommunizieren können oder zum Anpassen eines PG5-Programmierungssystems, eine Saia PG5 Visi.Plus-Überwachung auf einem anderen als das direkt von diesen Geräten unterstützten Netzwerks.

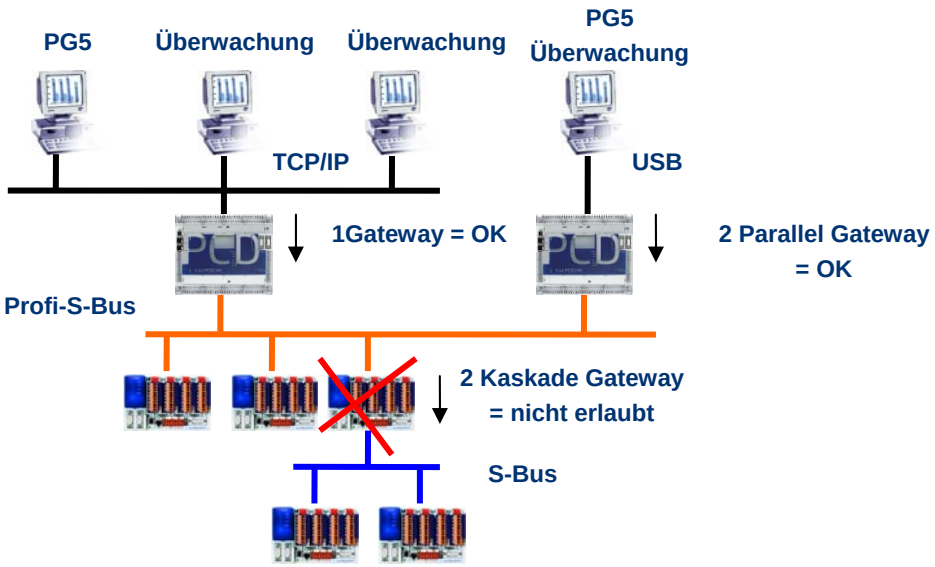
12.7.1 Anwendung



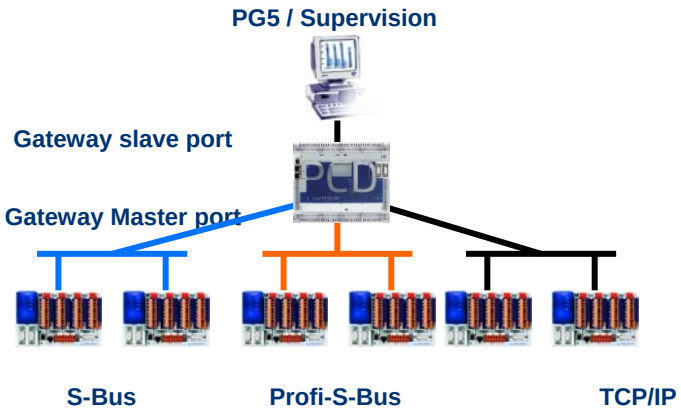
Die *Gateway*-Funktion ermöglicht die Durchführung einer Brücke zwischen zwei Netzwerken. Beispielsweise ein Ethernet-Netzwerk mit dem Profi-S-Bus-Netzwerk verbinden. Auf diese Weise tauschen die PCD-Systeme die Daten auf einem Bus aus dem Bereich Automatisierung aus, der vom EDV-Netz des Unternehmens getrennt ist. Aber die mit der PG5—Software oder einem Saia PG5 Visi.Plus Überwachungssystem bestückten Computer können Daten mit verschiedenen PCD austauschen.



Die Funktion *Gateway* kann als Schnittstelle zwischen dem Kommunikationsnetzwerk und der Außenwelt dienen. Beispielsweise, eine Kommunikation über Modem herstellen oder eine USB-Kommunikationsschnittstelle.



Um die Timingvorgaben der Kommunikation zu respektieren, können nicht zwei kaskadische Gateways definiert werden. Aber es ist möglich, zwei parallele Gateways im gleichen Netz zu definieren.



Falls erforderlich, kann die Funktion Gateway eine Verbindung zu verschiedenen Unterkommunikationsnetzen herstellen.

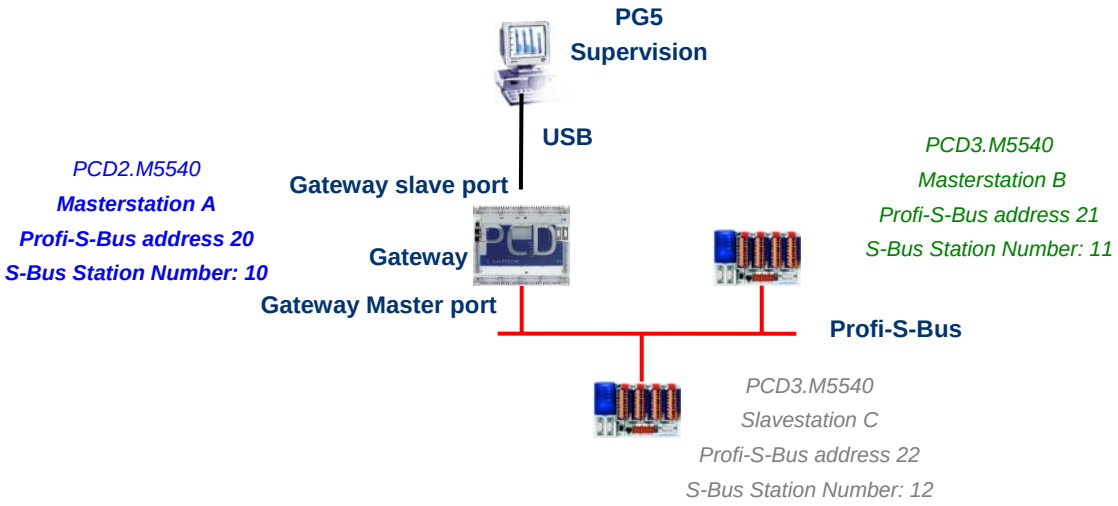
12.7.2 Konfiguration der Gateway PGU-Funktion

Die *Gateway*-Funktion lässt sich sehr leicht einrichten, sie benötigt kein Programm, sondern nur ein paar Parameter in den *Device Configurator* der PCD.

Im Allgemeinen muss man nur einen *Gateway Slave Port* und einen *Gateway Master Port* konfigurieren, anschließend wird alles automatisch von der *Gateway*-Funktion unterstützt.

Wenn sich die Daten auf dem *Gateway Slave Port* befinden und sie nicht die lokale Station (*Gateway*) betreffen, werden die Daten anhand der gültigen, für jedes Unternetzwerk definierten Adressen, auf eines der auf dem *Gateway Master Port* angeschlossenen Unternetzwerken übertragen.

Beispiel: Gateway USB; Profi-S-Bus

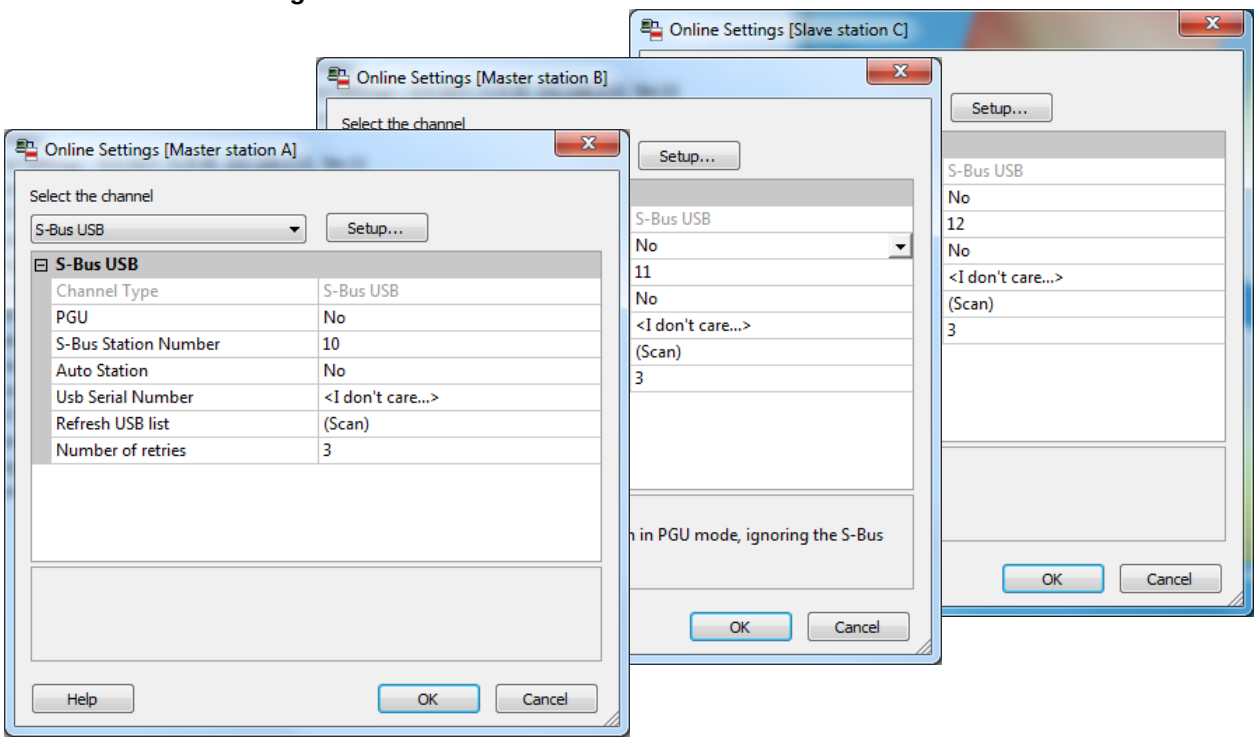


Onboard Communication, properties **der Masterstation A**

Profi-S-Bus Master Gateway	
Use Profi-S-Bus For Gateway	**Yes**
First S-Bus Station Profi-S-Bus	0
Last S-Bus Station Profi-S-Bus	253
Response Timeout	0

Der Gateway USB ist eine Ausnahme und erfordert keinen Parameter für den *Gateway Slave port*, nur der *Gateway Master port* muss parametriert werden. (Nicht vergessen, die neue Konfiguration in den Master A herunterzuladen!)

Online Settings der Device



Zum Erstellen der USB-Kommunikation mit jeder PCD muss man noch die *Online Settings* jeder Device des Projekts mit dem USB-Kanal und der Nummer der S-Bus-Station angleichen.

Testen des einwandfreien Betriebs der Gateway-Funktion

Slave station C - PCD3.M5540 - Station 12

Aktivieren eines der Device, *Master B* oder *Slave C*, des Projekts und zum Testen der Kommunikation mit der Station online gehen.

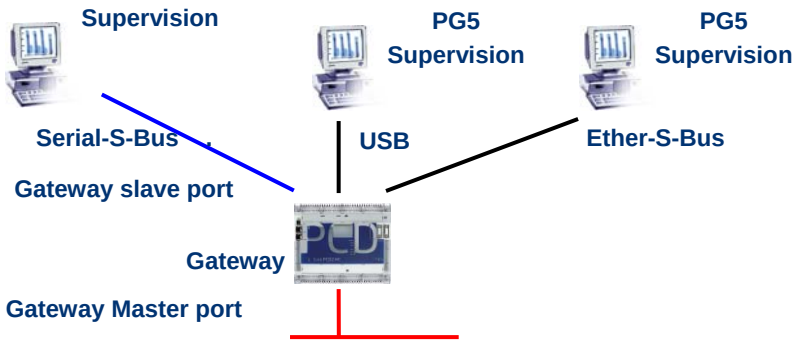
Falls erforderlich, kann man über den *Online Configurator* die Nummer der Station prüfen, die online ist. Man kann somit das Programm in der aktiven Device aufladen und testen und gleichzeitig mit dem USB-Kabel mit der Station *Master A* verbunden bleiben.

Master station B - PCD3.M5540 - Station 11

Um mit einer anderen Netzwerkstation zu kommunizieren, die Device aktivieren und online gehen.

Anmerkung:
Mit der *Gateway*-Funktion wird nur die Station S-Bus Slave definiert, die Nummer der Profi-S-Bus-Station wird nicht berücksichtigt, weil die Telegramme an alle anderen Profi-S-Bus (Broadcast) Stationen adressiert sind.

12.7.3 Konfiguration eines zusätzlichen Slave-Gateway-Slave-Port

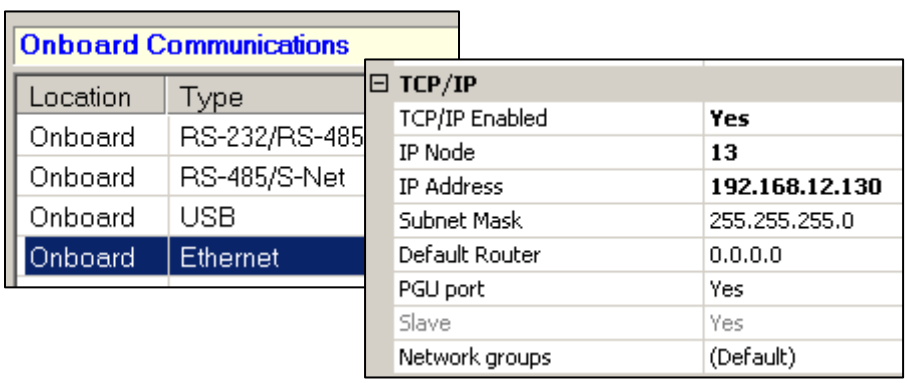


Der Gateway Slave port ist ein Mittel, um von außen in das Netzwerk zu gelangen. Falls erforderlich, kann ein zweiter oder ein dritter *Gateway Slave port* definiert werden.

***Device Configurator* Parameter**

Im Allgemeinen unterstützt die PCD nur einen einzigen PGU-Slavekanal. Aber die neuen PCD2 und PCD3.Mxxxx Steuerungen können mehrere auf demselben PCD unterstützen. Die Konfiguration des zweiten Gateway Slave PGU wird vollständig von den *Device Configurator.* unterstützt.

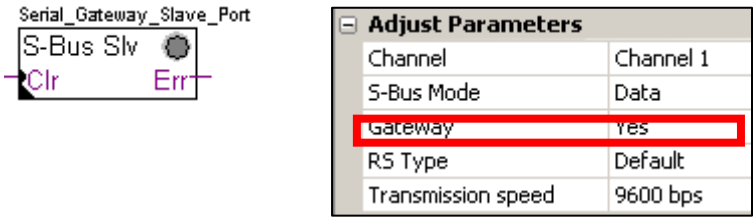
Beispiel: Hinzufügen eines zweiten Gateway Ether-S-Bus, Profi-S-Bus



Der zweite *Gateway Slave port PGU* wird durch Konfigurieren der *Device Configurator* mit dem Knoten und der TCP/IP-Adresse hinzugefügt.

Fupla- oder IL-Programm
Es ist auch möglich, eine zusätzliche Fbox/SASI-Anweisung zu verwenden, um einen zweiten *Gateway Slave port* hinzuzufügen.
Diesen *Gateway slave port*, ohne PGU-Funktionalität, unterstützt das PG5-Programmierungstool nicht, dafür aber Terminal- oder ein Überwachungsfunktionen. Das Lesen und Schreiben der PCD-Daten wird nicht unterstützt: Register, Indikatoren, …

Fupla-Beispiel: Hinzufügen eines dritten seriellen S-Bus, Profi-S-Bus



Der *Gateway*-Einstellungsparameter muss daher mit der Option *Yes* definiert werden. Je nach Kanaltyp müssen die anderen Parameter im Einstellungsfenster ebenso korrekt definiert werden.

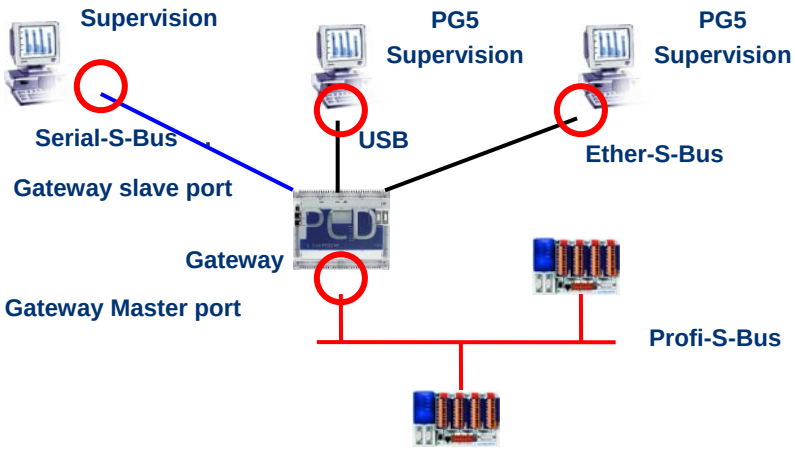
Beispiel I: Hinzufügen eines dritten seriellen S-Bus, Profi-S-Bus

Benutzen Sie nachfolgenden Text zur Zuweisung des Kanals:

```
$SASI
TEXT 11   "UART:9600;MODE:GS2;DIAG:F1110,R0501;"
$ENDSASI
```

Indikator und Diagnoseregister
Mode S-Bus Gateway Slave Data mode
Übertragungsgeschwindigkeit

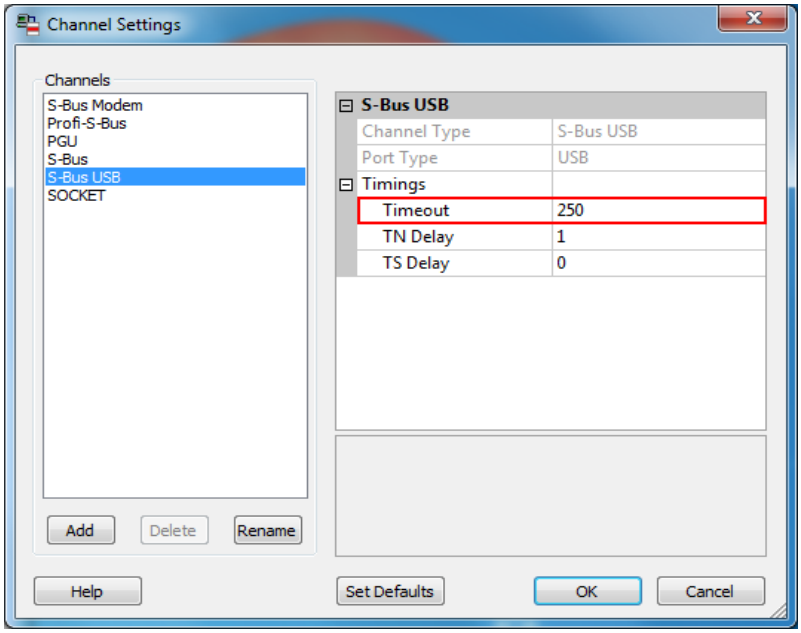
12.7.4 Kommunikationstimings



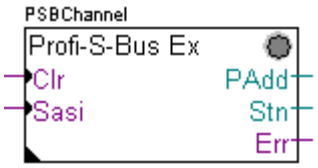
Im Allgemeinen werden die Kommunikations-*timings* mit den Standardwerten definiert und dies funktioniert einwandfrei. Aber die Benutzung der *Gateway*-Funktion erhöht die zum Datenaustausch erforderliche Reaktionszeit. Es ist daher manchmal erforderlich, das Timeout der Masterstationen mit Hilfe des *Gateway* anzupassen.

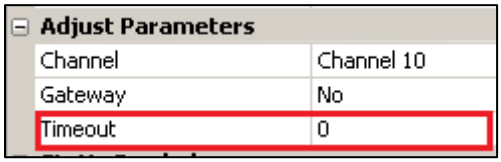
Nachstehende Abbildung zeigt die Masterkanäle, deren Timeout eingestellt werden müsste.

Zum Einstellen des *Timeout* des PG5 die *Online Settings* der *Master Station A* benutzen:



Zum Einstellen des *Timeout* des Datenaustauschprogramms auf dem PCD-*Gateway* die Fbox benutzen: *SASI Profi-S-Bus Extended*





12.8 Andere Referenzen

Weitere Informationen können Sie in folgenden Handbüchern finden:

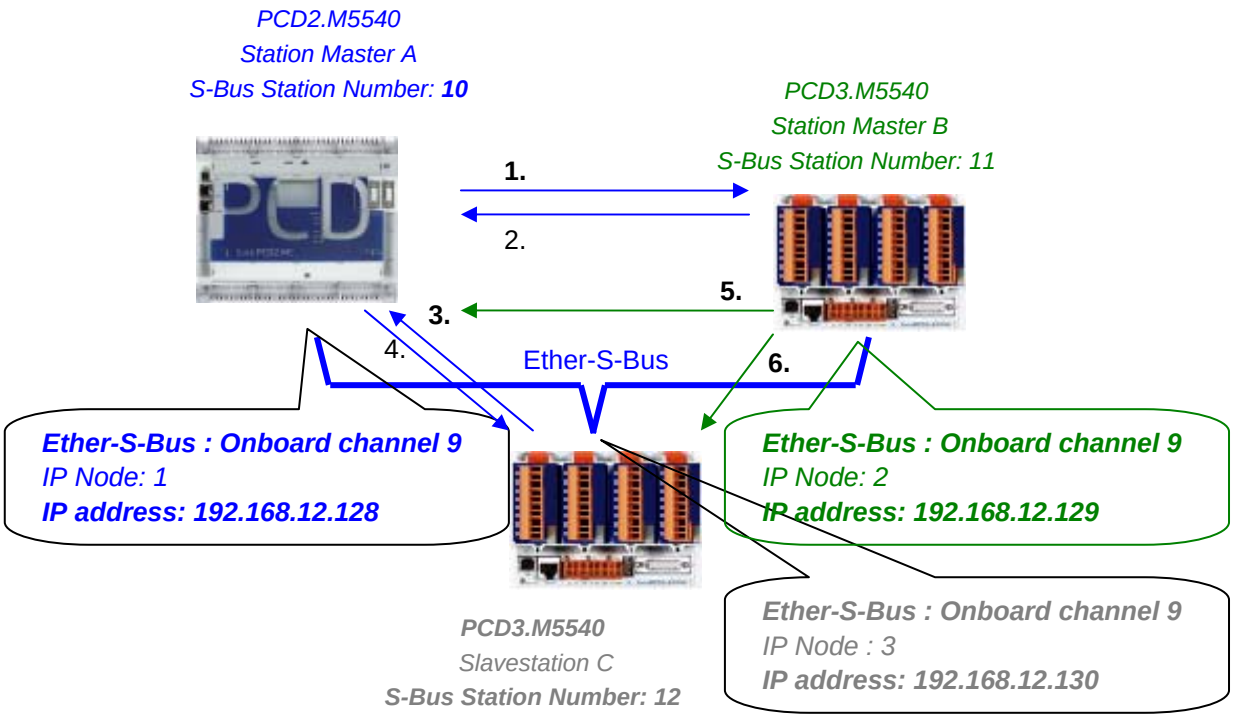
- Leitfaden zu den Anweisungen 26/133
- Profi-S-Bus (in Vorbereitung)
- Beispiel des Profi-S-Bus-Projekts, das mit Ihrem PG5 installiert ist.

Inhaltsverzeichnis

13 Ether-S-Bus

Dieses Beispiel zeigt, wie man ein paar Daten, wie beispielsweise Register und Indikatoren zwischen den am Ether-S-Bus-Netz angeschlossenen PCD austauscht.

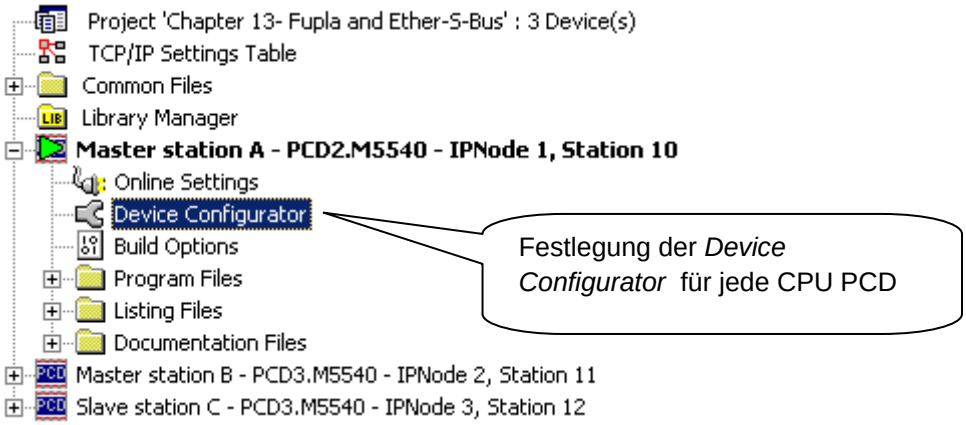
13.1 Beispiel eines Ether-S-Bus-Netzes.



13.2 Beispiele eines Ether-S-Bus-Datenaustausch.

	Master with data exchanges	***Data on the network***	***Passive master or slave***
	Master station A		***Master station B***
1	*Blinker0 .. 7* *F 0 .. 7*	*Write 8 flags in the Master station B*	*Station_A.Blinker0 .. 7* *F 100 .. 107*
2	*Master_B .Value100* *R 125*	*Read 1 register in the Master station B*	*Value100* *R 25*
			Slave station C
3	*Slave_C.Binary0 .. 7* *F 100 .. 107*	*Read 8 flags in the slave station C*	*Binary0 .. 7* *F 20 .. 27*
4	*Value0 .. 5* *R 0 .. 5*	*Write 6 registers in the slave station C*	*Master_A. Value0 .. 5* *R 20 .. 25*
	Master station B		***Master station A***
5	*Temperature1 .. 4* *Dynamic registers*	*Write the temperature measures to the slave C*	*Master_B.Temperature1 .. 4* *R 100 .. 104*
			Slave station C
6	*Temperature1 .. 4* *Dynamic registers*	*Write the temperature measures to the master A*	*Master_B.Temperature1 .. 4* *R 100 .. 104*

13.3 Projekt PG5



Der *Saia PG5 Project Manager* gibt einen Überblick über die PCD eines Applikationsprojektes sowie die Parameter für die Kommunikationsnetze. Beginnen wir mit dem Hinzufügen einer device in das Projekt für jede auf dem Netzwerk verfügbare Station.

13.4 Device Configurator

Die Konfigurationen der *Device Configurator* einer Masterstation und der Slaves sind fast gleich.

13.4.1 Festlegen der PCD-Parameter

Device

Type	Description
PCD2.M5540	CPU with 1M Bytes RAM, 8 I/O slots, 2 communication slot.

PCD-Type

Festlegen des Device-Typs

13.4.2 Festlegen der S-Bus-Stationsnummer auf dem Netzwerk

S-Bus	
S-Bus Support	Yes
Station Number	**10**

Device properties:

Station Number
Die S-Bus-Stationsnummer ist für alle Kommunikationskanäle des PCD gleich.

13.4.3 Festlegen des Ether-S-Bus-Kommunikationskanals

Onboard Communications

Location	Type
Onboard	RS-232/RS-485
Onboard	RS-485/S-Net
Onboard	USB
Onboard	Ethernet

TCP/IP	
TCP/IP Enabled	**Yes**
IP Node	**1**
IP Address	**192.168.12.128**
Subnet Mask	255.255.255.0
Default Router	0.0.0.0
PGU port	Yes
Slave	Yes
Network groups	(Default)

Onboard Communication, properties:

IP Node
Nummer des TCP/IP-Knotens. Der Knoten wird in den Fbox SEND und RCV benutzt, um eine Slavestation zu definieren, mit der die Daten ausgetauscht werden.

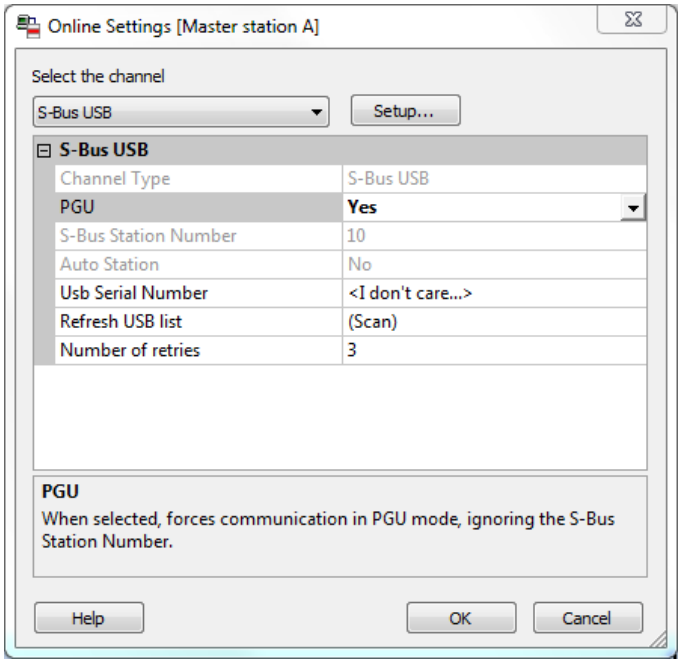
IP Address
Die mit dem Kanal verbundene Ether-S-Bus-Stationsnummer

PGU Port
Festlegen des Kanals als Slave oder PGU. Diese Festlegung kann mit der Masterfunktion zusammen erfolgen, indem man eine Fbox SASI Master in das Fupla-Programm aufnimmt.

PGU + Slave
Unterstützt den Datenaustausch mit den Masterstationen, den Überwachungssystemen und Terminals. Unterstützt aber auch das Hilfssystem zur Programmierung und Inbetriebnahme von PG5.

Slave
Unterstützt nur den Datenaustausch mit den Masterstationen, den Überwachungssystemen und Terminals.

13.4.4 Laden der Device Configurator Parameter auf den Device

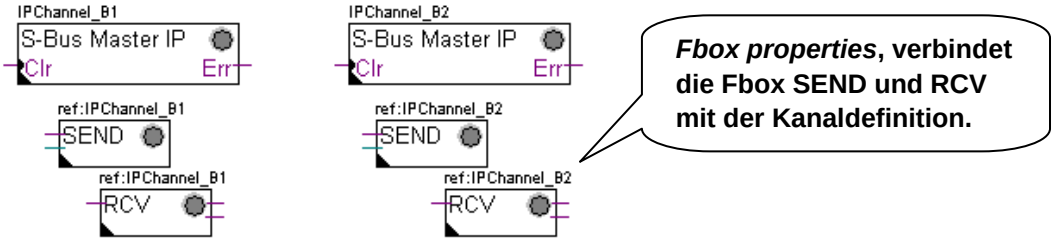


Mit den neuen Systemen PCD2 und PCD3, können die *Device Configurator Settings* über die USB-Schnittstelle heruntergeladen werden. Man muss nur prüfen, ob die *Online Settings* mit einem *Profi-S-Bus + PGU*-Kanal definiert worden sind.

Die Parameter, über die Schaltfläche *Download* im Fenster *Device Configurator*, in die PCD herunterladen.

13.5 Fupla-Programm

13.5.1 Zuweisung des Kanals mithilfe einer Fbox SASI



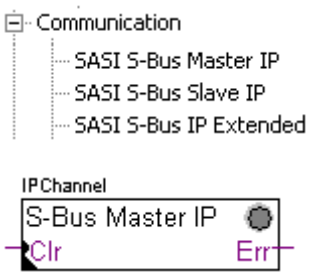
Die Zuweisung eines Kanals erfolgt über eine zu Beginn der Fupla-Datei platzierte FBox SASI. Jedes Kommunikationsnetz verfügt über seine eigene Fbox SASI, weil sich die Zuweisungsparameter von einem zum anderen Netz voneinander unterscheiden. Dasselbe gilt für eine Slave- oder Masterstation.

Wenn die PCD mehrere Kommunikationskanäle benutzt, sollte jeder einzelne Kanal über die entsprechende Fbox SASI definiert werden. Anschließend den Cursor mit der Maus auf die Fbox SASI setzen, das Kontextmenü markieren und für jede Fbox, je nach benutztem Kanal, den Parameter *Name* anders festlegen. Dieser Name ermöglicht es, verschiedene Fboxen zum Austauschen von *SEND* und *RCV* mit der dem benutzten Kanal entsprechenden Fbox SASI zu verbinden.

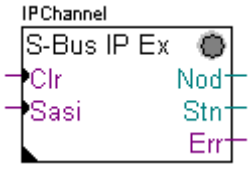
Je nach Netzwerk, können die Kommunikationsparameter teilweise im Einstellungsfenster der Fbox SASI festgelegt und in den *Device Configurator* vervollständigt werden.

Aber die Nummer des Kommunikationskanals wird immer über das Einstellungsfenster der FBox SASI festgelegt. Die Kanalnummer hängt von der PCD-Hardware und der verwendeten Kommunikationshardware ab : Slot B1, B2, serielle Schnittstelle PCD7.F, …

13.5.2 Zuweisung des Masterkanals



Masterstation



Master/-Slavestation
+ Definition der Timings

Die Zuweisung des Masterkanals erfolgt über die Vervollständigung der *Device Configurator* mit einer der obigen Fboxen.

Properties Parameters des Einstellungsfensters:

Kanal
Legt die Nummer des am Netzwerk angeschlossenen Kanals fest. Hängt von der PCD und ihrer Hardware ab.

Timing
Das Timeout wird allgemein mit dem Standardwert (0) festgelegt und wird nur für ganz bestimmte Applikationen (Gateway) verändert.

13.5.3 Zuweisung des Slavekanals

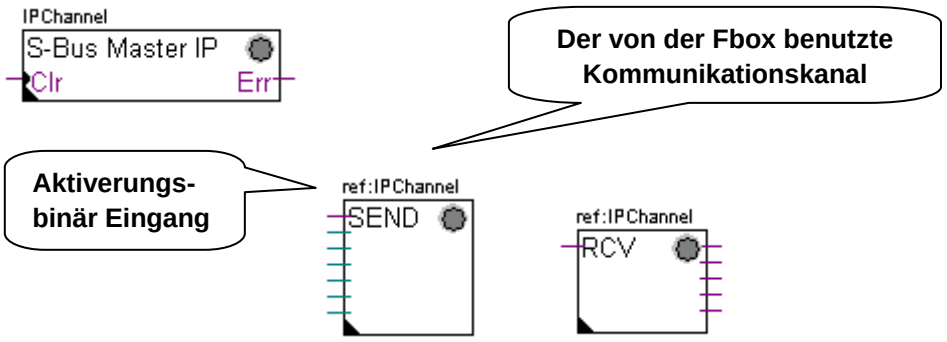
Für die Slavestation eines Ether-S-Bus-Netzes ist keine FBox SASI erforderlich. Alle erforderlichen Festlegungen wurden bereits in den *Device Configurator* gemacht.

13.5.4 Prinzip eines Datenaustauschs auf einem Multimasternetzwerk.

Das Multimaster-Kommunikationsnetzwerk setzt sich aus mehreren Master- und Slavestationen zusammen. Die Masterstationen sind die Stationen, die alleine dazu berechtigt sind, die Daten anderer Master- aber auch Slavestationen zu lesen oder zu schreiben. Der Datenaustausch zwischen den Slaves ist nicht erlaubt.
Mit einem Multimaster-Kommunikationsmodus werden die über das Netzwerk ausgeführten Daten unter mehreren Mastern aufgeteilt. Ein einziger Master verfügt jedoch an einem bestimmten Moment über einen Token, der ihn dazu berechtigt, Daten mit sämtlichen Master- oder Slavestationen innerhalb des Netzwerks

auszutauschen. Wenn der Master die Übertragung seiner Daten auf dem Netzwerk beendet hat, wird dieser Token einem anderen Master zugewiesen, der nun die Möglichkeit hat, Daten mit allen Master- oder Slavestationen auszutauschen. Der Token kreist automatisch durch die Masterstationen, wobei die Slaves niemals den Token erhalten und daher auch keine Daten anderer Stationen innerhalb des Netzwerks lesen oder schreiben können.

13.5.5 Datenaustausch zwischen Master- und Slavestationen.



Der Benutzer kontrolliert den Datenaustausch zwischen den Stationen mithilfe verschiedener Fupla-Fboxen, die in den Fupla-Seiten platziert und in dem *Fbox Selector* zur Verfügung stehen. Wir finden hier Fboxen zum Schreiben (*SEND*) oder zum Lesen (*RCV*) eines Datenpaketes, aber auch zur Unterstützung verschiedener Datenformate: Binär, Ganzzahl, gleitend, Datenblöcke, usw.

Die Fboxen *SEND* oder *RCV* können mit mehr oder weniger Ein- und Ausgängen ausgedehnt werden, was ihr ermöglicht, die Größe des mit einer anderen Station auszutauschenden Datenpakets zu definieren.

Die Adresse des von der Datenübertragungs-Fbox verwendeten Kommunikationskanals, wird über das oben links von der Fbox angezeigte Symbol definiert und verbindet sie mit der Fbox SASI desselben Namens, in der die Kanaladresse festgelegt ist. Dieses Symbol kann bearbeitet werden, indem man den Cursor auf die Fbox setzt und das Kontextmenü *properties* markiert.

Jede Fbox *SEND* und *RCV* ist mit einem binären Eingang für den Datenaustausch ausgestattet. Wenn sich dieser Eingang ständig im Zustand “high” befindet, dann wird der Datenaustausch so schnell wie möglich wiederholt. Wenn ein kurzer Impuls auf diesen Eingang gebracht wird, dann erfolgt der Datenaustausch mindestens einmal, es ist aber immer möglich, diesen über die Schaltfläche *Execute* oder beim Kaltstart des PCD mit der Option *Initialization* im Einstellungsfenster zu forcieren.

Die auf den Eingängen einer Fbox *SEND* vorhandenen Daten der Masterstation, werden zu der im Einstellungsfenster festgelegten Slavestation geschickt. Die auf den Ausgängen einer Fbox *RCV* vorhandenen Daten stammen aus der Slavestation anhand der Parameter im Einstellungsfenster: Adresse der Slavestation, Quellelemente und Basisadresse.

Nur die Masterstationen werden über die Fbox *SEND* und *RCV* programmiert! Die Slavestationen dürfen nur mit dem Kommunikationskanal zugewiesen werden.

Je nach verwendeter Fbox kann man im Einstellungsfenster festlegen, zu welchen Slavestationen die Daten der Masterstation (*SEND*) geschickt werden oder in welchen Slavestationen der Master liest. (*RCV*)

Properties Parameters des Einstellungsfensters:

IP Node
Festlegung der Knotennummer Ether-S-Bus Bus Slave

Source, destination station
Festlegung der Stationsnummer S-Bus Bus Slave

Source, destination element
Festlegung des im Slave zu schreibenden oder zu lesenden Datentyps.

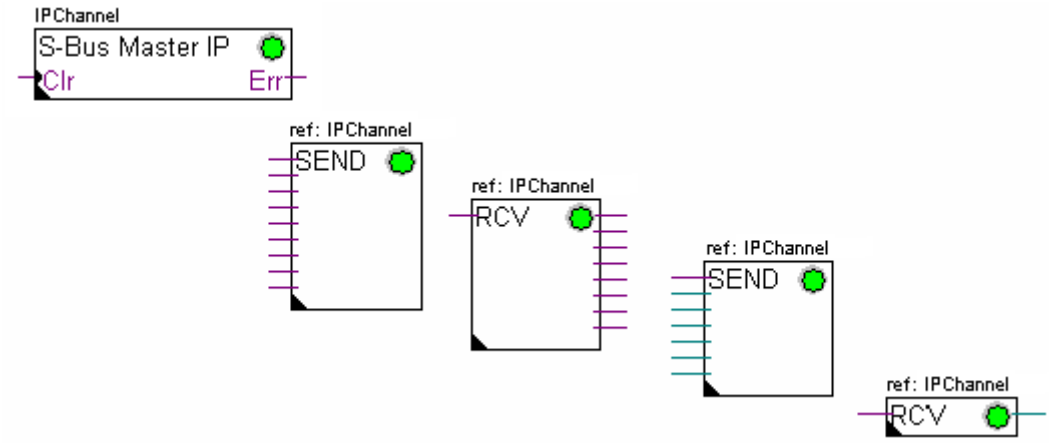
Source, destination address
Festlegung der Adresse des ersten im Slave zu schreibenden oder zu lesenden Datensatzes.
Die Anzahl der ausgetauschten Daten hängt von der Anzahl der Ein- oder Ausgänge der Fbox *SEND*, *RCV* ab.

13.5.6 Diagnosen

Wenn das Programm online ist, wird eine grüne oder rote LED oben rechts der Fbox *SASI*, *SEND* und *RCV* angezeigt. Grün bedeutet, dass die Datenübertragung OK ist, rot zeigt einen Fehler an.

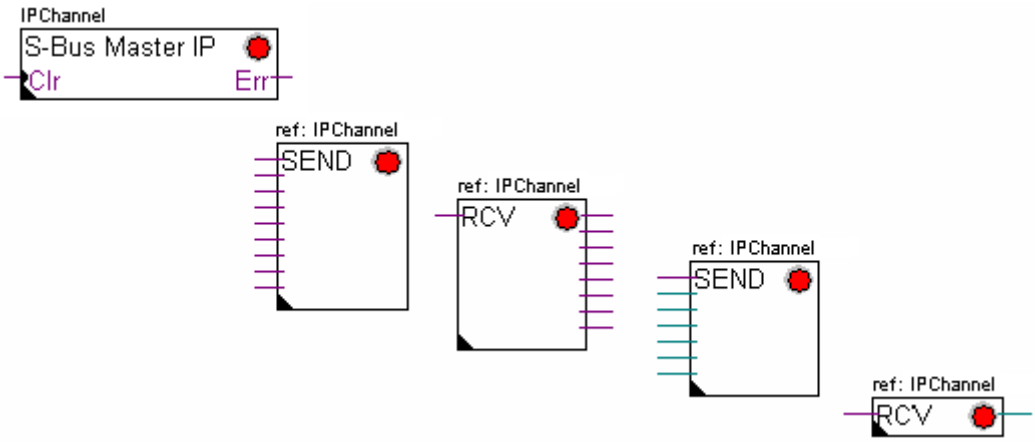
Einwandfreie Funktion

Alle Fboxen sind grün, der Datenaustausch funktioniert richtig.



Es können keine Daten über das Netzwerk ausgetauscht werden.

Die Fboxen *SASI* , *SEND* und *RCV* sind rot, es können keine Daten über das Netzwerk ausgetauscht werden.

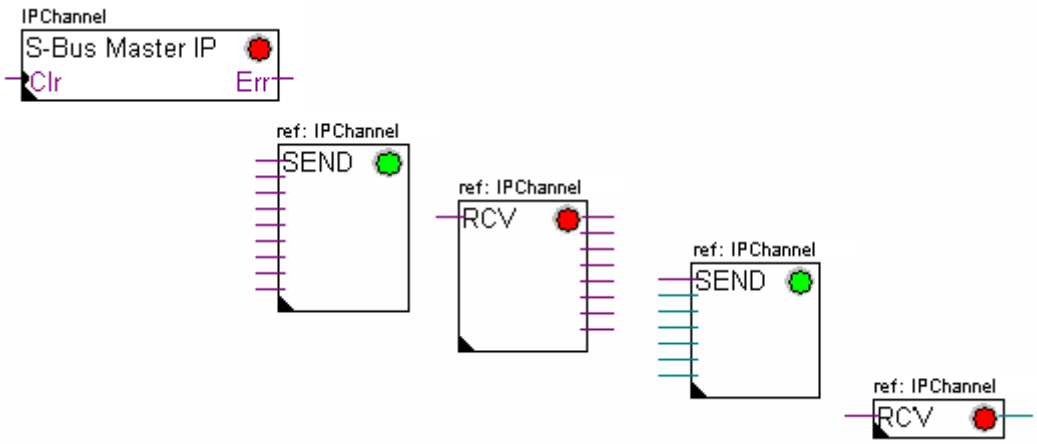


Mögliche Korrekturmaßnahmen auf der Master- oder Slavestation:

- *Device Configurator* prüfen.
- Prüfen, ob die *Device Configurator* in die PCD heruntergeladen worden sind.
- Prüfen, ob der mit den *Device Configurator* festgelegte Kommunikationskanal und die *SASI*-Funktion identisch sind (dieselbe Kanalnummer)
- Prüfen, ob die PCD mit der erforderlichen Kommunikationshardware bestückt ist.
- Prüfen, ob die Stationen am Netz angeschlossen sind und unter Spannung stehen.
- Netzwerkverkabelung prüfen
- Prüfen, ob die Firmwareversion Ether-S-Bus unterstützt.

Einige Daten können nicht über das Netzwerk ausgetauscht werden.

Die Fbox *SASI* sowie ein paar Fbox *SEND* und *RCV* sind rot. Die grünen Fbox tauschen einwandfrei die Daten aus.



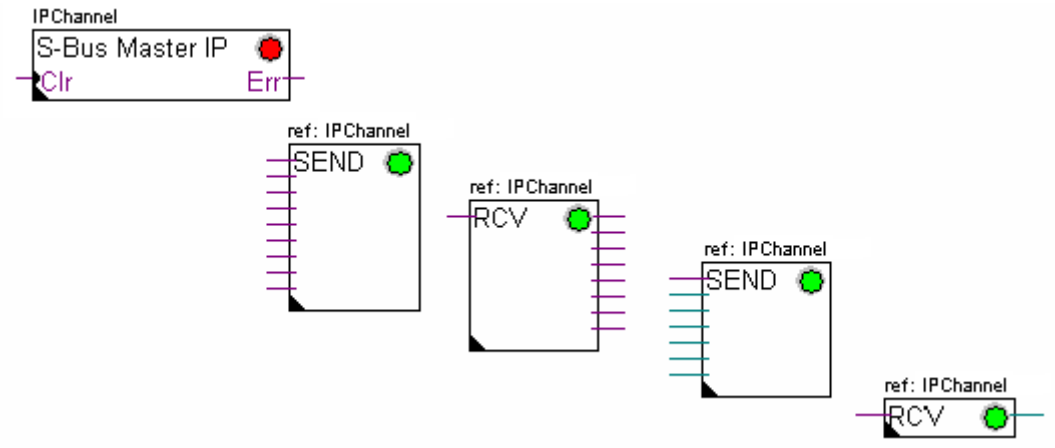
Mögliche Korrekturmaßnahmen auf der Masterstation
Parameter des Einstellfensters der Fbox *SEND* und *RCV* prüfen, deren Diagnose rot anzeigt. Prüfen, ob die Slaveadresse auch im Netzwerk vorhanden ist.

Mögliche Korrekturmaßnahmen auf der Slavestation
Für jede fehlerhafte Fbox *SEND* und *RCV* die Slavestationsnummer notieren und die entsprechenden Stationen prüfen.

- Prüfen, ob die *Device Configurator* richtig festgelegt worden sind.
- Prüfen, ob die PCD mit der erforderlichen Kommunikationshardware bestückt ist.
- Prüfen, ob die Station am Netz angeschlossen ist und unter Spannung steht.
- Netzwerkverkabelung prüfen
- Prüfen, ob die Firmwareversion Ether-S-Bus unterstützt.

Nur die Fbox SASI leuchtet rot

Einstellungsfenster der Fbox *SASI* öffnen und den letzten Fehler über die Schaltfläche *Clear* auf Null zurückstellen.

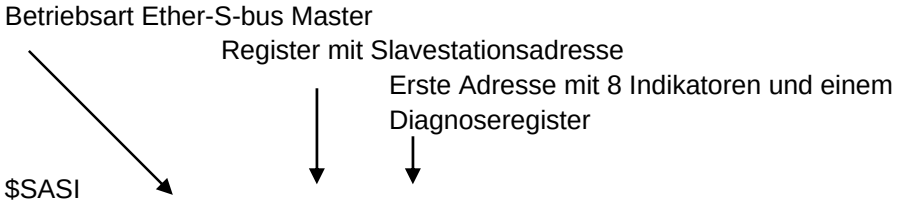


Diagnose-Fbox

Wenn die *SASI*-Anzeige rot leuchtet, dann ist es immer möglich, eine Diagnose zu erhalten, indem man sich die Funktion *SASI Diagnostic* im Einstellungsfenster ansieht. Diese Funktion muss direkt nach der *SASI*-Funktion platziert werden.

13.6 IL-Programm

13.6.1 Zuweisung des Masterkanals mithilfe einer SASI-Anweisung




```
$SASI
TEXT 10   "MODE:EM,R0500;DIAG:F1000,R0501;"
$ENDSASI
```

```
$SASI          8              ;Adresse des Kommunikationskanals
       10      ;Adresse des Texts mit den Zuweisungsparametern.
```

Die Zuweisung eines Kanals erfolgt über eine zu Beginn des Programms platzierte SASI-Anweisung: Graftec-Initialisierungssequenz oder XOB 16-Initalisierungsblock.

Die SASI-Anweisung besteht aus zwei Parametern: Der Adresse des Kommunikationskanals und der Adresse eines Textes mit allen für den Kanal erforderlichen Parametern.

Die Parameter des Zuweisungstexts sind je nach Kommunikationsnetzwerk verschieden. Die Parameter des Zuweisungstexts sind auch für eine Slave- oder Masterstation verschieden.

Wenn die PCD mehrere Kommunikationskanäle benutzt, jeden einzelnen Kanal mithilfe einer SASI-Anweisung und einem Zuweisungstext definieren.

Je nach Netzwerk können die Kanalparameter über die *Device Configurator* vervollständigt werden.

13.6.2 Zuweisung des Slavekanals

Für die Slavestation eines Ether-S-Bus-Netzes ist keine SASI-Anweisung erforderlich. Alle erforderlichen Festlegungen wurden bereits in den *Device Configurator* gemacht.

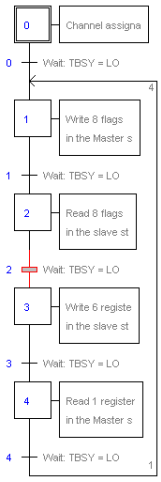
13.6.3 Prinzip eines Datenaustauschs auf einem Multimasternetzwerk.

Das Multimaster-Kommunikationsnetzwerk setzt sich aus mehreren Master- und Slavestationen zusammen. Die Masterstationen sind die Stationen, die alleine dazu berechtigt sind, die Daten anderer Master- aber auch Slavestationen zu lesen oder zu schreiben. Der Datenaustausch zwischen den Slaves ist nicht erlaubt.
Mit einem Multimaster-Kommunikationsmodus werden die über das Netzwerk ausgeführten Daten unter mehreren Mastern aufgeteilt. Ein einziger Master verfügt jedoch an einem bestimmten Moment über einen Token, der ihn dazu berechtigt, Daten mit sämtlichen Master- oder Slavestationen innerhalb des Netzwerks auszutauschen. Wenn der Master die Übertragung seiner Daten auf dem Netzwerk beendet hat, wird dieser Token einem anderen Master zugewiesen, der nun die Möglichkeit hat, Daten mit allen Master- oder Slavestationen auszutauschen. Der Token kreist automatisch durch die Masterstationen, wobei die Slaves niemals den

Token erhalten und daher auch keine Daten anderer Stationen innerhalb des Netzwerks lesen oder schreiben können.

13.6.4 Datenaustausch zwischen Master- und Slavestationen.

Datenaustauschs



Anfangsetappe: Kanalzuweisung

Etappe: Datenaustausch

Übergang: Wartet auf das Ende des

Der Datenaustausch zwischen den Stationen ist ein sequenzielles Programm: Die Zuweisung des Kommunikationskanals wird nur ein einziges Mal bearbeitet, der Datenaustausch über das Netzwerk findet erst statt, wenn der vorherige Datenaustausch beendet ist. Deshalb schlagen wir vor, den Datenaustausch IL mit dem Graftec-Editor zu bearbeiten.

Die Anfangsetappe ermöglicht die Zuweisung des Kommunikationskanals bei Kaltstart des PCD.

Die anderen Etappen werden in der Schleife bearbeitet und unterstützen jede den Austausch eines Datenpakets.

Jede Etappe wird durch einen Übergang getrennt, der den TBSY-Diagnoseindikator testet und festlegt, ob der Datenaustausch beendet ist. Wir dürfen erst dann Daten über nachfolgende Etappe austauschen, wenn sich TBSY im unteren Zustand befindet.

Datenaustausch mithilfe einer Etappe

Vor dem Datenaustausch müssen wir die Adresse der Slavestation in dem hierzu über den Zuweisungstext deklarierten Register definieren.

Definition der Slavestationsadresse

```
LDL     R 500   ; Adresse des Registers mit Slavestationsadresse
        11      ; Adresse S-Bus

LDH     R 500   ; Adresse des Registers mit Slavestationsadresse
        2       ;IP-Knoten
```

Der Datenaustausch zwischen den Stationen wird mithilfe von zwei Anweisungen unterstützt:
STXM zum Schreiben der Daten in einer Slavestation (*SEND*)
SRXM zum Lesen der Daten einer Slavestation (*RCV*)

Jede dieser Anweisungen besteht aus vier Parametern: Kanaladresse, Anzahl der auszutauschenden Daten und Adresse der ersten Quelldaten, Bestimmungsort

Eintragen der 8 Indikatoren (F 0,..,F 7) in eine Slavestation (F 200,..,F 207)

```
STXM  8      ;Kanaladresse
      8      ;Anzahl der auszutauschenden Daten
      F 0    ;Adresse der ersten Quelldaten (lokale Station)
      F200   ;Adresse der ersten Bestimmungsdaten (lokale Station)
```

Einlesen eines Registers (R 25) einer Slavestation (R 125)

```
SRXM  8      ;Kanaladresse
      1      ;Anzahl der auszutauschenden Daten
      R 25   ;Adresse der ersten Quelldaten (lSlavestation)
      R 125  ;Adresse der ersten Bestimmungsdaten (lokale Station)
```

Anmerkung:
Nur die Masterstationen werden mit STXM und SRXM programmiert! Die Slavestationen dürfen nur mit dem Kommunikationskanal zugewiesen werden.

Warten auf das Übertragungsende mithilfe eines Übergangs

```
STL    F 1003  ; Prüft, ob sich TBSY im unteren Zustand befindet
```

Der Zuweisungstext legt einen Bereich mit 8 Diagnoseindikatoren zur Kommunikation fest, wobei der dritte in den oberen Datenübertragungszustand versetzt wird und in den unteren fällt, wenn der Austausch beendet ist.

13.6.5 Diagnosen

Kanalzuweisungen
Bei einem Kommunikationsproblem prüfen, ob die Kanalzuweisung richtig durchgeführt worden ist. Das Programm in der Betriebsart Step-by-Step bearbeiten und prüfen, ob die SASI-Anweisung keine Fehlerflag enthält. Ansonsten kann die Kanalzuweisung nicht richtig durchgeführt werden und die Kommunikation funktioniert nicht.

Mögliche Korrekturmaßnahmen auf der Master- oder Slavestation:

- *Device Configurator* prüfen.
- Prüfen, ob die *Device Configurator* in die PCD heruntergeladen worden sind.
- Prüfen, ob die Stationen alle dasselbe Profil verwenden. S-Net, DP
- Prüfen, ob alle Stationen mit derselben Geschwindigkeit kommunizieren.
- Prüfen, ob der mit den *Device Configurator* festgelegte Kommunikationskanal und die *SASI*-Anweisung identisch sind (dieselbe Kanalnummer)
- Prüfen, ob die PCD mit der erforderlichen Kommunikationshardware bestückt ist.
- Prüfen, ob die Stationen am Netz angeschlossen sind und unter Spannung stehen.
- Netzwerkverkabelung prüfen
- Prüfen, ob die Firmwareversion Ether-S-Bus unterstützt.

Die Daten werden nicht über das Netzwerk ausgetauscht

Der Zuweisungstext definiert einen Bereich mit 8 Diagnoseindikatoren zur Kommunikation, wobei der fünfte (*TDIA :Transmitter diagnostic*) bei einem Datenübertragungsfehler in den oberen Zustand versetzt wird. Der Step-by-Step - Test des Kommunikationsprogramms ermöglicht Ihnen festzustellen, welche der STXM- und SRXM-Anweisungen einen Fehler enthalten.

Vorsicht! Wenn sich ein Kommunikationsfehler ereignet, dann bleibt das TDIA-Diagnoseflag ganz oben, während das Diagnoseregister nicht wieder auf Null zurückgestellt wird.

Mögliche Korrekturmaßnahmen auf der Masterstation
Prüfen, ob die Parameter der STXM- und SRXM-Anweisungen fehlerhaft sind.
Prüfen, ob die Slaveadresse auch im Netzwerk vorhanden ist.

Mögliche Korrekturmaßnahmen auf der Slavestation
Für jede fehlerhafte STXM- und SRXM-Anweisung die Slavestationsnummer notieren und die entsprechenden Stationen prüfen.

- Prüfen, ob die *Device Configurator* richtig festgelegt worden sind.
- Prüfen, ob die PCD mit der erforderlichen Kommunikationshardware bestückt ist.
- Prüfen, ob die Station am Netz angeschlossen ist und unter Spannung steht.
- Netzwerkverkabelung prüfen
- Prüfen, ob die Firmwareversion Ether-S-Bus unterstützt.

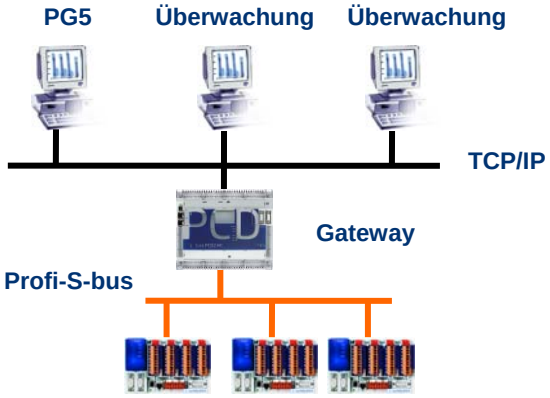
Diagnoseregister

Das Diagnoseregister kann mehr Informationen zur Art des Kommunikationsfehlers geben. Lassen Sie den Binärinhalt des Registers anzeigen und vergleichen Sie ihn mit den Beschreibungen in der Betriebsanleitung der PCD oder des Kommunikationsnetzwerks.

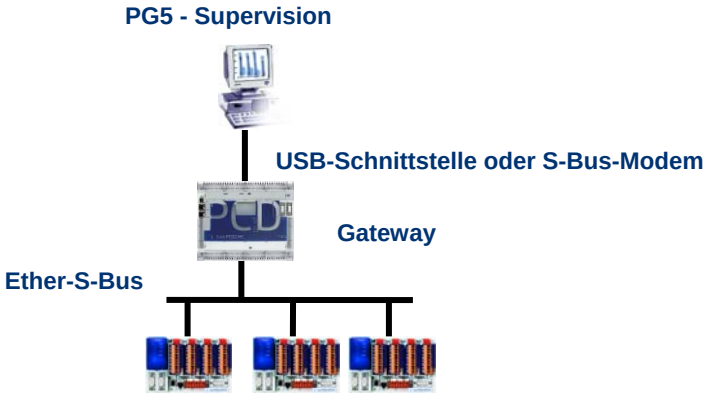
13.7 Gateway-Funktion

Die *Gateway*-Funktion wird häufig verwendet, damit zwei verschiedene Kommunikationssysteme zusammen kommunizieren können und zum Anpassen eines PG5-Programmierungssystems, eine Saia PG5 Visi.Plus-Überwachung auf einem anderen als das direkt von diesen Geräten unterstützten Netzwerks.

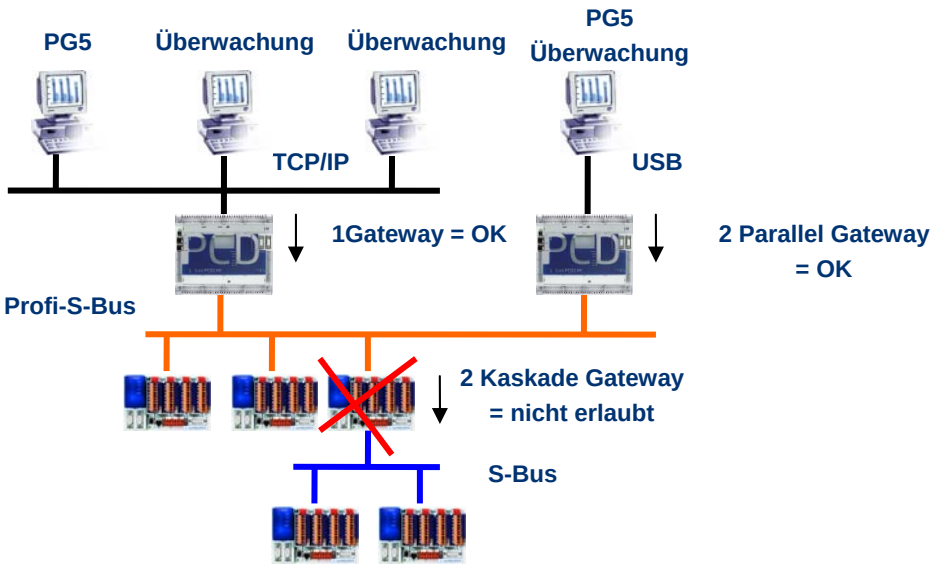
13.7.1 Anwendung



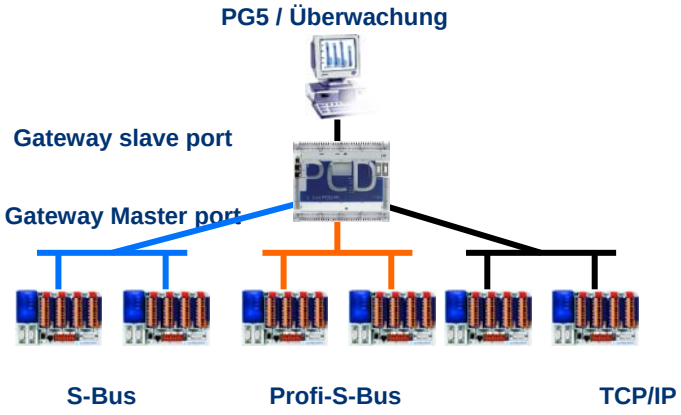
Die *Gateway*-Funktion ermöglicht die Durchführung einer Brücke zwischen zwei Netzwerken. Beispielsweise ein Ethernet-Netzwerk mit dem Profi-S-Bus-Netzwerk verbinden. Auf diese Weise tauschen die PCD-Systeme die Daten auf einem Bus aus dem Bereich Automatisierung aus, der vom EDV-Netz des Unternehmens getrennt ist. Aber die mit der PG5—Software oder einem Saia PG5 Visi.Plus Überwachungssystem bestückten Computer können Daten mit verschiedenen PCD austauschen.



Die Funktion *Gateway* kann als Schnittstelle zwischen dem Kommunikationsnetzwerk und der Außenwelt dienen. Beispielsweise, eine Kommunikation über Modem herstellen oder eine USB-Kommunikationsschnittstelle.



Um die Timingvorgaben der Kommunikation zu respektieren, können nicht zwei kaskadische Gateways definiert werden. Aber es ist möglich, zwei parallele Gateways im gleichen Netz zu definieren.



Falls erforderlich, kann die Funktion Gateway eine Verbindung zu verschiedenen Unterkommunikationsnetzen herstellen.

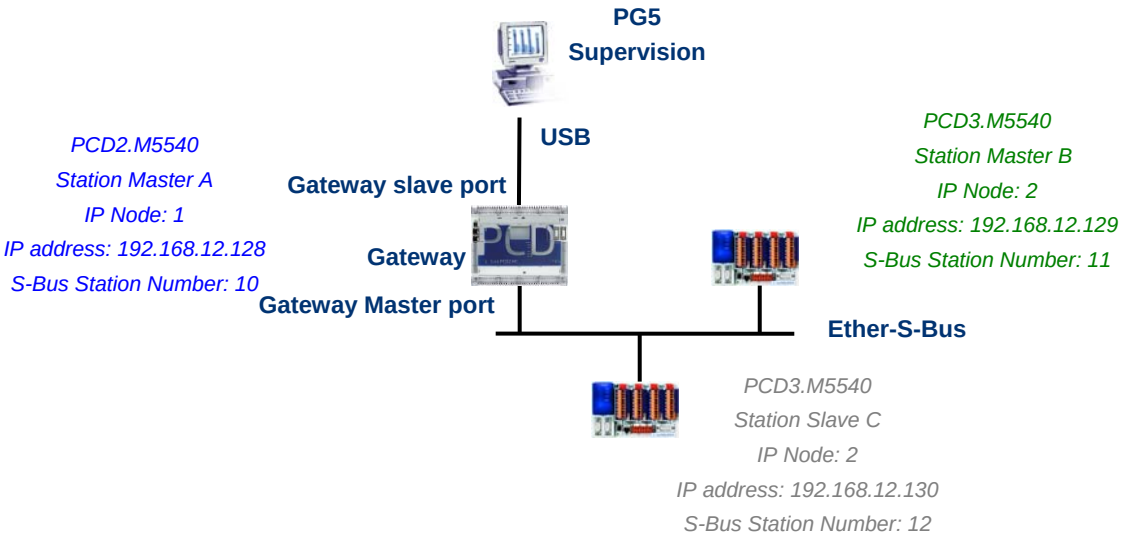
13.7.2 Konfiguration der Gateway PGU-Funktion

Die *Gateway*-Funktion lässt sich sehr leicht einrichten, sie benötigt kein Programm, sondern nur ein paar Parameter in den *Device Configurator* der PCD.

Im Allgemeinen muss man nur einen *Gateway Slave Port* und einen *Gateway Master Port* konfigurieren, anschließend wird alles automatisch von der *Gateway*-Funktion unterstützt.

Wenn sich die Daten auf dem *Gateway Slave Port* befinden und sie nicht die lokale Station (*Gateway*) betreffen, werden die Daten anhand der gültigen, für jedes Unternetzwerk definierten Adressen, auf eines der auf dem *Gateway Master Port* angeschlossenen Unternetzwerken übertragen.

Beispiel: Gateway USB; Ether-S-Bus

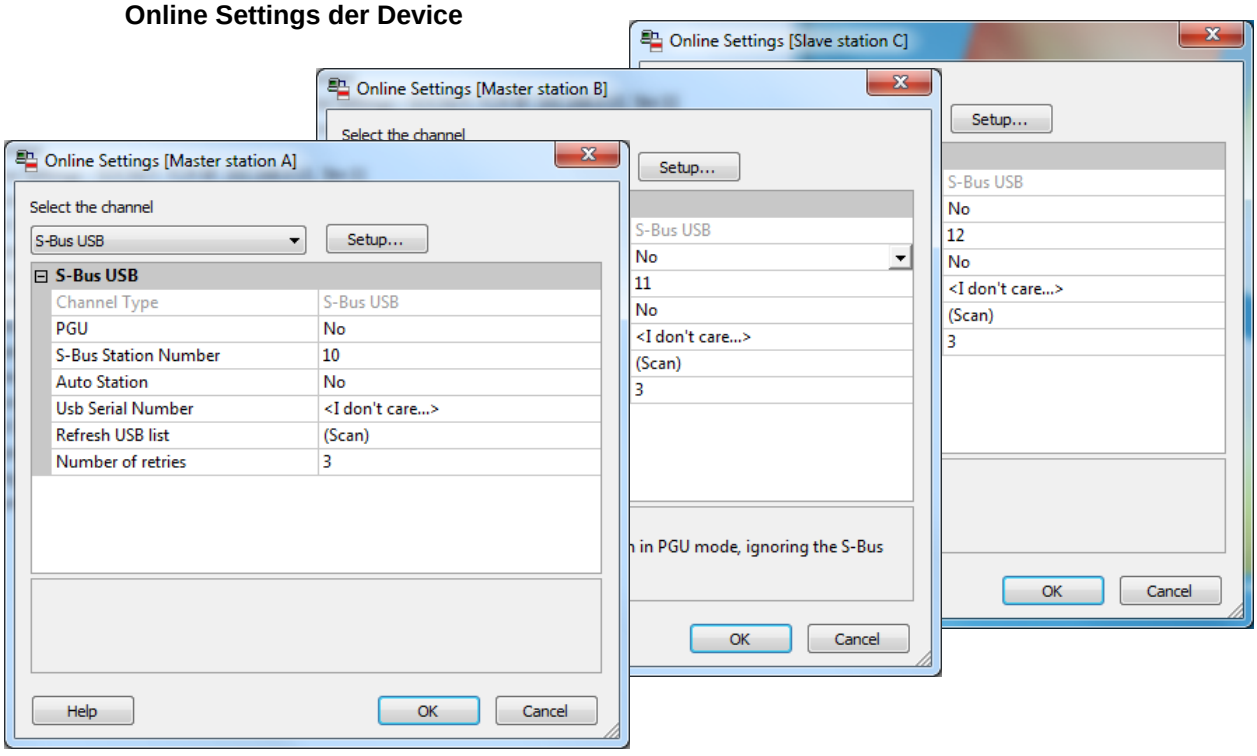


Onboard Communication, properties **der Masterstation A**

TCP/IP S-Bus Master Gateway	
Use TCP/IP For Gateway	**Yes**
First S-Bus Station	0
Last S-Bus Station	253
Response Timeout	0

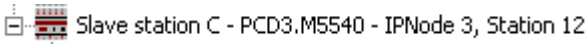
Der Gateway USB ist eine Ausnahme und erfordert keinen Parameter für den *Gateway Slave port*, nur der *Gateway Master port* muss parametriert werden. (Nicht vergessen, die neue Konfiguration in den Master A herunterzuladen!)

Online Settings der Device



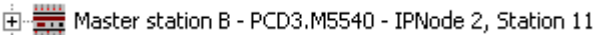
Zum Erstellen der USB-Kommunikation mit jeder der PCD muss man noch die *Online Settings* jeder Device des Projekts mit dem USB-Kanal und der Nummer der S-Bus-Station angleichen.

Testen des einwandfreien Betriebs der Gateway-Funktion



Aktivieren eines der device Device, *Master B* oder *Slave C*, des Projekts und zum Testen der Kommunikation mit der Station online gehen.

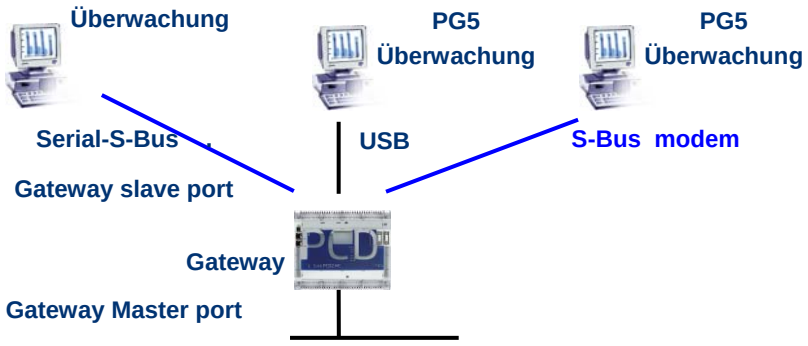
Falls erforderlich, kann man über den *Online Configurator* die Nummer der Station prüfen, die online ist. Man kann somit das Programm in der aktiven device aufladen und testen und gleichzeitig mit dem USB-Kabel mit der Station *Master A* verbunden bleiben.



Um mit einer anderen Netzwerkstation zu kommunizieren, die aktivieren und online gehen.

Anmerkung:
Mit der *Gateway*-Funktion wird nur die Station S-Bus Slave definiert, die Nummer der Ether-S-Bus-Station wird nicht berücksichtigt, weil die Telegramme an alle anderen Ether-S-Bus (Broadcast) Stationen adressiert sind.

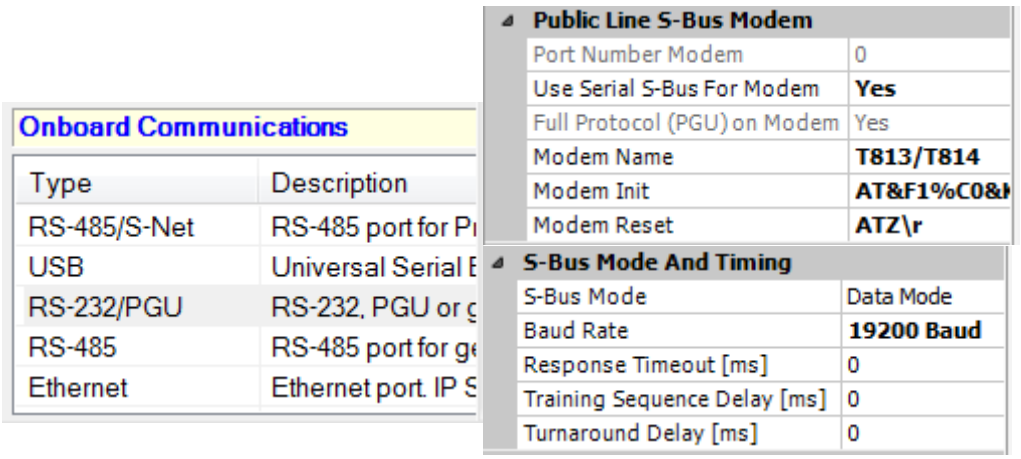
13.7.3 Konfiguration eines zusätzlichen Slave-Gateway-Slave-Port



Der Gateway Slave port ist ein Mittel, um von außen in das Netzwerk zu gelangen. Falls erforderlich, kann ein zweiter oder ein dritter *Gateway Slave port* definiert werden.

***Device Configurator* Parameter**
Im Allgemeinen unterstützt die PCD nur einen einzigen PGU-Slavekanal. Aber die neuen SPS PCD2 und PCD.3Mxxxx können mehrere auf demselben PCD unterstützen. Die Konfiguration des zweiten Gateway Slave PGU wird vollständig von den *Device Configurator* unterstützt.

Beispiel: Hinzufügen eines zweiten Gateway Ether-S-Bus

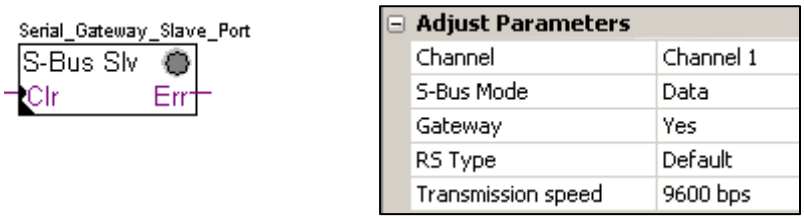


Der zweite *Gateway Slave port PGU* wird durch Konfigurieren der *Device Configurator* mit den Parametern für das Modem hinzugefügt.

Fupla- oder IL-Programm

Es ist ist es auch möglich, eine zusätzliche Fbox/SASI-Anweisung zu verwenden, um einen zweiten *Gateway Slave port*-hinzuzufügen.
Diesen *Gateway slave port* unterstützt das PG5-Programmierungstool nicht, aber nur einen Terminal oder ein Überwachungssystem. Nur das Lesen und Schreiben der PCD-Daten wird nicht unterstützt: Register, Indikatoren, …

Fupla-Beispiel: Hinzufügen eines dritten seriellen S-Bus, Ether-S-Bus



Der *Gateway*-Einstellungsparameter muss daher mit der Option *Yes* definiert werden. Je nach Kanaltyp müssen die anderen Parameter im Einstellungsfenster ebenso korrekt definiert werden.

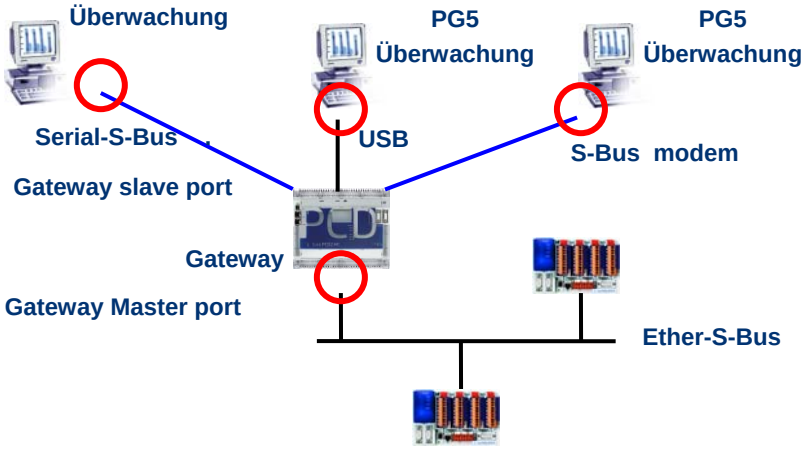
Beispiel I: Hinzufügen eines dritten seriellen S-Bus, Ether-S-Bus

Benutzen Sie nachfolgenden Text zur Zuweisung des Kanals:

```
$SASI
TEXT 11   "UART:9600;MODE:GS2;DIAG:F1110,R0501;"
$ENDSASI
```

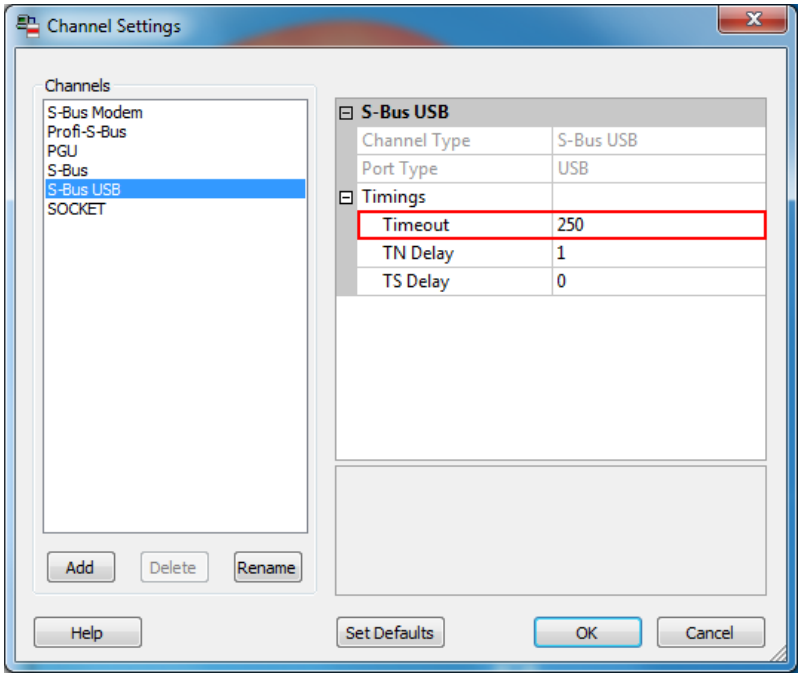
Indikator und Diagnoseregister
Mode S-Bus Gateway Slave Data mode
Übertragungsgeschwindigkeit

13.7.4 Kommunikationstimings

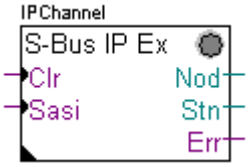


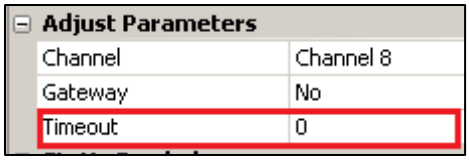
Im Allgemeinen werden die Kommunikations-*timings* mit den Standardwerten definiert und dies funktioniert einwandfrei. Aber die Benutzung der *Gateway*-Funktion erhöht die zum Datenaustausch erforderliche Reaktionszeit. Es ist daher manchmal erforderlich, das Timeout der Masterstationen mit Hilfe des *Gateway* anzupassen. Nachstehende Abbildung zeigt die Masterkanäle, deren Timeout eingestellt werden müsste.

Zum Einstellen des *Timeout* des PG5 die *Online Settings* der *Master Station A* benutzen:



Zum Einstellen des *Timeout* des Datenaustauschprogramms auf dem PCD-*Gateway* die Fbox benutzen: *SASI S-Bus IP Extended*





13.8 Andere Referenzen

Weitere Informationen können Sie in folgenden Handbüchern finden:

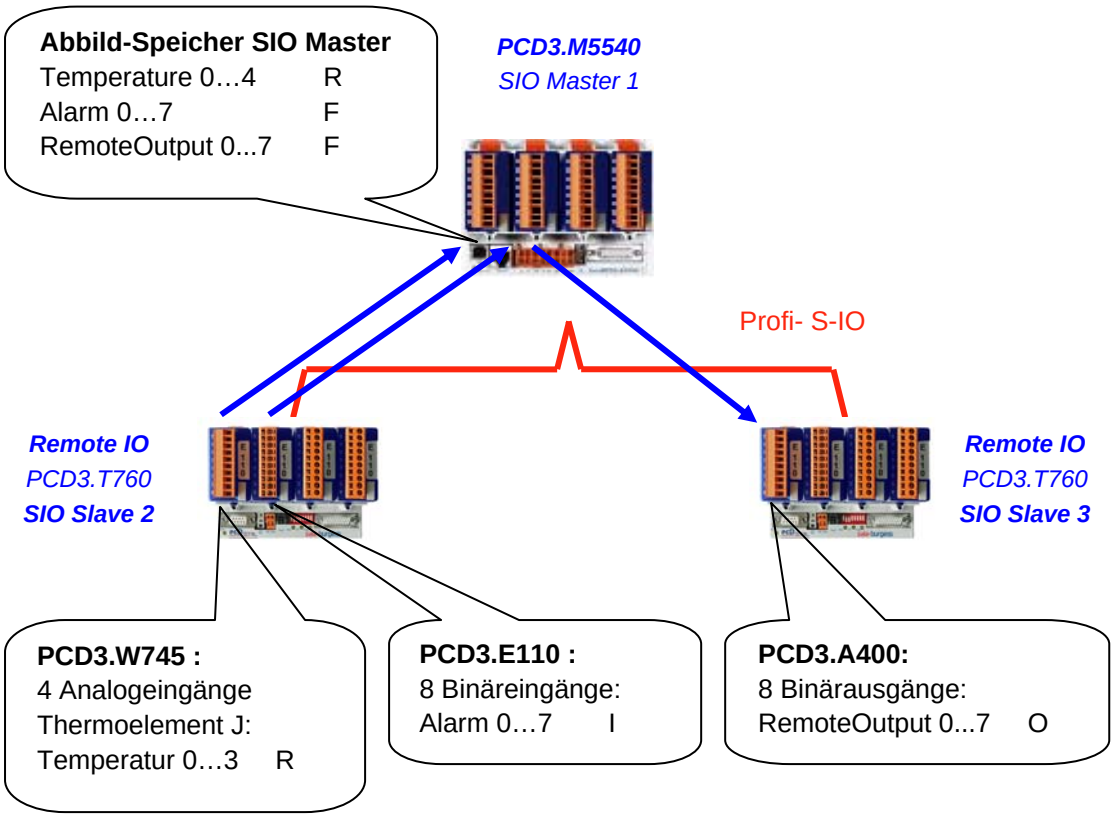
- Leitfaden zu den Anweisungen 26/133
- Ethernet TCP/IP 27/776
- Beispiel des Ether-S-Bus-Projekts, das mit Ihrem PG5 installiert ist.

Inhaltsverzeichnis

15 Profi-S-IO

Dieses Beispiel zeigt, wie man verlagerte binäre oder analogische Ein- und Ausgänge auf den RIO-Klemmen, Typ PCD3.T7xx benutzt.

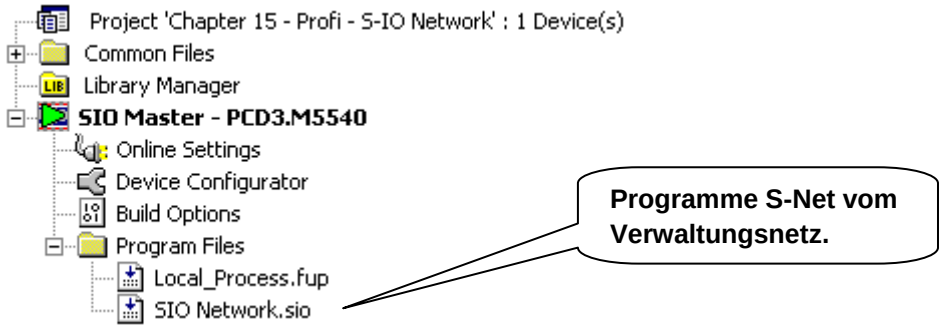
15.1 Beispiel eines Profi-S-IO.



15.2 Allgemeines Funktionsprinzip

Mit den Netzwerken Profibus DP und Profi-S-IO werden die Daten auf dem Netzwerk weder mit Hilfe des Editors Fupla oder IL definiert, noch mit Hilfe der *Device Configurator* PG5, sondern nur über den Konfigurator S-Net.
Dieser Konfigurator unterstützt die Definition der auf dem Netzwerk vorhandenen Stationen sowie der Ein- und Ausgangsmodule, die dort eingefügt sind. Für jedes Modul werden die Informationen der Ein- und Ausgänge mit einem Symbol definiert, das über das Programm der Masterstation bearbeitet wird.
Bei *build* des Projektes PG5, S-Net erstellt automatisch einen Abbild-Speicher der auf dem Netzwerk verfügbaren Daten. Sie werden über das auf der Masterstation vorhandene Fupla- oder IL-Programm bearbeitet.
Die Symbole, die diese Informationen auf den verlagerten Ein- und Ausgangsmodulen bezeichnen, sind dieselben wie die auf dem Abbild-Speicher der Masterstation. Daher verwendet das Fupla- oder IL-Programm die Symbole des Abbild-Speichers genau wie die anderen zur Masterstation gehörenden lokalen Symbole, wobei die Verwaltung des Datenaustauschs über das Netzwerk klar und deutlich von der Prozesskontrolle getrennt ist.

15.3 Projekt PG5



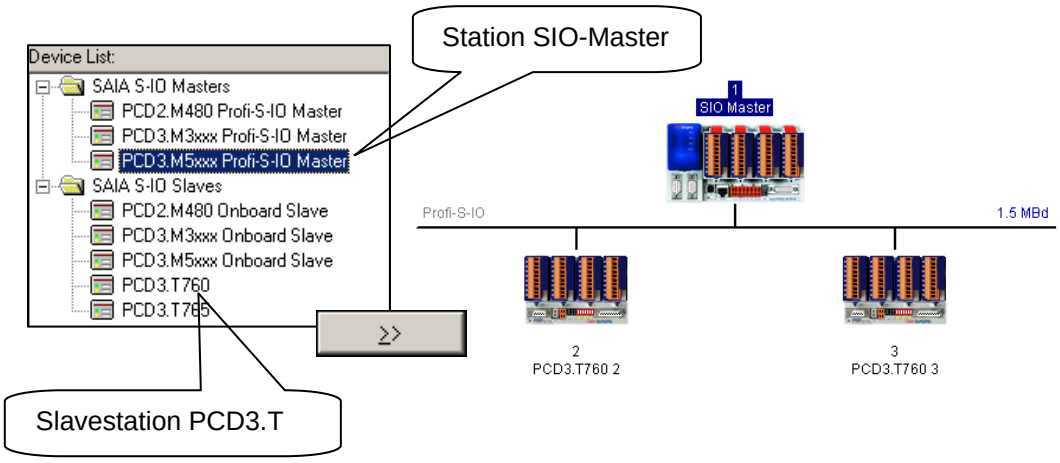
Die Datei S-Net wird auf dieselbe Weise in die Masterstation hinzugefügt, wie eine Fupla- oder IL-Datei. Jedoch muss eine Dateierweiterung .SIO (Profi-S-IO) oder .DP (Profibus DP) gewählt werden.

Tatsächlich ist die Benutzung des S-Net- Konfigurators für den Datenaustausch über ein Profi-S-IO und Profibus DP-Netzwerk identisch, mit Ausnahme von:

- Der Erweiterung der Konfigurationsdatei .SIO, .DP
- Die vom Netzwerk unterstützten Devices: SIO = devices Saia, DP = devices Saia + andere Lieferanten
- Die Netztimings: Profile Snet, DP

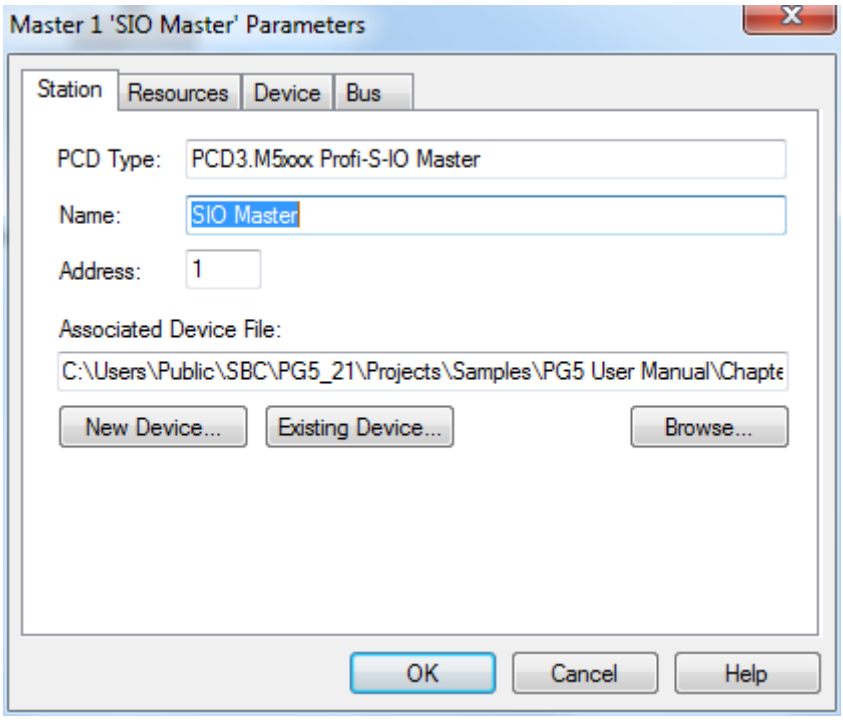
Anmerkung:
Wenn PCD7.T7xx auf dem Netzwerk vorhanden sind, immer das S-Net-Profil wählen.

15.4 Festlegung der Stationen auf dem Netzwerk



Für jede einzelne der auf dem Netzwerk vorhandenen Stationen, den Stationstyp in der Liste der *devices* wählen und über die dafür vorgesehene Schaltfläche dem Netz hinzufügen.

15.5 Konfigurieren der Masterstation

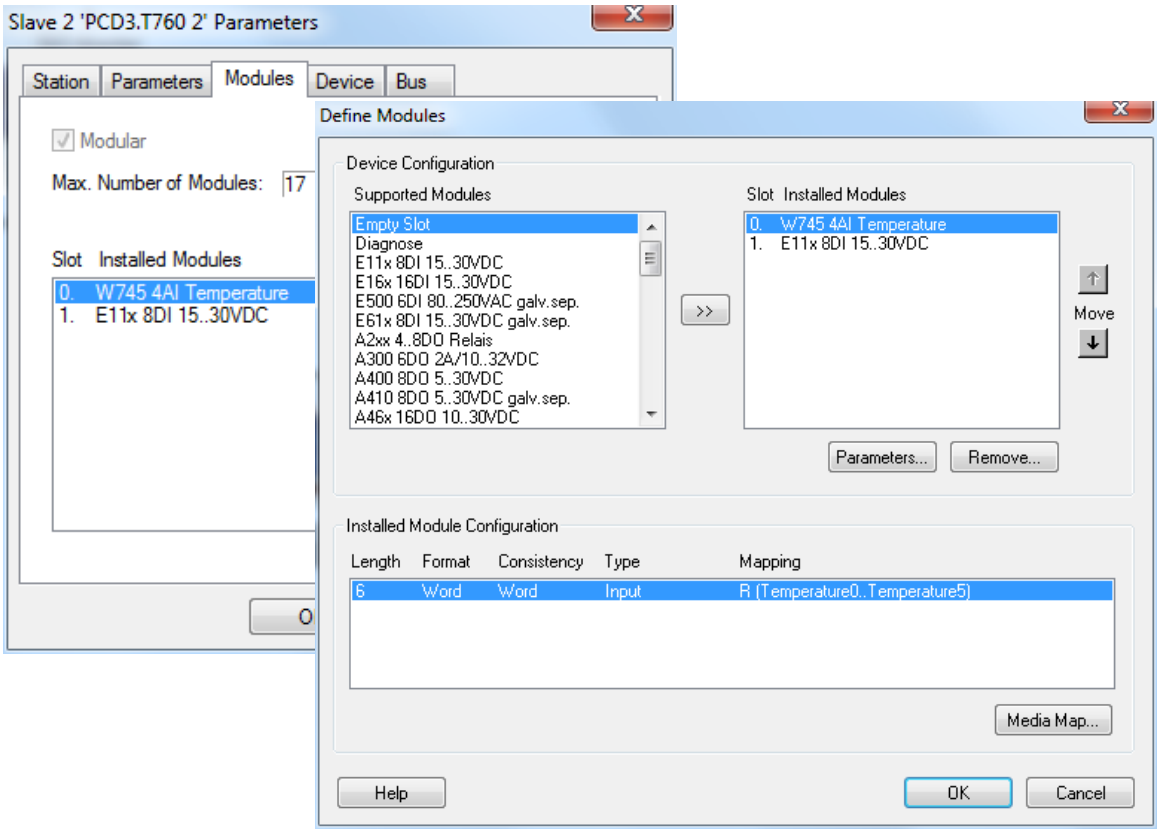


Eine einzige Information muss auf der Masterstation festgelegt werden. Dies ist der Zugangspfad zur Master-DEVICE in dem das S-Net das erforderliche Programm zur Ausarbeitung des Abbild-Speichers für die Masterstation einspeichern muss.

Wahlweise unterstützt dieser Dialog auch die Änderung von Namen und Adresse der DEVICE.

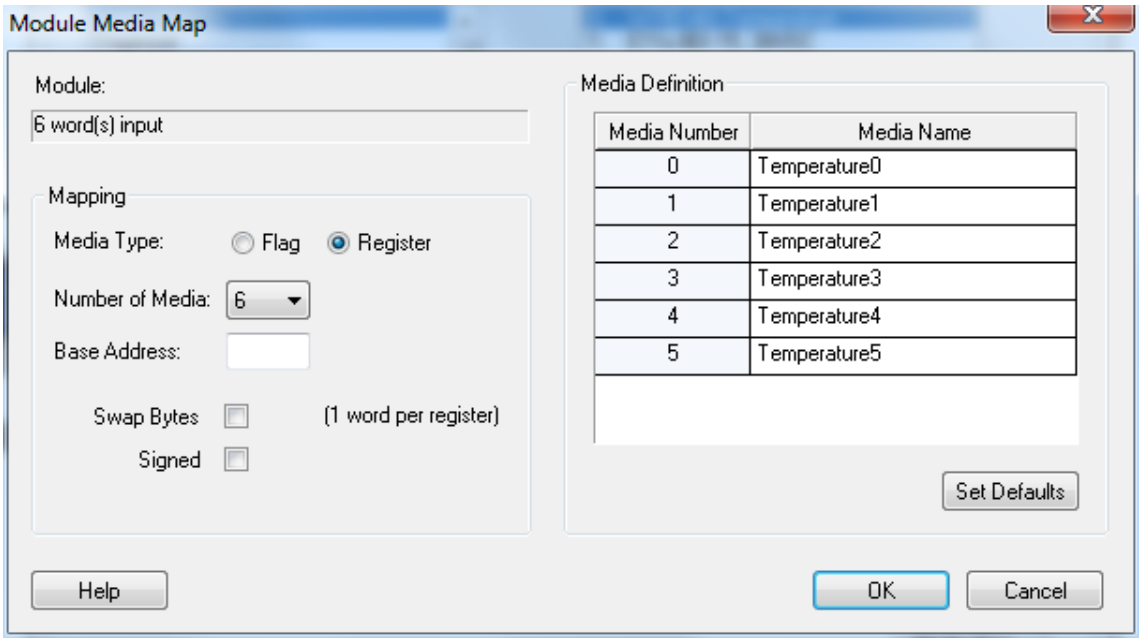
15.6 Konfigurieren der Slavestationen

15.6.1 Konfigurieren der Ein-/Ausgangsmodule.



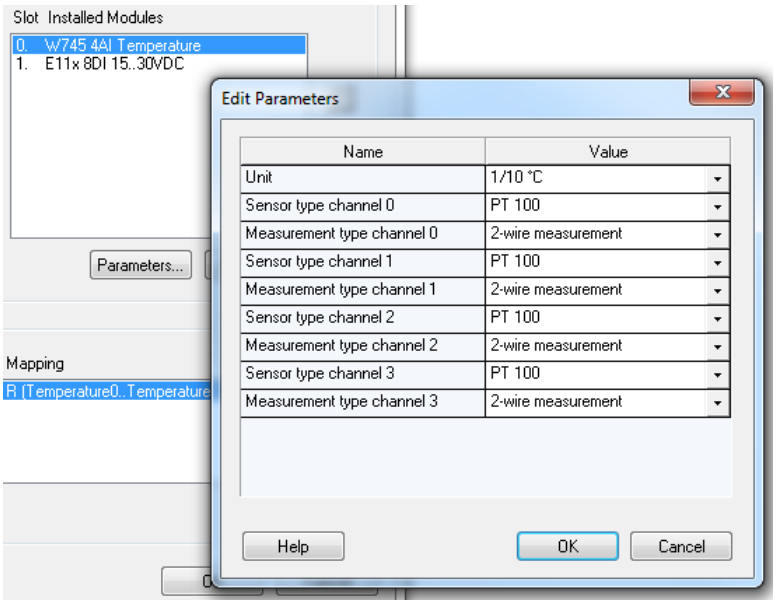
Für jedes Ein-/Ausgangsmodul auf der Slavestation, den Modultyp in der Liste *Supported modules* wählen und diesen im Fenster *Slot Installed Modules,* durch Klicken auf die entsprechende Schaltfläche hinzufügen. Achtung, die Nummer des *Slot* und der Typ des in die Slavestation eingefügten Moduls müssen immer der Nummer des *Slot* und der im Fenster *Slot Installed Modules* entsprechen!

15.6.2 Konfigurieren der Symbole der verlagerten Daten



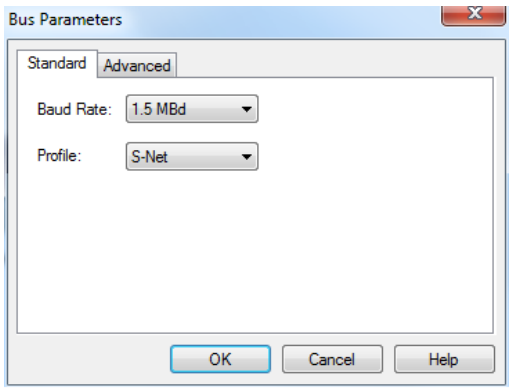
Für jedes im Fenster *Slot Installed Modules* vorhandene Modul, das Modul markieren und auf die Schaltfläche *Media Map* klicken, um die Symbole für jede vom Modul gelieferte Dateninformation zu definieren. Falls erforderlich, kann man die Adresse des ersten in der Masterstation zu benutzenden Flags definieren oder registrieren, um den Abbild-Speicher auszuarbeiten. Am einfachsten ist es, gar nichts zu definieren, dann sind die Adressen dynamisch.

15.6.3 Konfigurieren der Einstellungsparameter



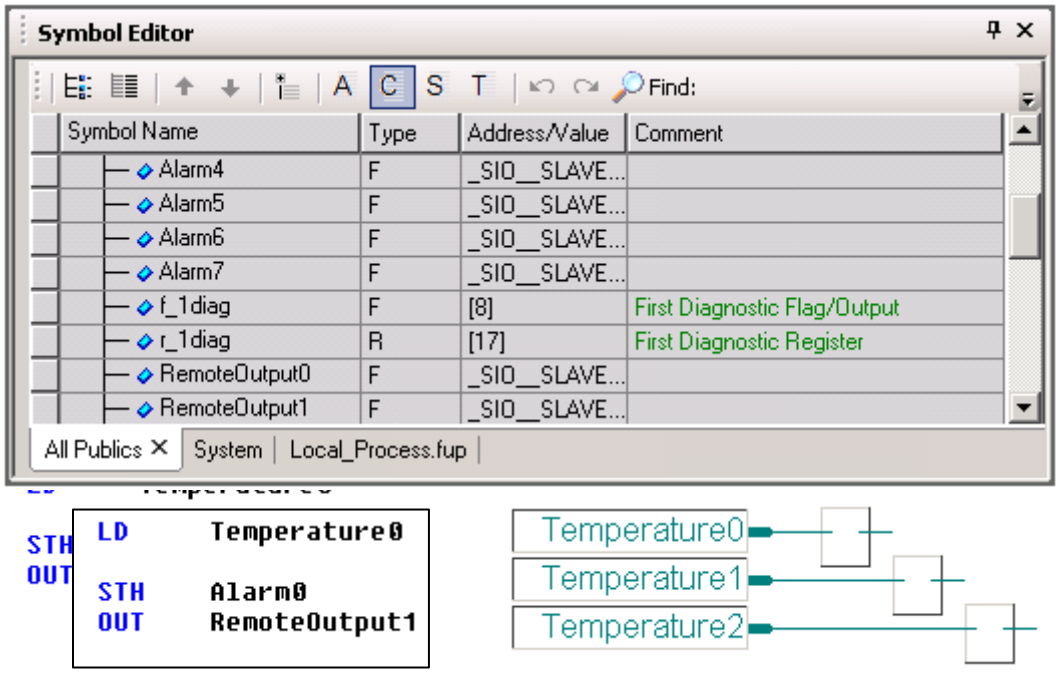
Mit einigen Modulen, wie beispielsweise Analogmessmodulen, kann es erforderlich sein, ein paar zusätzliche Parameter für die Maßumwandlung festzulegen: Maßeinheiten, Temperatursondentypen, ….
Diese Informationen können über das angezeigte Einstellungsfenster eingestellt werden, indem man das Modul markiert und auf *Parameters* klickt.

15.7 Konfigurieren der Netzwerkparameter.



Mit dem Menü *Edit Bus Parameter* kann man die Geschwindigkeit und das Profil des Kommunikationsnetzes festlegen.

15.8 Bearbeitung der Netzwerksymbole in einem Fupla- oder IL-Programm.



Mit der Kompilierung der Datei S-Net (Menu Project,Compile), zeigt der Symboleditor ein neues Blatt mit allen auf dem Netzwerk verfügbaren Symbole an. Diese Symbole können direkt in die Fupla- und IL-Dateien integriert werden.

15.9 Andere Referenzen

Weitere Informationen können Sie in folgenden Handbüchern finden:

- Profibus DP 26/ 765
- Profi-S-IO (in Vorbereitung)
- Beispiel des Profi-S-IO-Projekts, das mit Ihrem PG5 installiert worden ist.

Technische Daten

Technische Daten

Betriebssystem	PG5 2.1 läuft unter Windows 8.1, Windows 8 (64 bit), Windows 7 (32 und 64 bit) und Windows XP (32 bit). Microsoft .Net 4.0 Client Profile und Microsoft .Net 4.0 Extended. Microsoft .Net wird auf der Installations-CD mitgeliefert: CD :\Windows\ dotNetFx40_Full_x86_x64.exe
PC	Multi-core Prozessor mit minimal 2 GHz sowie mindestens 2 GB RAM (4 GB ist empfohlen). Die Software benötigt 600 MB freien Festplattenspeicher.
PCD-Befehlssatz	alle 150 Instruktionen der PCD werden unterstützt
Standard FBoxen	die Standard-FBoxen des PG5 umfassen mehr als 250 FBoxen
Programmiersprachen	PG5 enthält Editoren für Instruktionsliste, FUPLA und GRAFTECProgramme
Unterstützte CPUs	sämtliche SAIAPCD® CPU werden unterstützt (gilt nicht für die Serie xx7)
Kompatibilität	PG3-, PG4- und PG5 1.x-Dateien können im PG5 2.1 weiter verwendet werden.
Kommunikation	TCP/IP-, SAIA S-Bus®-, PROFIBUS DP-, BACnet- und LONWORKS®-Kommunikation sind im PG5 vorhanden

Saia-Burgess Controls AG
Bahnhofstrasse 18
CH-3280 Murten / Switzerland

Telephone ++41 26 672 71 11
Telefax ++41 26 670 44 43

E-mail: info@saia-pcd.com
Homepage: www.saia-pcd.com
Support: www.sbc-support.com

Your local contact:

Gedruckt in der Schweiz 26/732 D6 02. 2006 Änderungen von technischen Daten und Angaben vorbehalten.