



Das Programmier- Werkzeug der Steuerungen SAIA®PCD

Programmieren, Projektieren und Konfigurieren von SPS-basierten Systemen

Vorteile des Programmierwerkzeugs PG5

- **Programm-Portabilität:** PG5-Programme sind auf allen SAIA®PCD-Plattformen lauffähig.
- **Programm-Organisation nach Dateien** (mit mehreren Programmblöcken) erleichtert den gleichzeitigen Einsatz von Programm-Dateien in mehreren SAIA®PCD-Steuerungen.
- **Übernahme vorhandener PG3- und PG4-Programme.**
- **Programmier- und Debug-Umgebung** im selben Werkzeug vereint.
- **Einfaches Programmieren von Display-Anzeigen** mit integriertem HMI-Editor.
- **Starker Befehlssatz**, unterstützt mit einer Vielzahl von Assembler-Direktiven.

Die Eigenschaften des 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 SAIA®S-Bus, PROFIBUS DP und FMS, LONWORKS®.
- **Umfangreiche Zusatzbibliotheken** erweitern den PG5-Funktionsumfang.

Anwender- Handbuch

Inhaltsverzeichnis

	Vorwort	II
	Anpassungen	III
1	PCD - Inbetriebnahme, Quick Start	1-3
2	Projekt-Management	2-3
3	PCD - Ressourcen	3-3
4	FUPLA-Programmierung	4-3
5	Programmstrukturen	5-3
6	Programmieren in Graftec	6-3
7	Programmieren in IL	7-3
8	Zusätzliche-Werkzeuge	8-3
9	Saia-Netzwerke (S-Net)	9-2
10	Profi-S-Bus	10-2
11	Ether-S-Bus	11-2
12	Profi-S-IO	12-2

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 1.4
- Dokumentations CD 26/803
- 1 PCD2.M480¹⁾ 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 1.4 finden Sie auf der PG5 CD unter: \PG5\InstallationGuide_F.htm.

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⁾ oder eine andere SAIA®PCD

Anpassungen

Chronologie

Datum	Kapitel	Seite	Beschreibung

Inhaltsverzeichnis

1	PCD - INBETRIEBNAHME, QUICK START	3
1.1	Einleitung	3
1.2	Installation der Hardware	4
1.2.1	Beispiel: Treppenhausbeleuchtung	4
1.2.2	Anschlussschema für PCD2.M480	4
1.2.3	Bestücken der PCD2.M480	5
1.2.4	Verdrahtung	5
1.3	Programmerstellung	6
1.3.1	Software Installation	6
1.3.2	PG5 starten	6
1.3.3	Ein Projekt eröffnen	6
1.3.4	Konfiguration	8
1.3.5	Eine Programmdatei hinzufügen	10
1.3.6	Datei öffnen	11
1.3.7	Programm editieren	11
1.4	Programm aufbauen und in die PCD übertragen	16
1.4.1	Programm aufbauen (Build)	16
1.4.2	Programm in die PCD übertragen (Download)	16
1.5	Programmfehlersuche und Behebung (Debugging)	17
1.5.1	Programm korrigieren	18

1 PCD - Inbetriebnahme, Quick Start

1.1 Einleitung

Die Quick Start Anleitung erklärt mit einem einfachen Anwendungsbeispiel die Installation einer PCD, die Programmerstellung inklusive Programmtest mit dem PG5-Programmierungswerkzeug und die Inbetriebnahme der PCD.

Dafür werden der PG5-Projektmanager, der Fupla-Editor und verschiedene Online-Funktionen des PG5 benutzt.

Für die Ausführung dieser Anleitung sind keine Kenntnisse der Saia-Produkte erforderlich.

1.2 Installation der Hardware

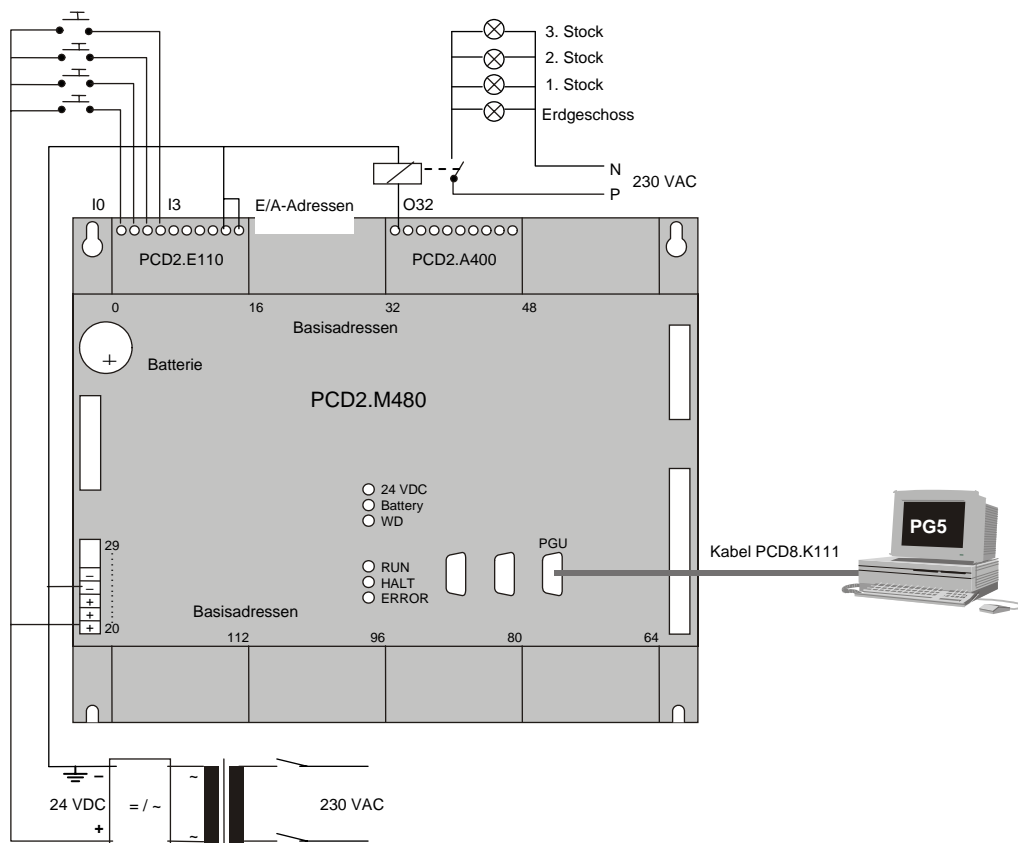
1.2.1 Beispiel: Treppenhausbeleuchtung

Die Inbetriebnahme einer 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.

1.2.2 Anschlussschema für PCD2.M480

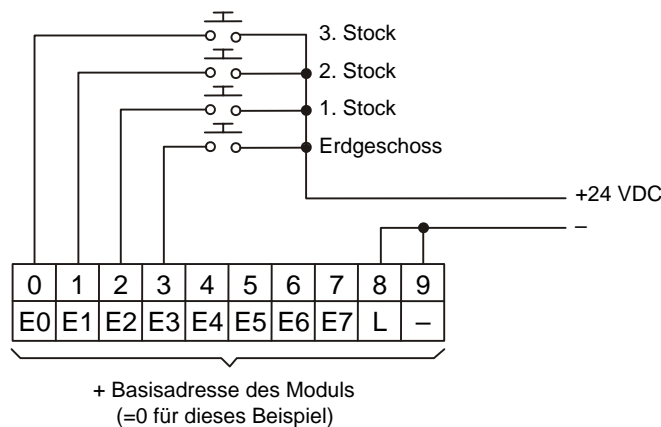


1.2.3 Bestücken der PCD2.M480

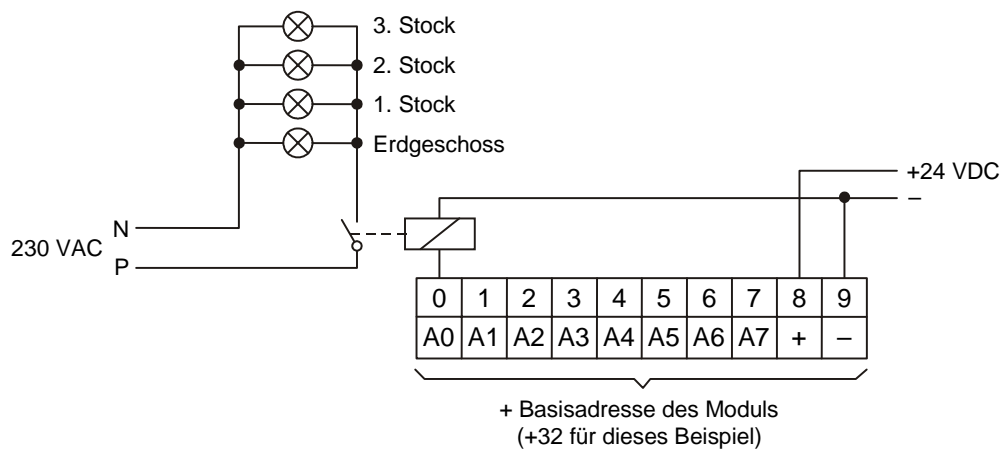
1. Die mitgelieferte Lithium-Batterie von 3.0 V einsetzen.
2. Ein Modul PCD2.E110 an Steckplatz 1 (Adressen 0 bis 15) einsetzen.
3. Modul gegen die Gerätemitte bis zur Endposition schieben und die Halteklinke einrasten. Damit stehen 8 digitale Eingänge für 24 VDC mit den Adressen E0 bis E7 zur Verfügung. 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.

1.2.4 Verdrahtung

1. Die Speisung 24 VDC an den Schraubklemmen 20 (+) und 23 (-) anschliessen.
Zulässige Speisespannungen sind:
24 VDC $\pm 20\%$ geglättet oder
19 VAC $\pm 15\%$ zweiweggleichgerichtet
2. Die Taster für die Treppenhausbeleuchtung gemäss folgendem Schema anschliessen.



3. Treppenhausbeleuchtung und Relais gemäss folgendem Schema anschliessen.



4. Die RS 232-Schnittstelle (COM-Port) des PC mit dem PGU-Stecker der PCD verbinden. Dafür ist das Kabel PCD8.K111 zu verwenden (siehe Anschlussschema).

1.3 Programmierstellung

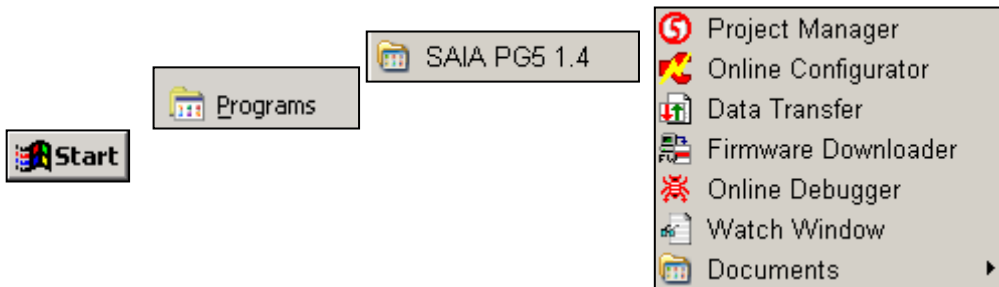
1.3.1 Software Installation

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

1.3.2 PG5 starten

Den PG5 Projekt Manager öffnen mit:

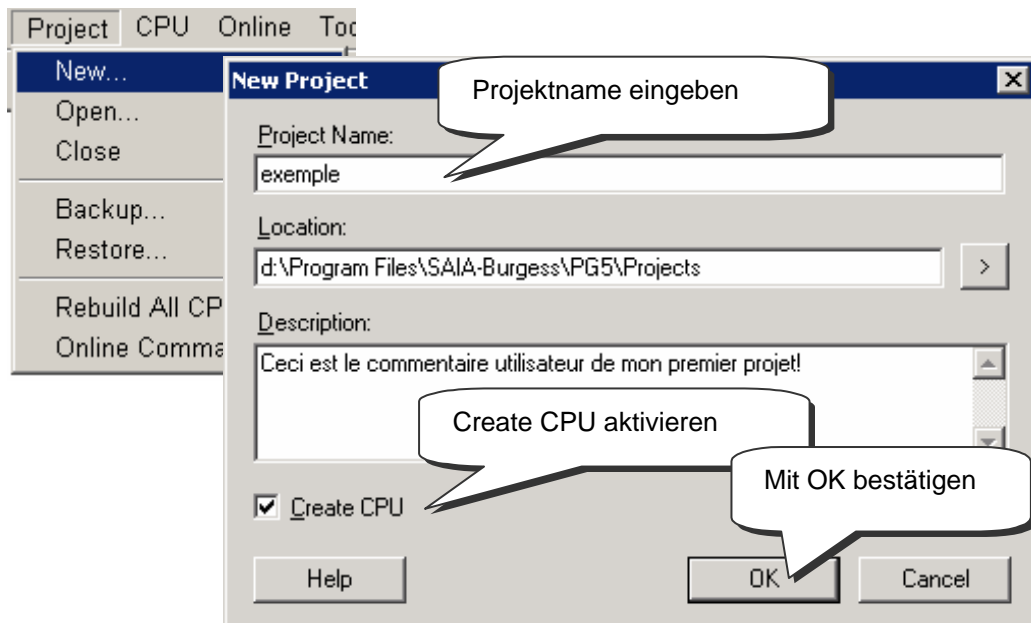
Start --> Programs --> SAIA PG5 V 1.4 --> Project Manager

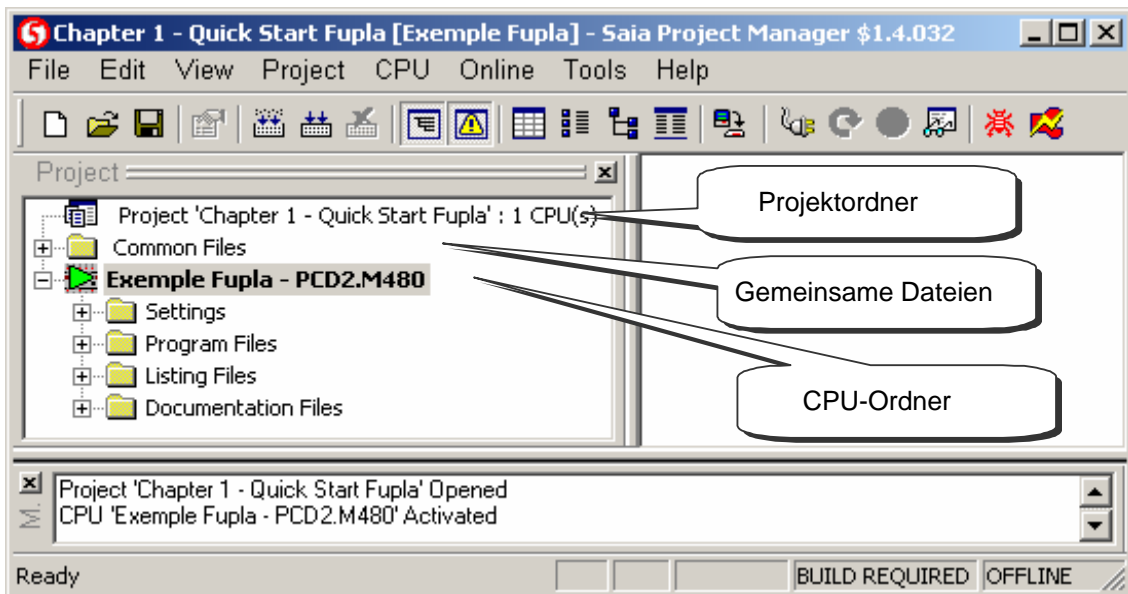


1.3.3 Ein Projekt eröffnen

Bevor mit dem Schreiben eines neuen Programms begonnen wird, ist ein neues Projekt oder ein bestehendes Projekt zu öffnen, das die notwendigen Definitionen und einige Konfigurationsparameter enthält, sowie die Dateien, welche für das Anwenderprogramm benötigt werden.

Ein neues Projekt öffnen





Das Fenster *SAIA Projekt Manager* für die Projektverwaltung wird bereits angezeigt. Der Fensterteil *Project (Projekt)* zeigt die Struktur für das neue Projekt (wenn nicht, kann sie via *View, Project Tree (Ansicht, Projekt-Menübaum)* angezeigt werden). Die Ordner im Fenster *Project* enthalten die Projektinformationen, nach bestimmten Kriterien geordnet:

- Die Bezeichnung des Hauptordners umfasst den Projektnamen und die Anzahl der im Projekt verwendeten CPU
- Im Ordner *Common Files (gemeinsame Dateien)* können die von den CPU gemeinsam benutzten Programme abgelegt werden

Darauf folgen die CPU-Ordner (jede CPU entspricht einer PCD). Jeder CPU-Ordner enthält die folgenden Ordner:

- *Settings (Einstellungen)* mit den für das Programmierwerkzeug und die PCD benötigten Konfigurationsparametern
- *Program Files (Programmdateien)* mit den PCD-Anwenderprogrammen
- *Listing Files (Listen)*. Sie enthalten bestimmte, beim Programmaufbau (*Build*) erzeugte Informationen. Diese sind für den Benutzer weniger interessant.

Ein bestehendes Projekt öffnen

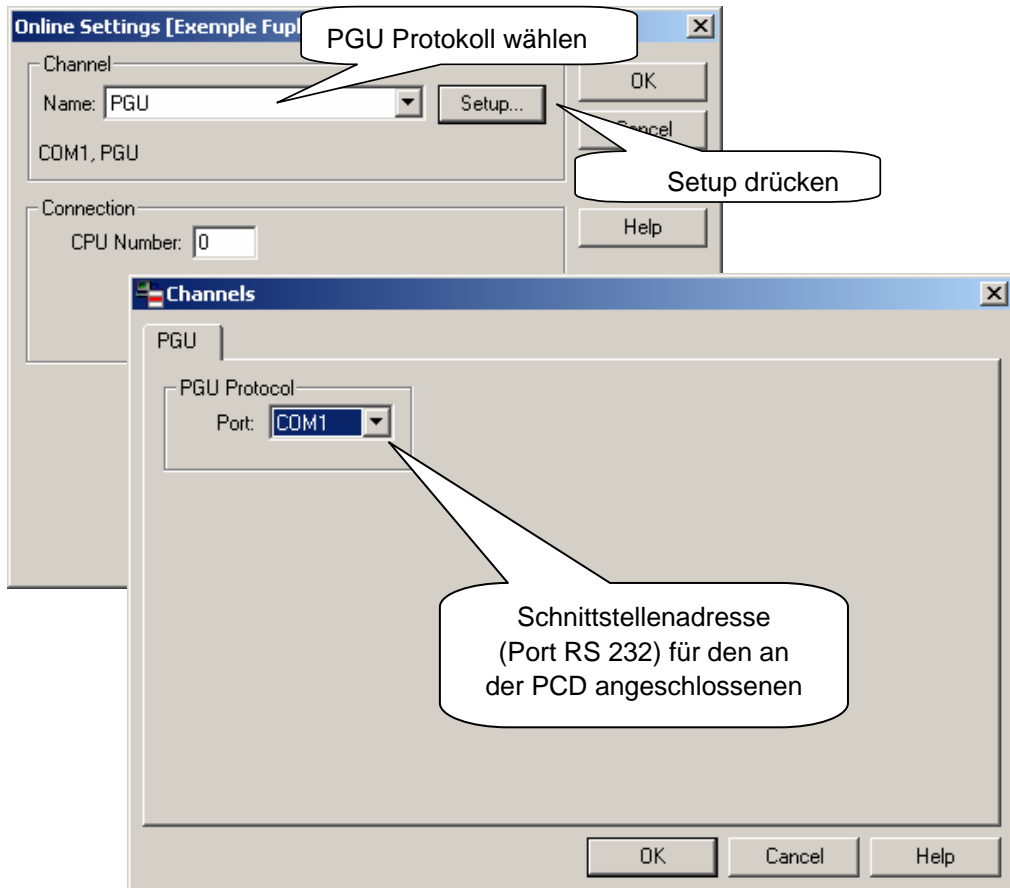
Ein bestehendes Projekt wird mit den Menü-Befehlen *Project, Open...* geöffnet. Damit werden alle Projektdateien (.5pj) im Projektverzeichnis gesucht und aufgelistet. Das gewünschte Projekt mit Doppelklick öffnen oder mit der Maus auswählen und mit *Open* öffnen. Mit *Browse* kann ebenfalls nach Projekten oder CPU-Dateien gesucht werden.

1.3.4 Konfiguration

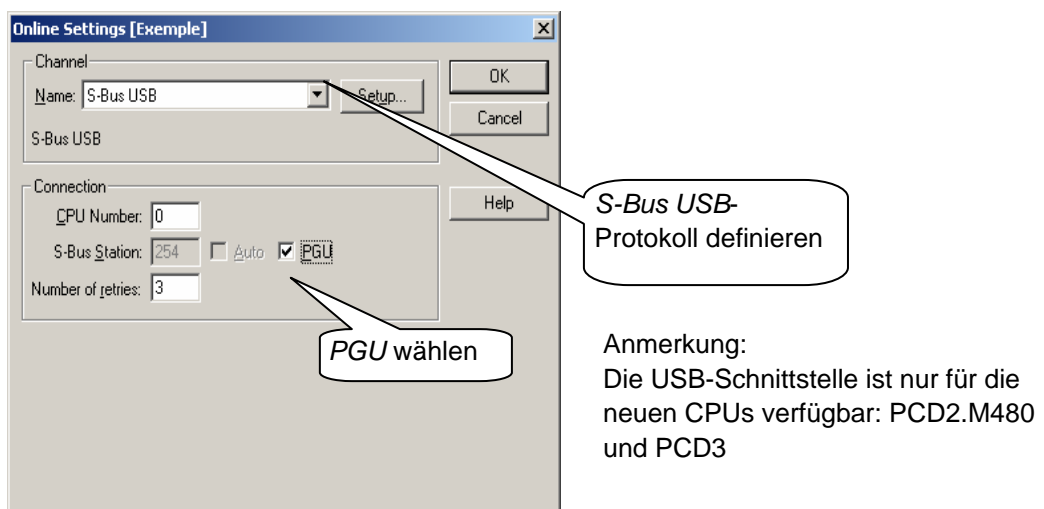
Bevor mit einer am Projekt beteiligten CPU gearbeitet werden kann, müssen die Konfigurationsparameter definiert werden, um zu gewährleisten, dass das Programmierwerkzeug und die PCD das zu erstellende Anwenderprogramm unterstützen.

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 (PGU) gewählt und es wird die Schnittstellenadresse (*Port*) des PC eingestellt.

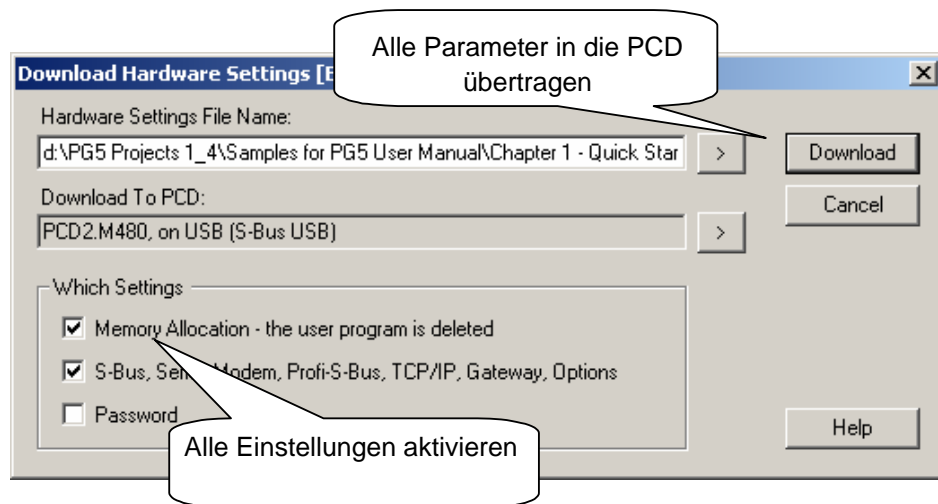
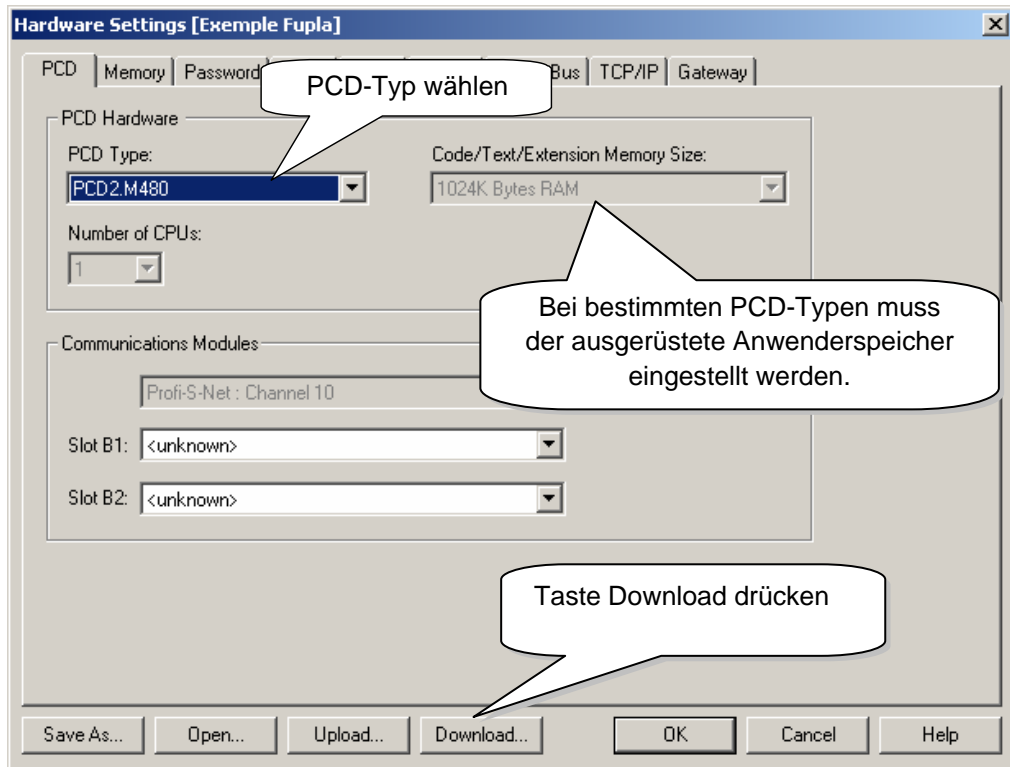
Channel PGU (RS 232)



Channel S-Bus USB



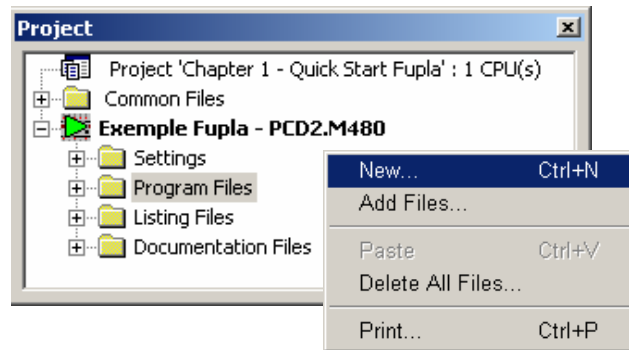
Unter *Hardware Settings (Hardware Einstellungen)* können die PCD-Parameter eingestellt werden. Die Funktionen *Memory*, *S-bus*, *Gateway*, *Modem* und *Password* werden für das Beispiel nicht benötigt. Wichtig sind jedoch der richtige PCD-Typ und die Grösse des ausgerüsteten Anwenderspeichers. Die PCD2.M480 wird immer mit 1024 KByte RAM geliefert.



1.3.5 Eine Programmdatei hinzufügen

Die PCD-Anwenderprogramme befinden sich in einer Datei. Es gibt mehrere Möglichkeiten, um eine Programmdatei einem Projekt hinzuzufügen:

Im Fenster *Project* den 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.

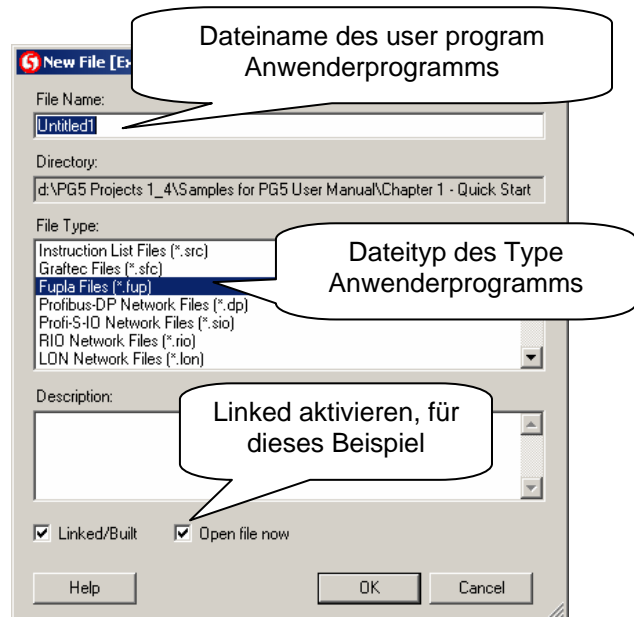
Für das Schreiben der PCD-Anwenderprogramme stehen mehrere Editoren zur Verfügung.

Der Benutzer kann den für das Anwenderprogramm am besten geeigneten Editor frei wählen.

Für dieses Beispiel ist es

Fupla File (.fup)*.

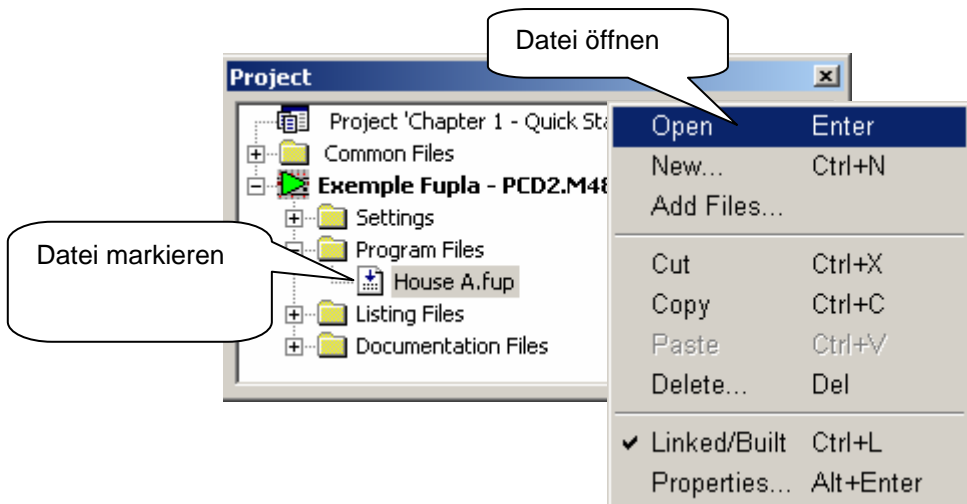
Dies ist ein universell verwendbarer Editor.



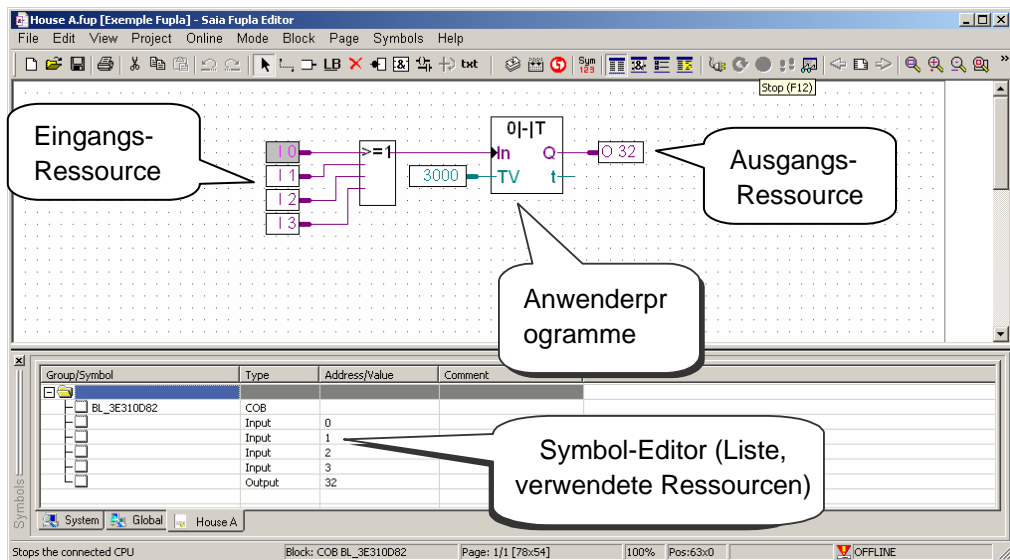
1.3.6 Datei öffnen

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.



1.3.7 Programm editieren

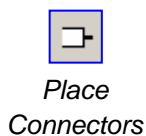


Ressourcen editieren

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).



Um einen Connector und sein Symbol zu einer Fupla-Seite hinzuzufügen, drücken Sie die Taste *Place 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.

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 Adresstyp (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..., Add Empty Side Connectors*.

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

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

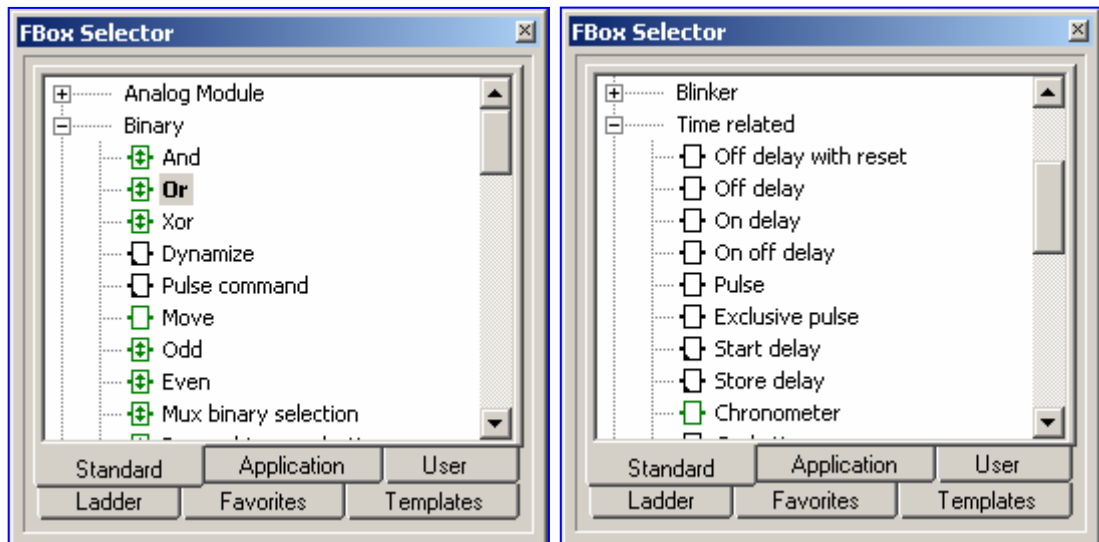
Programmfunktionen editieren

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.



Add Fbox



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 (*Timer*) 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.

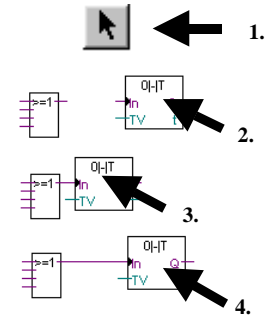
Nachdem im Fenster *Fbox Selector* ein Funktionsblock gewählt wurde, wird er mit der linken Maustaste auf der Fläche zwischen den Ressourcen-Spalten platziert.

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

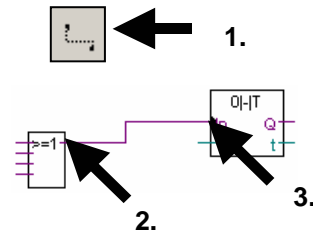
1. Möglichkeit, für gerade, horizontale Verbindungen)

1. Taste *Select 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 Mode* drücken und die zu löschende Linie, Funktionsblock, Symbol oder Connector mit der Maus markieren und Taste loslassen.



1.4 Programm aufbauen und in die PCD übertragen

1.4.1 Programm aufbauen (Build)



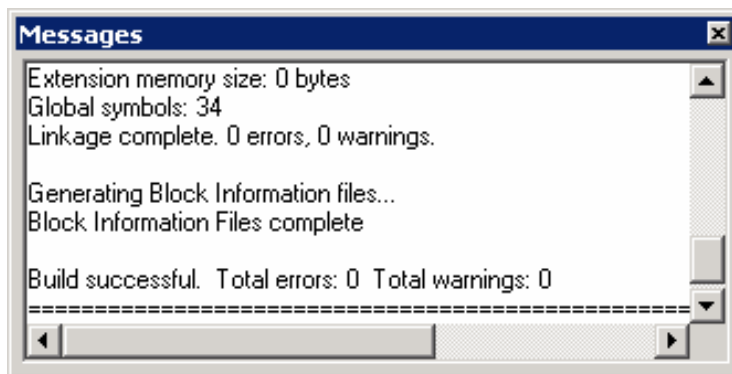
*Rebuild
All Files*

Damit das fertig editierte Programm von der PCD gelesen und ausgeführt werden kann, muss es im Project Manager via Menü *CPU, 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 Programmaufbereitungs-Schritte (*Compiler, Assembler, Linker* etc.) angezeigt. Wenn das Programm korrekt editiert wurde, wird die Build-Funktion mit der Meldung abgeschlossen: *Build successful. 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.



1.4.2 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 (*Settings Online* und *Settings Hardware*) sowie die PC <-> PCD-Verbindung mit dem Kabel PCD7.K111, USB zu überprüfen.

1.5 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.



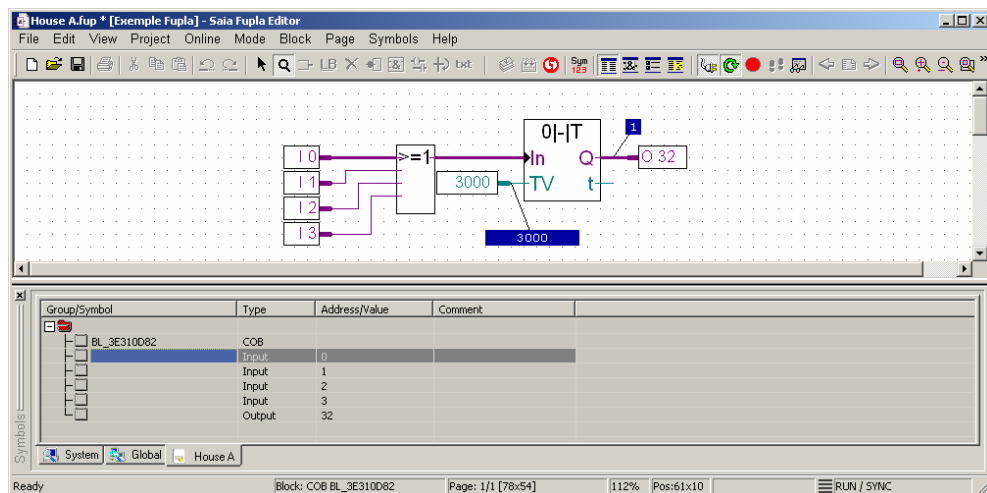
Nach dem Drücken der Taste *Stop* darf die LED *RUN* nicht mehr leuchten.

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 *Place Probe* und den Link mit der Maus.



1.5.1 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	PROJEKT-MANAGEMENT	3
2.1	Einleitung	3
2.2	Projektorganisation	4
2.2.1	Beispiel eines Anwender-Projekts	4
2.2.2	Speichern des Projekts im PC	6
2.2.3	Komprimieren eines Projekts oder einer CPU	6
2.2.4	Bestehendes Projekt öffnen	7
2.2.5	Neues Projekt anlegen	7
2.3	Projektfenster	8
2.3.1	Projektordner	8
2.3.2	Gemeinsamer Ordner	10
2.3.3	CPU-Ordner	10
2.3.4	Online Einstellungen	12
2.3.5	Verbindung PC zu PCD	13
2.3.6	Hardware-Einstellungen	14
2.3.7	Software-Einstellungen	19
2.3.8	Programmdateienordner	21
2.3.9	Dateitypen	22
2.3.10	Dateien verbinden mit "linked"	23
2.3.11	Gemeinsame Dateien	23
2.4	Programm verarbeiten (Building)	24
2.4.1	Rebuild All und Build	25
2.4.2	<i>Build</i> -Optionen	25
2.5	Messages-Fenster	26
2.6	Programm-Download (in die PCD)	27
2.6.1	Download-Optionen	28
2.6.2	Programm auf Flash Card laden (Backup-Speicher)	29
2.6.3	Backup-Speicher und Transfer des Anwenderprogramms	29
2.7	View-Fenster	30
2.7.1	Organisationsblockstruktur	30
2.7.2	Liste der Organisationsblocks	30
2.7.3	Symbolliste	31
2.7.4	Querverweis	31
2.8	Programm-Backup	32
2.9	Selbstladende Dateien	33
2.9.1	Erstellung einer „.sd5“-Datei	33
2.9.2	Herunterladen einer „.sd5“-Datei	34

2 Projekt-Management

2.1 Einleitung

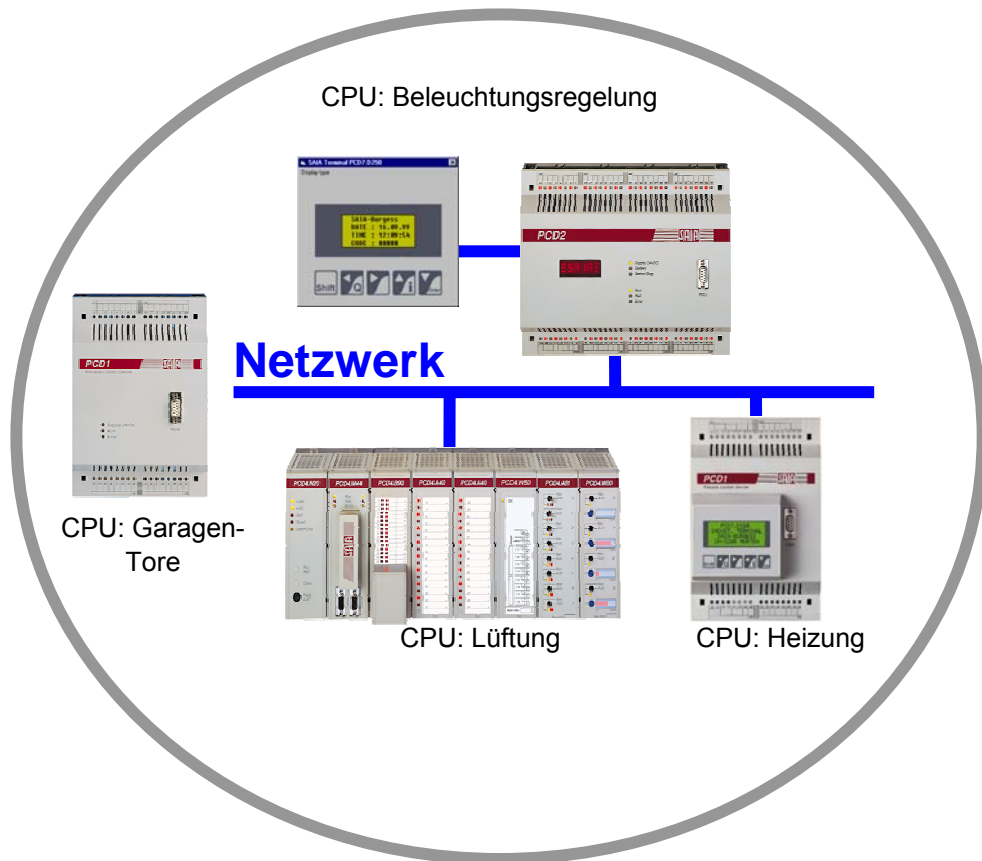
Moderne Automationsanwendungen bestehen häufig aus einer grossen Anzahl von Steuerungen, die in einem Netzwerk miteinander verbunden sind. PG5 verbindet daher in einem einzigen Projekt die Programmierung und Konfiguration aller PCD Steuerungen innerhalb einer Anwendung. Der PG5 Projekt Manager bietet dem Anwender eine umfassende Übersicht aller Informationen, die zu einem Projekt gehören.

2.2 Projektorganisation

2.2.1 Beispiel eines Anwender-Projekts

In der Praxis enthalten Automatik-Installationen fast immer eine Anzahl programmierbarer lokaler PCD Steuerungen, die in einem Kommunikations-netzwerk miteinander verbunden sind. Jede PCD unterstützt die Steuerung einer bestimmten Funktion innerhalb eines Gesamtprojekts, wie z.B.: die Beleuchtungsregelung, die Heizung, die Lüftung, oder die automatischen Tore einer Tiefgarage..

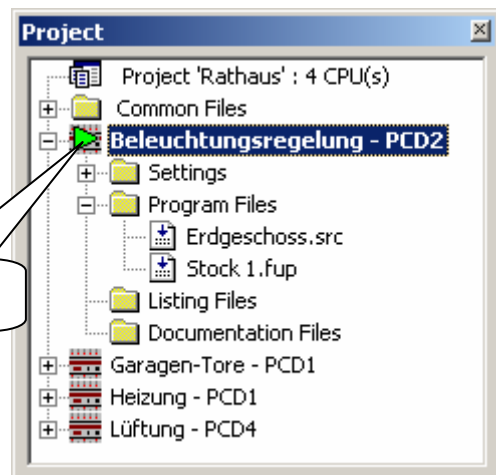
Projekt: Rathaus

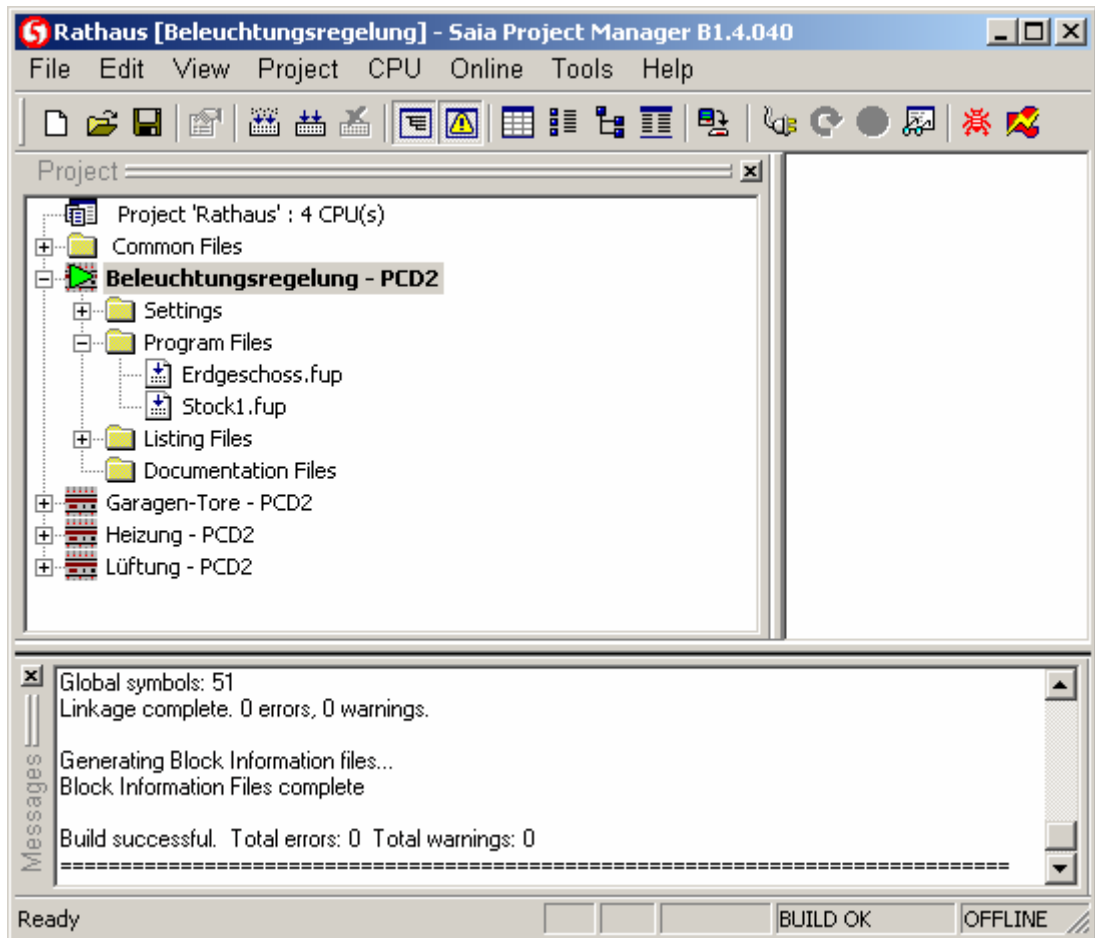


Das PG5 Programmier Werkzeug verbindet in einem einzigen PG5 Projekt alle PCD CPUs, die zu einer bestimmten Anwendung gehören.

Ein grünes Dreieck kennzeichnet die aktive CPU. Alle Befehle für das Verarbeiten, (Herunter-) laden und Testen des Programms nutzen die Konfigurations- und Pro-

Aktive CPU





Der Projekt Manager besitzt drei Fenster:



*Project
Tree*

Das *Project* Fenster zeigt die Struktur des Projekts mit all seinen PCD CPUs. Zum Anzeigen dieses Fensters, den Menüpfad *View, Project Tree* auswählen, oder den Knopf *Project Tree* anklicken.



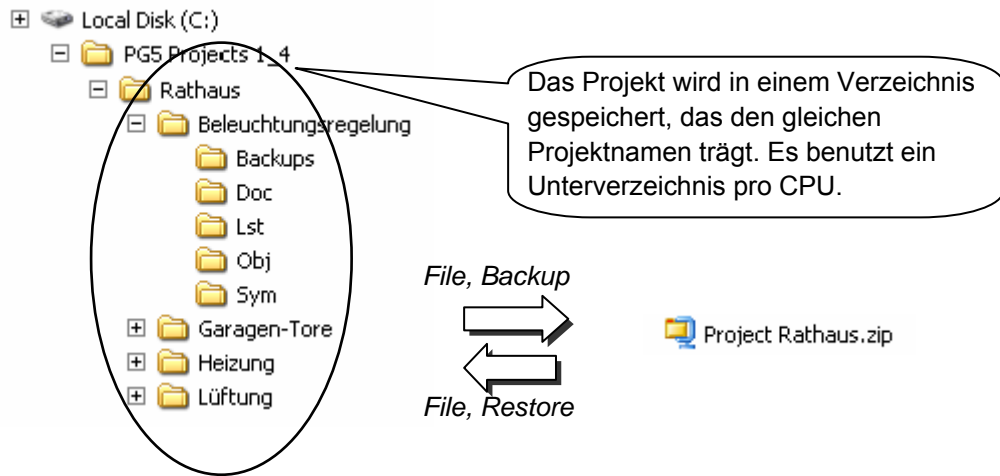
*Message
Window*

Im *Messages* Fenster werden Alarme und Fehlermeldungen angezeigt, die während der Programm-Verarbeitung erzeugt wurden. Zum Anzeigen dieses Fensters, den Menüpfad *View, Message Window* auswählen, oder den Knopf *Message Window* anklicken.

Das *View* Fenster zeigt die View Liste, die Block Liste, die Block Struktur sowie Text Dateien. Es erlaubt ausserdem Querverweise zu Symbolen.

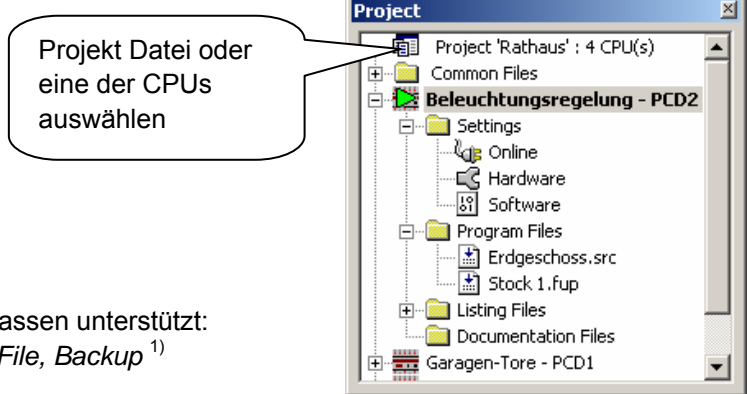
2.2.2 Speichern des Projekts im PC

Durch Voreinstellung werden Projekte im Verzeichnis **C:\PG5 Projects 1_4** abgelegt.

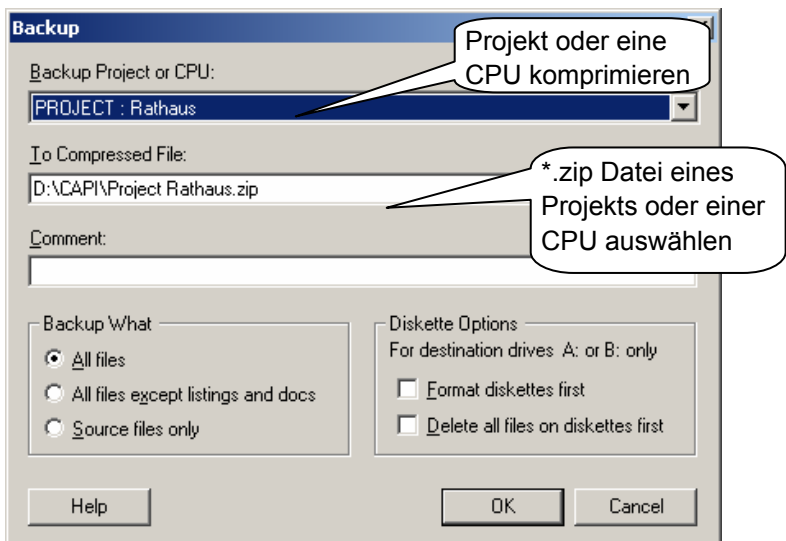


2.2.3 Komprimieren eines Projekts oder einer CPU

Wenn ein Projekt gespeichert wird, muss die ganze Struktur der Verzeichnisse und die darin enthaltenen Dateien konserviert werden. Am einfachsten komprimiert man die ganze Struktur mit dem *Backup* Befehl in eine *.zip Datei.



Dies wird folgendermassen unterstützt:
Menü Befehl *Project File, Backup* ¹⁾

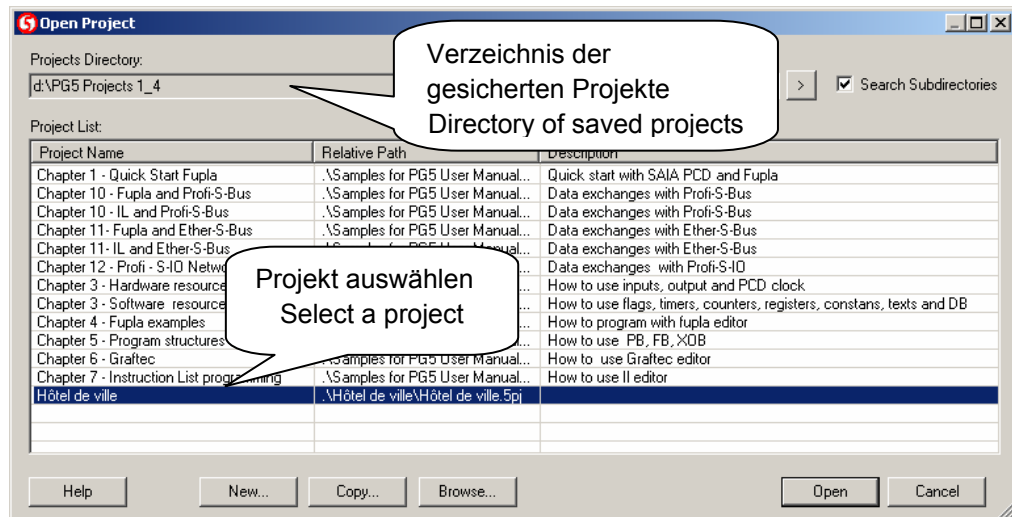


¹⁾ Der *Project, Restore* Menü Befehl stellt ein Projekt/eine CPU wieder her, das/die vorher in einer *.zip Datei gesichert wurde.

2.2.4 Bestehendes Projekt öffnen

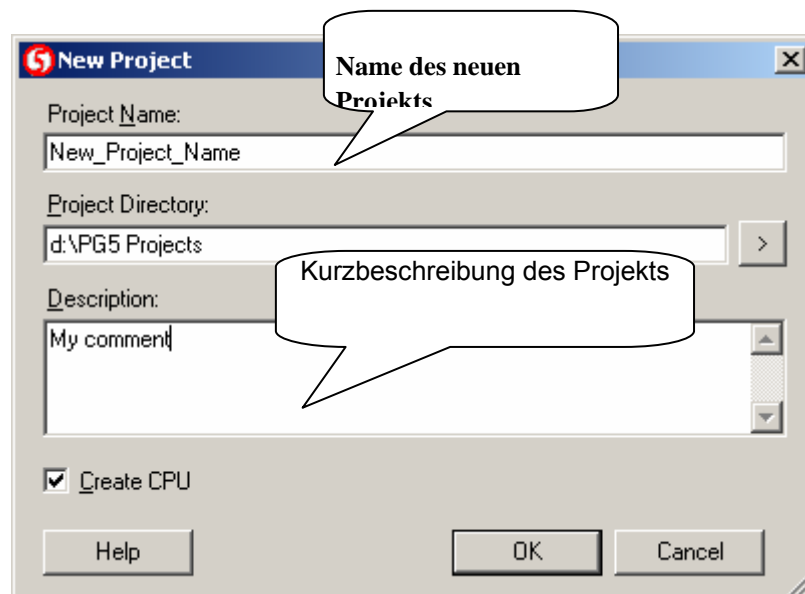
PG5 wird mit allen in diesem Handbuch beschriebenen Beispielen ausgeliefert. Der Menübefehl *Project, Open... File, Open Project* erlaubt diese zu öffnen und auszuprobieren.

Ein bereits bestehendes Projekt kann mit dem Befehl *Project, Open...* geöffnet werden. Alle Projekt-Dateien (.5pj) sind im Projekt-Verzeichnis abgelegt und werden in einer Liste angezeigt. Mit Doppel-Klick auf eines der Projekte in der Liste oder Auswahl eines Projektes und betätigen des *Open* Knopfes öffnet ein Projekt. Weitere Möglichkeit: *Browse* Knopf betätigen und direkt nach der Projekt Datei (.5pj) oder CPU Datei (.5pc) suchen.



2.2.5 Neues Projekt anlegen

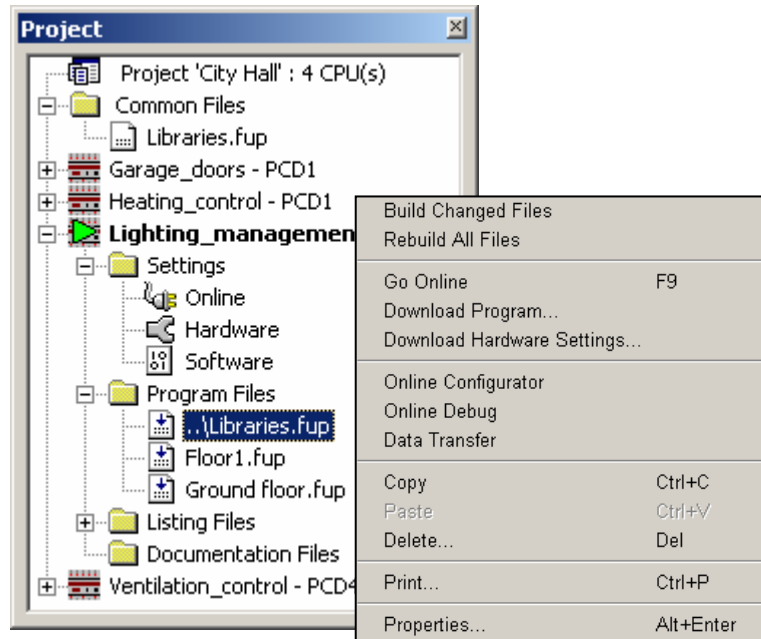
Zum Anlegen eines neuen Projektes den Menü Befehl *Project, New...*, benutzen, Namen des neuen Projektes im Feld *Project Name* festlegen, Option *Create CPU* auswählen und mit *OK* bestätigen.



2.3 Projektfenster



**Projekt
Tree**



Die Ordner im Projekt Fenster gruppieren die Projektinformationen nach bestimmten Organisationskriterien:

2.3.1 Projektordner



Der oberste Ordner zeigt das Projekt mit seinem Namen sowie die Anzahl der enthaltenen CPUs. Um Änderungen in diesem Ordner vorzunehmen, Ordner mit der Maus auswählen und mit der rechten Maustaste das Kontextmenü anzeigen.

<i>New CPU...</i>	Fügt eine neue CPU zum Projekt hinzu
<i>Import CPU...</i>	Importiert CPUs aus einem anderen PG5 Projekt oder aus einem alten PG4 Projekt
<i>Paste CPU</i>	Fügt eine CPU mit ihrem gesamten Inhalt ein
<i>Copy Project...</i>	Kopiert ein Projekt mit seinem gesamten Inhalt
<i>Backup...</i>	Sichert ein Projekt in einer ZIP-Datei (*.zip)
<i>Restore...</i>	Stellt das Projekt aus einer ZIP-Datei (*.zip) wieder her
<i>Rebuild All CPUs Online Commands</i>	Gibt den Befehl für die Implementation eines vollständigen Projekt- <i>Builds</i> oder - <i>Downloads</i>
<i>Print...</i>	
<i>Find...</i>	
<i>Properties</i>	Ermöglichen die Änderung von Informationen, die für den Projektordner spezifisch sind: Projektname, -beschreibung, ...

2.3.2 Gemeinsamer Ordner



Im *Common Files* Ordner sind Module abgelegt, die für mehrere CPUs im Projekt genutzt werden. Um eine Programm-Datei hinzuzufügen, den Ordner auswählen und *New File* aus dem Kontextmenü benutzen.

Mit *Add Files* im Kontextmenü kann eine beliebige PG5 Programm-Datei importiert werden, es können aber auch anwendungsspezifische Dateien und Wartungsdokumente (in Word, Excel, etc.) importiert werden. Diese Dateien werden im PG5 Projekt abgelegt und können mit Doppelklick geöffnet werden.

Anmerkung:


Common files benutzt dieselben *Lokalen Symbole* in jeder CPU, die auf diesen Ordner zugreift. Jede CPU kann jedoch auch eigene *Globale Symbole* benutzen, so können also *Globale Symbole* für jede CPU anders sein.

2.3.3 CPU-Ordner



Jeder CPU Ordner enthält die Konfigurationen und Programme für eine Steuerung im Projekt.

Um im CPU Ordner zu ändern, rechte Maustaste klicken, um das Kontextmenü anzuzeigen.

<i>Set Active</i> 	Wenn eine CPU im <i>Project</i> Fenster ausgewählt ist, wird sie durch diesen Befehl zur aktiven CPU. Die aktive CPU wird durch ein grünes Dreieck gekennzeichnet. Alle Menü- und Tastenbefehle können auf die aktive CPU angewandt werden.
<i>Rebuild Changed files</i> <i>Rebuild All Files</i>	Befehl für die Erstellung eines <i>Builds</i> der im <i>Project</i> Fenster ausgewählten CPU.
<i>Go Online</i> <i>Download Program</i> <i>Download Hardware Settings</i>	
<i>Online Configurator</i> <i>Online Debug</i> <i>Data Transfer</i>	
<i>Copy, Paste</i> <i>Delete</i>	Die im <i>Project</i> Fenster ausgewählte CPU kann hiermit kopiert, eingefügt oder gelöscht werden.
<i>Print...</i>	
<i>Properties</i>	Ermöglichen die Änderung von Informationen, die für den CPU-Ordner spezifisch sind: CPU-Name, -Beschreibung, ...

Anmerkung:

Wenn im Menü *Tools, Options, General page* die Option *Activate CPU according to Project Tree location* gewählt ist, wird der *SAIA Project Manager* die CPU automatisch entsprechend ihrem Verwendungskontext aktivieren. Der Befehl *Set*

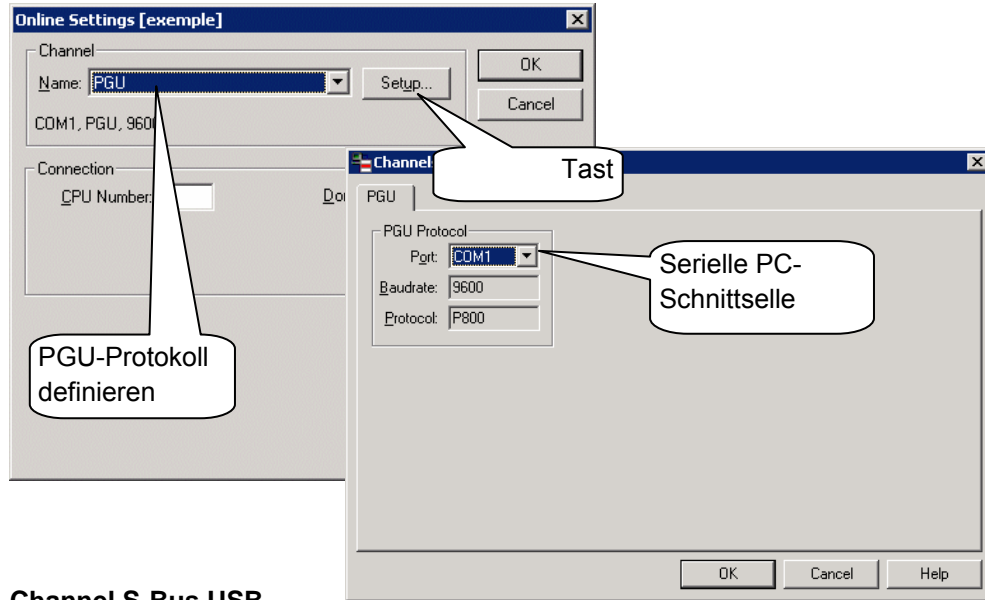
Active kann in diesem Fall nicht verwendet werden und erscheint nicht im Kontextmenü.

2.3.4 Online Einstellungen

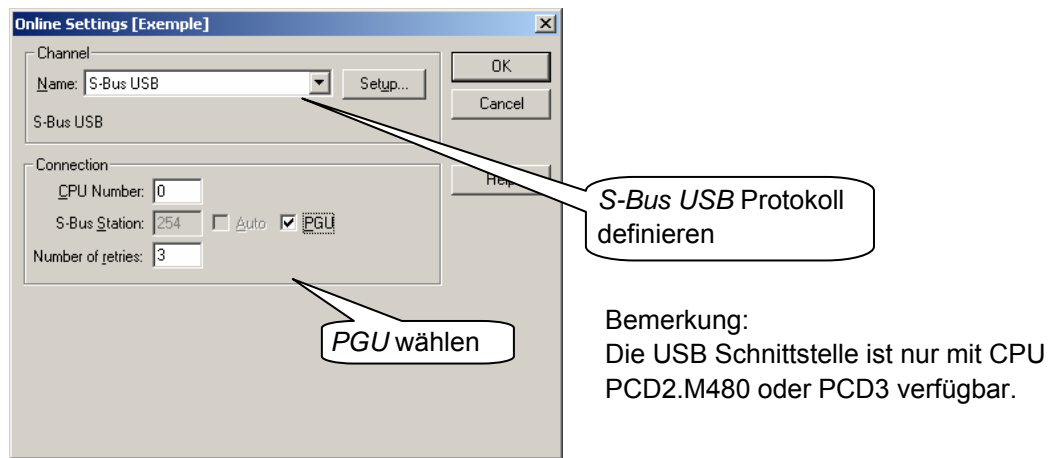


Im *Settings*, *Online* Ordner werden die CPU Kommunikationsparameter festgelegt. Folgende Kommunikationsprotokolle werden unterstützt: PGU, S-Bus, Ethernet, etc. Das PGU- und das *S-Bus USB* Protokoll erlaubt eine direkte Kommunikation mit einer PCD ohne konfigurierten PGU Port in den PCD *Hardware Settings*.

Channel PGU (RS 232)



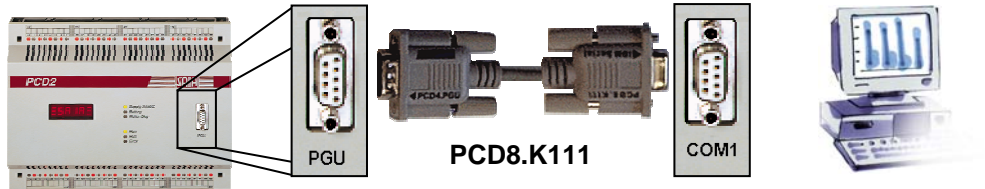
Channel S-Bus USB



2.3.5 Verbindung PC zu PCD

Ein PCD8.K111 Kabel stellt die RS 232 Verbindung zwischen PC und PCD sicher. Weitere Informationen über dieses Kabel sind im PCD Hardware Handbuch zu finden.

Channel PGU (RS 232)



Channel S-Bus USB

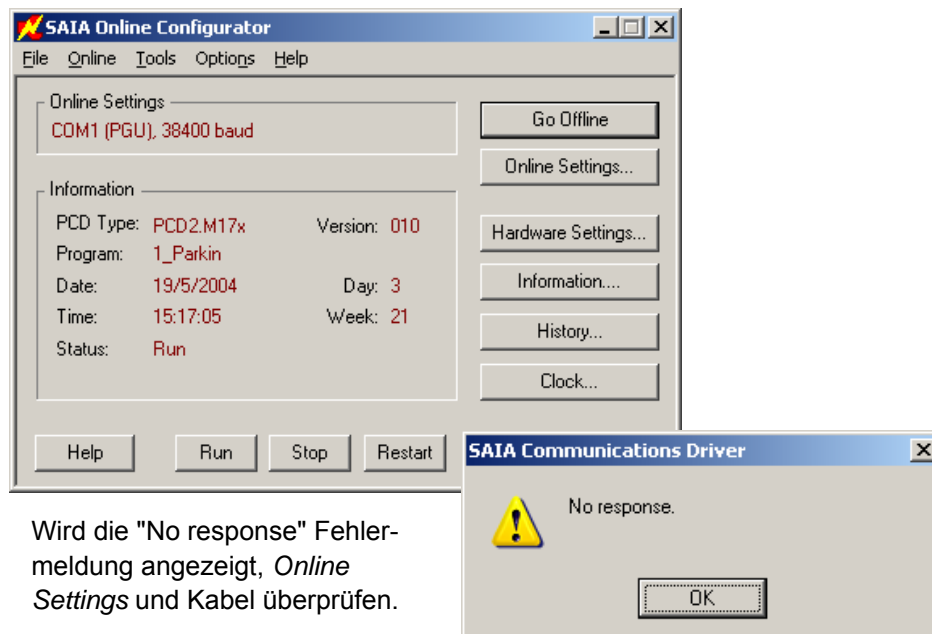
Die USB Schnittstelle ist nur mit der CPU PCD2.M480 oder PCD3 verfügbar.



Online Configurator

Verbindung überprüfen

Mit dem *Online Configurator* Knopf, oder den Menü Befehlen *Tools*, *Online Configurator* können die Einstellungen der angeschlossenen PCD angesehen werden. Erscheint die Information in rot, arbeitet die Kommunikation sauber.

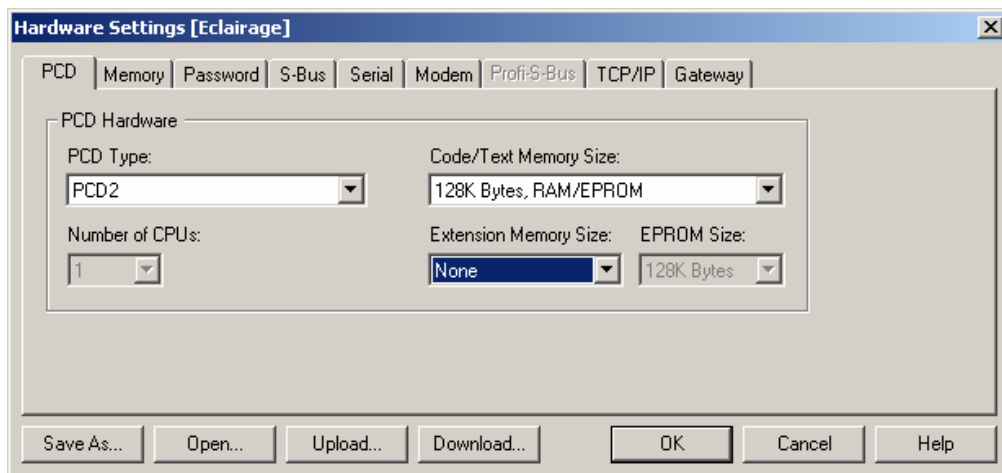


Wird die "No response" Fehlermeldung angezeigt, *Online Settings* und Kabel überprüfen.

2.3.6 Hardware-Einstellungen



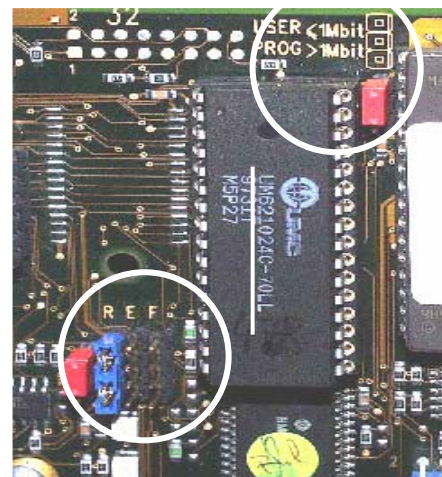
Im *Hardware Settings* Ordner werden Speichergrösse und Kommunikationsparameter der PCD Steuerung festgelegt.




Wird eine Steuerung das erste Mal eingesetzt, oder wird ein neuer Speicher hinzugefügt, muss der Speicher der PCD neu konfiguriert werden. Die Parameter des Fensters oben können auf zwei Arten festgelegt werden:

1. Den *Upload* Knopf betätigen und Einstellungen direkt aus der Steuerung auslesen.
2. Die Einstellungen im Fenster oben anhand der Tabellen auf den nächsten beiden Seiten festlegen. Das Beispiel oben zeigt die fettgedruckten Angaben in diesen Tabellen.

Die folgende Tabelle enthält Informationen über die Speicher-Jumper (Steckbrücken). Diese Jumper müssen auf der PCD CPU Karte gesteckt werden. Weitere Informationen sind im PCD Hardware Handbuch zu finden.



Speicher-Jumper in einer PCD2.M120 mit LM621024 Speicher

PCD Typ	Speicher	Bestell- nummer	Code/Text Memory Size	Extension Memory Size (RAM)	Jumper Position auf PCD
PCS1.C8	Flash 2 MBit		240 kByte	128 kByte	kein
	Flash 4 MBit		²⁾ 1008 kByte	896 kByte	kein
PCD1. M110 /120/150	Leerer Sockel		17 kByte	kein	R
	1 RAM 256 kBit	4 502 5414 0	32 kByte	13 kByte	R
	1 RAM 1MBit	4 502 7013 0	128 kByte	13 kByte	R
	1 Flash 1MBit	4 502 7141 0	112 kByte	13 kByte	E
	1 EPROM 512kBit	4 502 3958 0	64 kByte	13 kByte	E
	1 EPROM 1Mbit	4 502 7126 0	128 kByte	13 kByte	E
PCD2.M110 /120/150 	Leerer Sockel		32/128 kByte ¹⁾	kein	R ,<=1Mbit
	1 RAM 256 kBit	4 502 5414 0	32 kByte	24/128 kByte ¹⁾	R ,<=1Mbit
	1 RAM 1 Mbit		128 kByte	24/128 kByte ¹⁾	R ,<=1Mbit
	1 RAM 4 Mbits	4 502 7013 0	512 kByte	24/128 kByte ¹⁾	R , >1Mbit
	1 Flash 1 MBit	4 502 7175 0	112 kByte	24/128 kByte ¹⁾	F ,<=1Mbit
	1 Flash 4 MBit	4 502 7141 0	448 kByte	24/128 kByte ¹⁾	F , >1Mbit
	1 EPROM 512 kBit	4 502 7224 0	64 kByte	24/128 kByte ¹⁾	E ,<=1Mbit
	1 EPROM 1 MBit	4 502 3958 0		24/128 kByte ¹⁾	
	1 EPROM 4 MBit	4 502 7126 0	128 kByte	24/128 kByte ¹⁾	E ,<=1Mbit
		4 502 7223 0	512 kByte		E , >1Mbit
PCD4.M	2 RAM 62256	4 502 5414 0	64 kByte		RAM
	2 RAM 1 Mbit	4 502 7013 0	256 kByte	172 kByte,	RAM
	2 EPROM 256 kBit	4 502 5327 0	64 kByte	wenn Speicher	E256
	2 EPROM 512 kBit	4 502 3958 0	128 kByte	PCD7.R310	E512
	2 EPROM 1 MBit	4 502 7126 0	256 kByte	verwendet wird	E1M
PCD4.M170	RAM 1 MBit		1024 kByte		kein

¹⁾ 128 kBit für PCD2.M110/120 mit Hardware Version J oder höher, 128 kBit für alle PCD2.M150

²⁾ 1008 kByte für alle PCS1 mit Hardware Version E oder grösser
 Jumper: R= RAM, E = EPROM, F = Flash

Auf dem Speicherchip ist eine Referenzbezeichnung aufgedruckt mit der in der Tabelle unten Bestellnummer und Speichergrosse abgelesen werden können, falls die Speichergrosse der PCD nicht bekannt ist:

Speichergrosse	Bestellnummer	Referenz
RAM 256 kBit	4 502 5414 0	SRM 2B256SLCX70 HY62256ALP-70 GM76C256CLL-70 M5M5256DP-70LL TC55257DPL-70L
RAM 1 MBit	4 502 7013 0	BS62LV1025 PC-70 LP621024D-70LL SRM20100LLC70 HY628100ALP-70 GM76C8128CLL-70 M5M51008BP-70L TC551001BLP-70L
RAM 4 MBit	4 502 7175 0	BS62LV4006PC P55 BS62LV4007PC P55 HM628512LP-5 KM684000ALP-5L KM684000BLP-5L
Flash 1 MBit	4 502 7141 0	AM29F010-70PC
Flash 4 MBit	4 502 7224 0	AM29F040 auf Sockel
EPROM 256 kBit	4 502 5327 0	UPD27C256AD-10 M27C256B-10F1 TMS27C256-10JL
EPROM 512 kBit	4 502 3958 0	AM27C512-15XF1 AMC27C512-15XF1 AM27C512-90DC UPD27C512D-10 M27512-10XF1 M27512-10F1
EPROM 1 MBit	4 502 7126 0	AM27C010-90DC NM27C010Q-90 M27C1001-10F1
EPROM 4 MBit	4 502 7223 0	AM27C040-100DC M27C4001-10F1

PCD-Typ	Code/Text/ Extension Memory Size (RAM)	Backup mit internem Speicher (Flash)	Backup mit externem Speicher (Flash)
PCD4.M170 PCD2.M170 PCD2.M480	1024 kByte	Keines	PCD7.R400 - 1024 kByte
PCD3.M3020P CD3.M3120	128 kByte	128 kByte	PCD3.R500 - 128 kByte
PCD3.M3230 PCD3.M3330	256 kByte	256 kByte	PCD3.R500 - 256 kByte
PCD3.M5440 PCD3.M5540 PCD3.M6340 PCD3.M6540	512 kByte	256 kByte	PCD7.R500 - 512 kByte

Neue PCD-Systeme unterstützen interne oder externe Backup-Speicher: PCD7.R400/R500 (optional).

Der Backup-Speicher ermöglicht es Ihnen, eine RAM-Kopie (Code/Text/Extension) eines Anwenderprogramms auf einem Flash Speicher zu sichern, so dass im Fall eines Stromausfalls oder einer defekten Batterie kein Inhalt verloren geht.

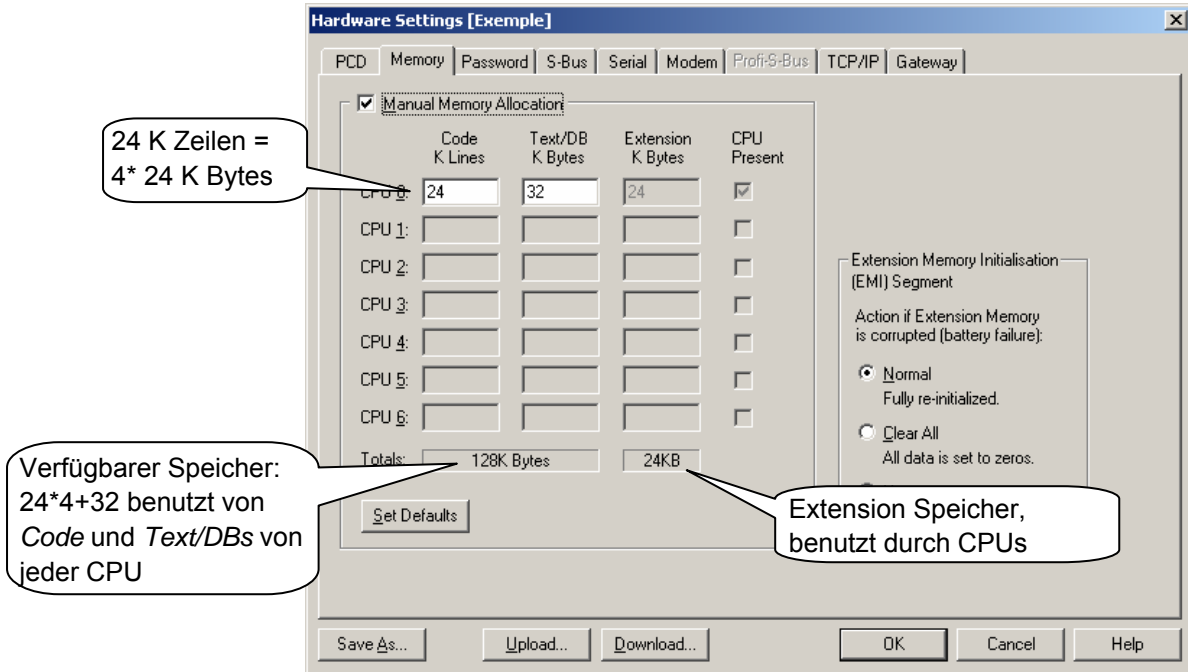
Wir empfehlen die Verwendung eines Backup-Speichers für Ihre PCDs, um sich gegen unerwünschten Datenverlust abzusichern.

Wenn das RAM-Anwenderprogramm (Code/Text/Extension) korrupt ist, stellt ein PCD-Kaltstart das Programm mit Hilfe des Backup-Speichers automatisch wieder her.



Ein externer Backup-Speicher ermöglicht Ihnen auch den Transfer von Anwendungen von einer PLC zur anderen sowie die Erstellung von RAM-Texten und DBs im Extension Memory bei laufender PLC (Adresse ≥ 4000).

Anmerkung:
Die Quelldateien des Projekts müssen, unabhängig von der Ablage im Backup-Speicher, stets gesichert werden. Die Quelldateien werden nicht im PCD Speicher gespeichert.



Der verfügbare Speicher, wie in den PCD Einstellungen festgelegt ist aufgeteilt zwischen Programm Code und Text in jeder CPU. Einige PCDs haben mehrere CPUs: PCD4.M44x und PCD6.Mxxx.

Für eine einzelne CPU wird dies entsprechend dem Anwenderprogramm automatisch festgelegt, so kann *Manual Memory Allocation* ungeprüft bleiben.

Die Vorgabeparameter sind für die meisten Anwendungen ausreichend. In Anwendungen, wo dies nicht der Fall ist, kommt eine Fehlermeldung beim herunterladen des Programms in die PCD:



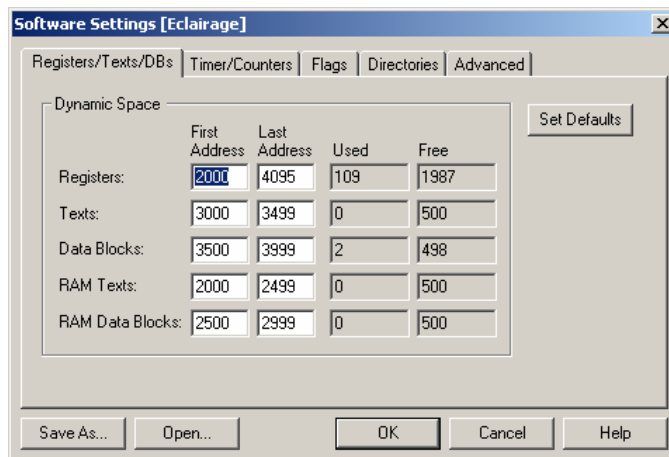
Es gibt mehrere Arten diesen Fehler zu beheben:

- *Manual Memory Allocation* zurücksetzen und PG5 die Aufteilung von Code/Text überlassen, wenn genügend Speicher vorhanden ist.
- *Manual Memory Allocation* setzen und den Speicherplatz gemäss den Angaben in der Fehlermeldung konfigurieren.
- PCD Speicherkapazität erhöhen.

Download...

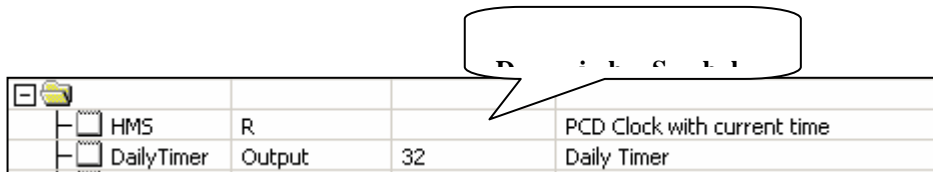
Sind die *Hardware Settings* fertiggestellt, nicht vergessen die Daten immer in die PCD herunterzuladen: *Download* Knopf betätigen, oder Menü Befehle *Online, Hardware Settings, Download* wählen.

2.3.7 Software-Einstellungen



In diesem Fenster werden Adressbereiche für Register, Zähler, Timer und dynamische Flags reserviert. Während der Programmverarbeitung, werden diese Adressen automatisch dynamischen Symbolen zugewiesen, definiert durch das Anwenderprogramm und Fupla FBoxes.

Ein dynamisches Symbol ist eines für das keine absolute Adresse definiert ist:

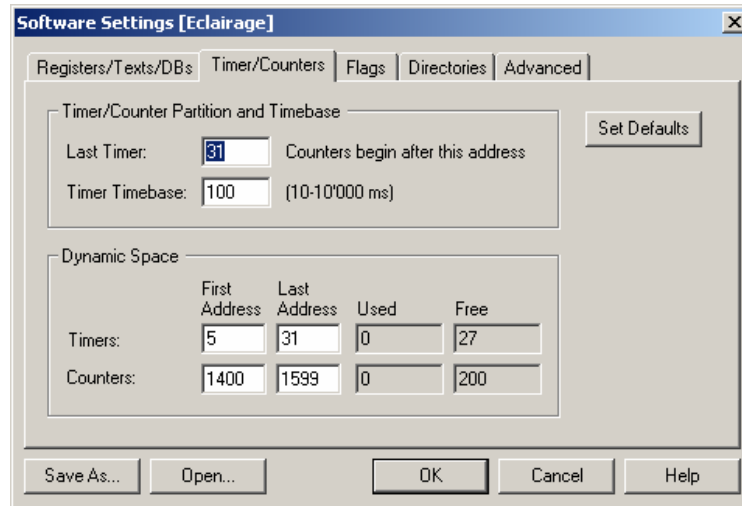


Es ist nicht immer notwendig die dynamischen Adressen zu ändern. Die vorgegebenen Einstellungen genügen für die meisten Anwendungen.

Wenn jedoch während der Verarbeitung eines umfangreichen Programms folgende Fehlermeldung erscheint:

Fatal Error 368: Auto-allocation/dynamic space overflow for type: R
 muss der dynamische Adressbereich für den Media Typ, der in der Fehlermeldung angezeigt wird, erweitert werden.

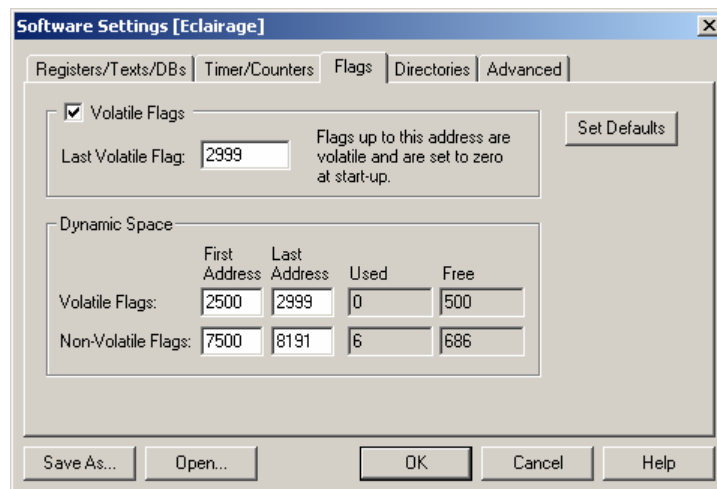
Ist die Steuerung mit EPROM oder Flash Speicher ausgerüstet, müssen den dynamischen Adressbereichen der *RAM Texte* und der *RAM View Blocks* Adressen ab 4000 aufwärts zugewiesen werden, so dass diese Texte und DBs im beschreibbaren RAM Speicherbereich liegen.



In den PCDs sind 31 Timer konfiguriert, von denen einige ihre Adressen dynamisch zugewiesen bekommen. Bei bestimmten Programmen kann es nötig werden, die Anzahl der Timer zu erhöhen..

Die Zeitbasis auf der Timer zählen ist 0.1 Sekunden (100ms). Wenn notwendig kann ein anderer Wert gesetzt werden. Zu beachten ist, dass die Zeitbasis keinen Einfluss auf Fupla Programme hat. Von diesem Parameter werden nur IL Programme berührt.

Eine unnötig hohe Anzahl Timer sowie eine unnötig kleine Zeitbasis verlangsamen die Programm Zykluszeiten.



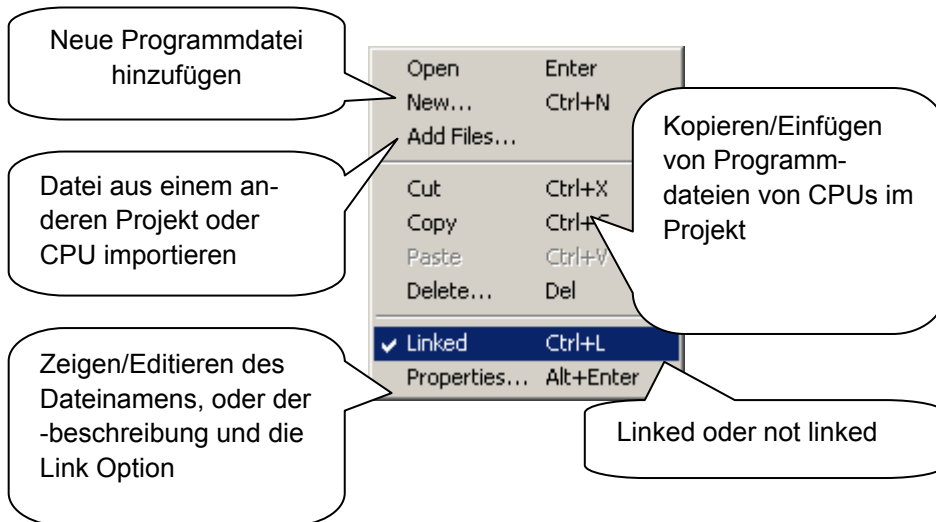
Per Voreinstellung sind alle Flags nicht flüchtig. Wenn nötig, kann mit dem Parameter *Last Volatile Flag* ein flüchtiger Bereich festgelegt werden. (Im Beispiel sind flüchtige Flags in den Adressen F 0 bis F 2999 vorgesehen.) Flüchtige Flags werden bei Programmstart immer auf 0 gesetzt, nichtflüchtige Flags behalten ihren Wert.

2.3.8 Programmdateienordner

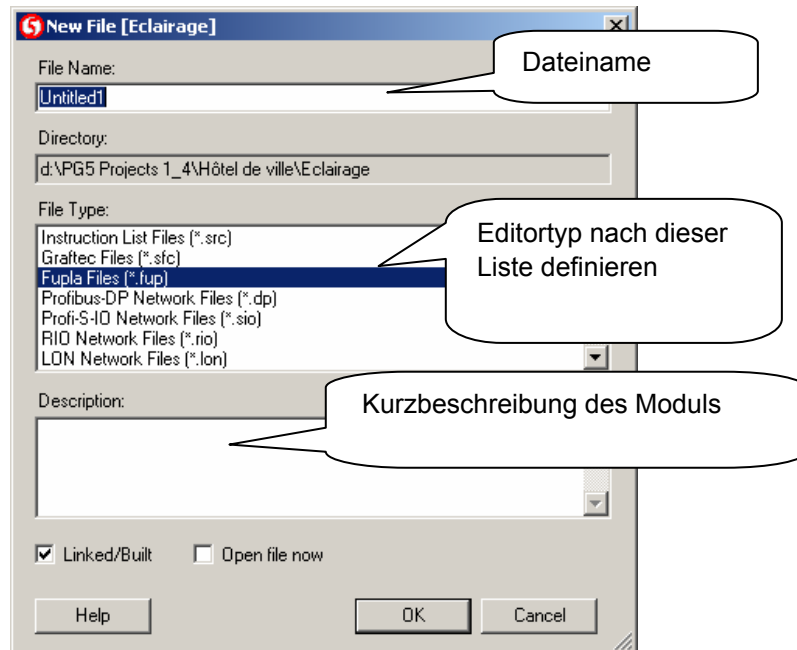


Dieser Ordner enthält die Dateien des CPU Programms.

Zum Ändern einer Datei im Programm Ordner, mit der rechten Maustaste auf den Ordner oder die Datei klicken, zum Anzeigen des Kontextmenüs.



Wird eine neue Datei dem Programm Ordner hinzugefügt, Dateiname und Typ definieren.



2.3.9 Dateitypen

Eine CPU kann mehrere Programm Dateien verschiedenen Typs enthalten. Jeder Datei-Typ besitzt einen entsprechenden anwendungsspezifischen Editor.

Instruction list Editor (*.src)

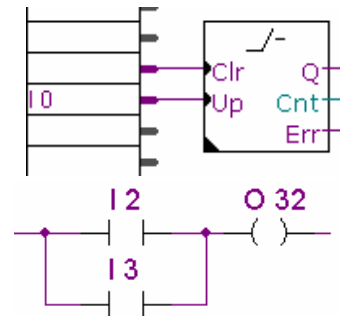
Programmierung in Textform mit einem Befehlssatz von 127 Instructions. Geeignet für alle Anwendungen, erfordert jedoch eine gewisse Programmiererfahrung.

```

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

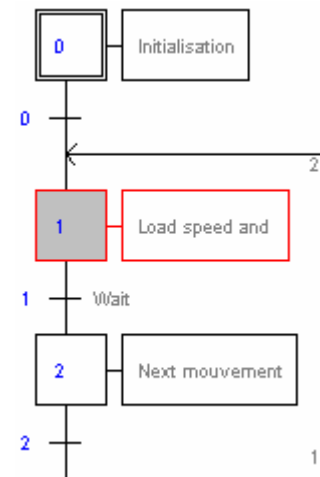
Fupla Editor (*.fup)

Programme in gezeichneter Form, wie Funktionspläne und Kontaktprogramme. Erfordert keine Programmiererfahrung. Viele Bibliotheken verfügbar für die rasche Implementation von HEAVAC Anwendungen und Kommunikations-Netzwerken (Modem, LON, Belimo, EIB, etc.).



Graftec Editor (*.sfc)

Dies ist ein Werkzeug für strukturierte Programmierung in IL (Instruction list) und Fupla. Besonders geeignet für sequentielle Anwendungen mit Warteschlaufen für interne oder externe Ereignisse. Das ideale Werkzeug für die Programmierung von Maschinen mit Befehlen für Motoren, Schalter, etc.



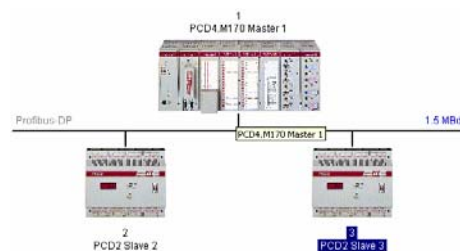
HMI Editor

Erlaubt die Konfiguration von Dialogen mit PCD7.D1xx und PCD7.D2xx Anzeigeeinheiten (zusätzlich im PG5 installiert)

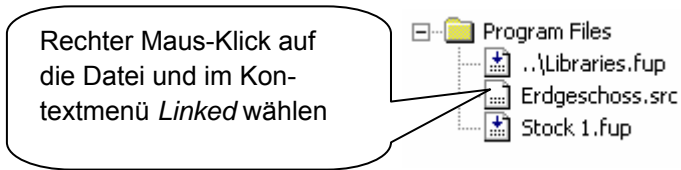



S-Net Editor (*.dp, *.lon, *.rio)


Unterstützt die Konfiguration von Kommunikations-Netzwerken: PROFIBUS-DP, LON und SRIO.



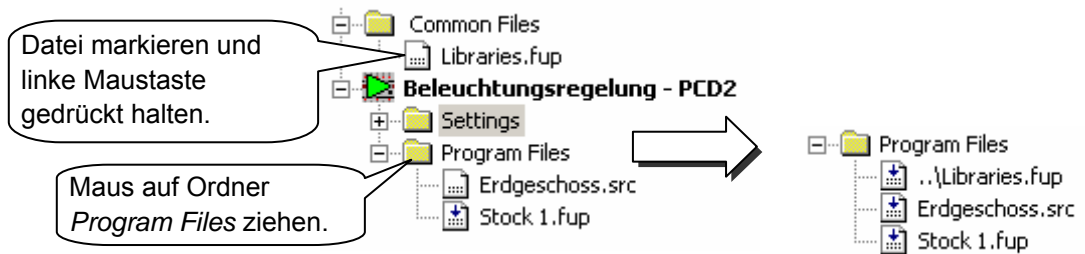
2.3.10 Dateien verbinden mit "linked"



 Dateien mit diesem Icon und einem Pfeil darin sind zu einem Programm verbunden (linked), das in den Speicher der PCD geladen werden kann.

 Dateien mit diesem Icon ohne Pfeil sind nicht Teil des Programms. Diese Dateien werden ignoriert und nicht in den PCD Speicher geladen. Dies kann hilfreich sein für Module zu Testzwecken, die jedoch nicht im endgültigen Programm erscheinen sollen.

2.3.11 Gemeinsame Dateien



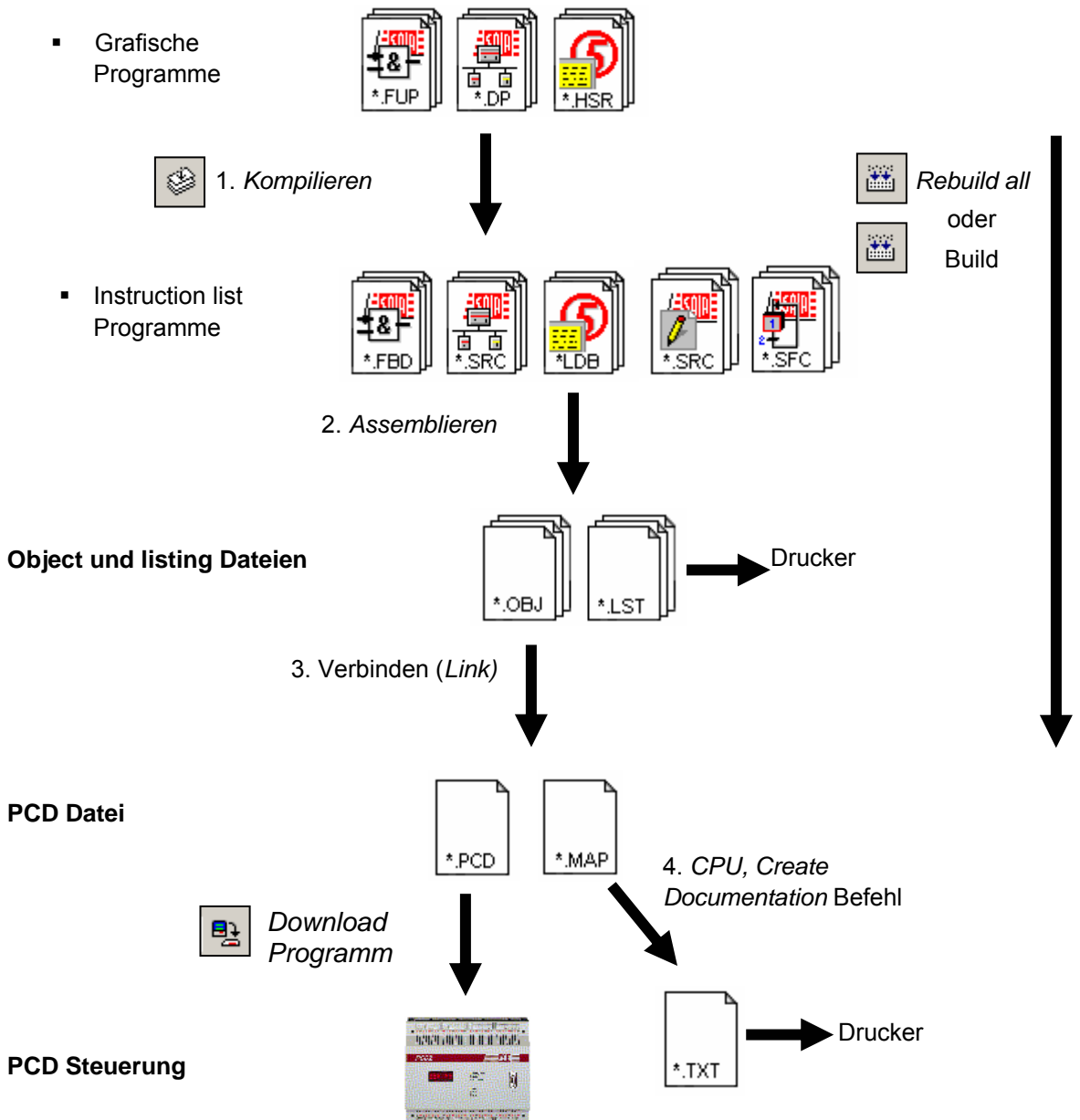
Dateien im *Common Files* Ordner können in den Programm Ordner einer CPU kopiert, eingefügt oder gezogen werden. Vor dem Dateinamen der kopierten oder gezogenen Datei sind zwei Punkte zu sehen, dies bedeutet, diese Datei stammt aus einem Ordner der nächst höheren Ebene.

Die Datei kann im *Common Files* Ordner oder im *Program Files* Ordner der CPU editiert werden. In beiden Fällen ändert der Anwender die gleiche Datei und die Änderungen wirken sich auf alle verbundenen (linked) CPUs aus.

2.4 Programm verarbeiten (Building)

Die PCD kann die Programme nach dem Editieren in Fupla, IL, Graftec, S-Net oder HMI nicht direkt ausführen. Die Dateien müssen zuerst in verschiedenen Schritten aufbereitet werden, wie dieses Diagramm zeigt:

Quelldateien:



1. Kompilieren wandelt grafische Dateien in Instruction list Dateien (*.fbd, *.src, *.hsr)
2. Assemblieren erzeugt binäre Object Dateien (*.obj) sowie einen Bericht (*.lst), dieser kann ausgedruckt und für die Fehlersuche bei bestimmten Assembler Fehlern benutzt werden.
3. Beim Verbinden (link) werden Object Dateien (*.obj) zu einer einzigen ausführbaren Datei (*.pcd) verbunden und in die Steuerung geladen.
4. Die Menü Befehle *CPU, Create Documentation* generieren eine Dokumentation, die im *Documentation Files* Ordner abgelegt wird.

2.4.1 Rebuild All und Build



Rebuild
all

Der *Rebuild All* Knopf start die Kompilierung, Assemblierung und das Verbinden (link) aller Dateien der aktiven CPU.

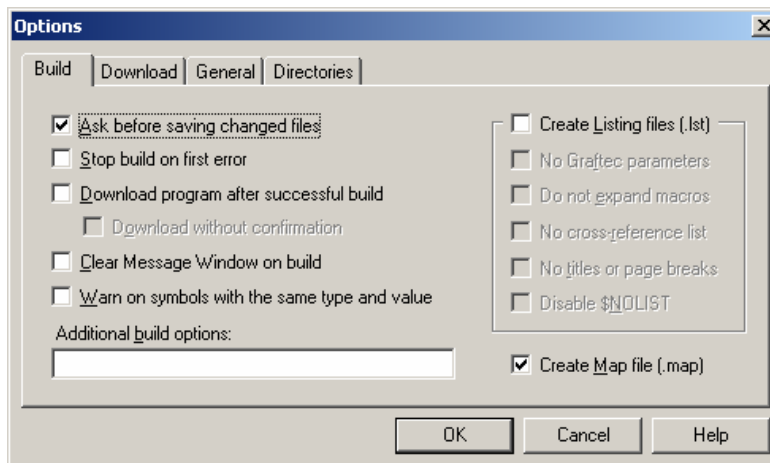


Build

Der *Build* Knopf bewirkt dasselbe, jedoch nur für Dateien, die seit der letzten Verarbeitung geändert wurden. Dies spart Zeit beim Verarbeiten eines umfangreichen Programms.

2.4.2 Build-Optionen

Mit den Menü Befehlen *Tools, Options* können weitere *Build* Optionen gesetzt werden:



Ask before saving changed files before a build

Wenn angehakt, fragt PG5, ob Quelldateien, die geändert, aber vor dem verarbeiten nicht gespeichert wurden, jetzt zu speichern sind. Wenn nicht angehakt, werden die Dateien automatisch gespeichert.

Stop build on first error

Ist diese Option ausgewählt, wird die Verarbeitung abgebrochen, sobald der erste Fehler im *Messages* Fenster erscheint.

Download program after successful build

Ist diese Option ausgewählt, erfolgt automatisch der *Download* des Programms in die PCD, jedoch nur, wenn während der Verarbeitung kein Fehler auftrat.

Download without confirmation

Normalerweise beginnt das Laden des Programms in den PCD Speicher mit einer Dialog Box, in der der Anwender vermerkt wird und die mit *OK* bestätigt werden muss. Ist diese Option ausgewählt, erfolgt der download des Programms direkt, ohne Anzeige der Dialog Box.

Clear Message Window on build

Das *Messages* Fenster wird bei Beginn jeder Verarbeitung gelöscht.

Create listing file

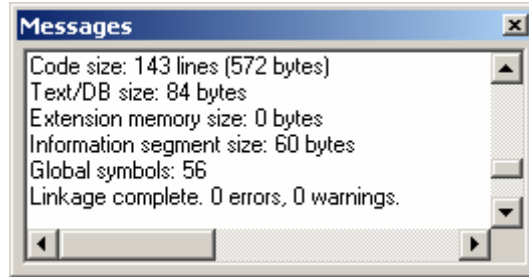
Generiert einen Bericht (*.lst)

Create map file

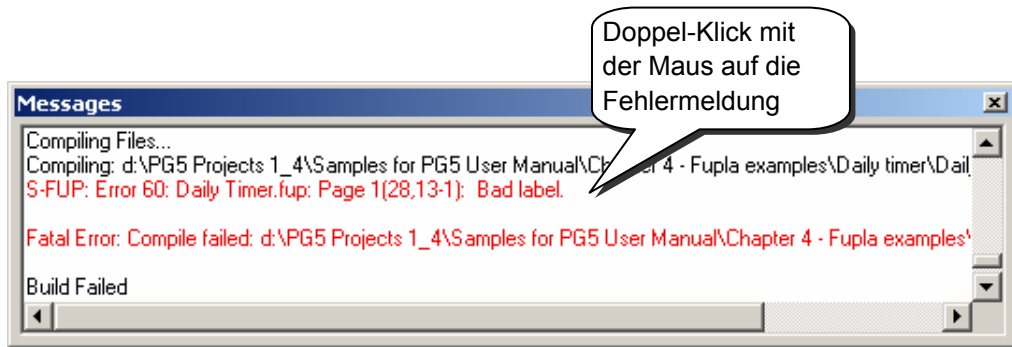
Generiert eine Datei, die den Speicherplatzbedarf zeigt, den eine Anwendung belegt sowie eine Liste der globalen Symbole.

2.5 Messages-Fenster

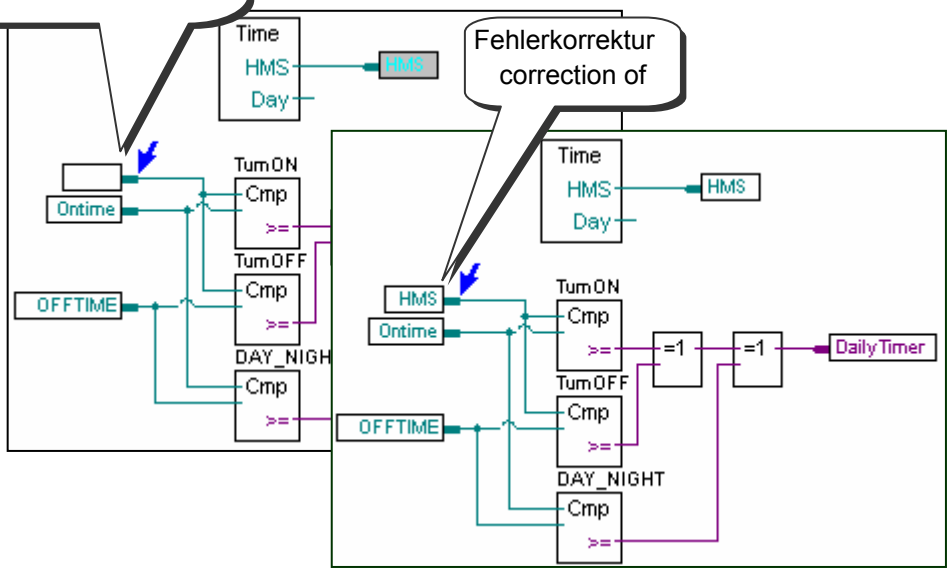
Das *Messages* Fenster enthält Informationen über den Fortschritt einer Programmverarbeitung (*build*). Es zeigt die verschiedenen Schritte der Verarbeitung: Kompilieren, Assemblieren und Verbinden (link). Wurde das Programm korrekt editiert, endet die Verarbeitung mit der Meldung: *Build successful. Total errors 0 Total warnings: 0*



Fehlermeldungen werden in roter Schrift angezeigt. Doppel-Klick mit der Maus auf die Fehlermeldung zeigt im Allgemeinen den Fehler im Anwenderprogramm.



Der Fehler wird rot oder mit einem Pfeil markiert
The error is marked in red or with an arrow

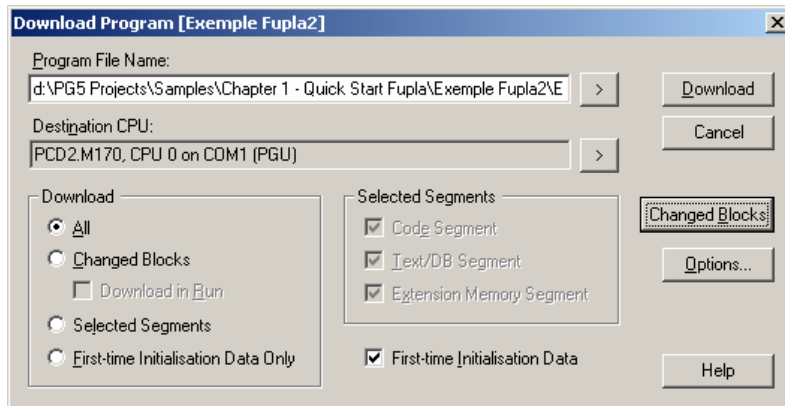


2.6 Programm-Download (in die PCD)



Download Program

Wurde die Verarbeitung fehlerfrei abgeschlossen, kann mit dem *Download Program* Knopf oder den Menü Befehlen *Online*, *Download Program* das Programm in den Speicher der PCD geladen werden.



Program File Name

Durch Voreinstellung ist dies der Programmname der aktiven CPU.

All

Lädt das gesamte Programm (Code Segment, Text/DB Segment, Extension Memory Segment)

Changed Blocks

Lädt nur Blöcke (COB,PB,FB,SB,ST,TR,XOB), die seit dem letzten *Download* geändert wurden. Dieser Option hilft bei geringfügigen Programmänderungen Zeit sparen. Mit dem *Changed Blocks* Knopf kann eine Liste der geänderten Blöcke angezeigt werden.

Download in Run

Erlaubt das laden von geänderten Programm-Blöcken ohne den Programm-Ablauf zu unterbrechen. Die korrekte Ausführung dieser Option hängt von den getätigten Programm-Korrekturen ab.

Selected Segments

Lädt nur Segmente die unter *Selected Segments* definiert sind:

Code Segment = Programm, *Text/DB Segment* = Text und DB 0...3999,

Extension Memory Segment = Text und DB 4000...7999.

First-time Initialisation Data Only

Lädt nur die unten beschriebenen Daten.

First-time Initialisation View

Diese Option initialisiert bestimmte Daten während der Programm Verarbeitung.

Diese Daten sind folgendermassen definiert:

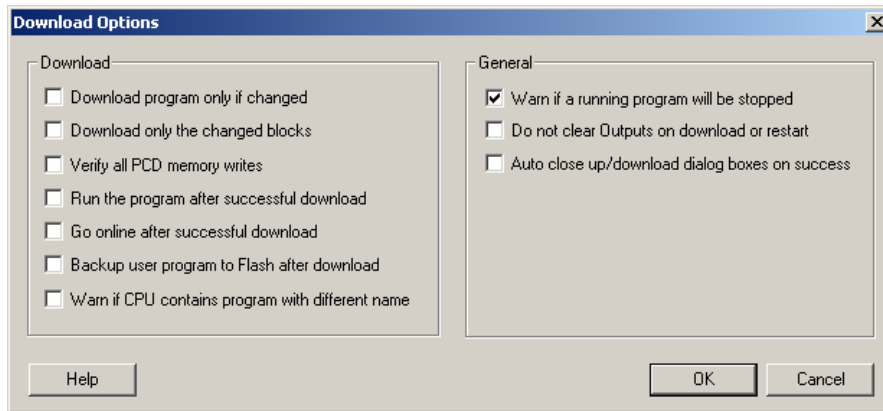
symbol type address := initialisation_value

Group/Symbol	Type	Address/Value	Comment
<ul style="list-style-type: none"> Symbol0 Symbol1 	R	10:= 314	First time initialisation value = 314
	R	11	

Sollen diese Daten nicht mit dem Laden des Programms initialisiert werden, so kann dies auch bei jedem Kaltstart durch einen Kode im XOB16 angestossen werden.

2.6.1 Download-Optionen

Die *Download* Optionen werden mit den Menü Befehlen *Tools, Options* oder mit dem *Options* Knopf in der *Download Program* Dialog Box festgelegt. Dies ermöglicht eine anwenderspezifische *Download* Prozedur.



Download program only if changed

Programm nur laden, wenn es sich von dem im PCD Speicher bereits vorhandenen Programm unterscheidet.

Download only the changed Blocks

Siehe vorhergehende Seite.

Verify all PCD memory writes

Alle zur PCD geschriebenen Daten werden zurück gelesen und miteinander verglichen. Diese Option verdoppelt die Download-Zeit und sollte deswegen nur bei Bedarf ausgewählt werden.

Run the program after successful download

Setzt die CPU nach einem Download automatisch in den Run-Zustand.

Vorsicht: Diese Option nur wählen, wenn das Programm korrekt arbeitet und ein mögliches Risiko für Mensch und Gut bei Ausfall ausgeschlossen werden kann.

Go online after successful download

Setzt die CPU nach einem Download automatisch Online.

Backup user program to Flash after download

Kopiert das Programm automatisch auf den Flash¹ Backup-Speicher.

Auch wenn diese Option nicht ausgewählt wird, kann nach dem Download eine Kopie erstellt werden. Verwenden Sie hierzu das Menü *Online: Flash Backup/Restore*.

Warn if CPU contains program with different name

Vergleicht den in der PCD vorhandenen Programmnamen mit dem Namen des geänderten Programms. Wenn sich diese Programmnamen unterscheiden, wird eine Meldung angezeigt, um das Herunterladen der falschen CPU zu verhindern.

Warn if a running program will be stopped

Programm Download hält die PCD an. Ist diese Option ausgewählt, wird bevor die PCD anhält, eine Warnmeldung angezeigt.

Do not clear Outputs on download or restart

Diese Option kann bei HEAVAC Anwendungen hilfreich sein. Sie verhindert während eines Downloads das Abschalten z. B. der Ventilation oder der Beleuchtung. Nicht bei anderen Anwendungen auswählen.

Auto close Up/Download dialog boxes on success

Wenn diese Option ausgewählt wurde, bleibt das *Up/Download* Dialogfenster nur sichtbar, wenn ein Fehler aufgetreten ist.

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

2.6.2 Programm auf Flash Card laden (Backup-Speicher)

Ist die PCD mit einer Flash card ¹⁾ ausgerüstet, kann mit den Menü Befehlen *Online*, *Flash Backup/Restore* ein Programm in den PCD RAM Bereich geladen und dann auf die Flash Card kopiert werden, und umgekehrt. Durch Auswahl der entsprechenden *download option* kann dies automatisiert werden.

2.6.3 Backup-Speicher und Transfer des Anwenderprogramms

Der Backup-Speicher kann zum Transfer eines Anwenderprogramms von einer PCD zu einer anderen PCD derselben Art verwendet werden:

- Programm auf Backup-Speicher laden
- Bevor der Backup-Speicher entfernt wird, wird die Verbindung zur PCD unterbrochen.
- Bevor der Backup-Speicher angeschlossen wird, wird die Stromversorgung der PCD unterbrochen.
- Batterie entfernen oder „Backup Memory“-Taste für 3 Sekunden gedrückt halten (nur bei PCD7.R400)
- Stromversorgung zur PCD wiederherstellen. Die LEDs blinken, während das Anwenderprogramm mit Hilfe des Backup-Speichers wiederhergestellt wird.
- Batterie einlegen, um Fehlermeldung *Battery Fail* zu vermeiden

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

2.7 View-Fenster

Informationen in diesem Fenster sind nur verfügbar, wenn die Programm-Verarbeitung (*build*) erfolgreich beendet wurde.

2.7.1 Organisationsblockstruktur

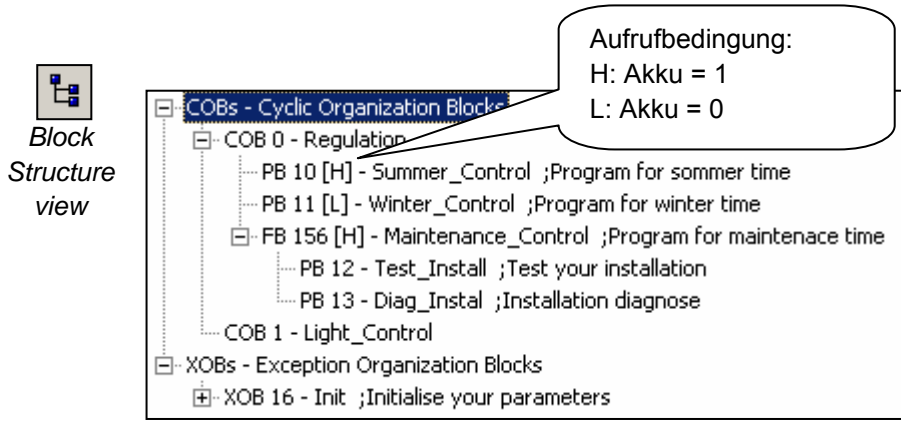
Das SAIA®PCD Programm ist in verschiedenartige Organisationsblöcke strukturiert, in die der Anwender seine Programme speichert.

Je nach Anwendung besitzt jede Blockart bestimmte Eigenschaften: so z. B. für zyklische Programmierung (COB), für sequentielle Programmierung (SB) für Unterprogramme (PB), Funktionen mit Parametern (FB), Ausnahme-Routinen (XOB).

Nach der Programm-Verarbeitung kann man mit dem *Block Structure view* Knopf oder dem Menü Befehl *View block Structure* die Gesamtstruktur der Organisationsblöcke, die das Programm bilden, zeigen.

Das Beispiel unten zeigt ein Programm, bestehend aus den Blöcken: COB 0, COB 1, XOB16 PB 10, PB11 und FB 156.

Anmerkung: COB 0 ruft die drei Unterprogramme (PB 10, 11 und FB 156) unter bestimmten Bedingungen auf. Die Bedingungen stehen in eckigen Klammern.



2.7.2 Liste der Organisationsblocks

Der *Block List view* Knopf, oder der Menü Befehl *Block List* zeigt eine Liste aller Blöcke, die das Programm bilden.

Block List view

Block Name	Type	ID	Description
Regulation	COB	0	
Summer_Control	PB	10	Program for sommer time
Winter_Control	PB	11	Program for winter time
Maintenance_Control	FB	156	Program for mainten...
Light_Control	COB	1	
Init	XOB	16	Initialise your parameters
Test_Install	PB	12	Test your installation
Diag_Instal	PB	13	Installation diagnose

2.7.3 Symbolliste

Die Menü Befehle *View, Global Symbols* und *View, View List* zeigen die im Programm benutzten Symbole:

- *Global Symbols* zeigt den *Symbol Editor*, der die Symbole festlegt, die von allen Dateien der aktiven CPU benutzt werden. Diese Symbole werden hier editiert.
- *View List view* zeigt alle Symbole, die von der aktiven CPU benutzt werden. Diese Liste ist nicht editierbar. Nicht benutzte Symbole werden nicht angezeigt.



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

2.7.4 Querverweis

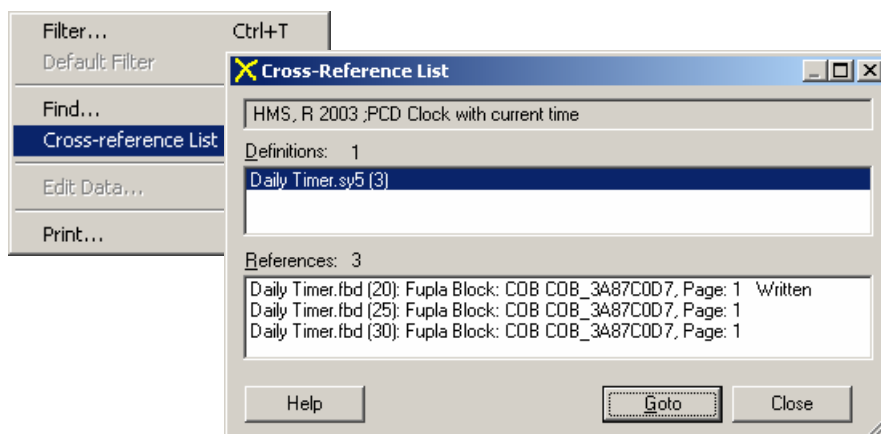
Global Symbols und *View List view* bieten die Möglichkeit, Symbole auszuwählen und ihre Querverweisliste anzuzeigen, z. B. eine Liste aller Programmpositionen, wo ein Symbol vorkommt.

Jeder Eintrag zeigt den Dateinamen und den Block, wo das ausgewählte Symbol vorkommt, versehen mit der entsprechenden Zeile oder der Seitenzahl. Ausserdem wird mit dem Wort *Written* angezeigt, ob dieses Symbol an dieser Position geändert werden kann.

Die *Definitions* Liste zeigt, wo das Symbol definiert ist, z. B. wo ist sein IL EQU Statement zu finden. Die *References* Liste zeigt, wo das Symbol im Programm vorkommt.

Für Blöcke: '>>' zeigt, wo ein bestimmter Block gefunden werden kann.

Zum Zeigen des Programms, wo das Symbol vorkommt, die Definition oder die Referenz auswählen und *Goto* drücken.



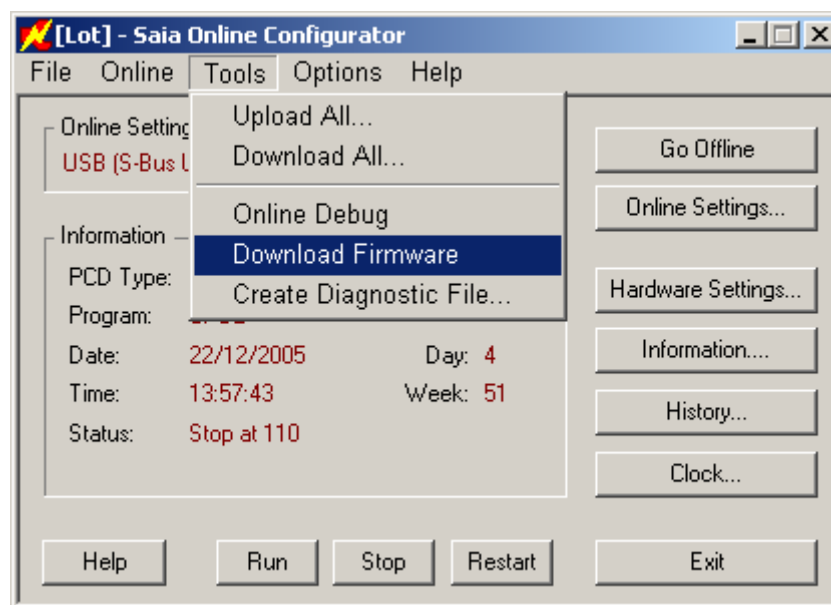
2.8 Programm-Backup

Eine Programmänderung führt manchmal zu einem unerwünschten Ergebnis. Zum Beispiel entsprechen die verfügbaren Quelldateien nicht der letzten SW-Version oder man ist nicht vertraut mit der Installation, etc.

Um solchen Problemen aus dem Wege zu gehen, ist es besser den gesamten Inhalt des PCD-Speichers zu sichern und bei Bedarf wieder zurückzu- speichern..

Der Online Configurator *Tools, Upload All* Befehl sichert den gesamten PCD Speicher in einer einzigen Datei (einschliesslich Programm, Hardware Einstellungen, Werte von Registern, Flags, Zähler, DBs und Texte).

Zum rückspeichern des Programms in den PCD-Speicher, dient der Befehl *Tools, Download All* und dann die Datei auswählen.



Anmerkung:

Backups können auch mit Hilfe des Backup-Speichers erfolgen:
PCD7.R400/R500

2.9 Selbstladende Dateien

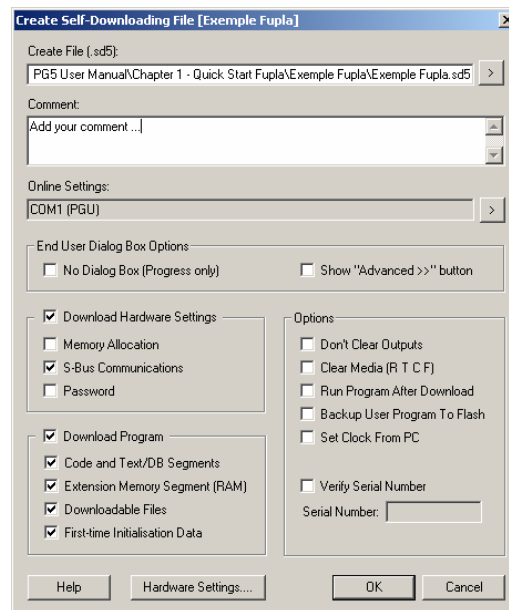
Die selbstladenden Dateien vereinfachen das Herunterladen von Programmen und *Hardware Settings* auf PCDs vor Ort.

Dieses PG5 Werkzeug erlaubt Ihnen die Erstellung einer *„sd5“-Datei*, die alle zur Aktualisierung von PCD-Programmen und -Konfigurationen notwendigen Informationen enthält. Der PG5 Programmierer schickt lediglich diese Datei per E-Mail an die zuständige Person vor Ort der PCD.

Wenn Sie eine *„sd5“-Datei* öffnen, wird das Dialogfenster für den Daten-Download angezeigt. 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 PCD *Hardware Settings* herunterzuladen. Das Werkzeug ermöglicht das Herunterladen von Programmen und *Hardware Settings* ohne die vorherige Installation von PG5 oder einer Benutzerlizenz. Das Paket *Stand Alone Online Tools* muss jedoch vor Ort installiert sein.

2.9.1 Erstellung einer *„sd5“-Datei*



Die Datei wird aus in der aktiven CPU enthaltenen Informationen erstellt, wie im *Project Tree* Fenster ersichtlich ist. Bitte stellen Sie sicher, dass die *Online Settings* und *Hardware Settings* korrekt konfiguriert sind, und führen Sie einen CPU *Build* durch, bevor Sie die *„sd5“-Datei* erstellen.

Das CPU-Menü: *Create Self-Downloading File* ermöglicht Ihnen die Konfiguration von Parametern und Optionen für den Download selbstladender Dateien vor Ort.

Diese Parameter und Optionen stimmen mit denjenigen überein, die wir bereits aus anderen Menüs kennen: *Online*, *Hardware Settings*, *Download* und *Online*, *Download Program ...*

Es wurden jedoch neue Parameter hinzugefügt:

Create Files (*.sd5)

Ermöglicht Ihnen die Definition des .sd5-Pfads und des Dateinamens.

Show "Advanced >>" button

Ermöglicht Ihnen das Ausblenden der Vorauswahl in diesem Dialogfenster während des Downloads der selbstladenden Dateien.

No Dialog box (Progress only)

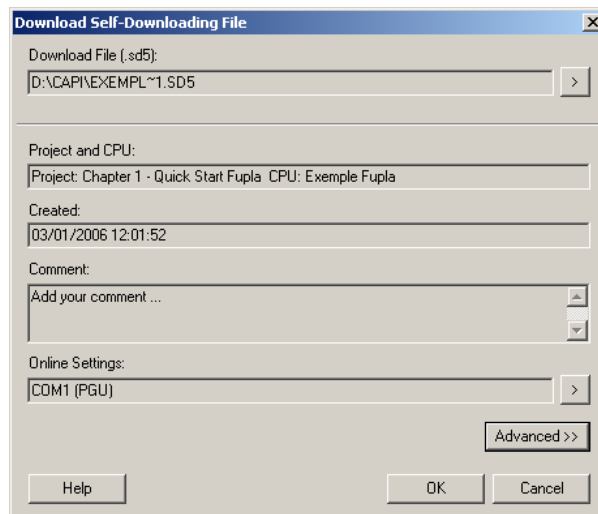
Lädt die „sd5“-Datei herunter, ohne ein Dialogfenster anzuzeigen (automatische Installation).

Verify Serial Number

Prüfung während des Downloads, mittels derer die Übereinstimmung der Seriennummer der PCD mit der im Feld *Serial Number* definierten Nummer sichergestellt wird. 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 neuen PCD3-Systemen unterstützt. Mit dem *Online Configurator* kann die Nummer online gelesen werden. Verwenden Sie dazu das Menü *Online, Information*.

2.9.2 Herunterladen einer „sd5“-Datei

Um die Inhalte von einer „sd5“-Datei herunterzuladen, muss zunächst sichergestellt werden, dass PG5 oder *Stand Alone Online Tools* auf dem PC installiert sind. Weitere Einzelheiten entnehmen Sie bitte der PG5 Installationsanleitung. Öffnen Sie die „sd5“-Datei per Doppelklick über den *Windows Explorer*. Es wird ein Dialogfenster ähnlich dem oben abgebildeten Fenster angezeigt. Sie können den Vorgang des Herunterladens der Dateiinhalte mit der Taste *Advanced* neu konfigurieren. Starten Sie den *Download* mit der Taste *OK*.

Inhaltsverzeichnis

3	PCD - RESSOURCEN	3
3.1	Einleitung	3
3.2	Hardware-Ressourcen	4
3.2.1	Digitale Ein- und Ausgänge	4
3.2.2	Hardware Uhr	5
3.2.3	Interrupt-Eingänge	6
3.3	Interne (Software-) Ressourcen	7
3.3.1	Flags	7
3.3.2	Register	8
3.3.3	Konstanten	9
3.3.4	Timer und Counter (Zeiten & Zähler)	10
3.3.5	Texte und Datenblöcke	13
3.3.1	Zusammenfassung	15
3.4	Symbol Editor	16
3.4.1	Elemente einer Ressourcenliste	16
3.4.2	Gruppieren der Symbole	17
3.4.3	Symbolbereiche	17
3.4.4	Lokale Symbole	18
3.4.5	Globale Symbole	18
3.4.6	global definieren	19
3.4.7	Netzwerk Symbole	19
3.5	Arbeiten mit Symbolen	20
3.5.1	Editieren einer Symbol-Liste	20
3.5.2	Mehrere Symbole im Symbol-Editor einfügen	21
3.5.3	Referenzierte Symbolnamen	22
3.5.4	Symbole aus einer EQUATE-Liste importieren	23
3.5.5	Symbole aus einem anderen Programm importieren	23
3.5.6	Symbole während dem Editieren in IL eingeben	23
3.5.7	Symbole während dem Editieren in FUPLA eingeben	24
3.5.8	Symbole übertragen	25
3.5.9	Automatische Erkennung von Symbolen	26
3.5.10	Automatische Zuweisung	26
3.5.11	Texteingabe	27
3.5.12	DB Eingabe	28
3.5.13	Suchen nach Symbolen	28
3.5.14	Anordnen der Symbole	29
3.5.15	Neu anordnen in "List View"	30
3.5.16	Symbole exportieren	31
3.5.17	Symbole importieren	33
3.5.18	Symbole initialisieren	35
3.5.19	Symbolische Namen	36
3.5.20	Reservierte Ausdrücke	36

3 PCD - Ressourcen

3.1 Einleitung

Dieses Kapitel gibt einen Überblick über alle Datentypen, die in einer Applikation benutzt werden können.

Die ersten zwei Unterkapitel fassen alle Elemente der SAIA[®]PCD Familie, wie Eingänge, Ausgänge und Flags sowie deren Adressbereich und Anwendung, zusammen.

Die beiden letzten Unterkapitel zeigen, wie diese Elemente im Symbol Editor benutzt werden.

3.2 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.

3.2.1 Digitale Ein- und Ausgänge

I/O 1 Informationsbit (0/1)

	Maximale Anzahl E/A
PCD1	32 (64 ³⁾)
PCD2.M120/M150 (+_2.C1_0)	64/96/128 (255 ³⁾) ¹⁾
PCD2.M170 + PCD3.C100	510 ²⁾ ³⁾
PCD2.M480 + PCD3.C100	1023 ¹⁾ ³⁾
PCD3.Mxxxx	1023 ¹⁾ ³⁾
PCD4	510 ²⁾
PCD6	5100 ²⁾

- 1) Die -Adresse 255 (und 511 bei der PCD2.M170) sind für den Watch-Dog reserviert
- 2) Die -Adressen 255, 511, 767, 1023 etc. bis 5119) sind für den Watch-Dog reserviert
- 3) mit Eingangsmodulen PCD2/3.E16x und/oder Ausgangsmodulen PCD2/3.A46x

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 entweder durch dessen Position (bei PCD1/2/3 und 4) oder mit Schaltern definiert (bei PCD 6).

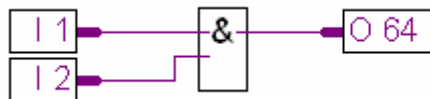
Das folgende Beispiel schaltet den Ausgang O 64 ein, wenn die Eingänge I 1 und I 2 = H sind. Eine andere Darstellung ist die boolsche Gleichung: $O\ 64 = I\ 1 * I\ 2$.

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)

3.2.2 Hardware Uhr

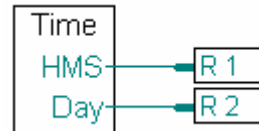
Die meisten PCD (PCD1.M120/130 und alle PCD2/3/4/6) haben eine eigene Uhr (RTC = Real Time Clock). Es kann mit einer Instruktion die Zeit und das Datum in je ein PCD-Register geladen werden.

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
R 2 = 073030210
```

09 Uhr 35 Minuten und 10 Sekunden
Woche 07, Tag.-Nr. 3 (Mittwoch),
10. Februar (20)03

3.2.3 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 ; Programme
      0 ; principal

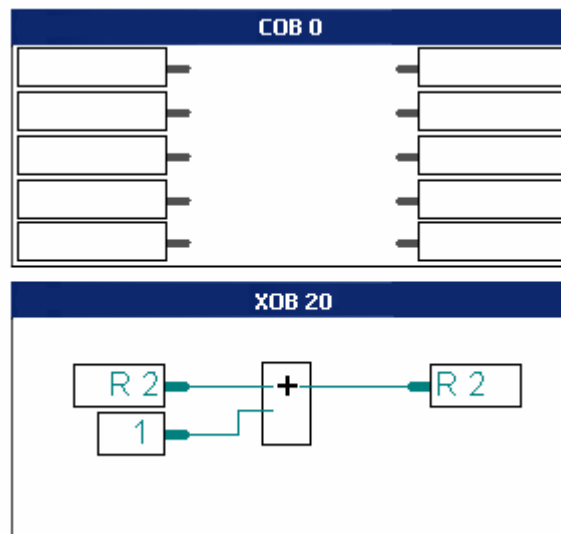
ECOB

XOB 20 ; interruption INB1
INC R 2 ; incrémentation
      ; du registre R2

EXOB
```

Funktionsplan (FUPLA)

Programm:



- 1) PCD1.M120/130, PCD2.M120/150, PCD2/4.M170, PCD2.M480 (4 Interrupt-Eingänge IN0 ... IN3), PCD3.M und PCD6.M3
- 2) Weitere Informationen siehe PCD-Handbücher

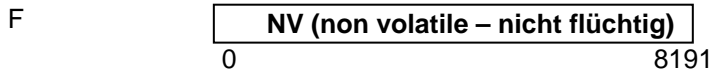


Die Begrenzung durch den Eingangsfilter (schützt einen digitalen Eingang gegen Störspitzen und Prellen eines mechanischen Kontaktes) verhindert das Zählen eines Impulses mit einer Frequenz grösser 50Hz. Deshalb stellen Interrupt-Eingänge eine interessante Alternative für diese Art der Anwendung dar. Sie stellen einen Ersatz für den Einsatz von PCD2.H1xx oder PCD4.H1xx Zählmodulen dar, die eine Zählfrequenz - abhängig vom Modultyp - von 10 bis 160kHz haben.

3.3 Interne (Software-) Ressourcen

3.3.1 Flags

1 Informations-Bit (H/L)



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.

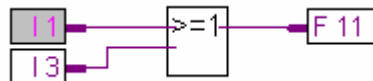
Das folgende Beispiel bringt das Flag F 11 in den logischen H-Zustand wenn die Eingänge 1 oder 3 = H sind. Die boolsche Gleichung lautet $F 11 = I 1 + I 3$

Verwendung von Flags in Programmen

Instruction List (IL)
Programm:

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

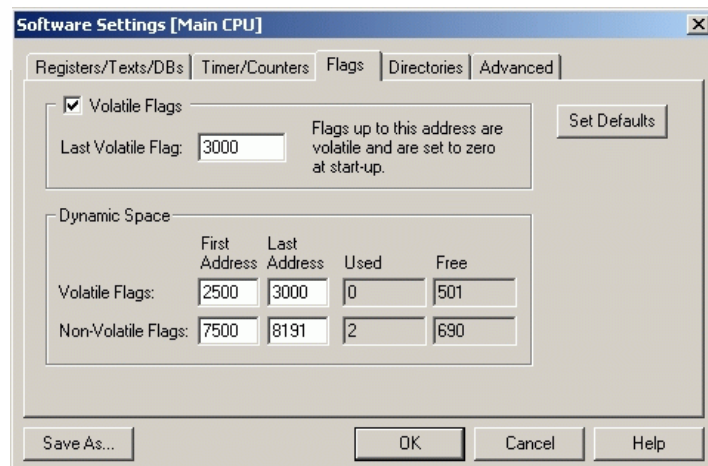
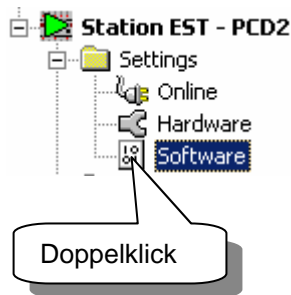
Funktionsplan (FUPLA)
Programm:



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

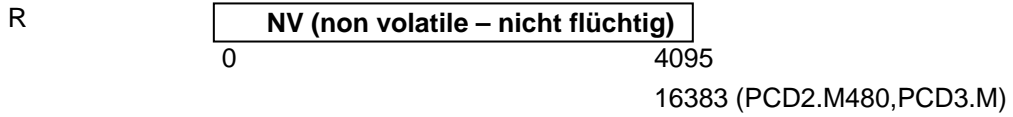


3.3.2 Register

32 Bit Wert

Integer : -2 147 483 648 to +2 147 483 647

Floating-Point : -9.22337E+18 to +9.22337E+18



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: $R\ 54 = R\ 12 + 113$

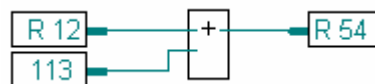
Instruction-List (IL):

Programm:

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

Funktionsplan (FUPLA)

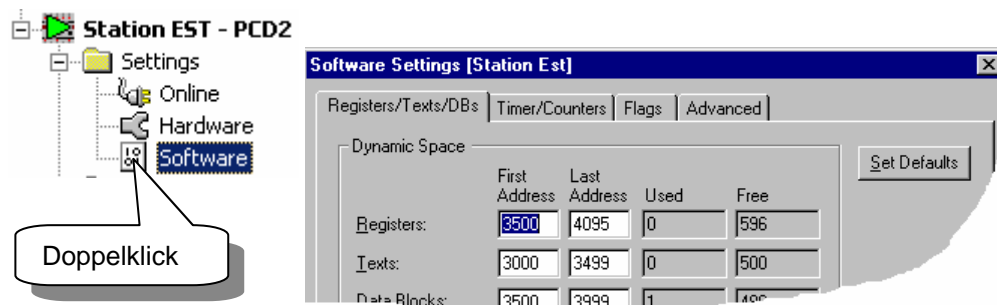
Programm:



FBox: Addition, (Ganzzahl Arithmetik)

Setup Registers

Bei der der Festlegung von dynamischen Ressourcen 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.

3.3.3 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) Programm:

```

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

COPY R 4 ; Konvertiere Wert
     R 35 ; von Integer
IFP R 35 ; zu Floating Point
    0

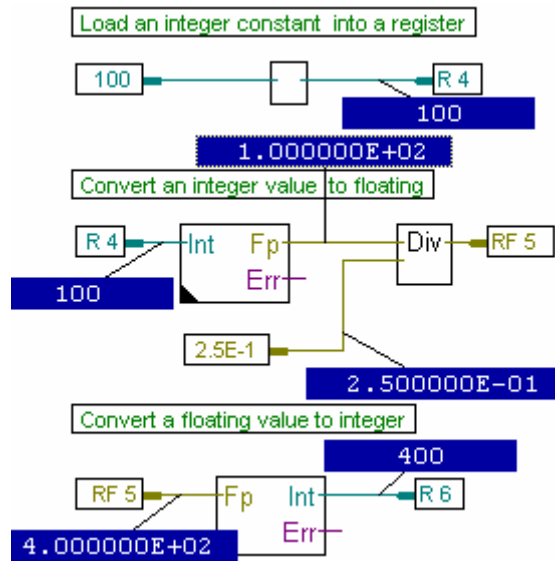
LD R 36 ; Lade 0,25 ins R 36.
   2.5e-1

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
    
```

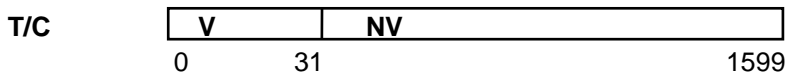
Funktionsplan (FUPLA) Programm:



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

3.3.4 Timer und Counter (Zeiten & Zähler)

31 Bit Wert (0 ... 2 147 483 648)



Timer (T) und Counter (C) können Werte zwischen 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

	First Address	Last Address	Used	Free
Timers:	5	31	2	25
Counters:	1400	1599	0	200



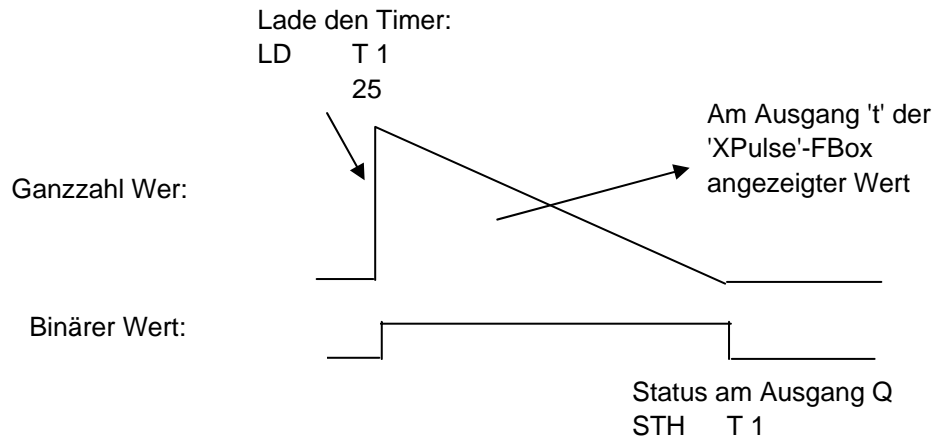
Technische Zusatzinformation

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.



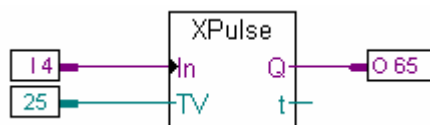
Lösung:

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 "Software Settings" 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.

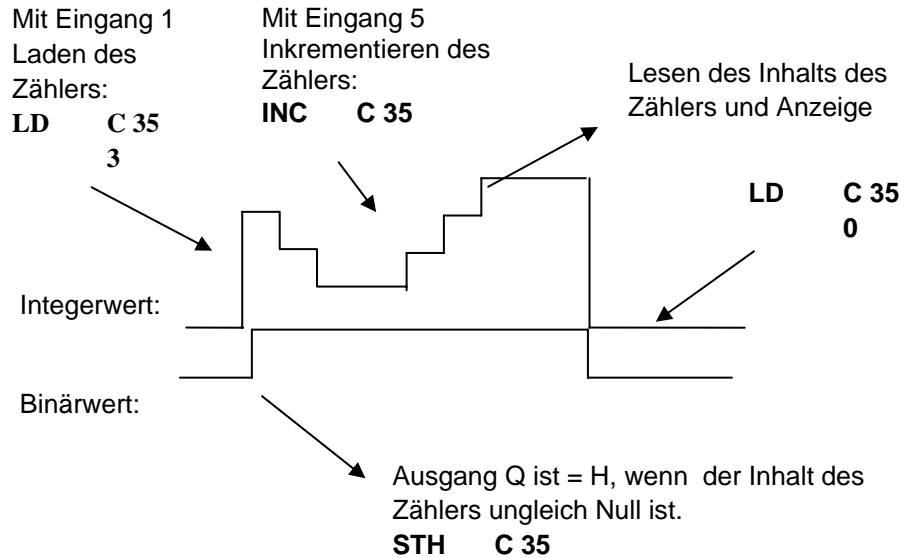
Format: T#nnnS|MS

BL_3DE393BA	COB		
DelayTime	K Constant	T#100MS	100 milliseconds
OneDay	K Constant	T#3600S	3600 secondes

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 verwirlich wirkt, macht dies nichts, es wird alles viel klarer, wenn Sie selbst eine praktische Aufgabe programmieren werden.



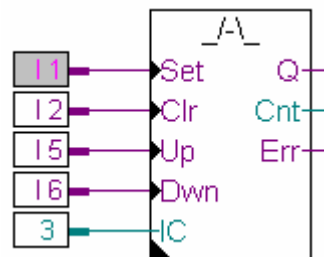
Lösung:

Instruction List (IL) 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)

3.3.5 Texte und Datenblöcke

TEXT/DB	Main memory		NV
	0		3999
	Extension memory		NV
	4000		(PCD4/6) 7999
			(PCD2) 5999
			(PCD1) 4999
			(PCD2.M480, PCD3.M) 8191

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-EPR0M 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 letzten 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

Beispiel: Datentransfer im Funktionsplan (FUPLA)

Datentransfer von einem Analogmodul in den DB 4010. Jedesmal wenn der Eingang 'Store' = H kommt, wird der Analogwert gelesen und in den DB4010 übertragen.

Grösse des DB

Vorgabewerte

Doppelklick

FBoxes:
 - PCD2.W2, (Analogmodule)
 - DB Logger, (Data blocks)

Beispiel: senden einer SMS im Funktionsplan (FUPLA)

Das folgende Beispiel zeigt das Senden einer SMS mit dem digitalen Zustand eines Eingangs oder Flags. Die Nachricht ist in Text 10 hinterlegt. Beachten Sie die schwarzen Dreiecke in der unteren linken Ecke der FBoxes. Sie weisen darauf hin, dass diese FBoxes ein Einstellungsfenster besitzt, in dem z.B. die Mitteilungszentralnummer und die verschiedenen Telefonnummern der Endempfänger eingetragen werden. Die Einstellungsfenster werden durch einen Doppelklick auf die FBox geöffnet.

Doppelklick auf die FBox öffnet das Einstellungsfenster

Doppelklick

Text für die SMS-Nachricht

FBoxes:
 - Modem, Modem Driver 14
 - Modem SMS, Call SMS
 - Modem SMS, Send SMS

3.3.1 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...8191	0,1		2) Nein
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
Registers	R	0...4095 5) 0...16383		-2 147 483 648...+2 147 483 647 -9.22337E+18...+9.22337E+18	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 "CPU -> Software Settings"
- 3) Im gleichen Bereich gespeichert wie Anwenderprogramme (RAM / EPROM / FLASH)
- 4) Im erweiterten Speicherbereich (extension memory) gesichert (RAM)
- 5) PCD2.M480, PCD3.M

3.4 Symbol Editor

Bevor wir mit der Programmierung beginnen, ist eine Liste mit allen Elementen, welche im Projekt verwendet werden sollen, zu erstellen (Anzahl Ein- und Ausgänge, Anzahl Timer usw.). Alle diese Elemente müssen dem PG5 bekannt sein. Diese Liste wird das Auffinden von Elementen im Programm bei Programmierfehlern oder während der Inbetriebnahme sehr erleichtern. Alle Ressourcen werden also in einer Liste zusammengefasst. Diese Liste wird 'Symbol-Editor' genannt.

Der Ausdruck 'Symbol' passt besser als 'Element', da betont werden soll, dass jedes Element einen (symbolischen) Namen hat. Sind alle Ressourcen mit einem Namen versehen, erleichtert dies das Lesen eines Programms sehr.

3.4.1 Elemente einer Ressourcenliste

Group/Symbol: Name der Ressource (kann bis zu 80 Zeichen lang sein)

Type: Hier wird der Typ der Ressource angegeben, z.B. Eingang, Flag, Register ...

Comment: Kommentare erhöhen die Lesbarkeit von Programmen.

Group/Symbol	Type	Address/Value	Comment
NormalRun	I	3	Machine is in normal run
Cond_Run	I	4	Machine is in conditional...
OilHigh	I	5	Oil Level is too high
Emergency	I	7	Emergency stop on the ...
IntermediatFlag	F		
OilPump	O	20	Oilpump
OilPumpProg	COB	3	

Global: Symbole in dieser Liste gelten für alle Programm Dateien

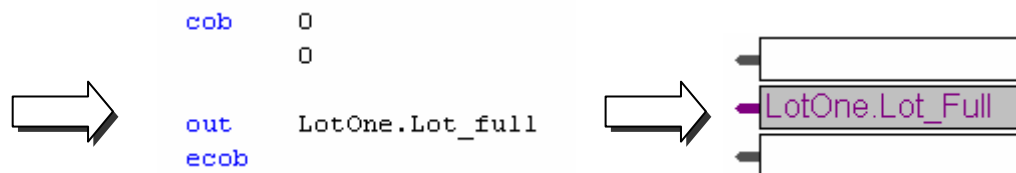
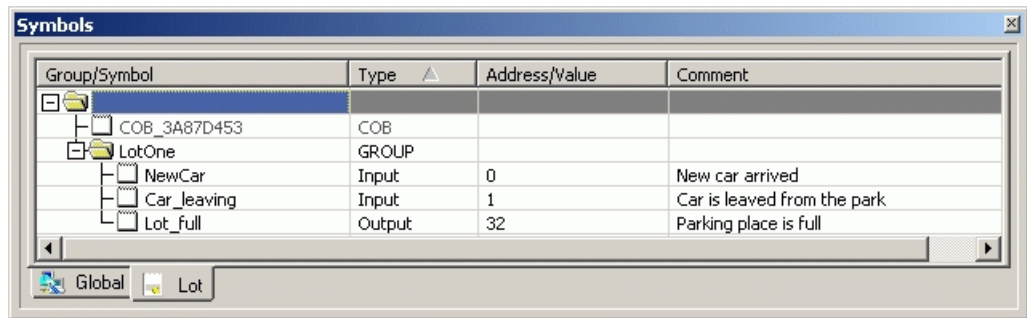
Dateiname: sagt aus, zu welcher Datei die Symbole gehören. Es können nur Symbole zu dieser Datei definiert

Address/Value
Es muss angegeben werden, welcher PCD-Eingang oder -Ausgang zu diesem Symbol gehört. Interne Ressourcen (alle, ausser Ein- und Ausgängen) sind in diesem Fall nicht zu spezifizieren. Das System bestimmt sie automatisch (Auto-Allocation).

3.4.2 Gruppieren der Symbole

Die Symbole können nach eigenen Ideen gruppiert werden. Dies erleichtert das Lesen des Programms. Es ist dabei die rechte Maustaste zum Eröffnen einer neuen Symbolgruppe zu klicken und danach sind die gewünschten Symbole in diese Gruppe hinein zu ziehen (Drag & Drop).

Beispiel: Die Gruppe "LotOne" enthält mehrere Symbole:



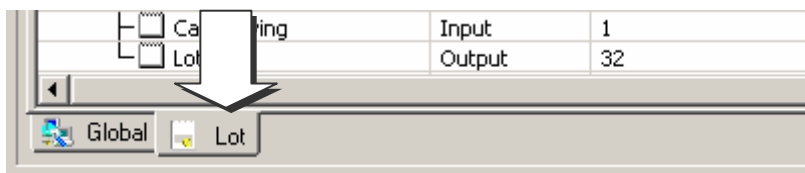
Im Anwenderprogramm wird dem Symbolnamen der Gruppenname „LotOne“ voangestellt. Gruppenname und Symbolname sind dabei durch einen Punkt getrennt.

3.4.3 Symbolbereiche

Symbole gelten normalerweise nur für eine einzige Programmdatei, der Symbolbereich ist somit lokal. Wird in einem Editor eine Datei geöffnet, wird gleichzeitig auch der Symboleditor mit den zugehörigen Symbolen geöffnet.

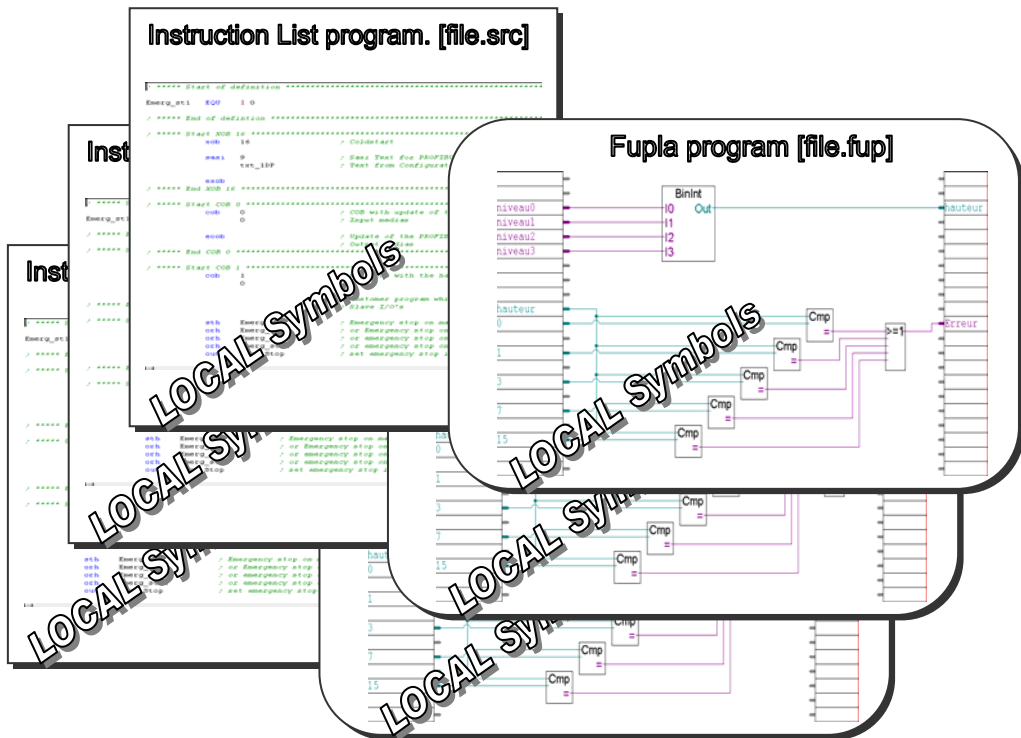
Beispiel:

Beim Öffnen der Programmdatei "Lot.src" wird automatisch im Symbol Editor die gleichnamige lokale Symbolliste geöffnet.



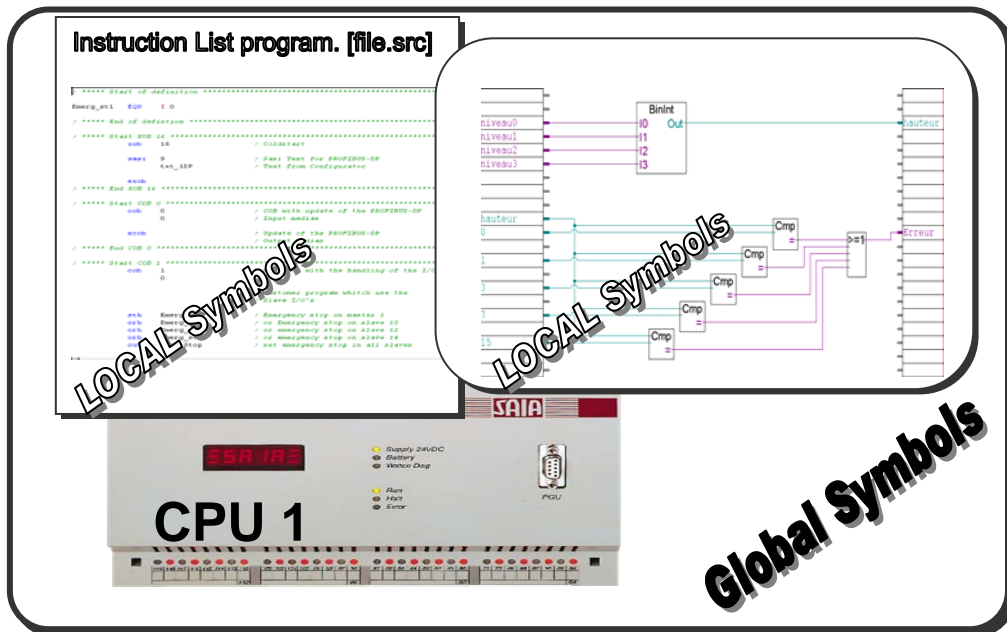
3.4.4 Lokale Symbole

„Local Symbols“ (lokale Symbole) sind nur innerhalb der jeweiligen Datei bekannt.

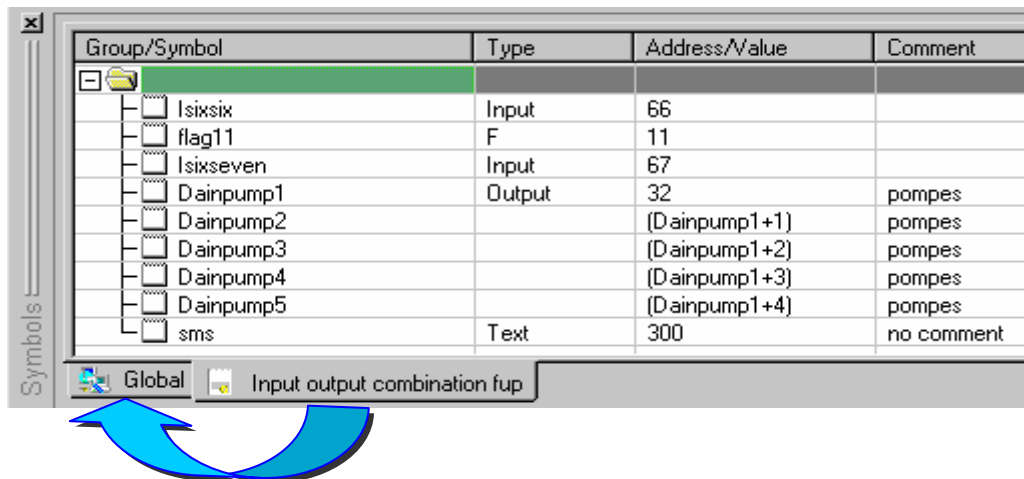


3.4.5 Globale Symbole

„Global Symbols“ können in allen Dateien einer CPU verwendet werden.

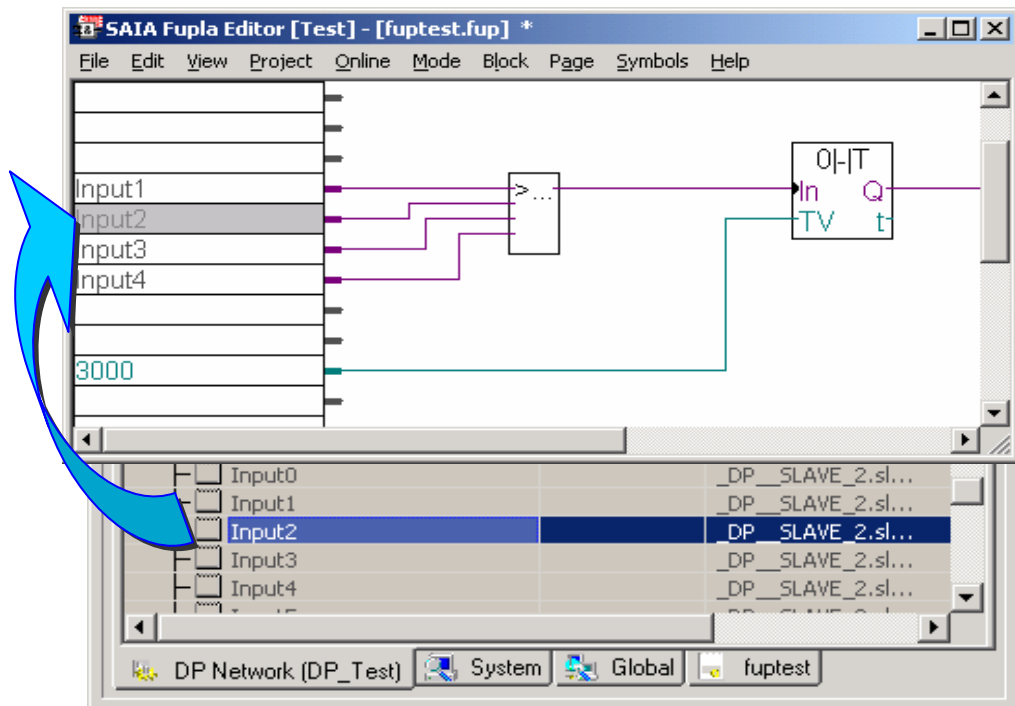


3.4.6 global definieren



Sollen die gleichen Symbole in verschiedenen Dateien verwendet werden, so sind die Symbole von der lokalen in die "globale" Liste zu verschieben (die betreffenden Symbole sind zu markieren und mit der Funktion „Advanced Make Global“ (rechte Maustaste) werden sie verschoben. Ist ein Symbol als 'global' definiert, kann dieses innerhalb des Projekts von überall her erreicht werden.

3.4.7 Netzwerk Symbole



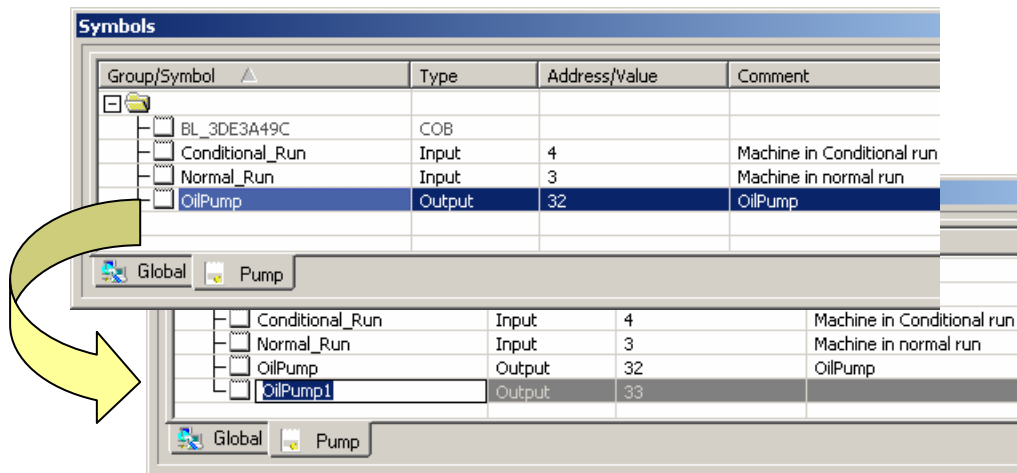
Zwischen PCD-Systemen werden Daten über serielle Schnittstellen oder Netzwerke ausgetauscht. Für die Konfiguration des Datenaustausches über die Netzwerke Profibus DP, Profibus FMS und LonWorks wird der Netzwerk-Editor verwendet. Alle im Editor definierten Symbole werden nach dem Programm „Build“ im Symbol-Editor in der Netzwerklste (im Beispiel: DP Network ...) aufgelistet. Die Symbole können nun wie globale Symbole in mehreren Dateien verwendet werden.

3.5 Arbeiten mit Symbolen

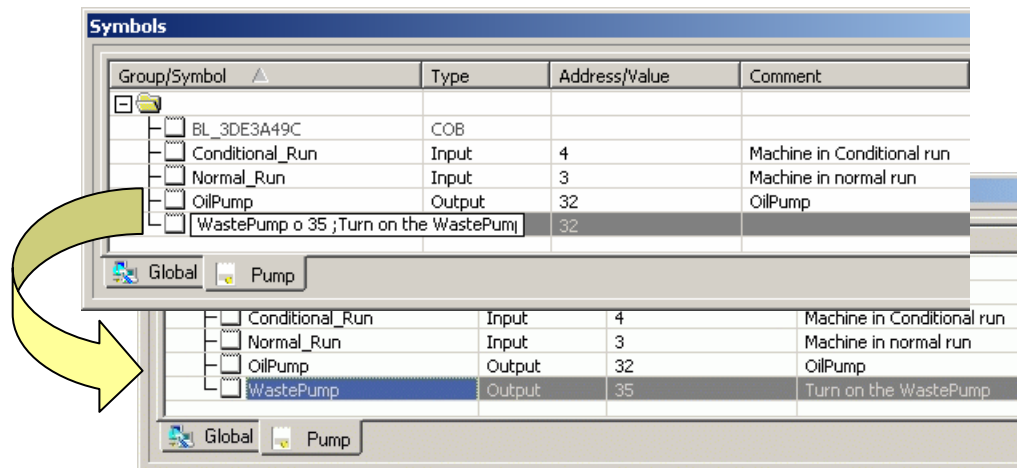
3.5.1 Editieren einer Symbol-Liste

Zuerst wird die zu bearbeitende Datei geöffnet. Damit wird auch der Symbol-Editor geöffnet. Mit der linken Maustaste das 'Group/Symbol' Fenster anklicken. Anschliessend wird mit der 'Insert'-Taste (von der Tastatur) ein neues Symbol eingefügt. Ein Symbol kann auch über das Kontext-Menü mit der rechten Maustaste eingefügt werden. Den Symbolnamen, Typ, Adresse/Wert und einen Kommentar eingeben und mit der 'Enter'-Taste bestätigen. Ein weiteres Symbol wird wieder mit der 'Insert'-Taste eingegeben.

Dabei kopiert der Editor automatisch den Symbolnamen vom vorangehenden Feld und erhöht die Adresse/Wert um 1 (siehe dazu auch nachfolgendes Bild). Der neue Eintrag kann so übernommen oder geändert und ein Kommentar beigefügt werden.



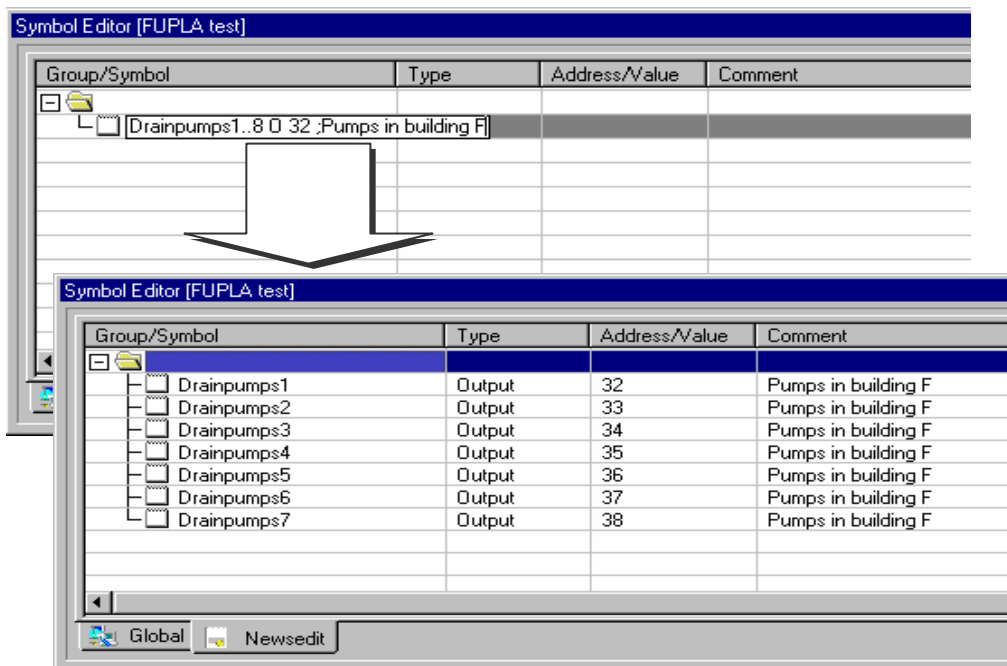
Ist bereits ein Liste eröffnet und diese soll mit einem Symbol ergänzt werden, so ist das letzte Symbol anzuklicken, danach die 'INSERT'-Taste betätigen und es erscheint ein neues Symbolfeld.



3.5.2 Mehrere Symbole im Symbol-Editor einfügen

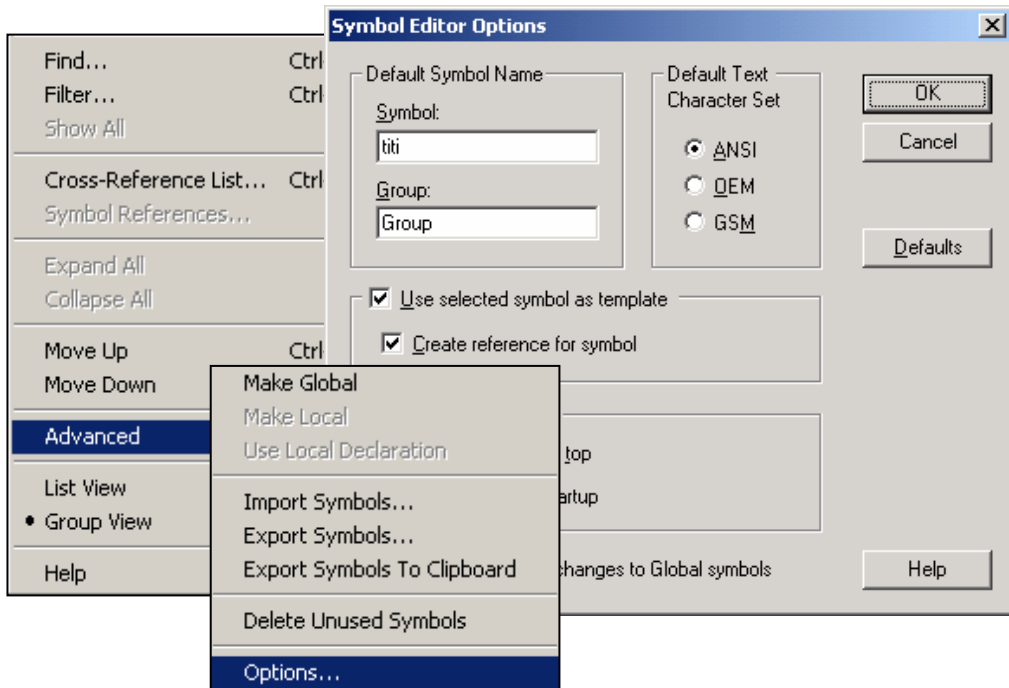
Es kann auch eine Reihe von Symbolen eingegeben werden. Es ist der Symbolname des ersten und die letzte Elementnummer wie im nachfolgenden Bild dargestellt, einzugeben:

(Drainpumps1..8 O 32 ;Pumps in building F) 8 ist die Anzahl der Symbole, O steht für Ausgang und 32 ist die Startadresse des Adressbereichs. Nach dem Betätigen der 'Enter'-Taste wird der Symboleditor die Liste automatisch ergänzen.

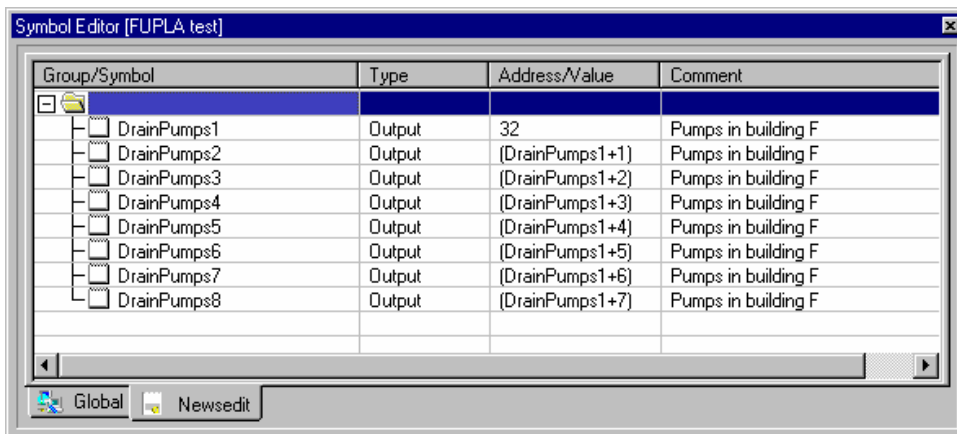


3.5.3 Referenzierte Symbolnamen

Eine Symbolreihe kann auch mit Referenz auf ein bestehendes Symbol eingegeben werden. Dazu muss über den Menü-Befehl 'Symbols-Advanced-Options..' die Option 'Create reference for symbol' angewählt und mit OK bestätigt werden.

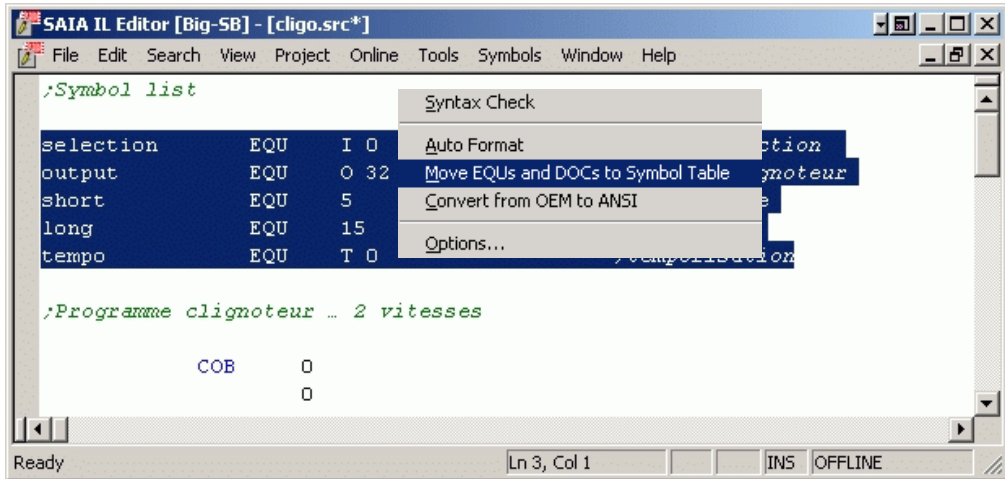


Im Symbol-Editor wird nun das Referenzsymbol mit der Maus selektiert. Mit der 'Insert'-Taste (von der Tastatur) wird ein neues Symbol (inkrementiert um 1) wie im nachfolgenden Bild dargestellt generiert. Dies kann dann interessant sein, wenn zusammenhängende Adressgruppen umadressiert werden sollen.



3.5.4 Symbole aus einer EQUATE-Liste importieren

Aus einer alten PG4/3 IL (Instruction List) Datei können EQU- oder DOC-Definitionen mit dem Menü-Befehl 'Tools, Move EQUs and Docs to Symbol Table' in die Symbol-Liste importiert werden.

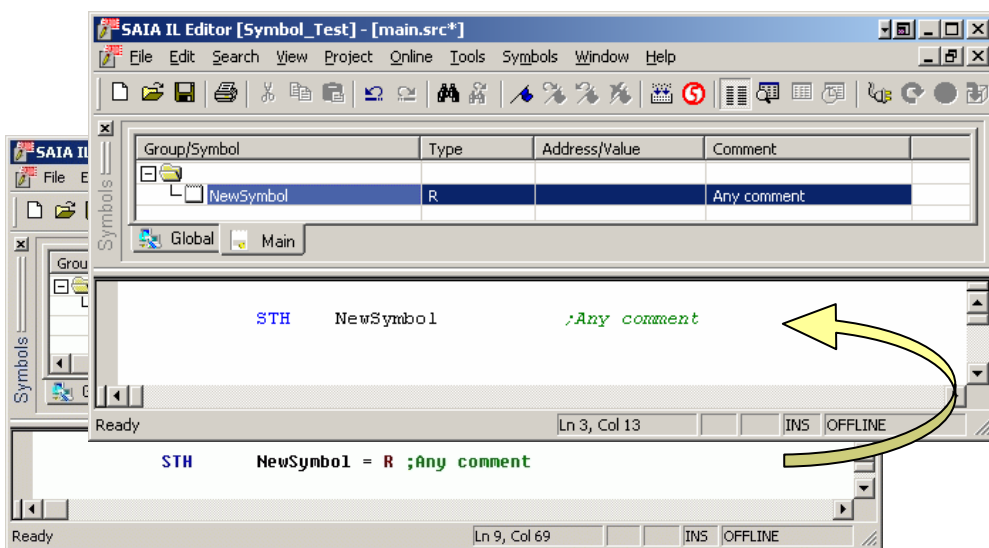


3.5.5 Symbole aus einem anderen Programm importieren

Es können auch Symbole aus einem andern Programm, z.B. Electro-CAD, in den Symbol-Editor des Projekts eingefügt werden. Dies macht die Dokumentation über das ganze Projekt durchgehend, d.h. die Symbole und Namen im Elektroschema werden die gleichen sein wie im Anwenderprogramm. Es ist dabei die Exportfunktion für Symbole in eine Textdatei des CAD-Systems zu verwenden. Diese Datei kann dann im Symbol-Editor importiert werden.

3.5.6 Symbole während dem Editieren in IL eingeben

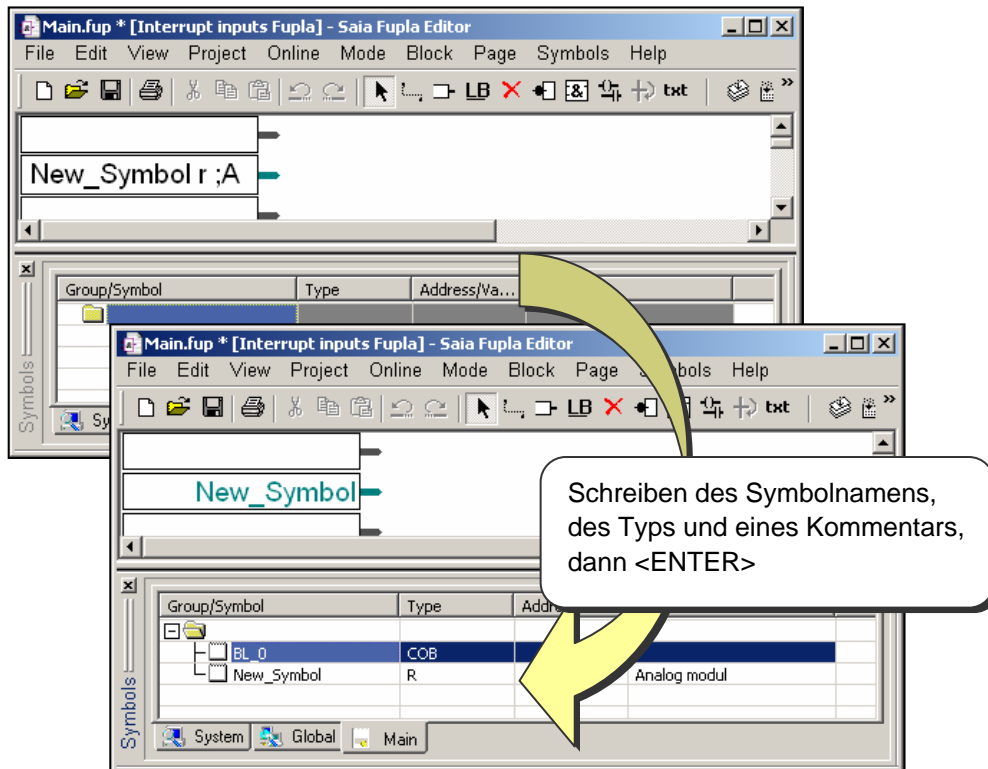
Wird während der Programmerstellung ein Symbol mit der Erweiterung "= Typ Adresse; Kommentar" eingegeben und mit <ENTER> abgeschlossen, so wird das Symbol automatisch in den Symbol Editor aufgenommen. Siehe dazu auch das nachfolgende Beispiel.



3.5.7 Symbole während dem Editieren in FUPLA eingeben

Im Fupla Editor können neue Symbole direkt von einem Eingangs/Ausgangsfeld mit der nachfolgenden Syntax in den Symbol Editor eingefügt werden.

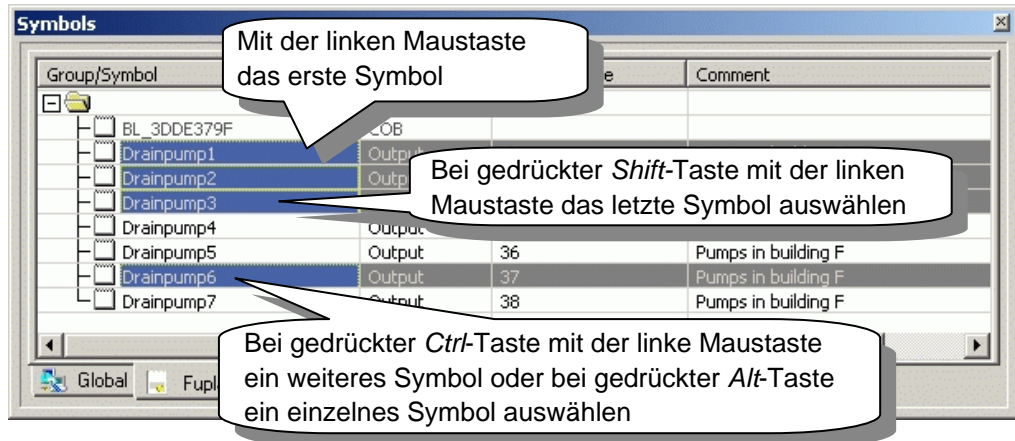
Syntax: *Symbolname Typ [Adresse] [;Kommentar]*



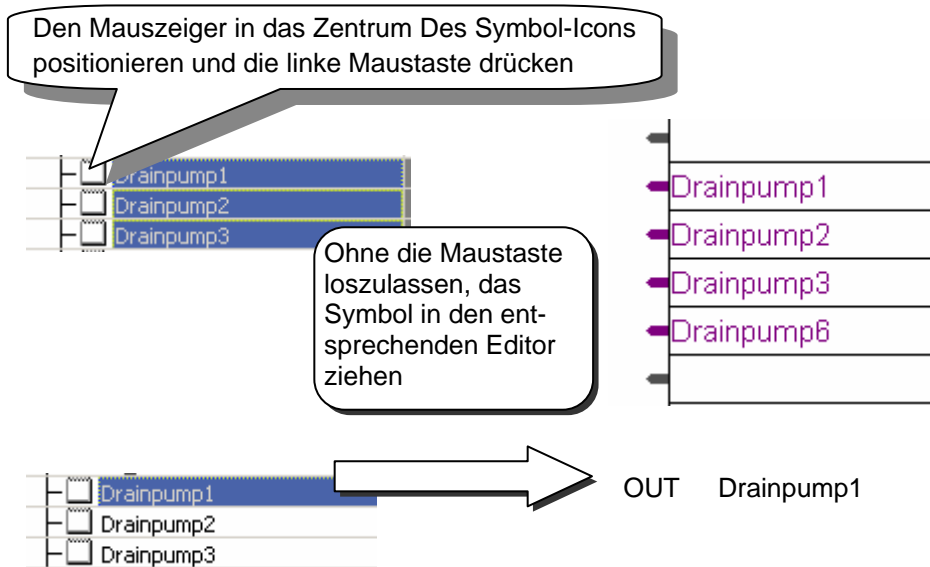
3.5.8 Symbole übertragen

Um die mehrfache Eingabe von gleichen Symbolnamen im Programm zu vermeiden (Risiko von Tippfehlern), können ein oder mehrere Symbole ausgewählt und in das entsprechende Fupla- oder IL-Programm gezogen werden.

Beispiel zur Auswahl mehrerer Symbole:



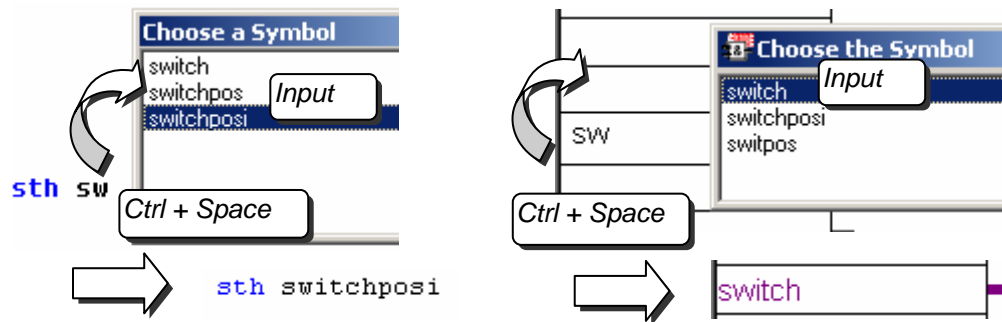
Beispiel zum Einfügen der Symbole in den Fupla- oder IL-Editor:



3.5.9 Automatische Erkennung von Symbolen

Um die Verständlichkeit des Programms zu verbessern, ist es häufig notwendig, lange Symbolnamen zu verwenden. Dabei wäre es mühsam, jedes Mal diese langen Symbolnamen einzutippen. Die Arbeit wird erleichtert, indem nur die ersten Buchstaben eines Symbols getippt werden müssen. Durch Betätigen von <Ctrl> und <Space> wird der Symbolname vervollständigt, da nach gleichen Buchstabenfolgen gesucht wird.

Beispiel:



3.5.10 Automatische Zuweisung

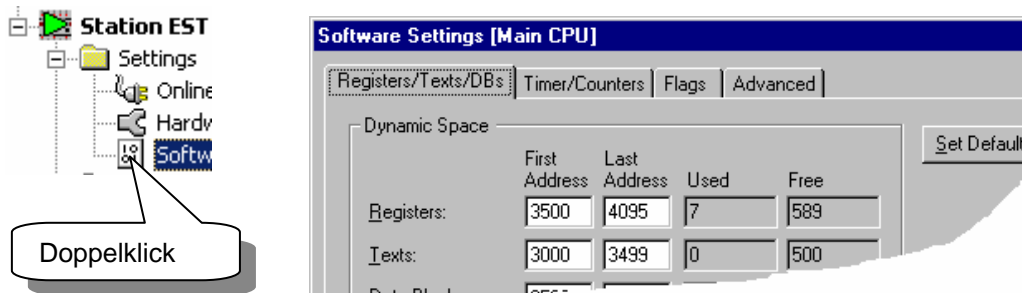
Bis jetzt wurden Elemente wie folgt deklariert:

	Symbol-Name	Typ	Adresse	Kommentar
Beispiel:	Pumpspeed	R	2000	;Speed in l/min

Beim Eingeben eines Symboltyps, der weder ein Eingang noch ein Ausgang ist, muss keine Adresse angegeben werden. Diese wird beim Bearbeiten der Datei mit der Funktion 'Build' automatisch vom PG5 zugeordnet. Ist diese Option gewählt, überprüft das PG5 in den 'Software-Settings' den definierten Bereich für das betreffende Element und weist ihm während des 'Build-Prozesses' eine entsprechende Adresse zu.

Beispiel:	Pumpspeed	R		;Discharge in l/min
------------------	-----------	---	--	---------------------

Es wird im Programm ein Register ohne Adresse deklariert:

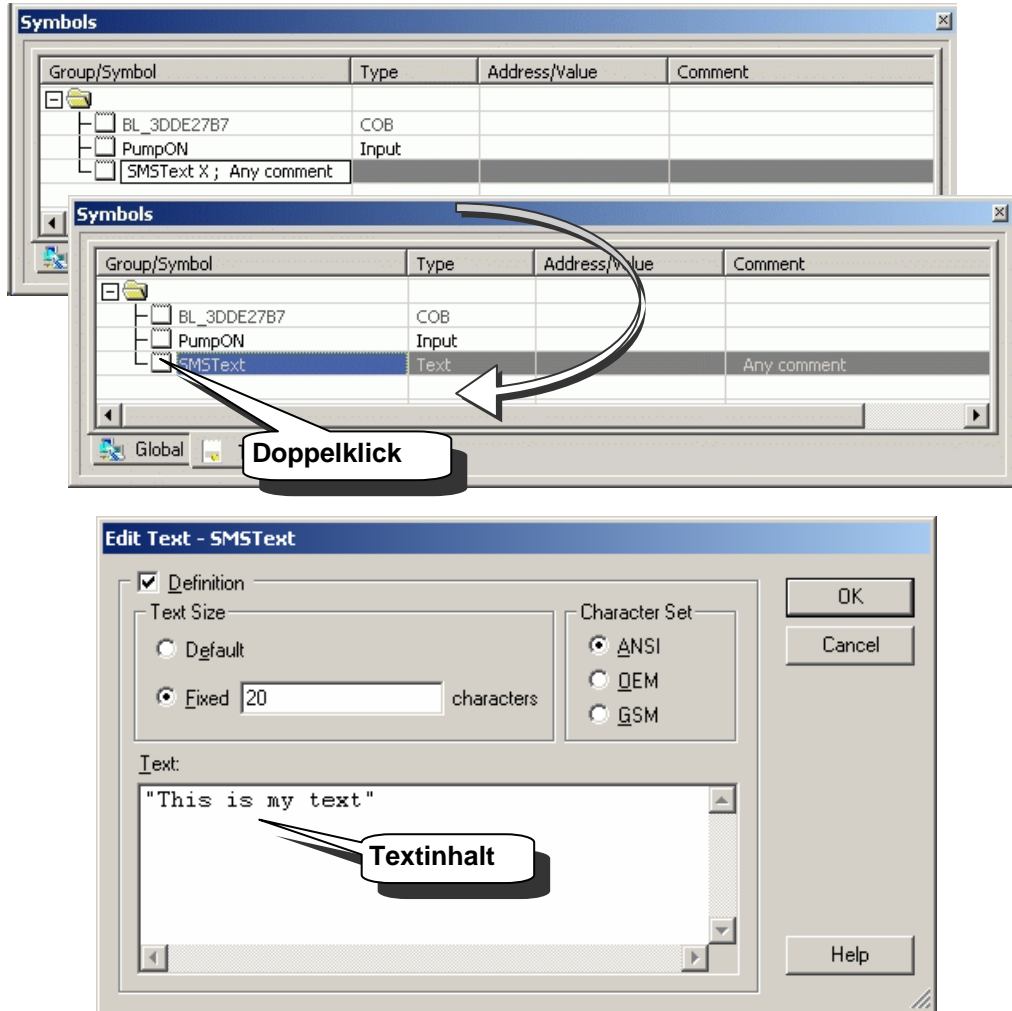


Das Register wird beim 'Build'-Prozess eine Adresse zwischen 3500 und 4095 erhalten, da in den 'Software-Settings' der dynamische Bereich für Register von 3500 bis 4095 definiert wurde.

3.5.11 Texteingabe

Soll ein Text in die PCD eingegeben werden, muss dieser zuerst deklariert werden. Dies kann durch die Eingabe eines 'X' nach dem symbolischen Namen geschehen.

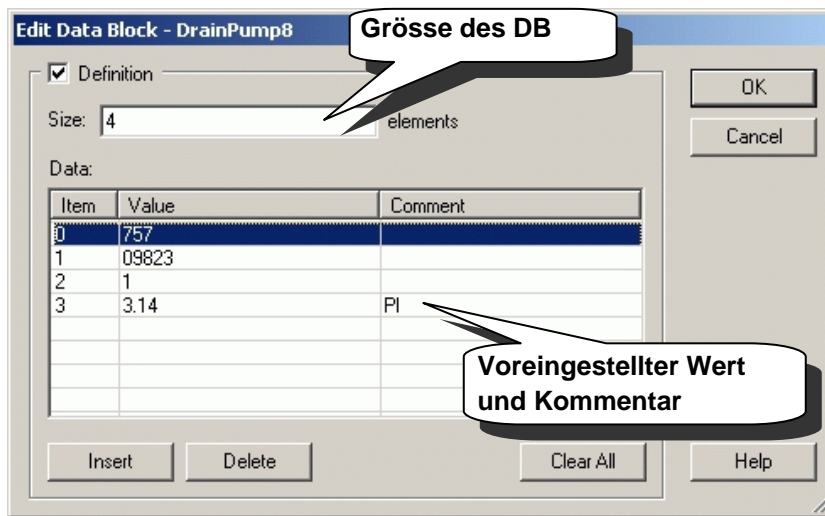
Beispiel:



Der Text muss zwischen doppelten Anführungszeichen stehen (" ... ") sonst wird dieser nicht als Text erkannt.

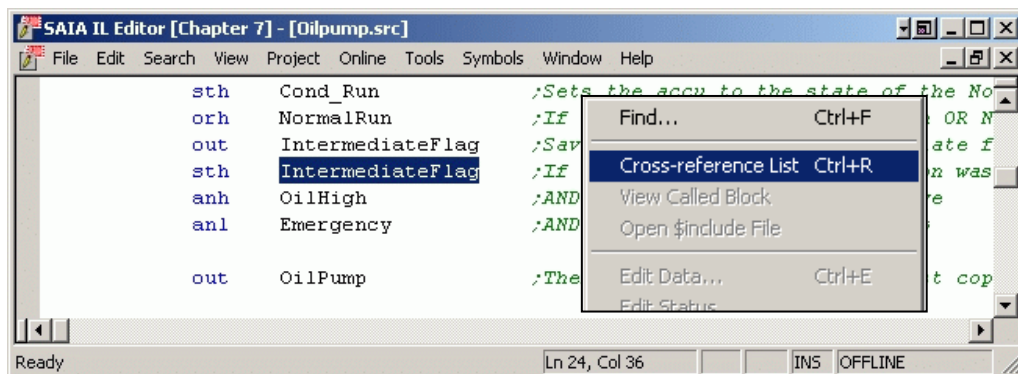
3.5.12 DB Eingabe

Für DB (Daten-Blöcke) steht ein spezieller Editor zur Verfügung. Weitere Informationen sind im 'Help' zu finden.

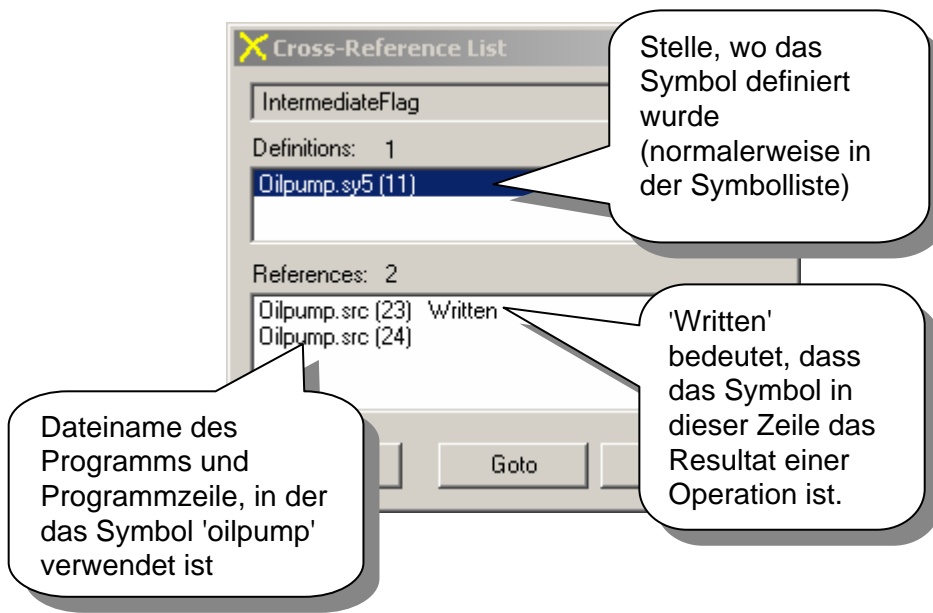


3.5.13 Suchen nach Symbolen

Oft wird ein Symbol in einem Programm an verschiedenen Stellen oder in verschiedenen Dateien verwendet. Nach einer erfolgreichen 'Build'-Operation kann nach dem Klicken der rechten Maustaste auf einem Symbol die 'Cross-reference List'-Funktion aufgerufen werden.

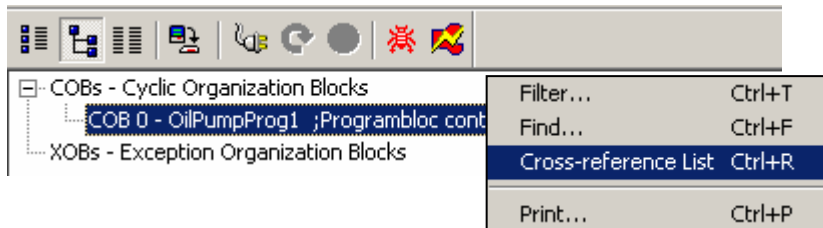


Die Cross-reference List (Querverweisliste) zeigt mit dem Dateinamen und der Zeilennummer an, wie oft das Symbol verwendet wurde.. Ein Doppelklick an eine Stelle der Liste wird das jeweilige Programm beim markierten Symbol öffnen.



Die Cross-Referenz (Querverweise) ist nicht nur im SEDIT und im FUPLA wirksam sondern auch in den verschiedenen 'Views' im Projekt-Manager.

Beispiel: Block Structure view

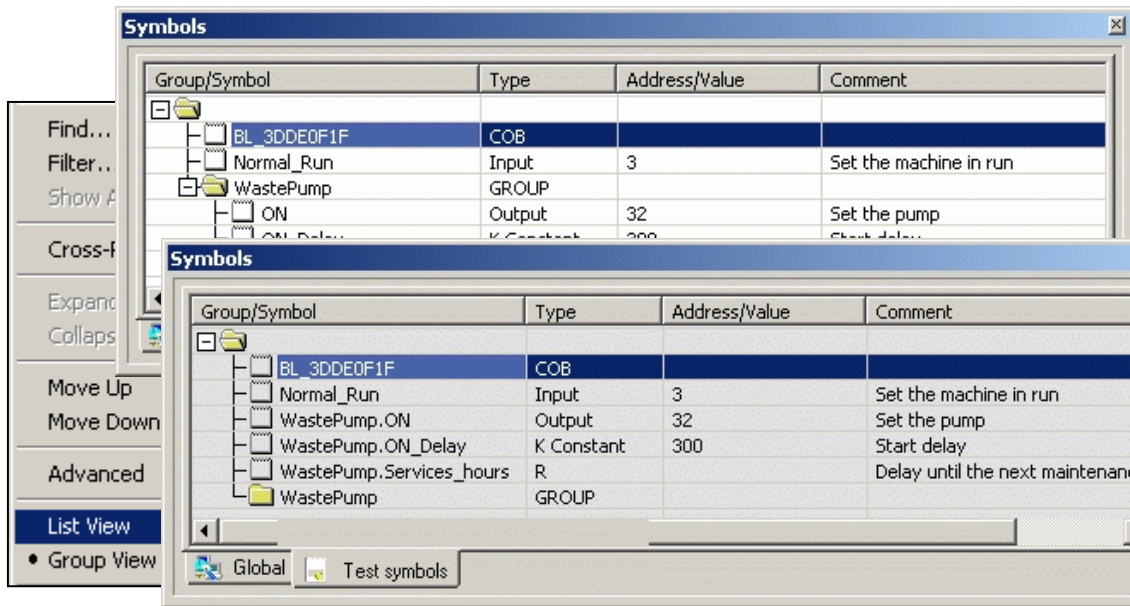


3.5.14 Anordnen der Symbole

Symbole werden dem Eintragsdatum nach geordnet. Auf diese Weise bleiben Symbole, die zur selben Zeit eingetragen wurden auch dann beieinander, wenn weitere Symbole später eingetragen werden.

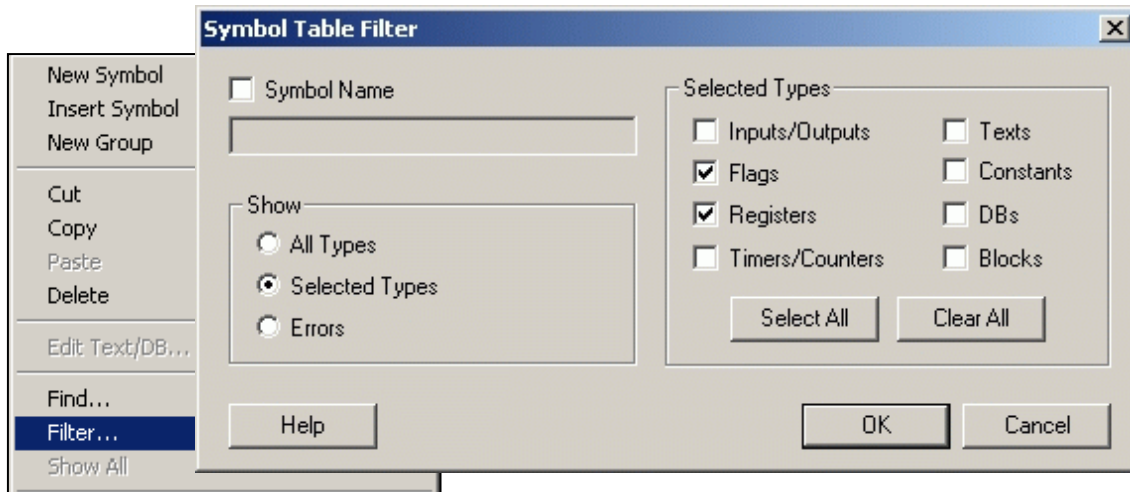
3.5.15 Neu anordnen in "List View"

Symbole können durch wechseln von 'Group view' zu 'List view' neu angeordnet werden. Durch Anlkicken der Tabellentitel 'Type', 'Address/Value' oder 'Comment' werden die Symbole entsprechend neu angeordnet



Gefilterte Anzeige

Wird zu 'Group view' zurückgeschaltet, ist die ursprüngliche Anordnung wieder hergestellt. Sind viel Symbole in der Liste, kann es vorteilhaft sein, nur bestimmte Typen oder nur gewisse Namen aufzulisten (Gefilterte Anzeige).



Die Filterfunktion bestimmt die Ansicht. Sobald die Filterfunktion gewählt ist, ändert das Symbolbild.

Beispiel: Filter nicht aktiv, alle Elemente anzeigen:
 Filter aktiv, nicht alle Elemente anzeigen:

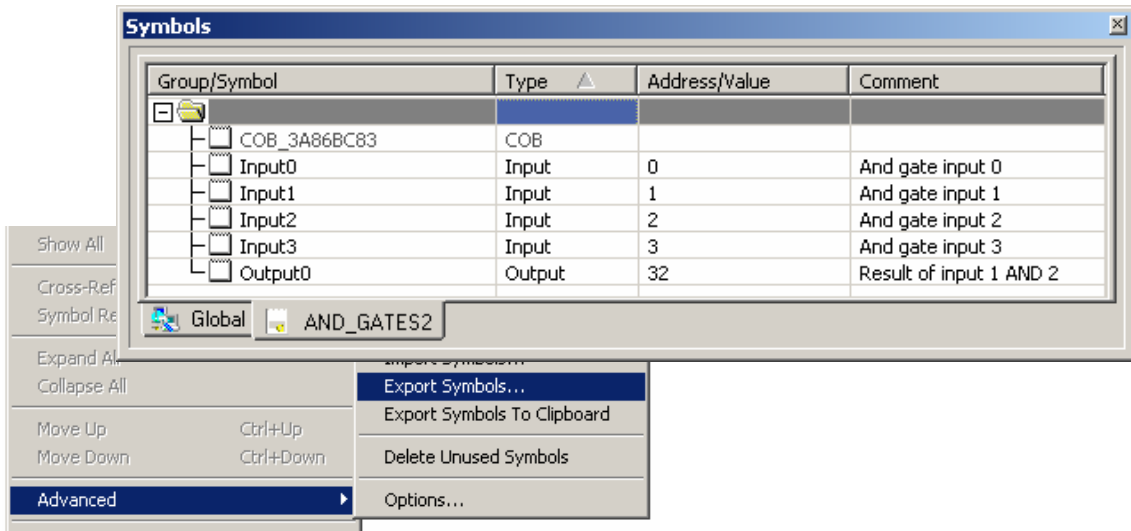


3.5.16 Symbole exportieren

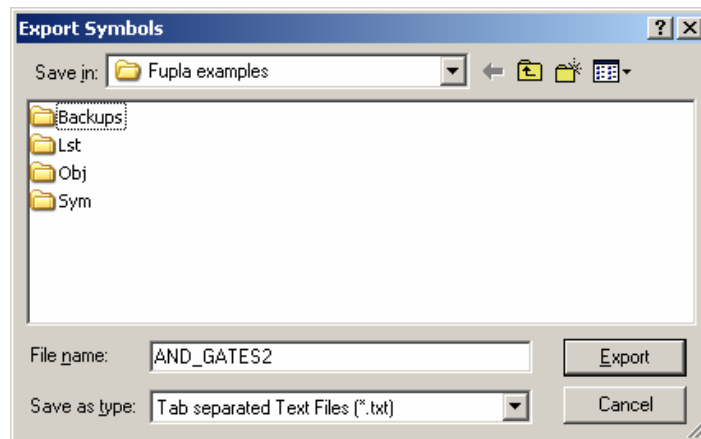
Die Symbolliste eines Anwenderprogramms kann in andere Programme, wie Excel, Visiplus oder Word, exportiert werden. Zum Beispiel als Liste der Ein-/ Ausgangsbelegung für eine Bedienungsanleitung.

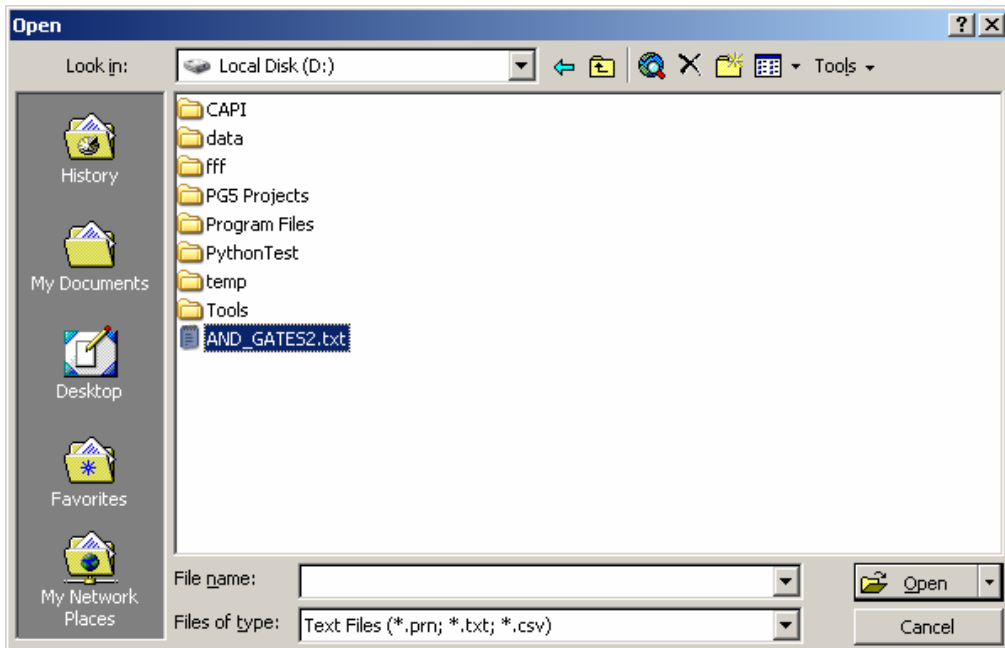
Beispiel: Export nach Excel:

Aus dem Menükontext des Symbol-Editors "Export Symbols..." wählen.



Beim Export der Symbolliste nach Excel wird ausdrücklich empfohlen, das „Tab separated Text Format“ (*.txt) zu wählen. Damit wird ein besseres Ergebnis erzielt, als mit dem „Excel File“ Format (*.xls).





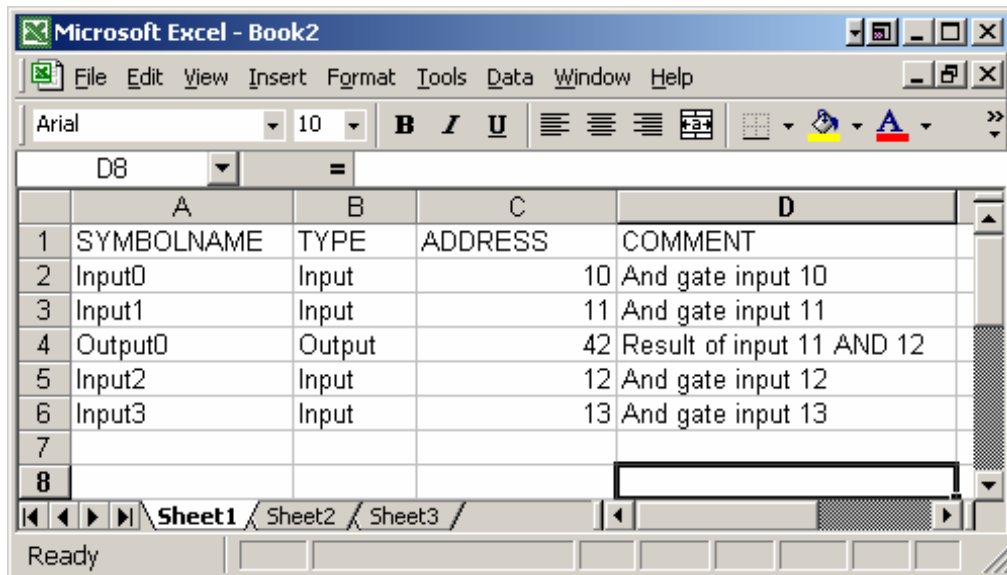
Excel starten und die Text-Datei (*.txt) mit den exportierten Symbolen öffnen.

The screenshot shows Microsoft Excel with the file 'AND_GATES2.txt' open. The spreadsheet contains the following data:

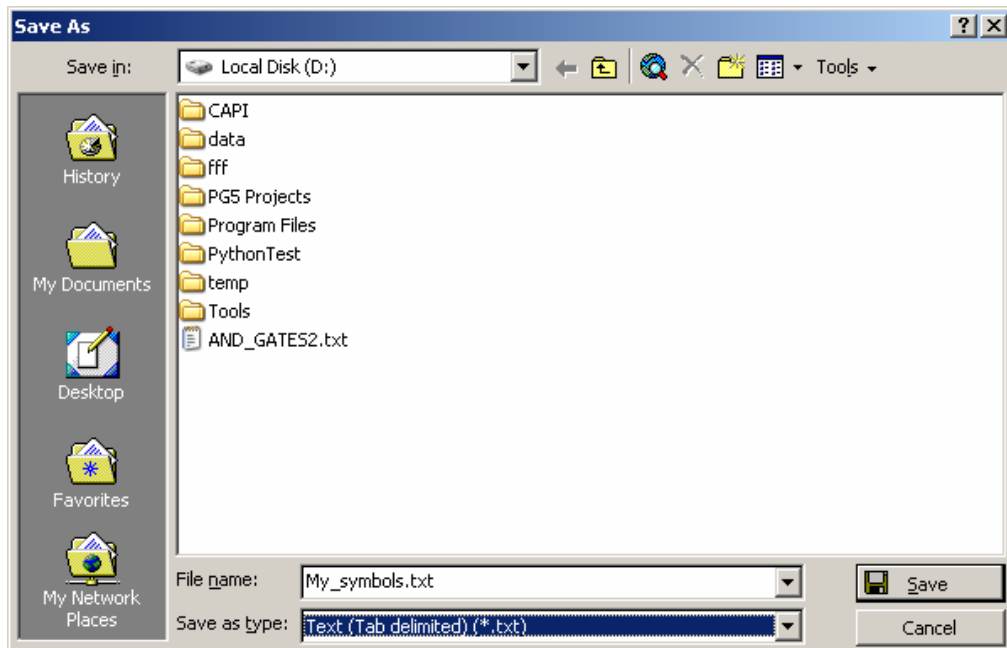
	A	B	C	D
1	SYMBOLNAME	TYPE	ADDRESS	COMMENT
2	Input0	Input	0	And gate input 0
3	Input1	Input	1	And gate input 1
4	Output0	Output	32	Result of input 1 AND 2
5	Input2	Input	2	And gate input 2
6	Input3	Input	3	And gate input 3
7	COB_3A86BC83	COB		
8				

3.5.17 Symbole importieren

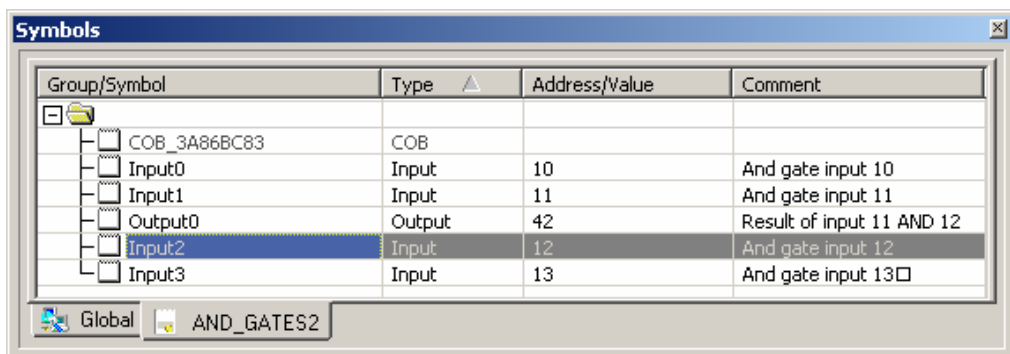
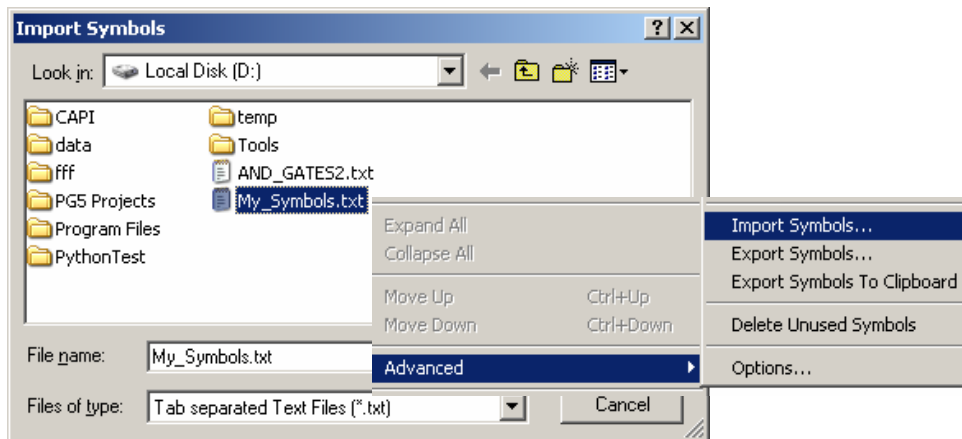
Es ist auch möglich, eine Symbolliste mit dem Excel-Editor zu erstellen und in ein PG5-Projekt zu importieren. Dazu muss die erstellte Symbolliste, wie im Folgenden gezeigt, als Text-Datei (*.txt) im Format "Tabstopp-getrennt" gespeichert werden.



	A	B	C	D
1	SYMBOLNAME	TYPE	ADDRESS	COMMENT
2	Input0	Input	10	And gate input 10
3	Input1	Input	11	And gate input 11
4	Output0	Output	42	Result of input 11 AND 12
5	Input2	Input	12	And gate input 12
6	Input3	Input	13	And gate input 13
7				
8				



Im Kontext-Menü des PG5 Symbol-Editors mit "Advanced, Import Symbols" die Text-Datei auswählen und importieren. Bitte beachten, dass die Excel-Datei richtig geschlossen wurde.



3.5.18 Symbole initialisieren

Es gibt 2 Möglichkeiten, die von der PCD benutzten Symbole zu initialisieren:

- Initialisierung beim Kaltstart (Speisung Ein)
- Initialisierung beim Programm-Download in die PCD

Beim Kaltstart

Die Initialisierung der Symbole beim Kaltstart wird im XOB 16 vorgenommen. Dieser Organisationsblock wird beim Kaltstart nur einmal durchlaufen. Der vom Anwender geschriebene IL- oder Fupla-Code zur Initialisierung der Symbole wird ausgeführt.

Beispiel: Initialisierung eines Flags und eines Registers beim Kaltstart.

Programm in IL

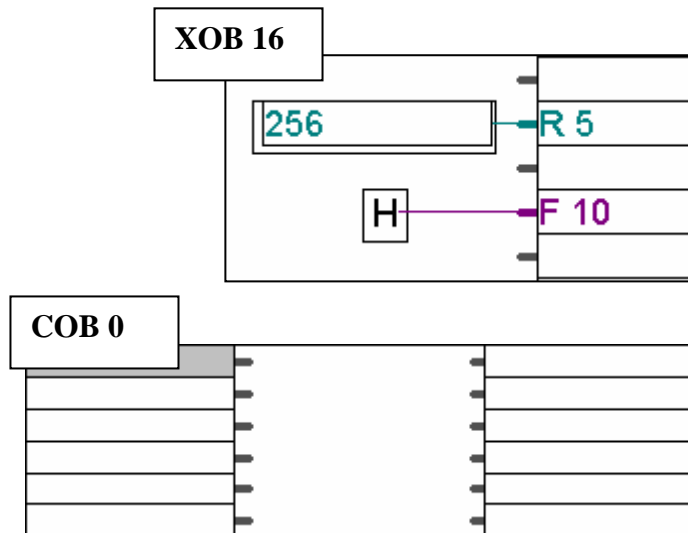
```

XOB 16      ;Kaltstart
            ; Block
LD   R 5    ; R 5 = 256
      256
SET  F 10   ;F 10 = 1

EXOB

COB 0      ;Zyklischer
      0    ;Block
...
;Anwenderprogramm
...
ECOB
    
```

Programm in Fupla



Weitere, detaillierte Informationen zu COBs und XOBs befinden sich im Kapitel 5 dieses Handbuchs.

Beim Programm-Download

Die Initialisierung der Symbole beim Programm-Download in die PCD erfordert, dass nach der Adresseangabe ein := (Doppelpunkt, Gleich) mit darauf folgendem Wert geschrieben wird, der dann als Initialisierungswert gilt.

Beispiel:

Group/Symbol	Type	Address/Value	Com
BL_3E315C55	COB	0	
SymbolA	R	5:= 256	
SymbolB	F	10:=1	



Achtung

Die Initialisierung beim Programm-Download wird nur wirksam, wenn diese Eigenschaft im Download-Menü angewählt ist.


 First-time Initialisation Data

3.5.19 Symbolische Namen

Symbolische Namen (Symbole) sind Bezeichnungen für Elemente in der PCD wie Eingänge, Ausgänge, Flags, Register, COBs usw. Symbolische Namen können bis 80 Charakter lang sein und sind nicht von Gross-/Kleinschreibung abhängig, solange keine länderspezifischen Zeichen verwendet werden. 'MotorOn' ist das Gleiche wie 'MOTORON' oder 'motoron' jedoch ist 'GRÜN' nicht das Gleiche wie 'grün'.

Symbole beginnen mit einem Buchstaben (a-z, A-Z). Zahlen werden nicht akzeptiert. Innerhalb der Symbole können Buchstaben und Zahlen sowie das Unterstreichzeichen '_' beliebig gemischt werden. Leerschläge (Space) dürfen nicht verwendet werden.

Reservierte Ausdrücke können nicht als Symbole verwendet werden.

3.5.20 Reservierte Ausdrücke

Die folgenden Ausdrücke sind reserviert und können nicht als Symbole verwendet werden:

- Assembler-Direktiven wie: PUBL, EXTN, EQU, DEF, LEQU, LDEF, MACRO, ENDM, EXITM usw..
- Medium Codes und Datentypen (I, O, F, R, C, T, K, M, COB, FB, TEXT, X, SEMA, DB).
- Spezialcodes des MOV-Befehls (N, Q, B, W, L, D).
- Codes für Bedingungen (H, L, P, N, Z, E).
- Alle Befehls-Mnemonics
- Vordefinierte Symbole
- Interne Symbole für die automatische Ressourcen-Verwaltung (Auto allocation), die mit Unterstrich beginnen, z. B.:
 _____TEXT, _____F.
- Internes Symbol __CSTART__ für die \$\$ Zuweisung.

Inhaltsverzeichnis

4	FUPLA-PROGRAMMIERUNG	3
4.1	Einleitung	3
4.2	Erstellen eines Fupla Projektes	4
4.2.1	Neues Projekt erstellen	4
4.3	Aufbau einer Fupla Seite	5
4.4	Symbols-Fenster	6
4.4.1	Neue Symbole der <i>Symbols</i> Liste hinzufügen	7
4.4.2	Symbolen Adressierung	8
4.4.3	Benutzen der Symbole aus der <i>Symbols</i> -Liste im Fupla Programm	9
4.4.4	Lokale und Globale Symbole	10
4.5	Editieren der Connectors	11
4.5.6	Platzierung der Connectors	11
4.5.7	Editieren des Symbols in einem Connector	11
4.5.8	Schnellverfahren für die Platzierung eines Symbols und seines Connectors	11
4.5.9	Ziehen, Kopieren/Einfügen, Löschen eines Symbols	11
4.5.10	Kopieren/Einfügen, Löschen eines Connectors	12
4.5.11	Strecken von Connectors	12
4.5.12	Vertikale Verschiebung eines Connectors	12
4.6	Editieren einer Fupla FBox	13
4.6.1	FBox selector	13
4.6.2	Editieren einer FBox	14
4.6.3	Editieren von erweiterbaren (ausziehbaren)FBoxen	14
4.6.4	Invertierung binärerer Signale	14
4.6.5	Dynamisierung (Flankentrigger)	15
4.6.6	Kommentare	15
4.6.7	FBox Hilfe	15
4.7	Editieren von Verbindungen zwischen den FBoxen und Connectors	16
4.7.1	Verbindung durch Verschieben einer FBox	16
4.7.1	Verbindung mit automatischem Routing	16
4.7.2	Mehrfachverbindung mit automatischem Routing	16
4.7.3	Verbindung aller Eingänge/Ausgänge einer FBox mit Connectors	16
4.7.4	Löschen von Linien, FBoxes, Connectors oder Symbolen	18
4.7.5	Vertikale Verschiebung einer FBox/eines Connectors ohne Aufhebung der Verbindungen	18
4.7.6	Einfügen einer FBox ohne Aufhebung der Verbindung	18
4.7.7	Wichtige Regeln	18
4.8	Editieren von FUPLA-Seiten	20
4.8.6	Fupla Seite einfügen	20
4.8.7	Fupla Seite löschen	20
4.8.8	Seitennavigation	20
4.8.9	Fupla Seiten-Dokumentation	21
4.8.10	Abarbeitung des Fupla Programmes in der PCD	21

4.9	Kopieren und Einfügen	22
4.9.1	Kopieren und Einfügen von Programmteilen	22
4.9.2	Kopieren und Einfügen von Symbolen	23
4.10	Export und Import von Fupla Seiten	24
4.10.1	Seiten Export	24
4.10.2	Seiten Import	25
4.11	Erstellen des ersten Fupla Programmes	27
4.11.1	Ziel	27
4.11.2	Vorgehensweise	27
4.11.3	Programmierung	29
4.12	Verarbeitung (build) des Programms	30
4.13	Laden des Programms in die PCD	31
4.14	Programmfehler finden und korrigieren (Debug)	31
4.14.1	Go On/Offline – Run – Stop - Step-by-step	31
4.14.2	Anhalte-Punkte (Breakpoints)	32
4.14.3	Anzeige von Symbolnamen oder absoluten Adressen	33
4.14.4	Anzeige des Symbolstatus in Fupla	33
4.14.5	Editieren von Symbolen online	33
4.14.6	Anzeige des Symbolstatus mit <i>Watch window</i>	34
4.14.7	Echtzeituhr der PCD einstellen	35
4.15	FBox Einstellfenster	36
4.15.1	Typen der Parametrierdaten	37
4.15.2	Initialisierung der HLK FBoxen	38
4.15.3	FBoxen mit Parametrierdaten	39
4.15.4	Kleinst HLK Applikation	39
4.15.5	Parametrierdaten nach dem Laden des Anwenderprogramms	40
4.15.6	On-line Überschreiben der Parametrierdaten.	40
4.15.7	Lesen der On-line Parametrierdaten	41
4.15.8	Parametrierdaten mit Standardwerten beschreiben	41
4.15.9	Definition von Symbolnamen für die Referenzierung von FBoxen	42
4.15.10	Definition der absoluten Adressen für die Parametrierdaten	43
4.16	Inbetriebnahme eines Analogmoduls	44
4.16.6	Erfassen einer Analogmessung	44
4.16.7	Beispiel für PCD2.W340 Analog-Eingangsmodule	45
4.16.8	Beispiel für PCD2.W610 Analog-Ausgangsmodule	46

4 FUPLA-Programmierung

4.1 Einleitung

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

Fupla, "*F*unction *P*LAn" 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.

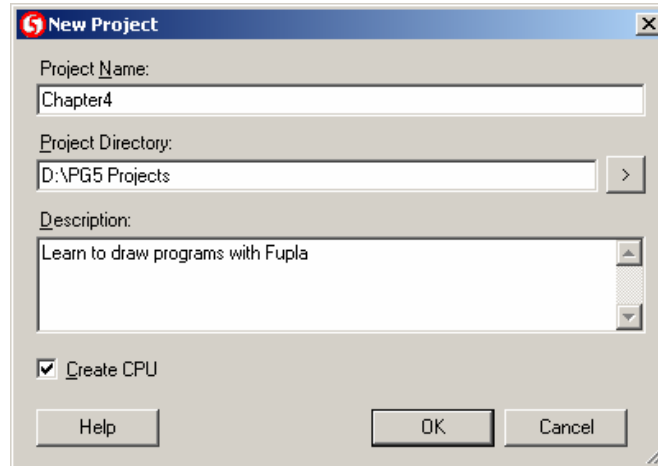
Der grosse Vorteil von Fupla liegt darin, dass der Anwender ein PCD Programm erstellen kann, ohne dass er eine Linie Programmcode schreiben muss, und dass er etwas über die Programmcode kennen muss.

4.2 Erstellen eines Fupla Projektes

Das Fupla Beispiel wird in einem neuen PG5 Projekt abgespeichert.

4.2.1 Neues Projekt erstellen

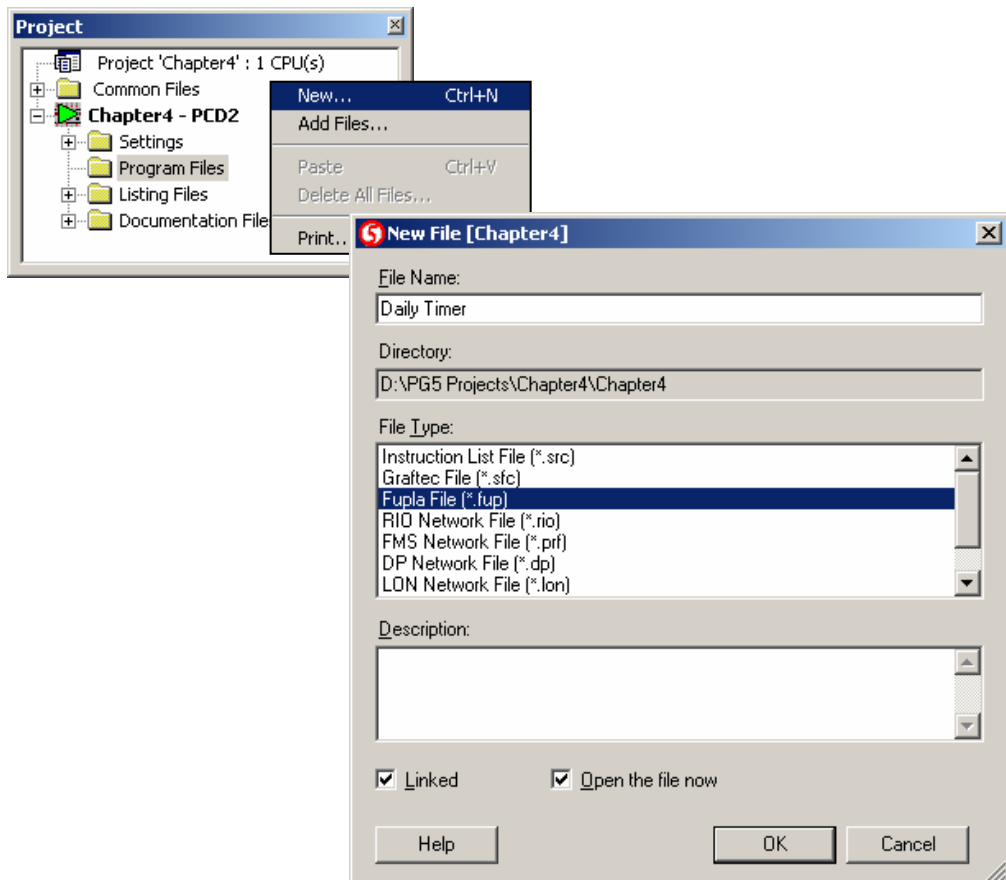
Im SAIA Project Manager Fenster, Auswahl von *File, Project, New...* und neues Projekt erstellen.



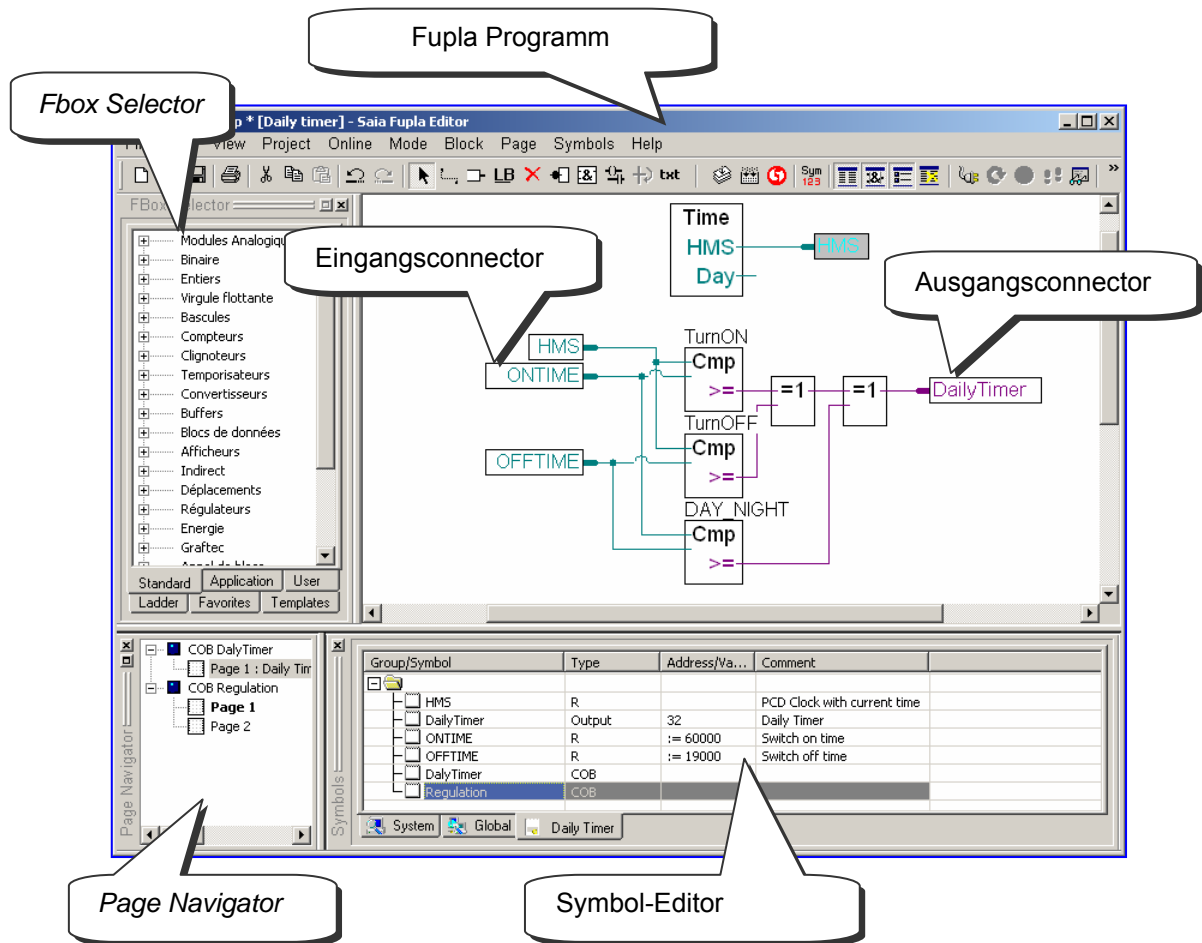
Zum Erstellen einer neuen Programmdatei ist "New File" oder die rechte Maustaste anzuklicken.



New File



4.3 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: *Standard, Wandler*).

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

4.4 Symbols-Fenster



Show Hide
Symbols Editor

Group/Symbol	Type	Address/value	Comment
HMS	R		PCD Clock ...
DailyTimer	Output	32	Daily Timer
ONTIME	R	:=60000	Switch on time
OFFTIME	R	:=19000	Switch off time

System Global Daily Timer

Das *Symbols* 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

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

Jeder Symbol-Typ besitzt einen bestimmten Bereich verfügbarer Adressen:

Ein-/Ausgänge: abhängig von den in der PCD gesteckten Modulen


Flags: F 0...F 8191

Register: R 0, ..., R 4095¹, 16383²

Timer/Counter: T/C 0...T/C 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

¹ Alle PCD

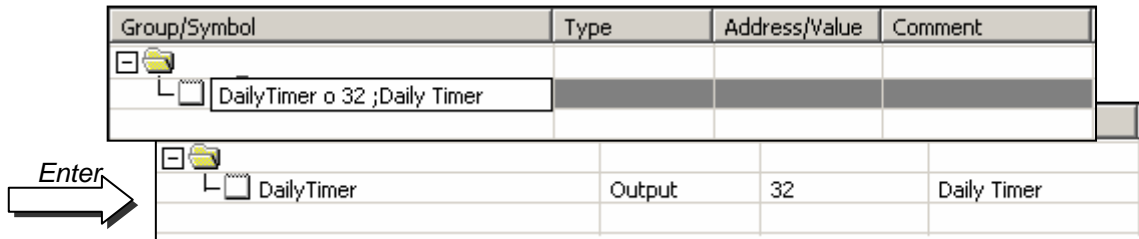
² PCD2.M480,PCD3

4.4.1 Neue Symbole der Symbols Liste hinzufügen

Einfache Methode:

Zum hinzufügen von neuen Symbolen zur Liste das *Symbols*-Fenster öffnen, die Maus in der Mitte des Fensters positionieren und mit rechtem Maus-Klick das Kontext-Menü *Insert Symbol* auswählen. Dann die Felder *Group/Symbol*, *Type*, *Address/Value* und *Comment* ausfüllen.

Schnelle Methode 1:

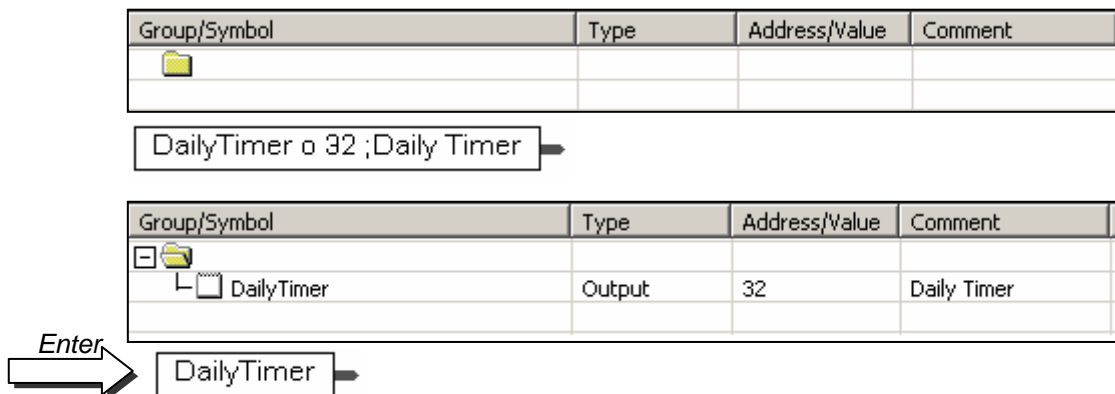


Eine weitere Möglichkeit ist die Eingabe von Variablen für die verschiedenen Informations- Felder vom *Group/Symbol* Feld. Das ist praktischer und schneller. Siehe Beispiel unten.

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-Code 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.

4.4.2 Symbolen Adressierung



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

Absolute addresses

Group/Symbol	Type	Address/Value	Comment
			
 	Output	32	Daily Timer



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

Group/Symbol	Type	Address/Value	Comment
			
 DailyTimer	Output	32	Daily Timer

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

Group/Symbol	Type	Address/Value	Comment
			
 HMS	R		PCD Clock with

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.

4.4.3 Benutzen der Symbole aus der Symbols-Liste im Fupla Programm

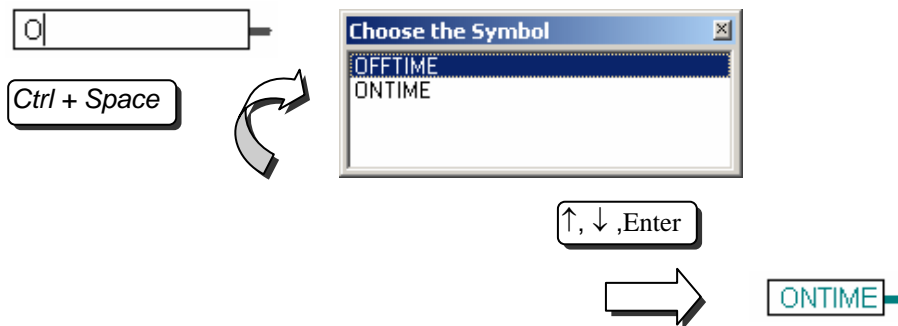
Nach erfolgter Programmierung können die im *Symbols*-Fenster festgelegten Symbole auf verschiedene Arten benutzt werden:

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.

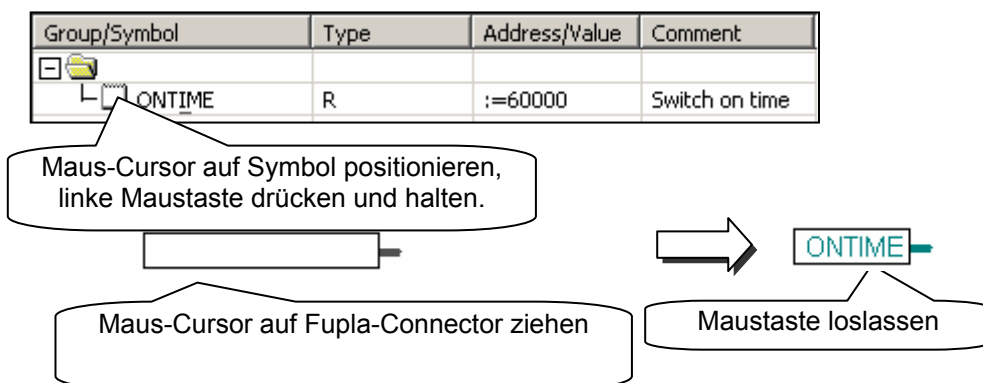
Symbol eingeben durch selektive Suche

Group/Symbol	Type	Address/Value	Comment
HMS	R		PCD Clock with current time
DailyTimer	Output	32	Daily Timer
ONTIME	R	:=60000	Switch on time
OFFTIME	R	:=19000	Switch off time



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 aller 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.

4.4.4 Lokale und Globale Symbole

Das *Symbols*-Fenster besteht aus zwei Ordnern: *Global* und *Lokal*



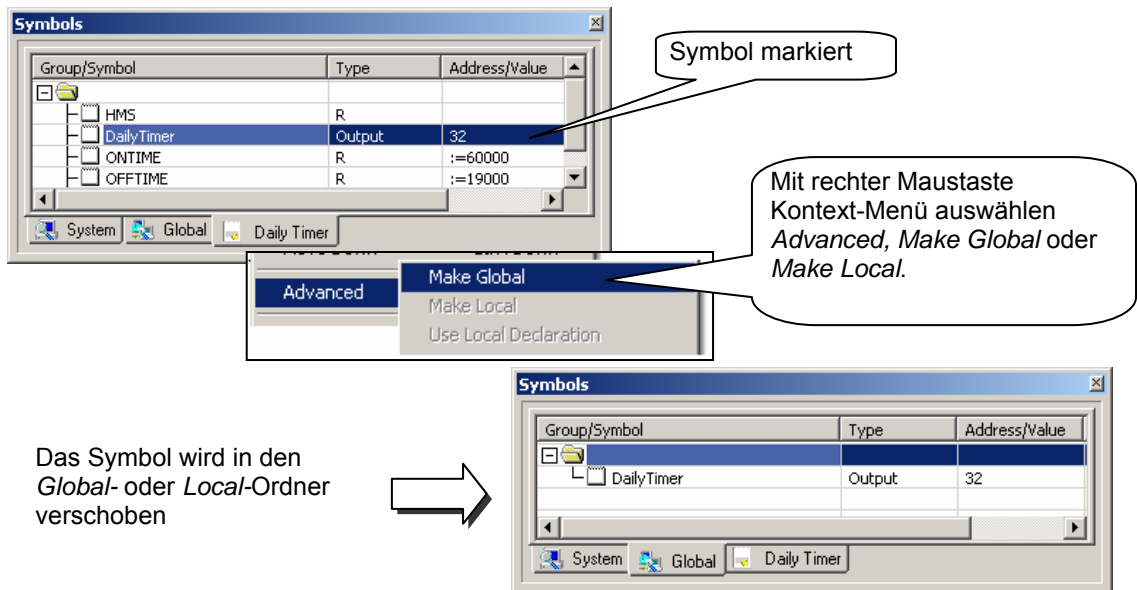
Definition

Lokale Symbole erscheinen in dem Ordner, der den Namen der Datei trägt die diese Symbole benutzt (*Parking lot.src*).

Globale Symbole im *Global* Ordner können von allen Dateien der CPU benutzt werden (*Parking lot.src* und *Ventilation.src*).

Lokal-Ordner in den Global-Ordner verschieben

Symbole im *Symbols*-Fenster können, wenn gewünscht, vom Lokal-Ordner in den Global-Ordner verschoben werden und umgekehrt.



N.B.:

Alle neuen Symbole, die direkt vom Fupla Editor definiert werden, werden abhängig von den Einstellungen der Option *Add symbols to Global table* entweder zum Lokal-Ordner oder zum Global-Ordner hinzugefügt. Siehe Kontextmenü *Advanced*, *Options* im *Symbols* Fenster.

4.5 Editieren der Connectors

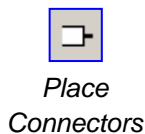
Die Eingangs- und Ausgangsconnectors können an einer beliebigen Stelle auf den Fupla-Seiten platziert werden und können die Symbole enthalten, die für die von den FBoxes beschriebenen Programmfunktionen notwendig sind.

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..., Layout, 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 Empty connectors*.

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

4.5.6 Platzierung der Connectors



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.

4.5.7 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* Fenster angezeigt wird.

4.5.8 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.

4.5.9 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.

4.5.10 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.

4.5.11 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*.

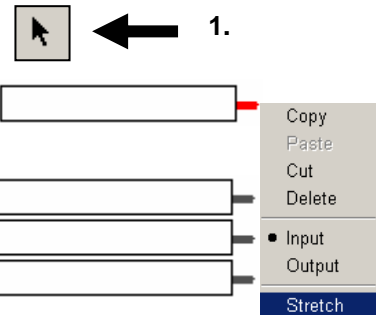
Wählen Sie einen Connector in einem rot markierten Bereich.

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.



4.5.12 Vertikale Verschiebung eines Connectors

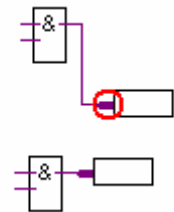
Um den Connector zu verschieben, platzieren Sie den Mauszeiger auf dem roten Kreis.

Drücken Sie die Umschalttaste und halten Sie sie gedrückt.

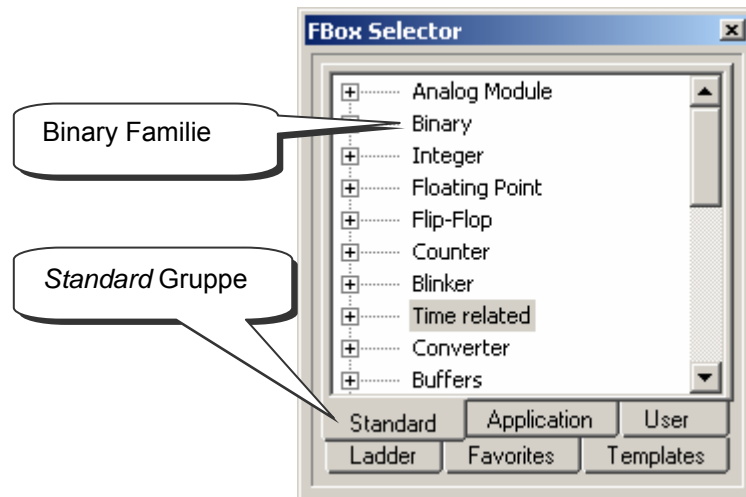
Drücken Sie die linke Maustaste und halten Sie sie gedrückt.

Ziehen Sie den Mauszeiger vertikal auf einen leeren Platz auf der Seite.

Lassen Sie Maus- und Umschalttaste los.



4.6 Editieren einer Fupla FBox



4.6.1 FBox selector



Alle, zur Programmgestaltung nötigen grafischen Funktionen (FBoxen) sind im FBox Selector Fenster aufgelistet. Die FBoxen sind in Gruppen aufgeteilt. Jede Gruppe hat eine eigene Registriertkarte.

- Die FBox Gruppe *Standard* enthält Basis FBoxen welche praktisch in allen Anwendung verwendet werden.
- Die Gruppe *Application* beinhaltet FBoxen für spezifische Applikationen wie z.Bsp HLK, Modem etc.
- Die Gruppe *Ladder* beinhaltet die grafischen Elemente welche für die Darstellung des Kontaktplans benötigt werden.

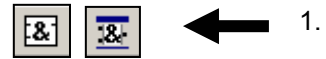
Jede FBox Gruppe ist in FBox-Familien unterteilt. FBoxen die einen bestimmten gemeinsamen Applikationsbereich abdecken werden in der gleichen FBox- Familie zusammengefasst. Zum Beispiel beinhaltet die Gruppe *Standard* die folgenden FBox-Familien:

<i>Binäre Funktionen</i>	FBoxen für logische Verknüpfungen
<i>Ganzzahl</i>	FBoxen für Ganzzahlarithmetik
<i>Arithmetik</i>	
<i>Fließpunkt</i>	FBoxen für Fließpunktarithmetik
<i>Arithmetik</i>	
<i>Counter</i>	FBoxen für Zählfunktionen
<i>Zeitfunktionen</i>	FBoxen für Zeitfunktionen
<i>Analogmodule</i>	FBoxen für die Handhabung Analogger Module
<i>Communication</i>	FBoxen für den Datenaustausch von Registern, Flags, etc. über S-Bus via Netzwerk, Ethernet oder Modem
<i>Wandler</i>	FBoxen für die Konvertierung von Binär Werten in Integer Wert, Integer Werte in Fließkomma Werte etc.

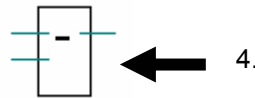
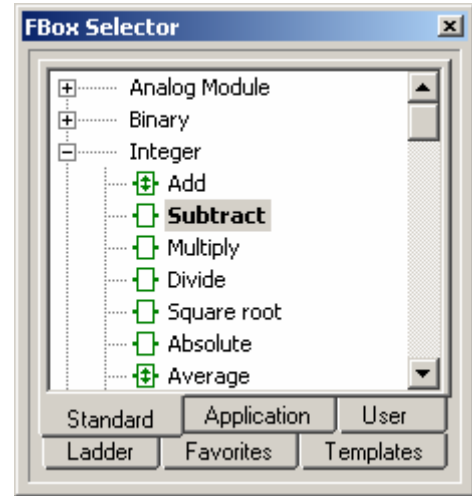
...

4.6.2 Editieren einer FBox

Die für die Programmgestaltung benötigten FBoxen werden zuerst im *FBox Selector* Fenster ausgewählt und dann in das Fupla Programm eingefügt.



1. Auswahl des *Add FBox* oder *Show/Hide FBox* Knopfes in der Werkzeugleiste.
2. Öffnen einer Funktionsfamilie.
3. Auswahl einer FBox.
4. Positionieren der FBox auf der Fupla Seite, danach linke Maustaste anklicken.

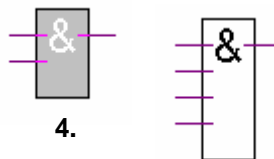
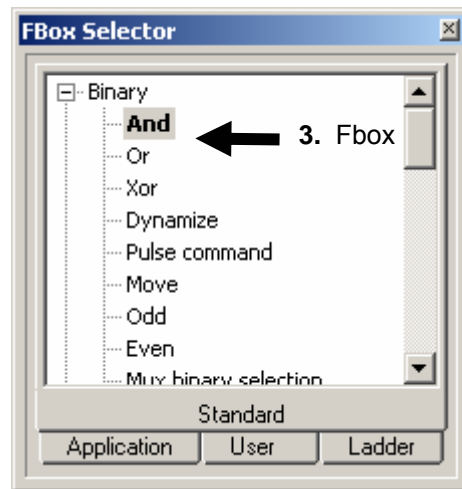


4.6.3 Editieren von erweiterbaren (ausziehbaren)FBoxen

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

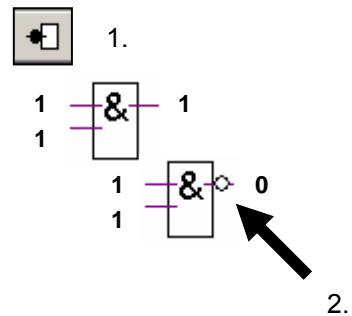


1. Auswahl des *Add FBox* oder *Show/Hide FBox* Knopfes in der Werkzeugleiste.
2. Öffnen einer Funktionsfamilie.
3. Auswahl einer FBox.
4. Positionieren der FBox auf der Fupla Seite, danach linke Maustaste anklicken.
5. Maus vertikal bewegen bis die gewünschte Anzahl Ein- oder Ausgangsanschlüsse sichtbar sind.
6. Linke Maustaste anklicken.



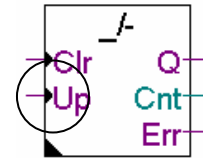
4.6.4 Invertierung binärer Signale

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



4.6.5 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.



4.6.6 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.



Daily Timer

4.6.7 FBox Hilfe

Um eine detaillierte Beschreibung jeder Funktion zu erhalten muss die entsprechende Fbox mit dem *FBox Selector* ausgewählt sein, und danach die F1 Taste betätigt werden.

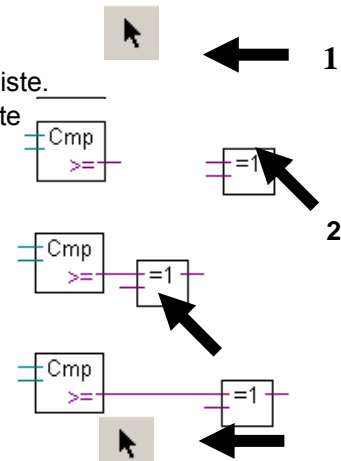
Eine andere Möglichkeit besteht darin, den Mauszeiger auf die entsprechende FBox im Fupla Editor zu positionieren und danach die linke Maustaste zu doppelklicken.

Eine unbekannte FBox kann rasch erkannt werden, indem das *FBox Selector* Fenster geöffnet wird, und danach im FUPLA Editor der Mauszeiger über die unbekannte FBox positioniert wird. Wird danach die linke Maustaste angeklickt, so erscheint im *FBox Selector* Fenster die entsprechende Funktion.

4.7 Editieren von Verbindungen zwischen den FBoxen und Connectors

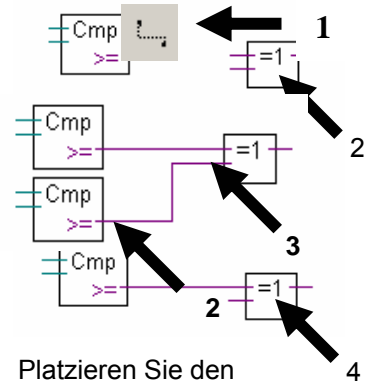
4.7.1 Verbindung durch Verschieben einer FBox

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 gewünschten Stelle ziehen.
4. Die Verbindungen zwischen den FBoxen werden erstellt, sobald sich zwei Kontaktstellen berühren.



4.7.1 Verbindung mit automatischem Routing

- 1 Klicken Sie auf die Taste *Auto Line Mode* auf der Werkzeugleiste.



- 2 Mauszeiger auf dem Ausgangsort und drücken Sie die linke Maustaste.

Platzieren Sie den

- 3 Mauszeiger auf dem Zielort und drücken Sie die linke Maustaste.

Platzieren Sie den

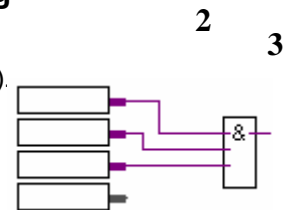
Anmerkung:

Es können auch Zwischenstationen ausgewählt werden.

Um die Arbeit an der Verbindung zu unterbrechen, drücken Sie die rechte Maustaste.

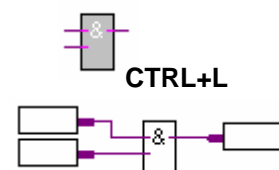
4.7.2 Mehrfachverbindung mit automatischem Routing

- 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.



4.7.3 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 All*.



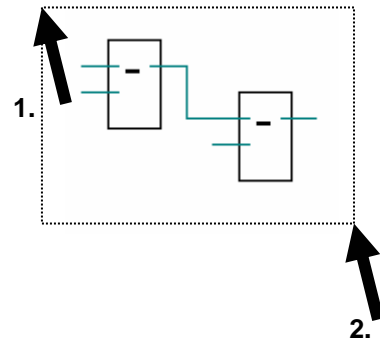
4.7.4 Löschen von Linien, FBoxes, Connectors oder Symbolen

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



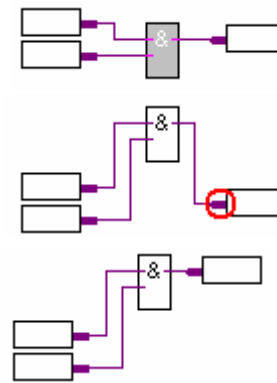
Im Schnellverfahren können Sie einen Bereich markieren und ihn löschen.

- 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*.



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

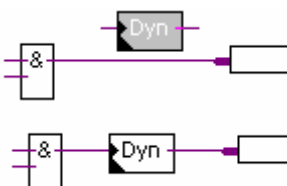
Platzieren Sie den Mauszeiger auf einer FBox.
 Drücken Sie die Umschalttaste und halten Sie sie gedrückt.
 Drücken Sie die linke Maustaste und halten Sie sie gedrückt.
 Ziehen Sie den Mauszeiger vertikal auf einen leeren Platz auf der Seite.
 Lassen Sie Maus- und Umschalttaste los.



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

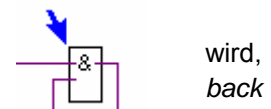
4.7.6 Einfügen einer FBox ohne Aufhebung der Ver

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



4.7.7 Wichtige Regeln

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



Es sind keine direkten Verbindungen zwischen Eingangs- und Ausgangsconnectors erlaubt. Es muss mit einer FBox

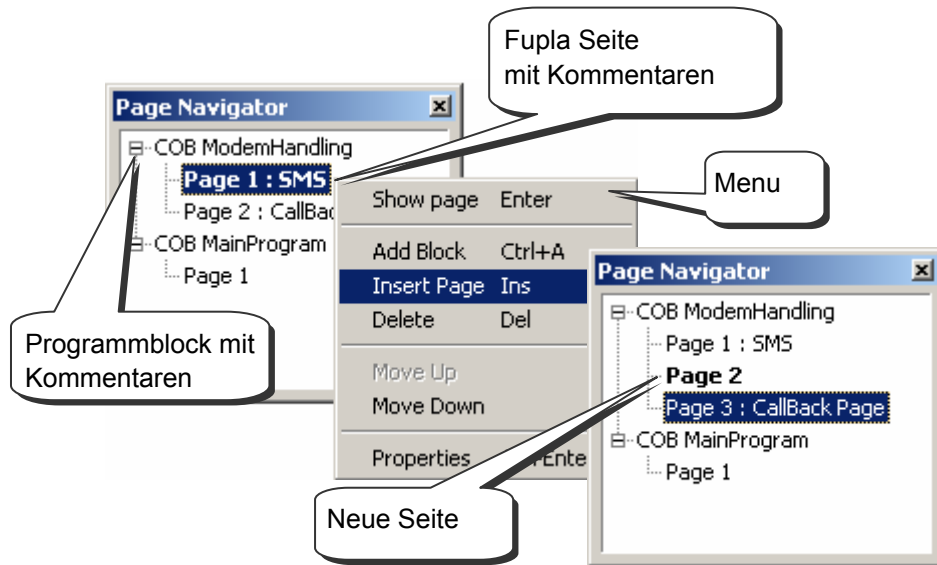


gearbeitet werden: *Binary, Direct transfer* oder *Integer, Direct transfer*.

Symbole von Ausgangsconnectors müssen stets mit einer FBox verbunden sein. Trifft dies nicht zu, wird eine Fehlermeldung angezeigt: Page 1: Error 53: Incomplete network



4.8 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 5 dieses Handbuches ersichtlich.

4.8.6 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* Taste der Tastatur betätigt wird oder über den Menüpunkt *Page* in der Werkzeugliste: *Page Insert After* (*Page Insert Before*)

4.8.7 Fupla Seite löschen

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

4.8.8 Seitennavigation



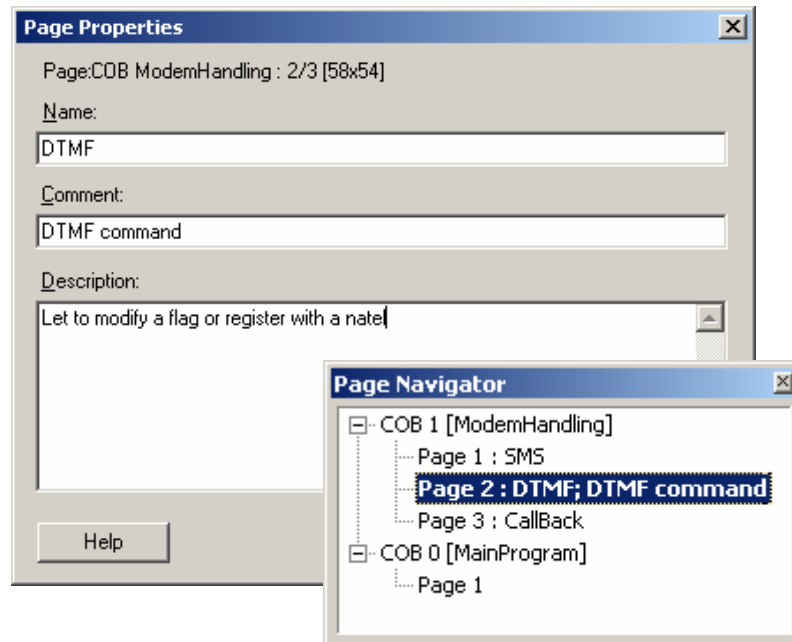
Goto Next
Page

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.

4.8.9 Fupla Seiten-Dokumentation

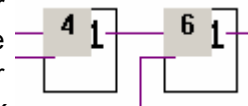
Es wird ausdrücklich empfohlen, jede Fupla Seite zu dokumentieren. Der Name und der Kommentar jeder Fupla Seite werden im *Page navigator* Fenster angezeigt und erleichtert das Navigieren zwischen den Seiten. Im Eingabefeld *Description* können nützliche Informationen über das Programm eingegeben werden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



4.8.10 Abarbeitung des Fupla Programmes in der PCD

In der PCD wird das Fupla Programm wie folgt abgearbeitet:

Die Fupla Seiten eines Programmblöckes 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, FBox priorities* der Werkzeugleiste angezeigt werden.

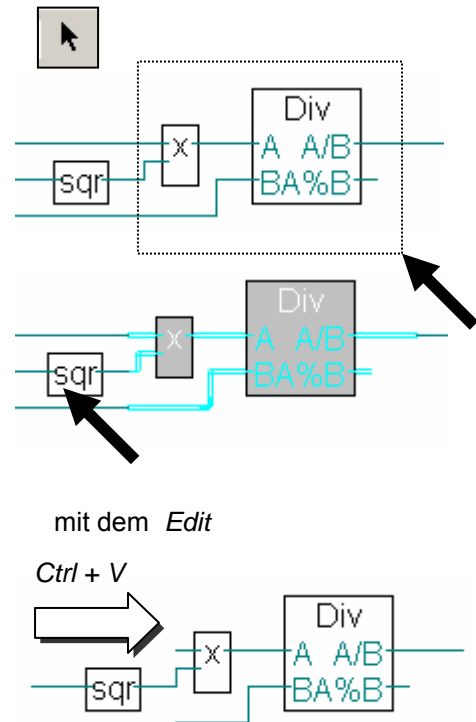


4.9 Kopieren und Einfügen

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.

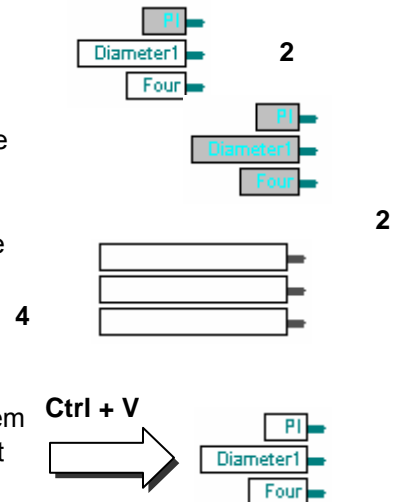
4.9.1 Kopieren und Einfügen von Programmteilen

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.
4. Kopieren des ausgewählten Bereiches *Copy* Menu der Werkzeugleiste, oder mit den *Ctrl + C* Tasten.
5. Einfügen des kopierten Bereiches mit dem *Edit Paste* Menu der Werkzeugleiste, oder der *Ctrl + V* Tasten.
6. 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.



4.9.2 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 letzte 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

4.10 Export und Import von Fupla Seiten

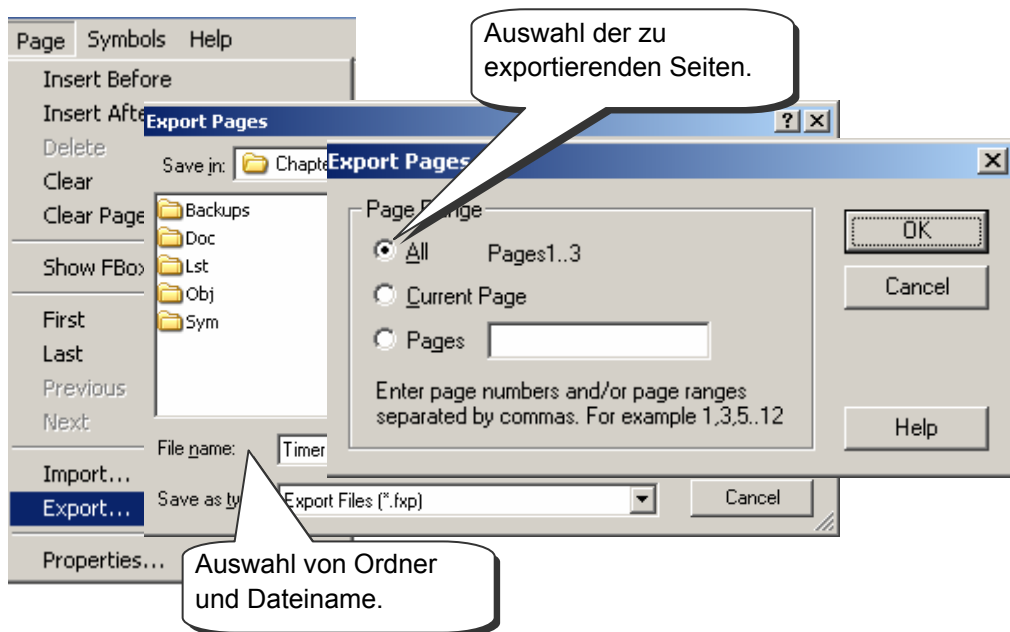
Die Seiten-Export/Import Funktionalität ist eine leistungsfähige Zusatzfunktion der Kopieren/Einfügen Funktion und bietet folgende Vorteile:

- Kopieren und Einfügen von mehreren Seiten
- Modifizieren von importierten Symbolen und absoluten Adressen
- Möglichkeit eine Komponentenbibliothek zu erstellen

4.10.1 Seiten Export

Die Funktion "Seiten Export" wird verwendet um eine oder mehrere ausgewählte Fupla Seiten in einer Export Datei (*.fxp) zu speichern. Die Fupla Seiten werden mit den FBoxen, Verbindungen, Kommentaren und Symbolen gespeichert. Diese Export Dateien (*.fxp) können dann zu einer Komponentenbibliothek zusammengefügt werden und können danach zur einfachen und schnellen Erstellung von zukünftigen Projekten verwendet werden.

Wird zum Beispiel in jedem Projekt die Zeitschaltuhr Funktionalität verwendet, welche jeden Tag, jeden Woche oder jeden Monat zu bestimmten Uhrzeiten digitale Ausgänge ein- oder ausschaltet, so ist es sinnvoll, wenn diese Funktionalität in einer Export Datei (*.fxp) gespeichert wird und bei Bedarf in das aktuelle Projekt kopiert wird. Die Export Datei enthält mehrere Fupla Seiten welche jede eine einzelne Zeitschaltuhr (Tag, Woche, Monat) enthält und hat den Namen: *Timer.fxp*.

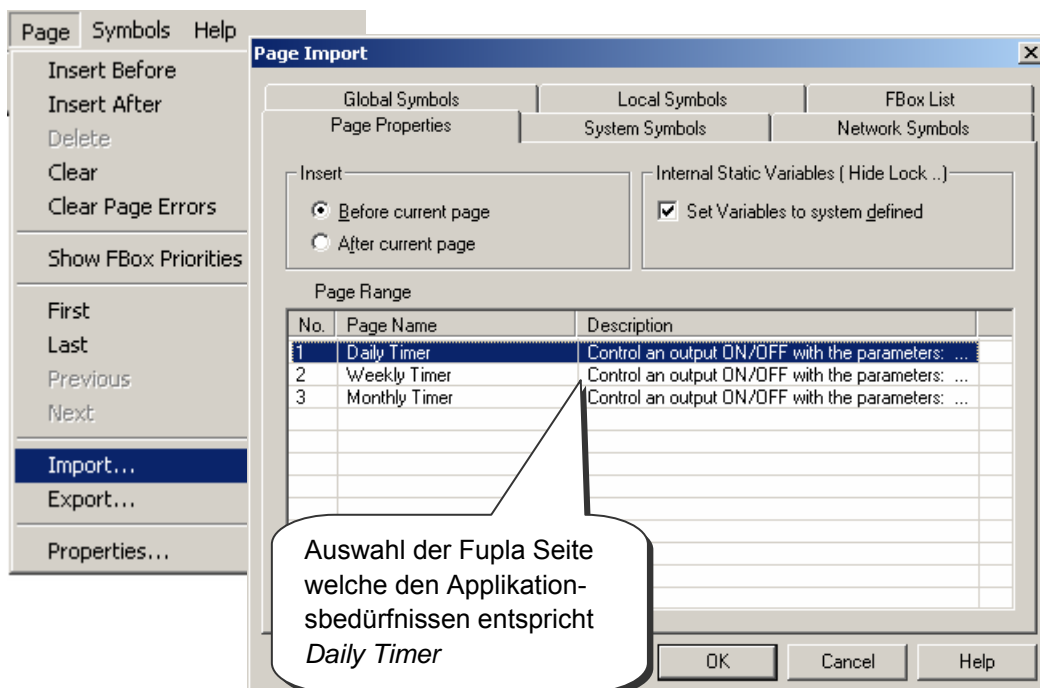


4.10.2 Seiten Import

Die Funktion "Seiten Import" wird verwendet um eine oder mehrere Fupla Seiten aus einer Export Datei (*.fxp) auszuwählen und die importieren Symbole dieser Seiten zu editieren.

In *Page Range* können die zu importierenden Fupla Seiten aus der Export Datei (*.fxp) ausgewählt werden.

Werden zum Beispiel in einem Projekt Zeitschaltfunktion verwendet, so können diese Zeitschaltfunktionen aus der im vorhergehenden Beispiel beschriebenen Export Datei *Timer.fxp* importiert werden. Beim Import in das Projekt können die, den Applikationsbedürfnissen entsprechenden Fupla Seiten ausgewählt werden und in das Projekt importiert werden.



In *Global Symbols* und *Local Symbols* werden die, in den ausgewählten Fupla Seiten verwendeten Symbole aufgelistet. Es ist möglich, den Namen und die Adresse oder den Wert jedes Symbols zu modifizieren.

Die schnellste Möglichkeit die Namen aller importierten Symbole zu wechseln besteht darin, die Symbole zu markieren und einer Gruppe zuzuordnen.

Diese Gruppenzuordnung ist mit dem Menüpunkt *Insert Pre-group* möglich.

Die Adressen Importierter Symbole kann auf effiziente Weise neu nummeriert werden, indem die entsprechenden Symbole markiert und im Menüpunkt *Renumber* entweder die Offset- oder Basisadresse angegeben wird.

In *FBox List* werden alle FBoxen aufgelistet, welche einen Namen (Referenz) in *FBbox properties* definiert haben. Diese Referenzen werden bei einigen FBoxen benötigt um mehrere FBoxen untereinander zu referenzieren. Der Name (Referenz) der FBoxen kann in dieser Liste modifiziert werden.

Ausgehend vom Beispiel mit den Zeitschaltfunktionen *Timer.fxp* wird nun die Komponente (Fupla Seite) *daily timer* in die Gruppe *Heating* integriert. Für jede Komponente *daily timer* wird ein neuer Gruppennamen definiert. Dadurch werden Komponentensymbole die mehrmals verwendet werden unterschiedlichen Gruppen zugeordnet und verhindern dadurch eine mehrfach- belegung der Ressourcen.

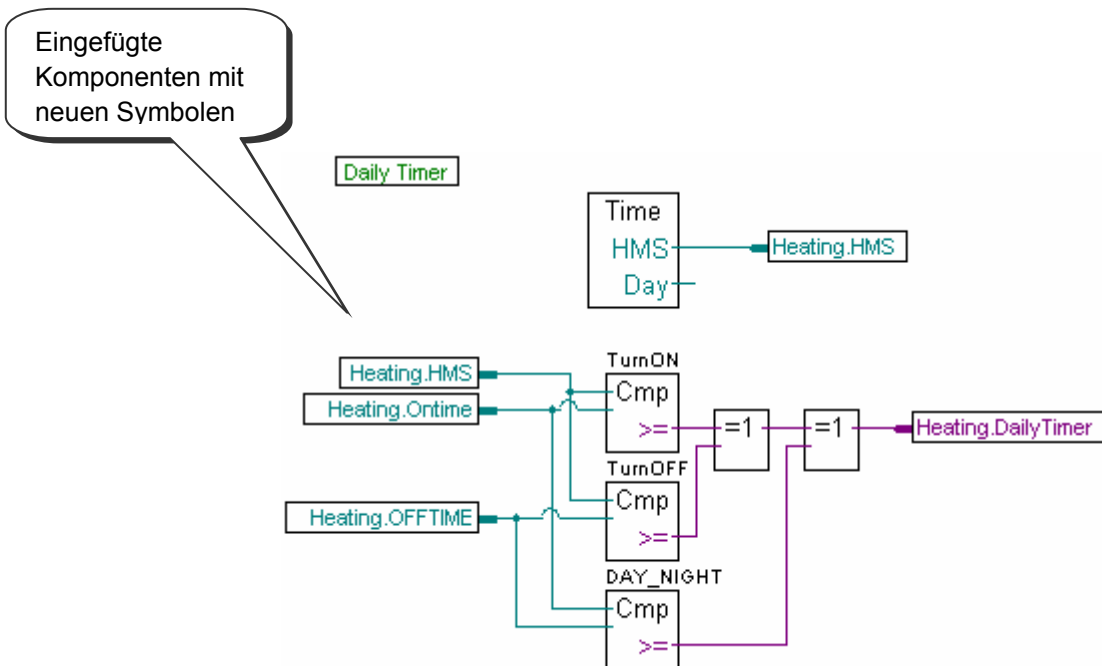
Page Import

Group	Symbol	Type	Address	Comment
	:=19000	R		Switch off time
	:=60000	R		Switch on time
	32	O		Daily Timer
		R		PCD Clock with c...

Page Import

Group	Symbol	Type	Address	Comment
Heating	OFFTIME	K	190000	Switch off time
	ONTIME	K	60000	Switch on time
	DailyTimer	F		Daily Timer
	HMS	R		PCD Clock with c...

Imported Symbol:



4.11 Erstellen des ersten Fupla Programmes

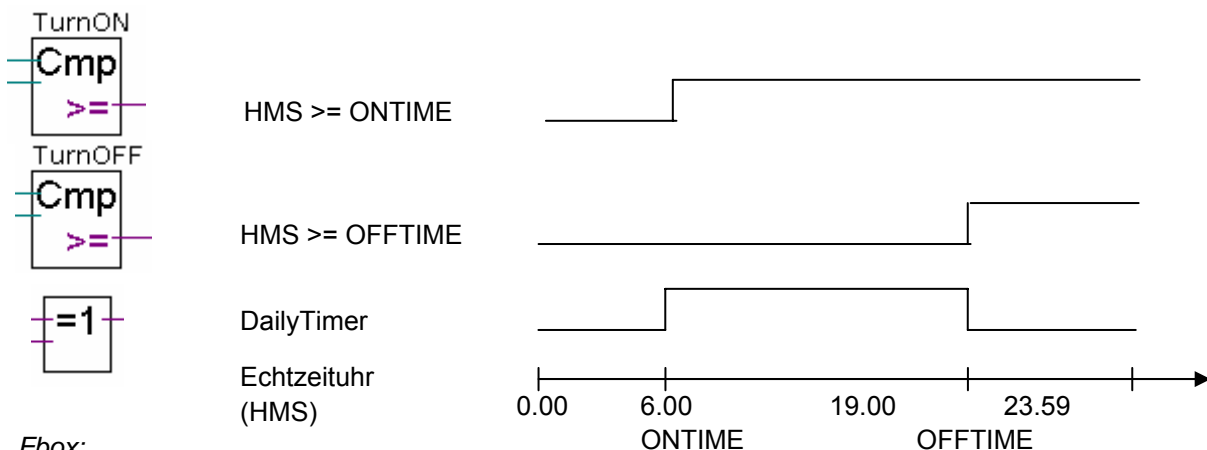
4.11.1 Ziel

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.

4.11.2 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.

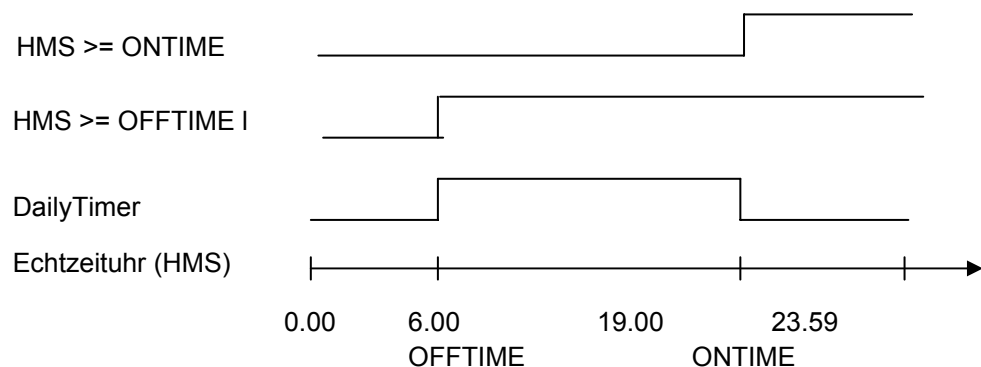
Für dieses Zeitschaltuhrbeispiel werden wir zwei Vergleichsfunktionen verwenden. Mit dem ersten Vergleich wird verglichen ob die Echtzeituhr (z.B. die Zeit unserer Uhr oder die Echtzeituhr in der PCD) in HMS (Stunden, Minuten, Sekunden) gleich oder grösser der Einschaltzeit *ONTIME* ist. Mit dem zweiten 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:

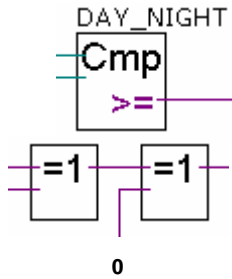
- Ganzzahl Arithmetik, Wert grösser/gleich
- Binäre Funktionen, Xor 2-10 Eingänge

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



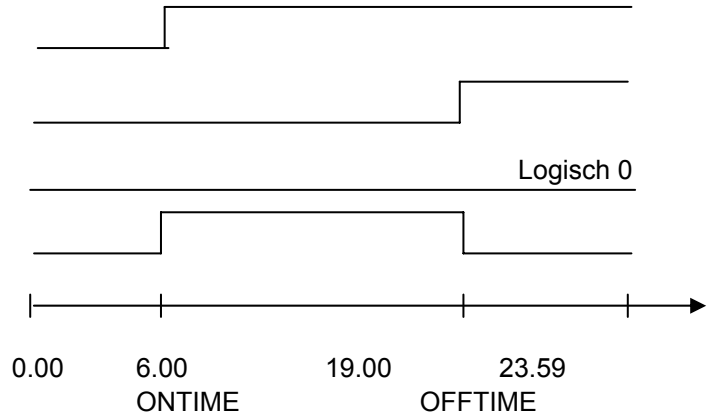
HMS >= ONTIME

HMS >= OFFTIME

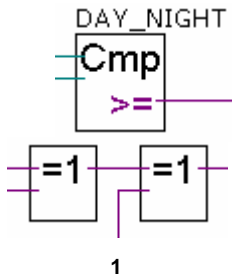
ONTIME >= OFFTIME

DailyTimer

Echtzeituhr (HMS)



Falls Ausgang bei Nacht geschaltet wird



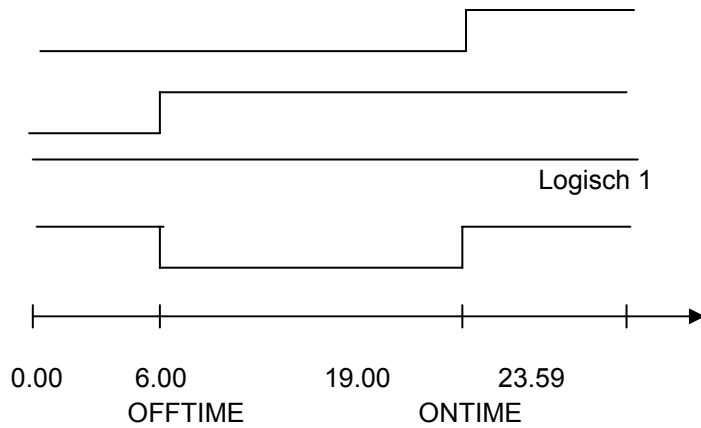
HMS >= ONTIME

HMS >= OFFTIME

ONTIME >=

DailyTimer

Echtzeituhr (HMS)



4.11.3 Programmierung

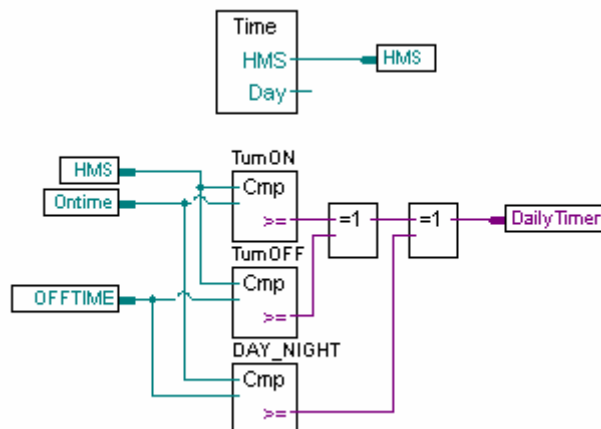
Es ist nun an der Zeit das Programm zu erstellen. Zu Beginn dieses Kapitels haben wir ein Projekt definiert welches eine Datei mit dem Namen *DailyTimer.fup* enthält. In diese Datei wird das Programmbeispiel geschrieben.

Group/Symbol	Type	Address/Value	Comment
HMS	R		PCD Clock with current time
DailyTimer	Output	32	Daily Timer
ONTIME	R	:=60000	Switch on time
OFFTIME	R	:=19000	Switch off time
COB_3A87C0D7	COB		

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 benötigten FBoxen sind in der *Standard* Gruppe des *FBox Selector* Fenster enthalten:

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

4.12 Verarbeitung (build) des Programms



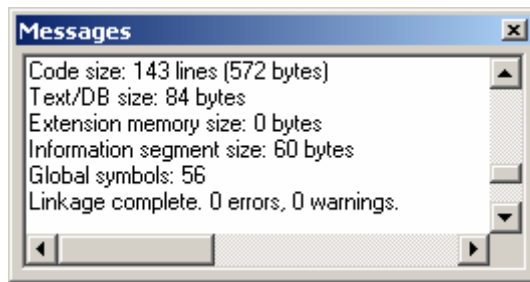
Build All

Bevor das erstellte Programm in der PCD abgearbeitet werden kann, muss es verarbeitet "build" werden. Dazu kann die Menüfunktion *CPU Build* oder *Build* 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.

Doppelklick mit Maustaste auf die Fehlermeldung

Compiling Files...
 Compiling: d:\PG5 Projects\Chapter4\Chapter4\Daily Timer.fup
S-FUP: Error 60: Daily Timer.fup: Page 1(1,27-1): Bad label.
Error: Compile failed: d:\PG5 Projects\Chapter4\Chapter4\Daily Timer.fup
 Build Failed

Fehler wird rot oder mit Pfeil markiert

Korrektur des Fehlers

4.13 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 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.

4.14 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.

4.14.1 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)



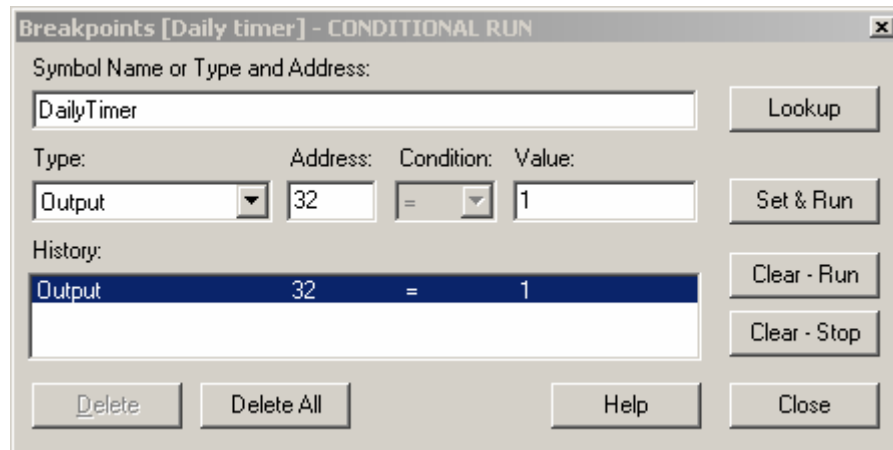
Die im nächsten Schritt abgearbeitete Fbox wird mit dem *Stop* Symbol angezeigt.

4.14.2 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.



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.

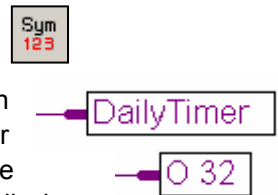
Wenn mehrere bedingte Anhalte-Punkte definiert sind, werden sie alle im Feld *History* aufgezeichnet. Es kann ein beliebiger Anhalte-Punkt mit der Maus ausgewählt und mit der Taste *Set & Run* aktiviert werden.

Anhalte-Punkt an einer Programm-FBox

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

4.14.3 Anzeige von Symbolnamen oder absoluten Adressen

Mit dem *Show Operand as symbol or value* 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.



4.14.4 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:

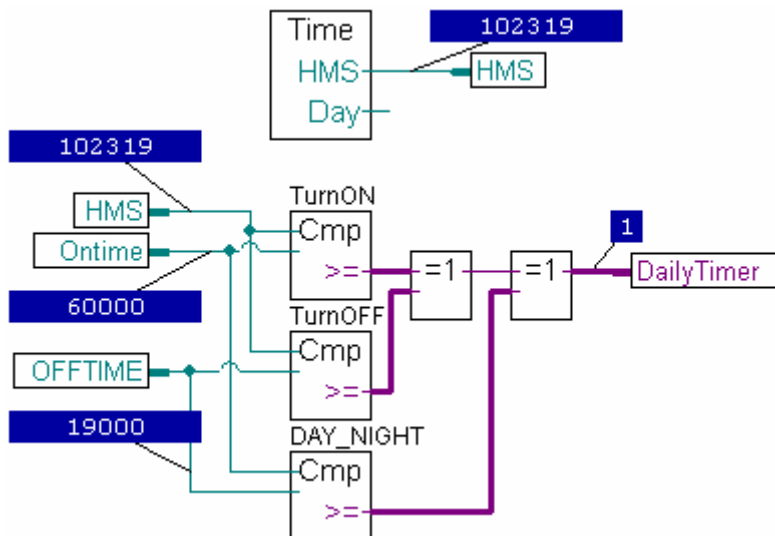
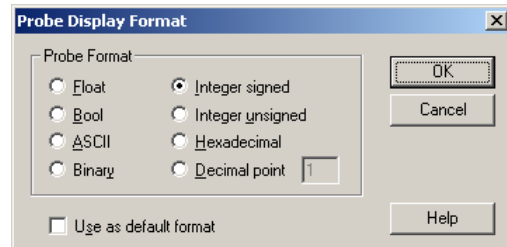


Place Probe

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.

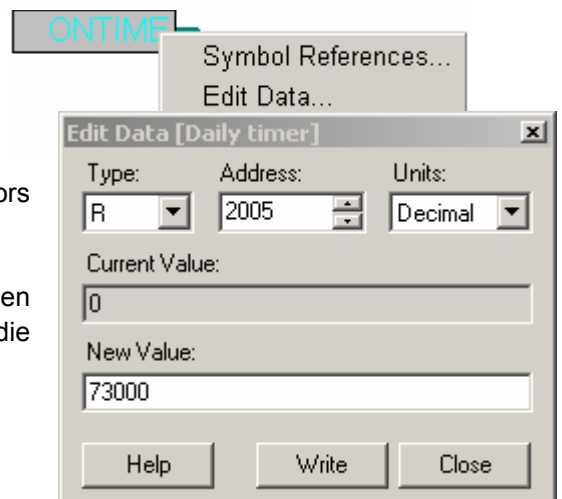
Ein Doppelklick auf eine Probe öffnet das *Probe Display Format* Fenster. Damit kann die Darstellung des Anzeigeformates der gewählten Ressource zwischen Integer, Hexadezimal, Binär, Fließkomma, Boolean oder ASCII geändert werden



4.14.5 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 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.

4.14.6 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:

1. Mauszeiger in der Mitte der Ikone platzieren. Linke Maustaste drücken.

2. Bei gedrückter Maustaste Symbol in das *Watch window* Fenster ziehen

3. Aufgelistete Symbole mit deren Status und Wert

4. Start/Stop Monitoring

Symbol	Address	Value	Modify Value	Symbol Comment
HMS	R 2113	103034		PCD Clock with current time
DailyTimer	O 32	1		Daily Timer
Ontime	R 2115	60000		Switch on time
OFFTIME	R 2114	19000		Switch off time

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

1. Start/Stop Monitoring

2. Platzieren Sie den Mauszeiger auf dem zu editierenden Wert.
 Führen Sie einen Doppelklick mit der linken Maustaste durch und geben Sie den neuen Wert ein.

3. Download Values

Symbol	Address	Value	Modify Value	Symbol Comment
HMS	R 2113	103034		PCD Clock with current time
DailyTimer	O 32	1		Daily Timer
Ontime		60000	43000	Switch on time
OFFTIME		19000		Switch off time

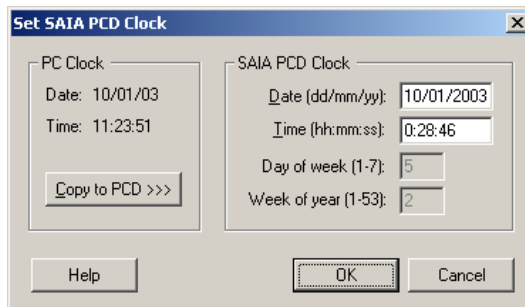
4.14.7 Echtzeituhr der PCD einstellen

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*.



4.15 FBox Einstellfenster

Verschiedene FBoxen enthalten Parametrierdaten welche über ein Dialogfenster modifizierbar sind. Solche FBoxen werden mit einem kleinen schwarzen Dreieck in der unteren linken Ecke gekennzeichnet. Die Parametrierdaten werden häufig in den FBoxen der HLK Bibliothek verwendet.

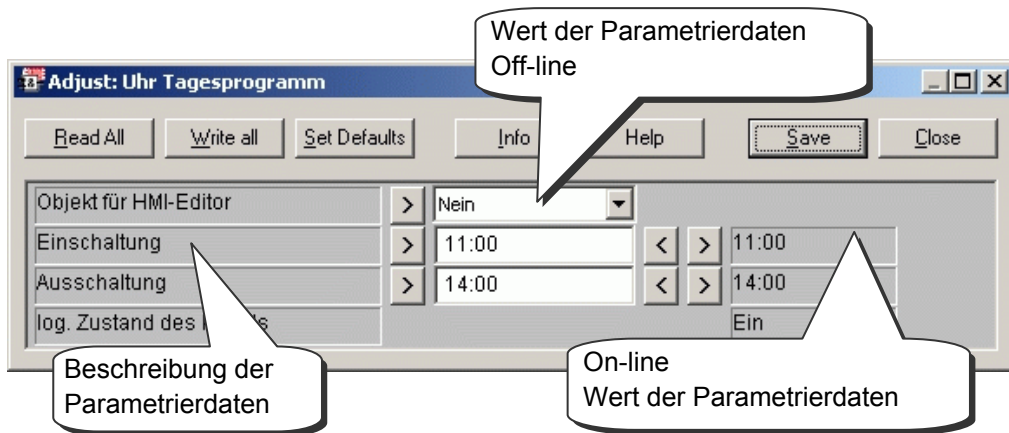


Fbox:

HLK Uhren, Uhr Tagesprogramm

Die Parametrierdaten werden z.B. benötigt um Eingangssparameter der FBox zu definieren. Im Gegensatz zu den Eingangssymbolen welche ausserhalb an die FBox angeschlossen werden bieten die Parametrierdaten aber folgende Vorteile: Beschreibung zu jedem Parameter, On-line Korrektur der Parameterwerte möglich, einfache Handhabung....

Doppelklick mit der rechten Maustaste auf die FBox öffnet das Dialogfenster.



Das Dialogfenster der Parametrierdaten ist in drei Spalten aufgeteilt:

Linke Spalte mit der Beschreibung der Parametrierdaten, welche die Funktionalität des Parameters beschreibt. Falls die Beschreibung mit drei Punkten endet, so bedeutet dies, dass noch Zusatzinformationen verfügbar sind. Ein Doppelklick mit der rechten Maustaste auf die Beschreibung der Parametrierdaten öffnet diese Zusatzinformationen.



Off-line modus

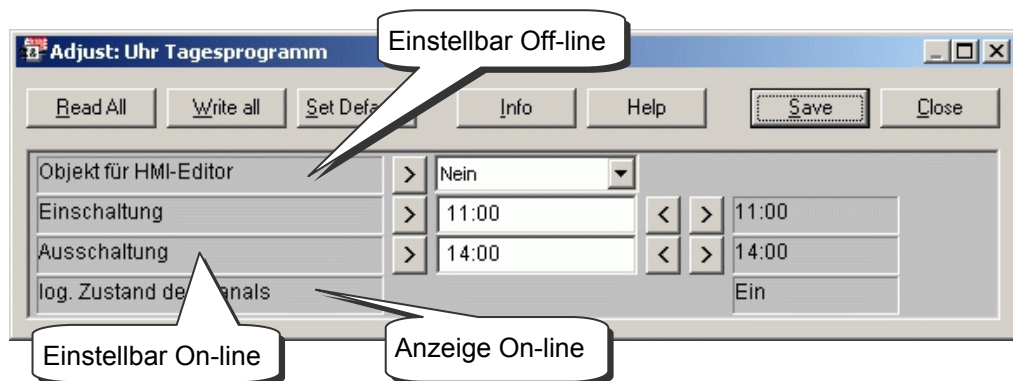
Mittlere Spalte mit dem Off-line Parametrierwert. Dieser Wert wird bei der Programmerstellung definiert und wird auch in der Fupla Datei gespeichert. Generell können diese Parametrierwerte als Eingabefeld, als Auswahlliste oder teilweise als Knopf dargestellt sein. Bei einigen FBoxen können diese Werte auch on-line geändert werden.



On-line modus

Rechte Spalte mit dem On-line Parametrierwert. Diese Spalte zeigt den aktuellen, auf der PCD vorhandenen Status der Parametrierdaten an. Dieser Wert wird von der PCD verwendet, wenn sich diese im Run Modus befindet.

4.15.1 Typen der Parametrierdaten



Die Parametrierdaten können in drei Gruppen unterteilt werden.

Parametrierdaten welche Off-line einstellbar sind.

Um auf der PCD wirksam zu sein muss nach jeder Modifikation eines solchen Typs das Programm neu verarbeitet (*Build All*) und in die PCD geladen (*Download Program*) werden.

Parametrierdaten welche On-line einstellbar sind.

Nach der Modifikation eines solchen Typs ist kein neues verarbeiten (build) und laden des Programms in die PCD nötig. Der neue Parametrierwert wird direkt in der PCD modifiziert.

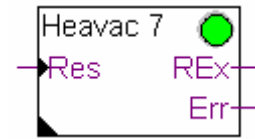
Parametrierdaten zur On-line Anzeige.

Dieser Typ kann nicht modifiziert werden und dienen zur Anzeige von Informationen über den Programmablauf innerhalb der Fbox.

4.15.2 Initialisierung der HLK FBoxen

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 dem 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.



Fbox: CVC Init, Initialization CVC

Die Parameter übernehmen die Standardwerte	
Manuelles Reset ...	Vor-/Reset
Automatisches Reset ...	Automatisch
Grund des Auto-Reset	
Reset-Eingang	Beim Aufstart

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 dreifarbig 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.

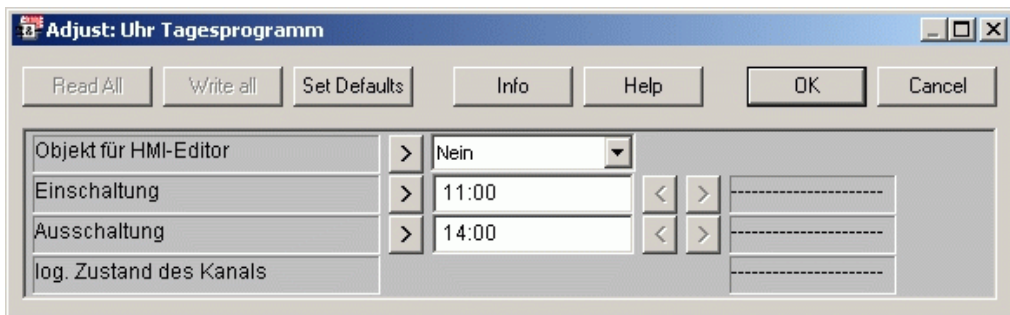
4.15.3 FBoxen mit Parametrierdaten

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 Parametrierdaten 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.



4.15.4 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.



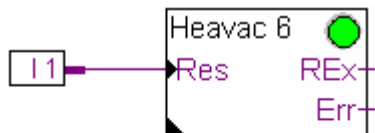
Rebuild All



Download Program



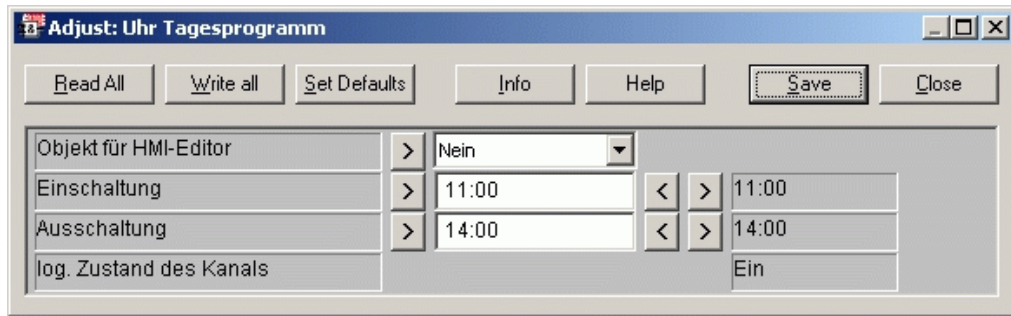
Go Online



Bemerkung:

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

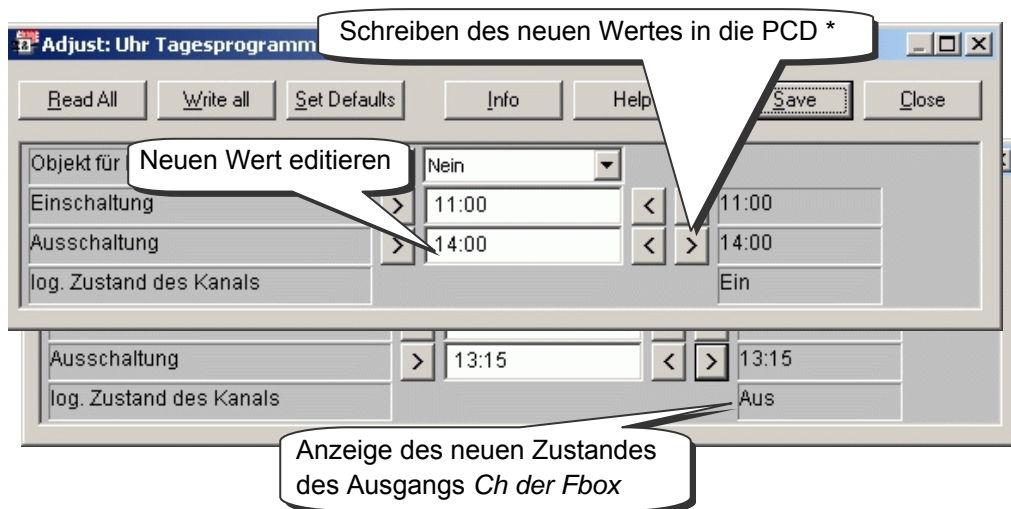
4.15.5 Parametrierdaten nach dem Laden des Anwenderprogramms



In der mittleren Spalte sind die Ein- und Ausschaltzeiten der Zeitschaltuhr dargestellt welche bei der Programmerstellung definiert worden sind (Off-line Parametrierwerte). Generell ¹⁾, entsprechen die Ein- und Ausschaltzeiten sowie der Kanal Status (*Switch on, Switch off, Channel state*) den Werten welche in der PCD verwendet werden und in der rechten Kolonne dargestellt sind.

4.15.6 On-line Überschreiben der Parametrierdaten.

Beim On-line Test des Programms können die Parametrierdaten der Ein- und Ausschaltzeiten direkt modifiziert werden:



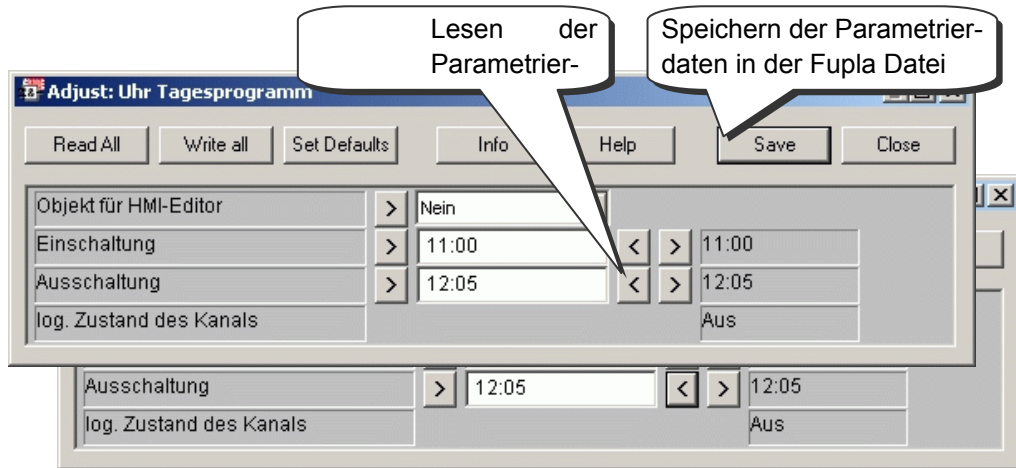
Mit dem Knopf *Write all* werden alle Parametrierdatenpunkte (Off-line Parametrierwerte) zusammen in die PCD geschrieben.

Sollen die neu editieren Werte für das nächste *Build All* verwendet werden, so muss beim schliessen des Fensters der Knopf *Save* betätigt werden. Sollen die neu editieren Werte nicht für das nächste *Build All* verwendet werden so muss das Fenster mit *Close* verlassen werden.

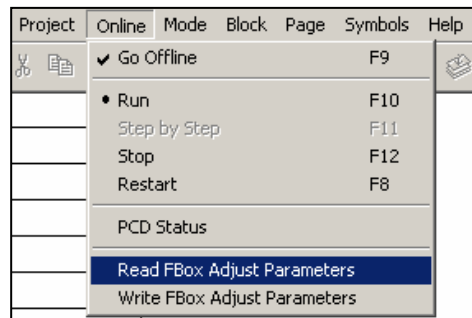
1) Die Werte können auch unterschiedlich sein. Je nach eingestellter Option in den Parametrierdaten der *Heavac 7* und dem Zustand des Einganges *Res* beim einschalten der PCD.

4.15.7 Lesen der On-line Parametrierdaten

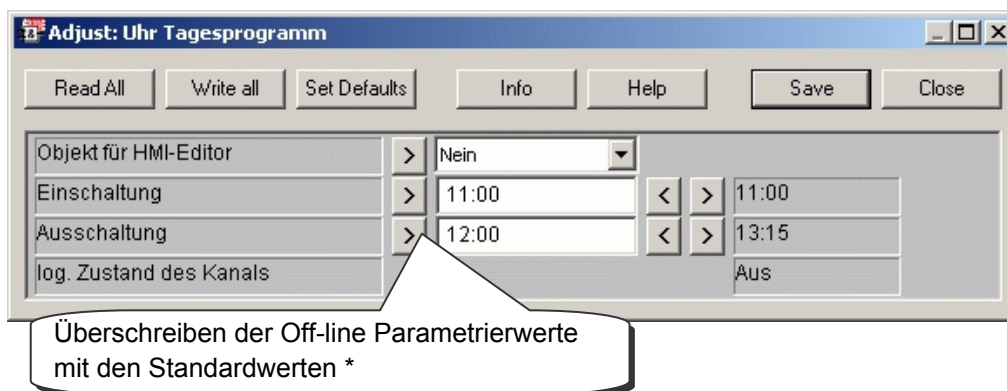
Es kann vorkommen, dass der Anwender die in der PCD verwendeten on-line Parametrierdaten für das nächste *Build All* lesen und speichern will. Untenstehend wird aufgezeigt wie die in der PCD verwendeten Parametrierdaten von der PCD in die Fupla Datei übertragen werden.



- * Mit dem Knopf *Read all* werden alle Parametrierdatenpunkte (On-line Parametrierwerte) gemeinsam aus der PCD gelesen.
- * Alle Parametrierdatenpunkte aller Fupla FBoxen können gemeinsam aus der PCD gelesen werden indem der Menüpunkt *Online, Read Fbox Adjust parameter* ausgewählt wird



4.15.8 Parametrierdaten mit Standardwerten beschreiben



- * Mit dem Knopf *Set Defaults* werden alle Parametrierdatenpunkte (Off-line Parametrierwerte) mit den dazugehörigen Standardwerten überschrieben.

4.15.9 Definition von Symbolnamen für die Referenzierung von FBoxen

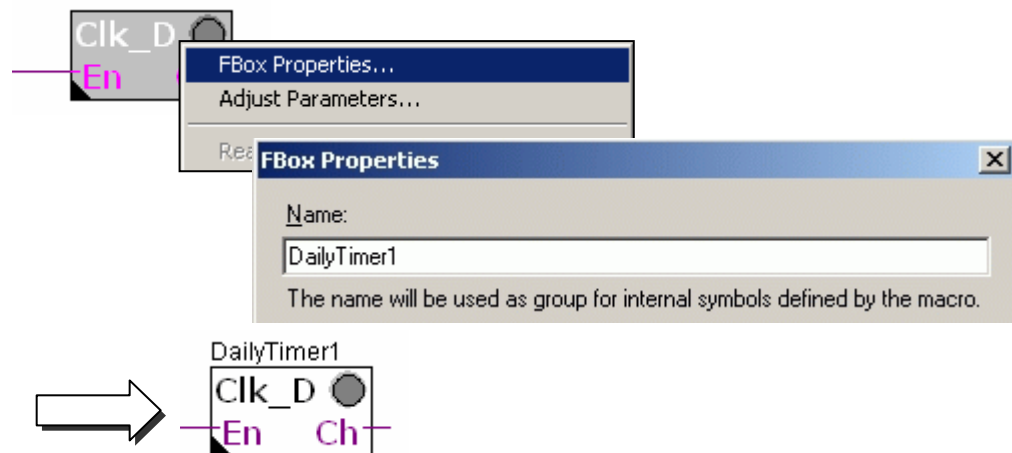
Teilweise ist es erforderlich, dass Parametrierwerte der FBoxen im Fupla Programm, von SCADA Systemen oder anderen Geräten gelesen oder geschrieben werden müssen.

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.



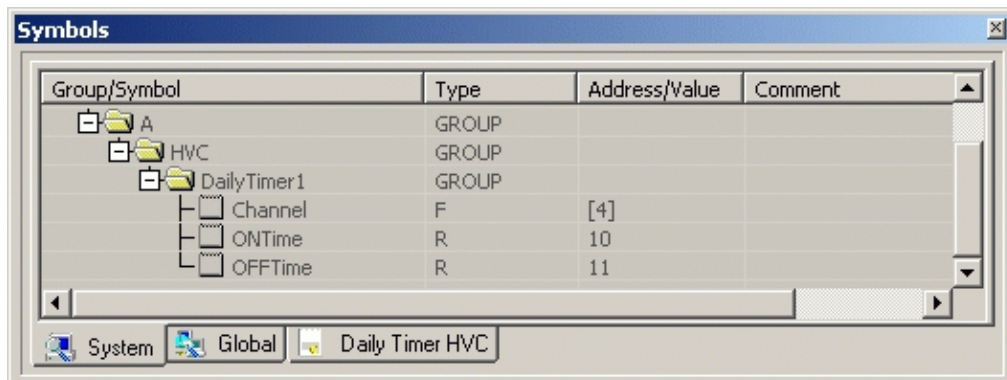
Rebuild All

Öffnen des Symboleditors, nachdem das Programm verarbeitet (build) wurde. Eine neue Registrierkarte *System* ist nun ersichtlich. In dieser Registrierkarte sind die Systemsymbole der PCD enthalten

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.



Show/Hide Symbol Editor



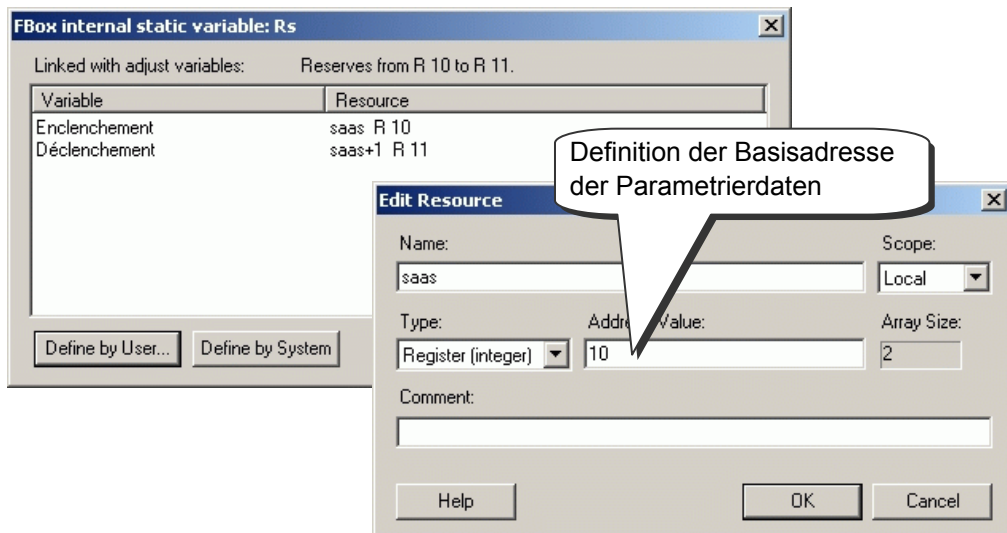
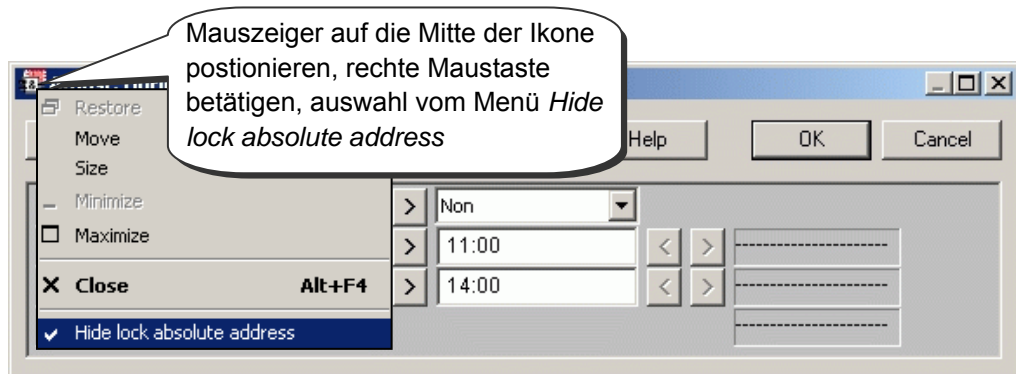
A.HVC.DailyTimer.ONTime

A.HVC.DailyTimer.OFFTime

Nun müssen nur noch diese Systemsymbole in Fupla verwendet werden.

4.15.10 Definition der absoluten Adressen für die Parametrierdaten

Definition eines Symbolnamens für die FBox wie zuvor aufgezeigt und Zuordnung einer absoluten Adresse wie folgt:

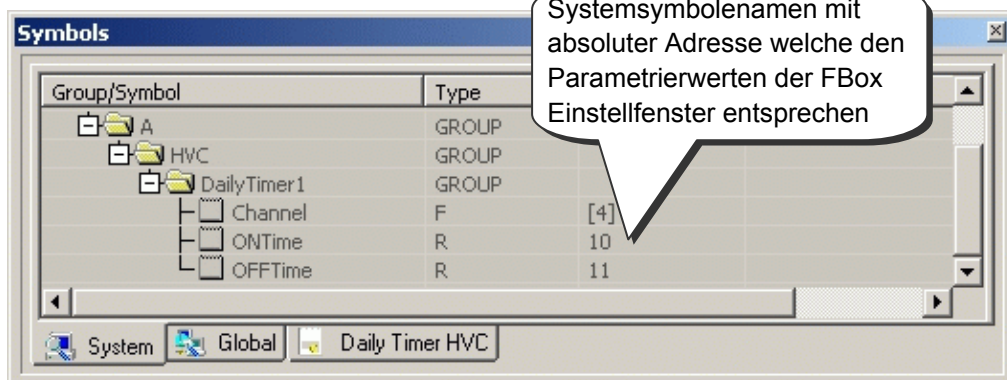


Rebuild All

Öffnen des Symboleditors, nachdem das Programm verarbeitet (build) wurde. In der Registrierkarte *System* sind die Systemsymbole nun mit den zuvor definierten absoluten Adressen versehen.



Show/Hide Symbol Editor



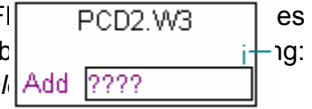
4.16 Inbetriebnahme eines Analogmoduls

4.16.6 Erfassen einer Analogmessung

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



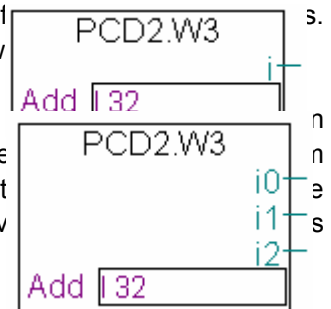
Mit analogen Ein-/oder Ausgangsmodulen muss eine FBox des Analogwertes benutzt werden. Diese FBoxen stehen mit Bibliothek *Standard, Analogmodulen, Applikationen und HEAVAC-Analog*



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.

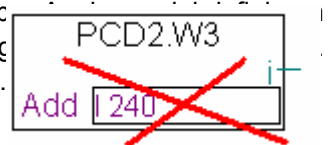
Das Feld *Add* ermöglicht die Basisadresse des zu definierenden Moduls. Diese Adresse gibt an, wo das Modul im PCD eingesetzt wird.



Analogmessungen sind in den FBox-Eingängen i 0 bis i 2 direkt mit anderen FBoxen verbunden werden oder die Register gespeichert werden. Das Speichern eines Wertes ist eine gute Lösung, vor allem, wenn der Wert auf vielen verschiedenen Programmen benutzt wird.

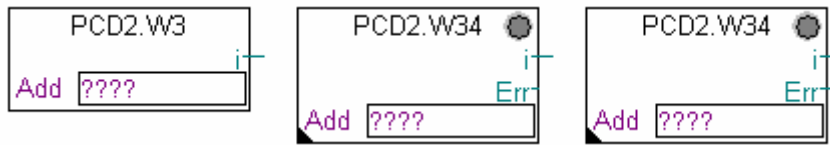
Achtung:

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



4.16.7 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.



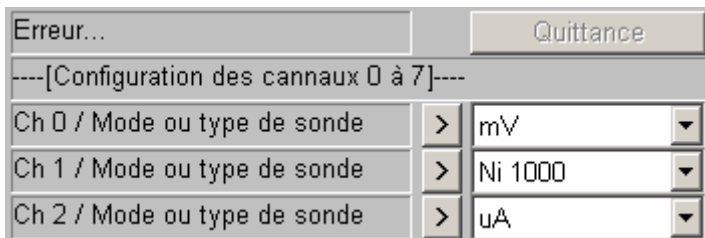
FBoxen: PCD2.W3, PCD2.W34, PCD2.W34 with error

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.



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 Störfalle.



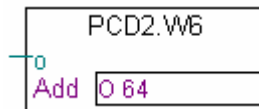
4.16.8 Beispiel für PCD2.W610 Analog-Ausgangsmodule

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.

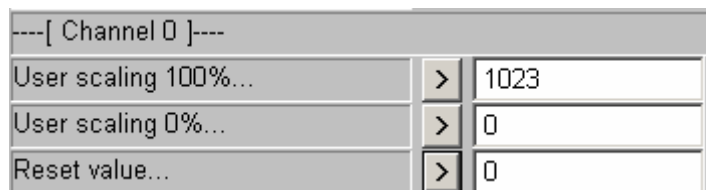


FBox: PCD2.W6

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

5	PROGRAMMSTRUKTUREN	2
5.1	Einleitung	2
5.2	Cyclic Organization Block (COB 0 bis 15)	3
5.2.1	Definition	3
5.2.2	Beispiel	4
5.2.3	Programmblöcke hinzuaddiert	4
5.2.4	Überwachungszeit	5
5.3	Program Block (PB 0 bis 299)	6
5.3.1	Definition	6
5.3.2	Beispiel	6
5.4	Function Blocks (FB 0 bis 999)	8
5.4.1	Definition	8
5.4.2	Aufrufbeispiel eines FBs	8
5.5	Ansicht der Block-Struktur	9
5.6	Exception Block (XOB)	10
5.6.1	Definition	10
5.6.2	Alle XOBs der PCD-Familie in einer Kurzübersicht	11
5.6.3	Anwendung der XOBs	12
5.6.4	History Table	15
5.6.5	Beschreibung der XOBs	16
5.7	Sequential Blocks (SB 0 , 96)	20
5.7.6	Zusammenfassung	20

5 Programmstrukturen

5.1 Einleitung

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).

5.2 Cyclic Organization Block (COB 0 bis 15)

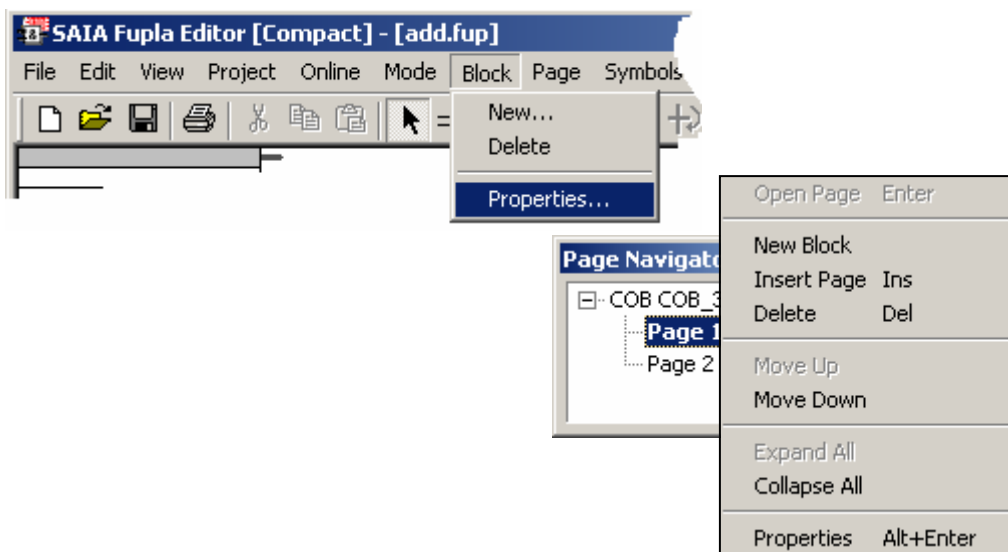
5.2.1 Definition

Cyclic Organization Blocks COB sind Bestandteile des Programms, die ohne Berücksichtigung von Kontrollstrukturen – wie z.B. das Warten auf interne oder externe Events – ausgeführt werden. Nach erfolgreichem Startup der PCD wird immer zuerst der COB 0 ausgeführt und nachfolgend – falls im Programm vorhanden – dann die weiteren COBs 1-15 in aufsteigender Reihenfolge. Der zyklische Aufruf der COBs erfolgt hierbei automatisch und bildet die fortlaufende Programmschleife.

Alle Signalabfragen – von z.B. von Motorendschaltern oder Initiatoren – die den Programmfluss beeinflussen, müssen sich innerhalb eines COBs befinden.

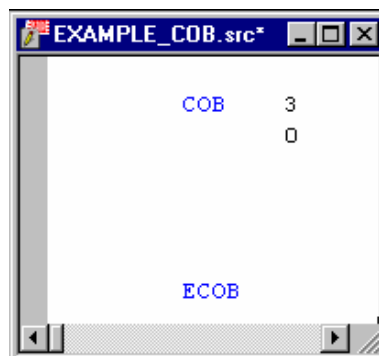
Innerhalb des SPS-Programms muss mindestens ein COB vorhanden sein !

Das richtige Verständnis des Cyclic Organization Block Konzepts ist wichtig. Um z.B. das regelmässige Ueberwachen von wichtigen Applikationssignalen zu garantieren, sollten Warteschleifen innerhalb des Programmes auf ein Minimum begrenzt, besser vermieden, werden.



Bei der Verwendung des Fupla Editors wird das Programm innerhalb eines automatisch generierten COBs geöffnet, wobei sich z.B. Block-Typ und Kommentare über das *Block*-Pulldown-Menü: *Properties* ändern lassen.

Bei der Programmierung in Anweisungsliste (kurz AWL, aber in diesem Workshop wird der englische Ausdruck IL, also InstructionList verwendet) ist das anders. Hier wird der COB seitens des Anwenders durch entsprechende Befehle erzeugt, die die eigentlichen Programmanweisungen einschliessen.



5.2.2 Beispiel

Nachfolgend ist ein Beispiel-Programm in sowohl IL als auch FUPLA abgebildet, das 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.

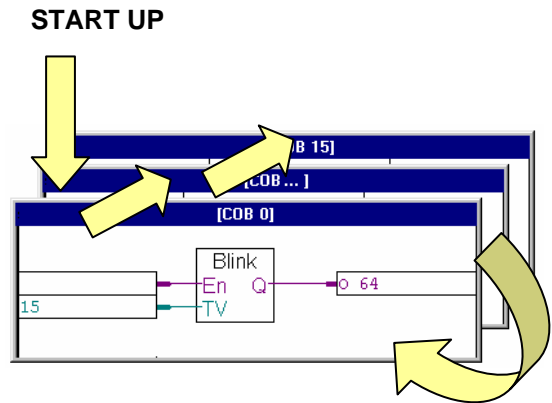
IL Programm

```

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
    
```

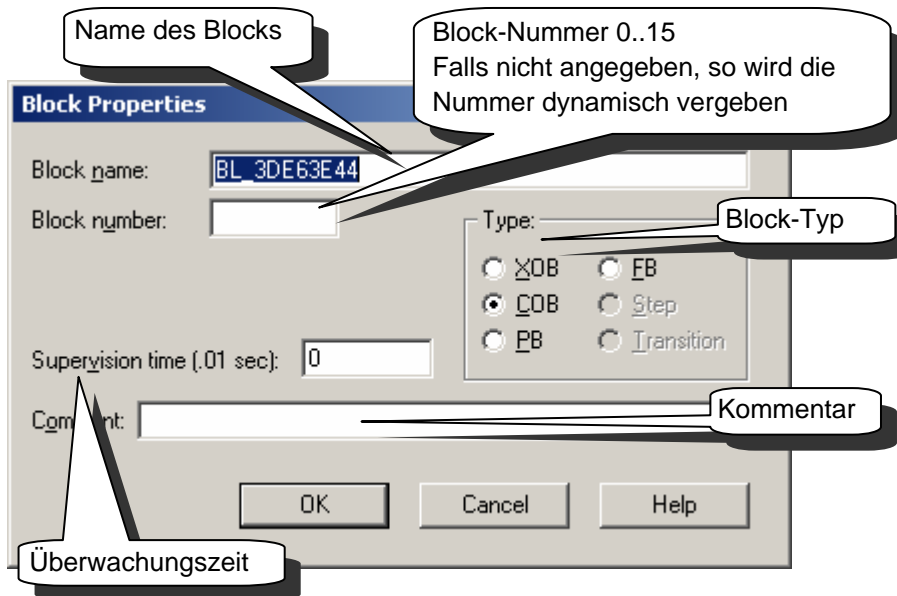
Fupla Programm



Fbox: *Blinker, Blinker symmetrisch*

5.2.3 Programmblöcke hinzuaddiert

Ein Fupla Programm kann mehrere Programmblöcke enthalten, die über das *Block-Pulldown-Menü: Properties* hinzuaddiert, gelöscht oder editiert werden können.



5.2.4 Überwachungszeit

Die Überwachungszeit definiert hierbei die Maximalzeit, die bei der Ausführung eines COBs vom Start bis Ende zulässig ist. Erfolgt z.B. eine Zeitüberschreitung während der Ausführung von COB X, so sind zwei Reaktionen möglich:

Ist XOB 11 nicht definiert, so wird COB X sofort verlassen und die Error-Led eingeschaltet. Nach der regulären Ausführung der nachfolgend definierten COBs erfolgt dann der Rücksprung nach COB X an die Stelle, die vorher aufgrund der Zeitüberschreitung verlassen wurde. Die weitere Ausführung des COBs wird dann erneut zeitlich überwacht.

Ist XOB 11 hingegen definiert, so wird dieser nach dem sofortigen Verlassen von COB X ausgeführt. Danach erfolgt auch hier – wie bereits oben spezifiziert – die Fortsetzung von COB X. Da in diesem Fall der Fehler über XOB 11 behandelt wurde, wird die Error-Led nicht eingeschaltet.

Wird die Überwachungszeit auf Null gesetzt, so ist die Funktion der Zeitüberwachung deaktiviert.

5.3 Program Block (PB 0 bis 299)

5.3.1 Definition

Auch Program Blocks (PBs) können für die hierarchische Organisation des SPS-Programms verwendet werden. Die Abarbeitung erfolgt hierbei nur nach Aufruf aus einem COB, PB, FB oder SB (Sequential Block).

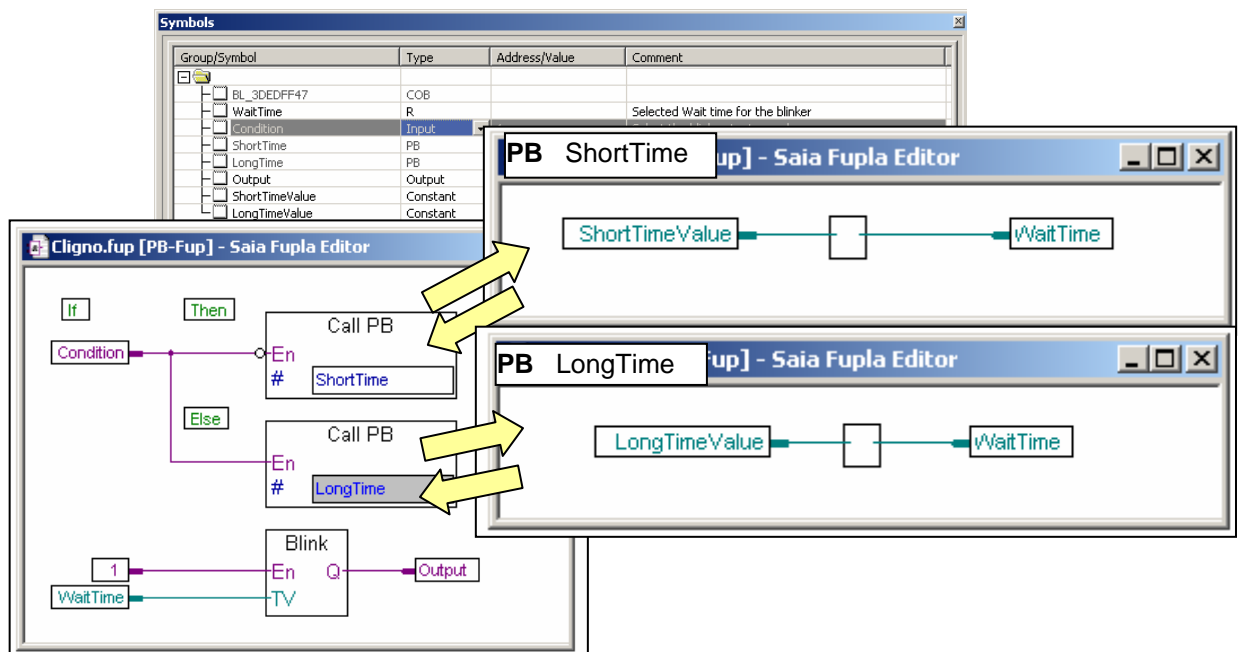
Der PB-Aufruf kann über einen Conditional oder Unconditional Call erfolgen. Während der erstgenannte immer ausgeführt wird, hängt die Ausführung des Conditional Calls vom Ergebnis einer zuvor erfolgten logischen Operation ab.

PBs lassen sich mehrfach aus einem Programm aufrufen, wobei auch Aufrufe aus einem PB bis zu einer Verschachtelungstiefe von 7 möglich sind. Bei höheren Verschachtelungen erfolgt die Fehlerbehandlung über XOB 10.

5.3.2 Beispiel

Nachfolgend ist ein Beispiel-Programm in FUPLA abgebildet, das anhand eines Eingangswertes die Blinkgeschwindigkeit eines Ausgangs verändert.

Fupla Programm:



Ist darin 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 Initialisierung von *WaitTime* nach einem Kaltstart.

IL Programm:

The screenshot shows the SAIA IL Editor interface. At the top, a menu bar includes File, Edit, Search, View, Project, Online, Tools, Symbols, Window, and Help. Below the menu is a 'Symbols' table with the following data:

Group/Symbol	Type	Address/Value	Comment
ShortTimeValue	Constant	5	0.5 s
LongTimeValue	Constant	15	1.5 s
Tempo	Timer		

Below the symbols table, the main editor area displays the following IL code:

```

COB      0
          0
          Condition           ;IF (Condition is high)
          CPB      L ShortTime ; THEN Call PB ShortTime
          CPB      H LongTime  ; ELSE Call PB LongTime
          ECOB

          PB      ShortTime
          STL     Tempo         ;IF tempo is low
          LD      Tempo         ; load it with a short value
          COM     ShortTimeValue
          EPB     Output        ; invert the output

          PB      LongTime
          STL     Tempo         ;IF tempo is low
          LD      Tempo         ; load it with a long value
          COM     LongTimeValue
          EPB     Output        ; invert the output
    
```

5.4 Function Blocks (FB 0 bis 999)

5.4.1 Definition

Function blocks sind den PBs ähnlich, da auch sie Programmstücke enthalten, die von anderen Blöcken aufgerufen werden. Dieser Aufruf kann gleichfalls Conditional oder Unconditional sein. Der einzige Unterschied zum PB ist, dass hier beim Aufruf ein zusätzlicher Parametersatz mit übergeben werden kann.

FBs bieten somit eine ideale Möglichkeit, Programm-Bibliotheken zu erstellen, die für verschiedene Projekte nutzbar sind. Hierbei ist jedoch zu beachten, dass die Parameterübergabe an FBs nur in IL möglich ist.

FBs lassen sich mehrfach aus einem Programm aufrufen, wobei auch Aufrufe aus einem FB bis zu einer Verschachtelungstiefe von 7 möglich sind. Bei höheren Verschachtelungen erfolgt die Fehlerbehandlung über XOB 10.

5.4.2 Aufrufbeispiel eines FBs

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 DEF  = 1          ; [T]  Adresse Timer
delay  DEF  = 2          ; [W]  Adresse Pausenzeit zwischen 2 Blinkzyklen
blinker DEF  = 3          ; [O]  Adresse Blinker

          STL      = tempo          ; Wenn Timer tempo abgelaufen ist (= low)
          LDL      = tempo          ; Dann lade Timer tempo mit Pausenzeit delay
          = delay
          COM      = blinker        ; Und invertiere den Ausgang Blinker
          EFB

          COB      0
          0

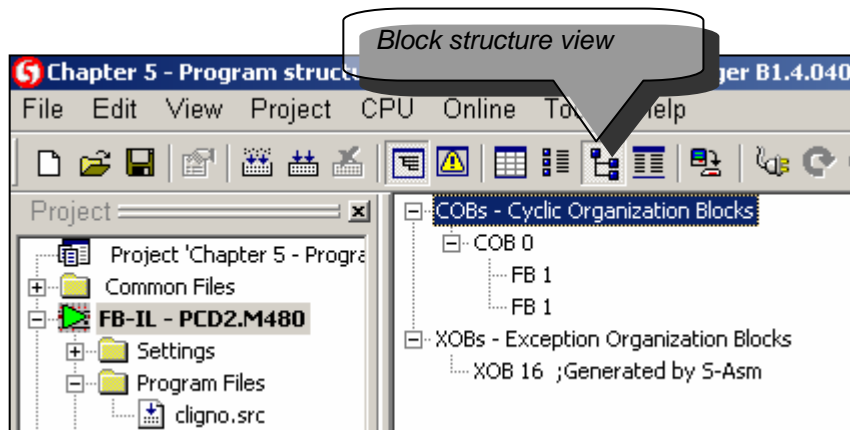
          CFB      1          ; Erster Aufruf des FB 1
          T 1
          15
          O 64

          CFB      1          ; Zweiter Aufruf des FB 1
          T 2
          30
          O 65
          ECOB

```


5.5 Ansicht der Block-Struktur

Die Struktur eines Programms lässt sich mit Hilfe des „Block-Struktur Ansicht“- Icons in der 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.



5.6 Exception Block (XOB)

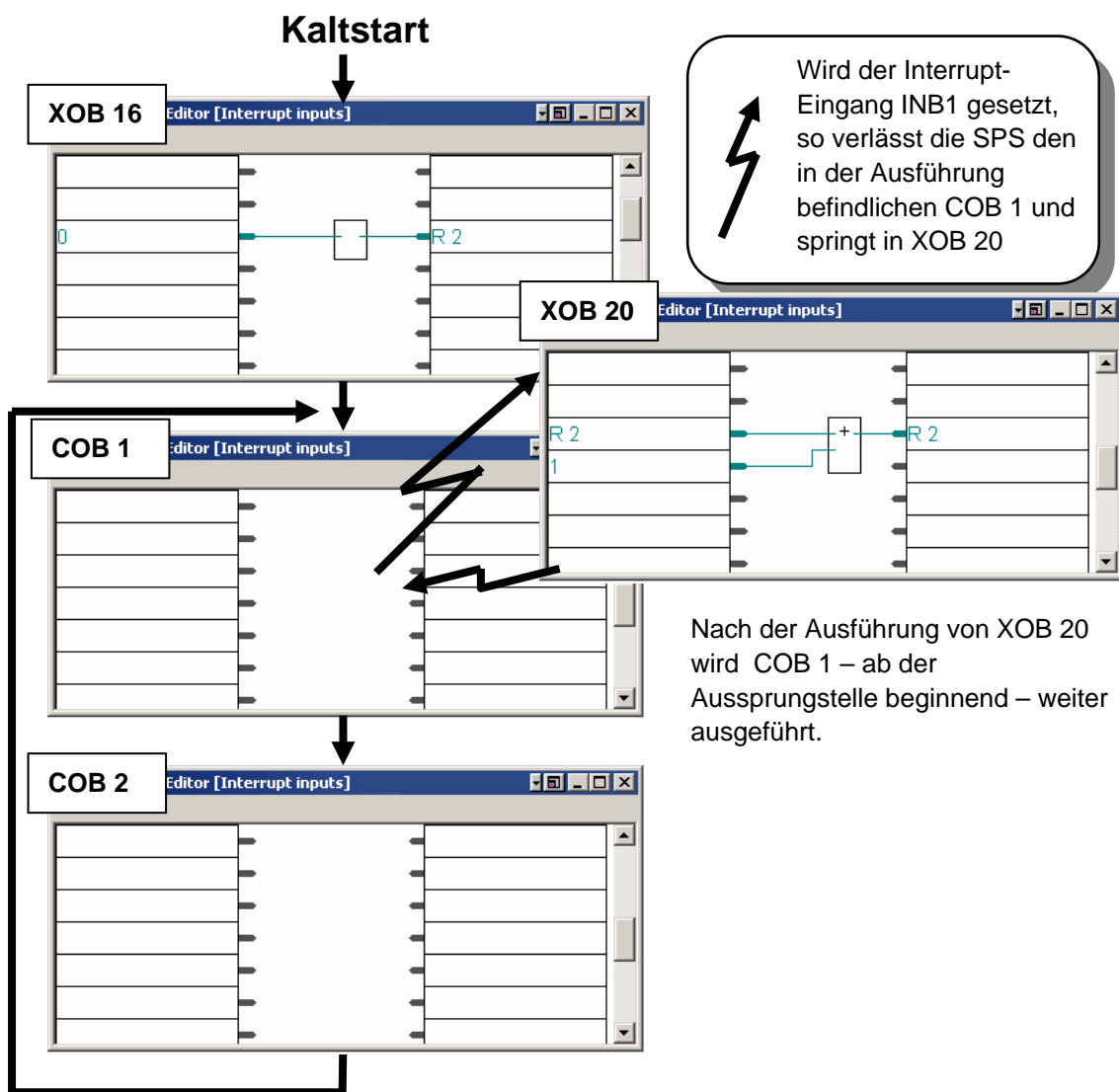
5.6.1 Definition

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.



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.

5.6.2 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
25	Interrupt via INB2	3
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.

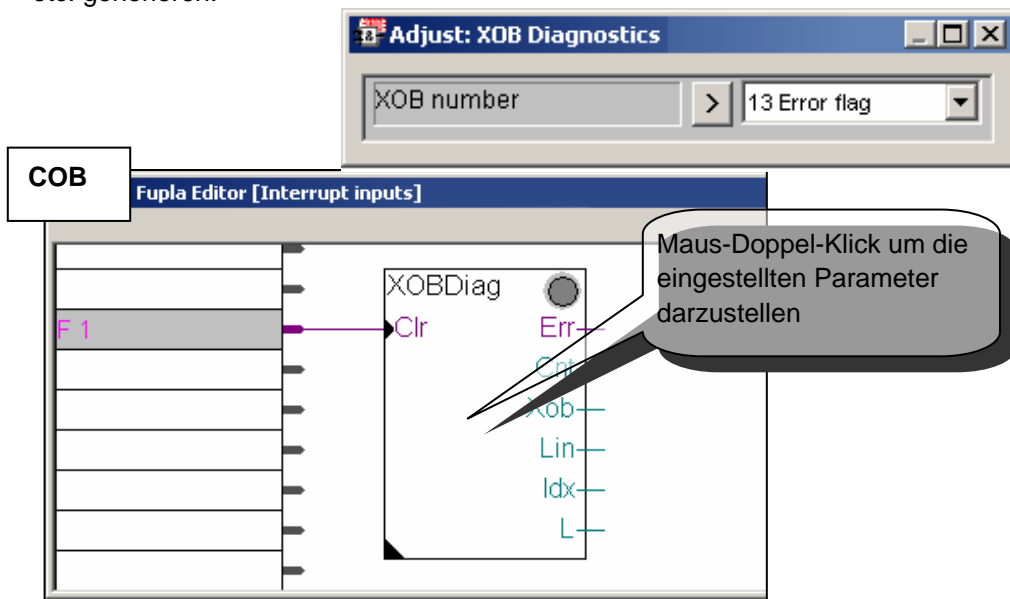
5.6.3 Anwendung der XOBs

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

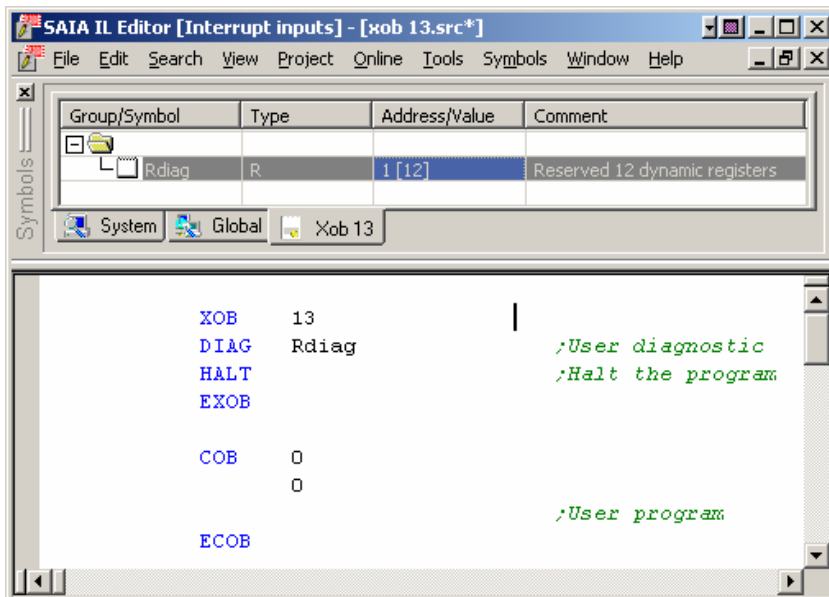
Beispiel in Fupla:

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.

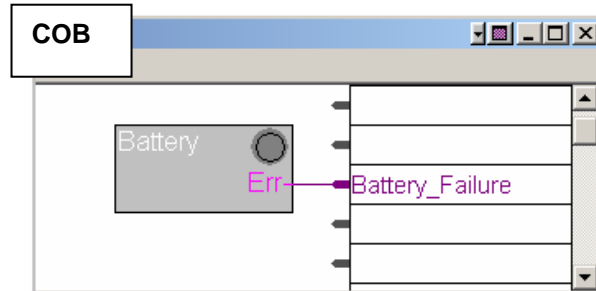


Überwachung der SPS:

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.

Group/Symbol	Type	Address/Value	Comment
Battery_Failure	Timer	1	

```

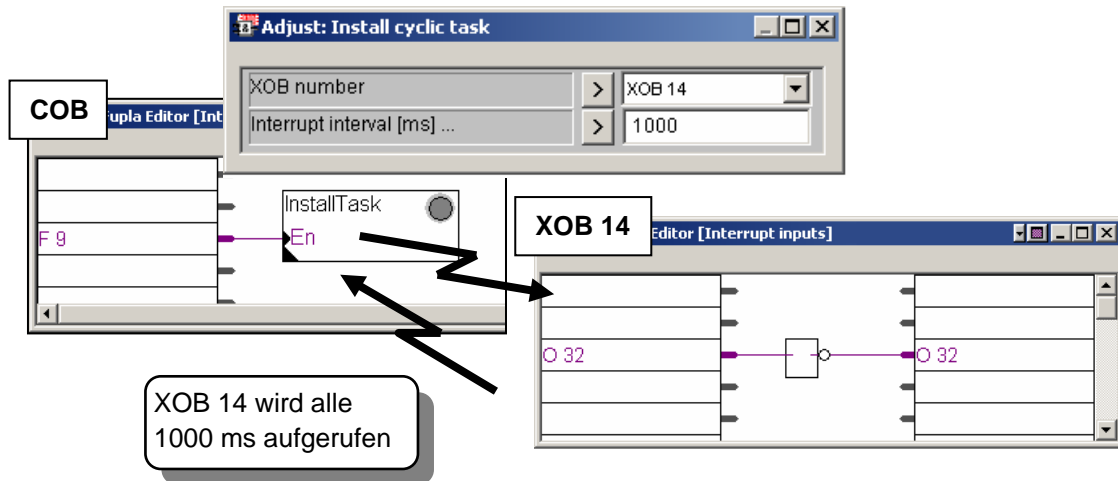
XOB 2 ;Alarm Battery
LD Battery_Failure
k 10
EXOB
COB 0 ;User program
0
STH Battery_Failure
ECOB
    
```

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 Funktionen*, *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
  
```

5.6.4 History Table

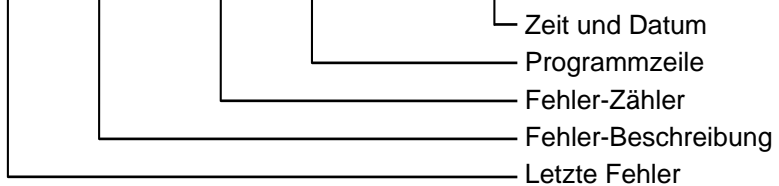
Die Funktion *PCD History Table* listet alle Hardware- und Software-Fehler auf, die vorher aufgetreten sind. Auch wenn die XOBs nicht programmiert sind, läuft diese Funktion ständig im Hintergrund mit.



Online Configurator

Der Aufruf der History Table erfolgt über die *Online Configurator* Schaltfläche oder über das Menü *Tool, Online Configurator*

Reason	Address	Time	Date
>7 CALL LEVELS	30	14:09:43	06/01/2003
>7 CALL LEVELS	30	14:09:43	06/01/2003
>7 CALL LEVELS	30	14:09:43	06/01/2003
>7 CALL LEVELS	30	14:09:43	06/01/2003
>7 CALL LEVELS	30	14:09:44	06/01/2003
>> >7 CALL LEVELS	30	14:09:44	06/01/2003
>7 CALL LEVELS	30	14:09:44	06/01/2003
>7 CALL LEVELS	30	14:09:44	06/01/2003
>7 CALL LEVELS	30	14:09:44	06/01/2003
>7 CALL LEVELS	30	14:09:44	06/01/2003
>7 CALL LEVELS	30	14:09:44	06/01/2003
>7 CALL LEVELS	30	14:09:44	06/01/2003
>7 CALL LEVELS	30	14:09:44	06/01/2003
BATT FAIL	816 0	14:09:43	06/01/2003
IR OVERFLOW	0 0	12:00:00	06/01/2003
ERROR FLAG	772 6	14:09:44	06/01/2003



Beachte:

- Jede CPU hat ihre eigene Fehler-Historie
- Die Zeile *BATT FAIL* existiert nur bei CPU 0.
- Wenn der Fehler einer Programmzeile zugeordnet werden kann, so wird die Zeilennummer angezeigt. Anderenfalls wird die Fehler-Beschreibung in hexadezimal ausgegeben.
- XOB 0 wird nur aufgerufen, sofern er auch programmiert ist

5.6.5 Beschreibung der XOBs

XOB 0: Spannungsversorgungs-Fehler auf der Hauptplatine

Der Spannungsmonitor des Versorgungsmoduls hat einen schwerwiegenden Spannungseinbruch auf der Hauptplatine detektiert.

Darauffin werden alle Ausgänge wie folgt auf Low gesetzt:

- bei einer PCD4: sofort
- bei einer PCD6: nach 1,5 ms.

Anschliessend wird XOB 0 gestartet und ca. 5 ms später sämtliche CPUs in den HALT-Zustand gebracht. Die Zeitspanne kann zur Datensicherung genutzt werden.

XOB 1: Spannungsversorgungs-Fehler im Erweiterungsgehäuse (PCD6)

Der Spannungsmonitor des Versorgungsmoduls hat einen schwerwiegenden Spannungseinbruch im Erweiterungsgehäuse (PCD 2 oder PCD6) detektiert.

In diesem Fall werden alle Ausgänge des Erweiterungsmoduls innerhalb von 2 ms auf Low gesetzt und XOB 1 ausgeführt. Wenn die Ausgänge eines „toten“ Erweiterungsmoduls von irgendeiner CPU noch weiter gelesen und/oder geschrieben werden, so wird der XOB 4 und/oder XOB 5 gleichfalls 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 einem Buserweiterungsmodul 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 dem Modul 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.

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 Programmieretechniken, wie z.B. „Timeslice“ werden im Kapitel <Andere Programmieretechniken> 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. der Merker 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, Fehlermeldung auf einen Drucker usw.) getroffen werden können. Dieser XOB 13 wird also immer aufgerufen, wenn das ERROR-Flag gesetzt wird, egal ob die Ursache ein Rechenfehler, ein Datentransferfehler oder ein Kommunikationsfehler ist. Soll die Diagnose bezüglich des ERROR-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 und müssen, 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 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*.

XOB20 & XOB 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 PCD2 Hardware-Manual.

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.

5.7 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 exzellentes Tool zur Erstellung von sequentiellen Programmen und wird näher im nächsten Kapitel beschrieben.

5.7.6 Zusammenfassung

Service	Media	Operand	Bemerkung
Cyclic Organization Block	COB	0...15	Mindestens ein COB pro Programm
Program Block	PB	0...299	Unterprogramm, aufrufbar aus einem COB, PB,FB,SB oder XOB
Function Block	FB	0...999	Parametrierbares Unterprogramm aufrufbar aus einem COB, PB, FB, SB oder XOB
Sequential Block	SB	0...32 0...96 ¹	Sequentielles Unterprogramm, aufrufbar aus einem COB, PB oder FB (bedingt auch SB und XOB)
Step	ST	0...1999 0...5999 ¹	
Transition	TR	0...1999 0...5999 ¹	

¹ PCD2.M170/480, PCD4. M170 und PCD3

Inhaltsverzeichnis

6	PROGRAMMIEREN IN GRAFTEC	3
6.1	Einleitung	3
6.2	Sequentielle Blöcke (SB 0 to 31, 96¹)	4
6.3	Zyklische Blöcke	5
6.3.1	Ablauf der zyklischen Programme	5
6.3.2	Zykluszeit	5
6.4	Erstellen einer neuen Graftec-Datei	6
6.4.1	Eröffnen eines neuen Projekts	6
6.4.2	Neue Fupla- oder IL-Datei eröffnen	6
6.4.3	Aufruf eines SB innerhalb eines COB	7
6.4.4	Neue Graftec-Datei eröffnen	7
6.5	Die Organisation des SB	8
6.5.1	<i>Block Navigator</i>	8
6.5.2	Generelle Struktur eines SB	9
6.5.3	Grundregel für die Reihenfolge	9
6.5.4	Transitionen (TR 0 to 1999 ¹)	10
6.5.5	Steps (ST 0 to 1999 ¹)	11
6.6	Typische SB-Strukturen	12
6.6.1	Einfache Sequenz (Schrittfolge)	12
6.6.2	Alternativ-Verzweigung (ODER)	12
6.6.3	Simultan-Verzweigung (UND)	12
6.6.4	Überspringen von Sequenzen	12
6.6.5	Wiederholen einer Sequenz	12
6.7	Editieren einer Sequenz	13
6.7.6	Editieren einer einfachen Sequenz	13
6.7.7	Editieren einer Verbindung	13
6.7.8	Editieren einer Alternativ-Verzweigung (ODER)	14
6.7.9	Schliessen von alternativen Sequenzen	14
6.7.10	Editieren von Simultan-Verzweigungen (UND)	14
6.7.11	Synchronisieren einer Simultan-Verzweigung	14
6.7.12	Zufügen eines Kommentars	14
6.7.13	Sequenz einfügen	15
6.7.14	Sequenz löschen	15
6.7.15	Kopieren/Einfügen einer Sequenz	16
6.8	Editieren des ersten sequentiellen Blocks	17
6.8.1	Öffnen einer Datei	17
6.8.2	Zeichnen der Grundstruktur	17
6.8.3	Wahl von S-Edit oder Fupla zum editieren des Programm-Code	18
6.8.4	Vorbereiten der Symbole	19
6.8.5	Editieren des Programm-Code	19
6.8.6	Das Programmieren einer Transition:	19

6.8.7	Benutzen von Timer in sequentiellen Blöcke	20
6.8.8	Warten bis die Zeit abgelaufen ist.	20
6.8.9	Wiederholen der Sequenz für die Pause des Blinkers	21
6.8.10	Dekrementieren eines Zählers	22
6.8.11	Alternative Verzweigung	22
6.9	Kompilation (build) und Debuggen eines Programms	23
6.9.1	Messages-Fenster	23
6.9.2	Online-Werkzeuge	23
6.10	Struktur eines Graftec-Programms mit Pages	24
6.10.1	Was sind Pages?	24
6.10.2	Erstellen einer Page	25

6 Programmieren in Graftec

6.1 Einleitung

Der Graftec-Editor dient zur Erstellung sequentieller Programme unter Verwendung der Fupla- bzw. IL-Sprachen. Dieses Kapitel stellt eine Einführung in die Arbeit mit Graftec zum Editieren von Programmen dar, die sequentielle Blöcke (SBs), Steps und Transitionen enthalten.

6.2 Sequentielle Blöcke (SB 0 to 31, 96 ¹⁾)

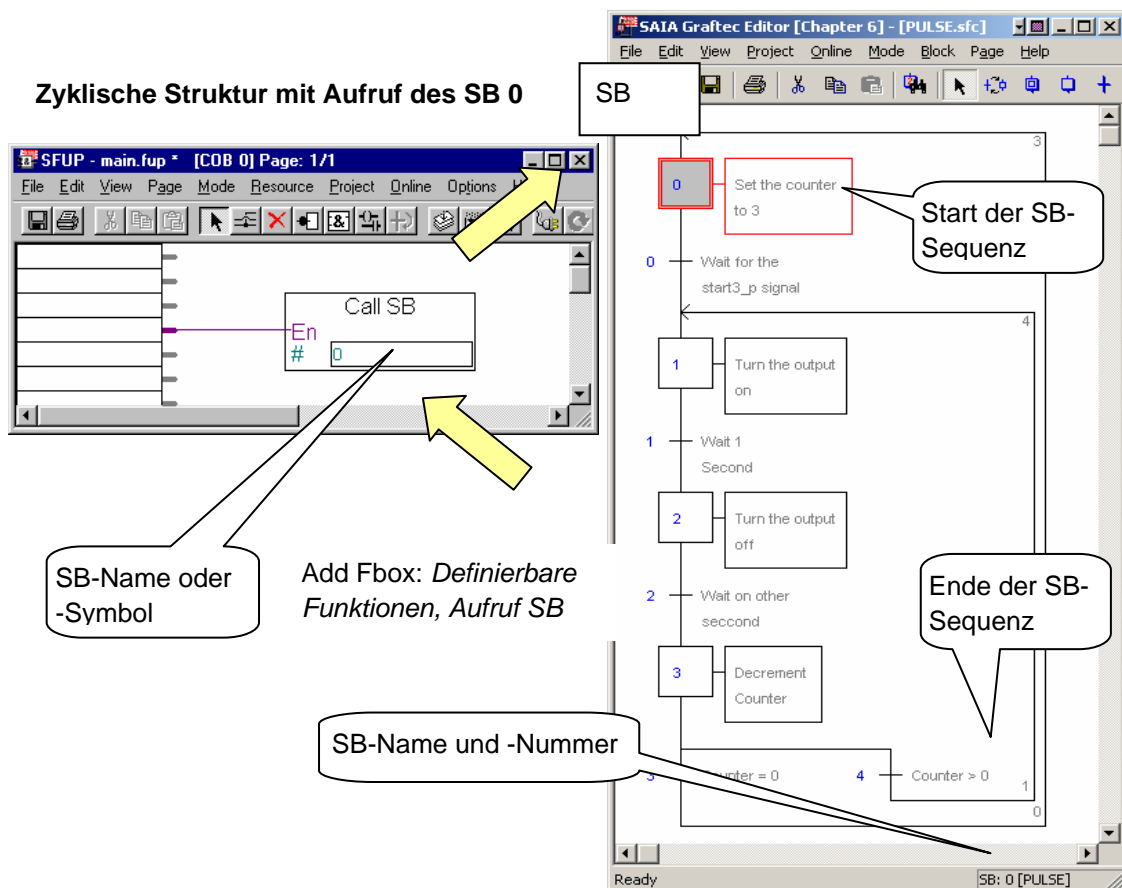
SB (Sequential Blocks) arbeiten nicht wie die bereits beschriebenen Programm- oder Funktions-Blöcke. SB eignen sich vorzüglich, wenn es darum geht, in einer Anlage Abläufe zu programmieren, welche genau definiert nacheinander aktiviert werden - intern zeitlich- oder extern ereignisgesteuert.

Zyklisch umlaufende Programme führen logische Verknüpfungen aus, SB hingegen bearbeiten vorzugsweise abwechselnde Folgen von Ereignissen und Aktionen. Es ist deshalb wichtig zu entscheiden, mit welchem Mittel eine Steueraufgabe realisiert wird.

Das Warten auf ein Ereignis, welches den weiteren Ablauf eines Prozesses bestimmt, darf in keinem Fall ein zyklisches Programm, das gleichzeitig verarbeitet wird, auf längere Dauer unterbrechen. Steht kein neues Ereignis an, so wird das zyklische Programm weiter verarbeitet. Parallel ablaufende Prozesse können in einer Struktur von 32 SB eingebettet werden, die bei jedem Durchgang des zyklischen Programms aufgerufen werden.

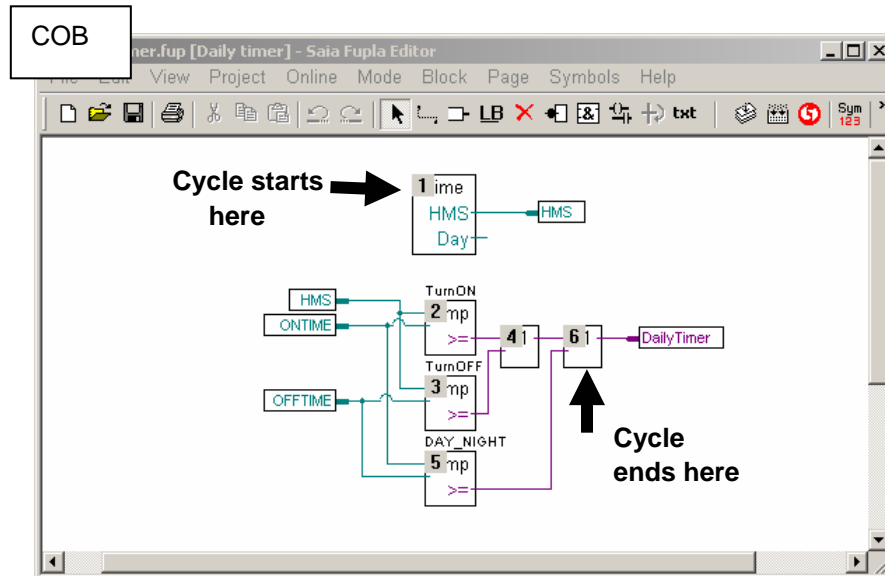
Jeder der 32 SB erlaubt die gleichzeitige Verarbeitung von mehreren Sequenzen, die zu einander zeitlich völlig unabhängig ablaufen.

Typische Graftec-Struktur



1) PCD2/4.M170, PCD2.M480, PCD3 bearbeiten bis zu 6000 ST/TR und 96 SB

6.3 Zyklische Blöcke



6.3.1 Ablauf der zyklischen Programme

Bis jetzt wurden alle Programme in einer endlosen Schleife abgearbeitet. Für die Erstellung der Programme sind Anweisungslisten oder graphische Darstellungen benutzt worden welche von der Steuerung nacheinander, so schnell wie möglich, vom Anfang bis zum Ende bearbeitet werden. Danach startet die Steuerung einen neuen Programmzyklus beginnend beim ersten COB.

6.3.2 Zykluszeit

Die Zeit, welche vom Beginn des ersten bis zum Ende des letzten COB verstreicht wird Zykluszeit genannt. Sie ist von der Anzahl und der Art der programmierten Befehle und der FBoxen abhängig.

Die Zykluszeit liegt normalerweise im Millisekundenbereich. Ist die Zykluszeit bekannt, kann ermittelt werden, wie lange es vom Wechsel eines Eingangssignals bis zur Reaktion an einem PCD-Ausgang dauert.

Zyklische Programme werden mit Elementen wie COB, PB, FB, SB und XOB aufgebaut.

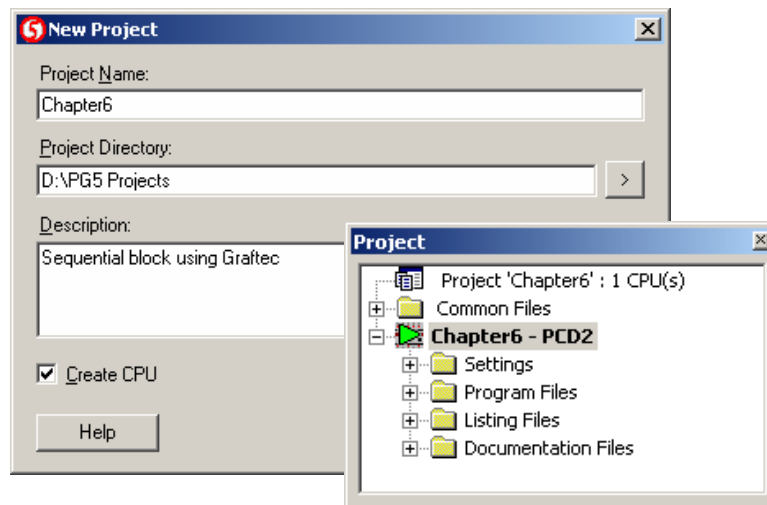
6.4 Erstellen einer neuen Graftec-Datei

Als Beispiel wird das Eröffnen eines neuen Projekts empfohlen, in welchem die Graftec-Programme eingefügt werden:

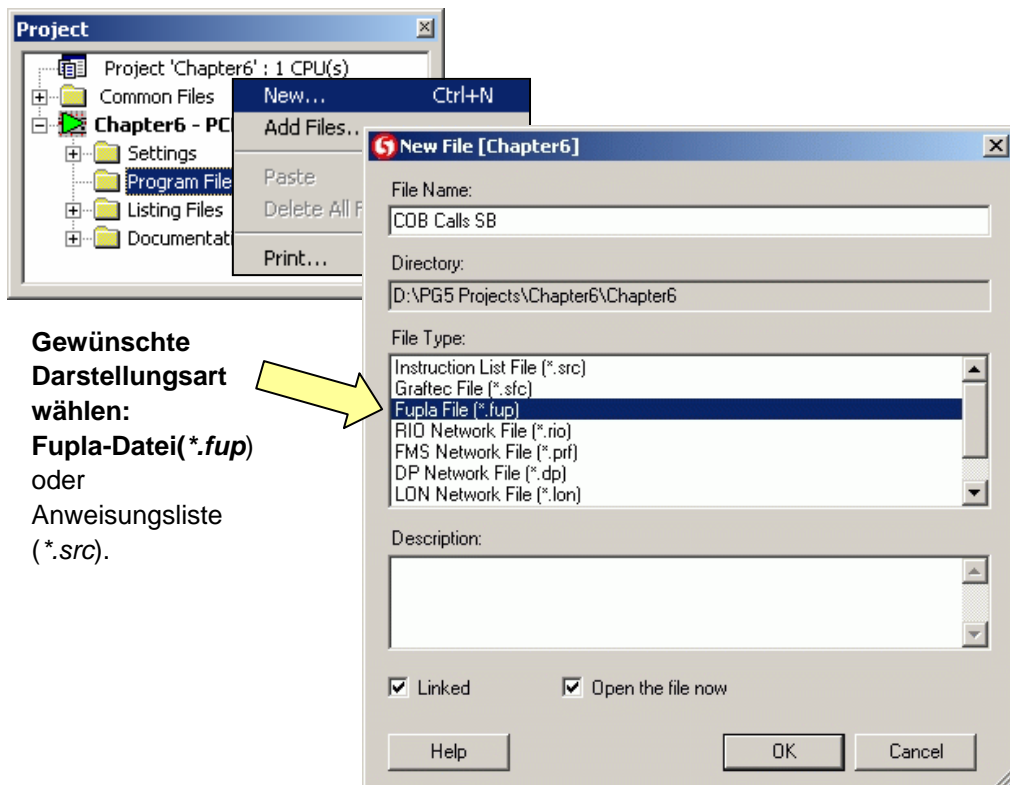
- Für ein Programm in graphischer Darstellung, eine Fupla-Datei und eine zweite Datei für Graftec eröffnen.
- Für ein Programm in Instruction bzw. Anweisungslisten, eine IL- /AWL-Datei und eine zweite Datei für Graftec eröffnen.
-

6.4.1 Eröffnen eines neuen Projekts

Im Fenster SAIA Project Manager die Schalter *Project, New... File, New Project...* anklicken, und das neue Projekt eröffnen.



6.4.2 Neue Fupla- oder IL-Datei eröffnen



6.4.3 Aufruf eines SB innerhalb eines COB

Aus einer Anweisungsliste wird Graftec mit CSB aufgerufen. Im Fupla benutzt man dazu die Funktionsbox *Aufruf SB*. Das Resultat ist dasselbe. Die neue Datei eröffnen und das Programm wie folgt schreiben:

In Anwendungslisten

```
COB 1 ;Start des COBs
    0
```

```
CSB 0 ;Auruf des SBs 0
```

```
ECOB ;Ende des COBs
```

Mit Fupla



Add FBox: *Definierbare Funktionen, Aufruf SB*

6.4.4 Neue Graftec-Datei eröffnen

Project 'Chapter6' : 1 CPU(s)

- Common Files
- Chapter6 - PCD2
 - Settings
 - Program Files
 - COB Call SB.fup
 - Pulse.sfc
 - Listing Files
 - Documentation Files

New... Ctrl+N
Add Files...
Paste Ctrl+W
Delete All Files...
Print... Ctrl+P

New File [Chapter6]

File Name: Pulse

Directory: D:\PG5 Projects\Chapter6\Chapter6

File Type:

- Instruction List File (*.src)
- Graftec File (*.sfc)
- Fupla File (*.fup)
- RIO Network File (*.rio)
- FMS Network File (*.prf)
- DP Network File (*.dp)
- LON Network File (*.lon)

Description:

Linked Open the file now

Help OK Cancel

Graftec File (*.sfc) anklicken

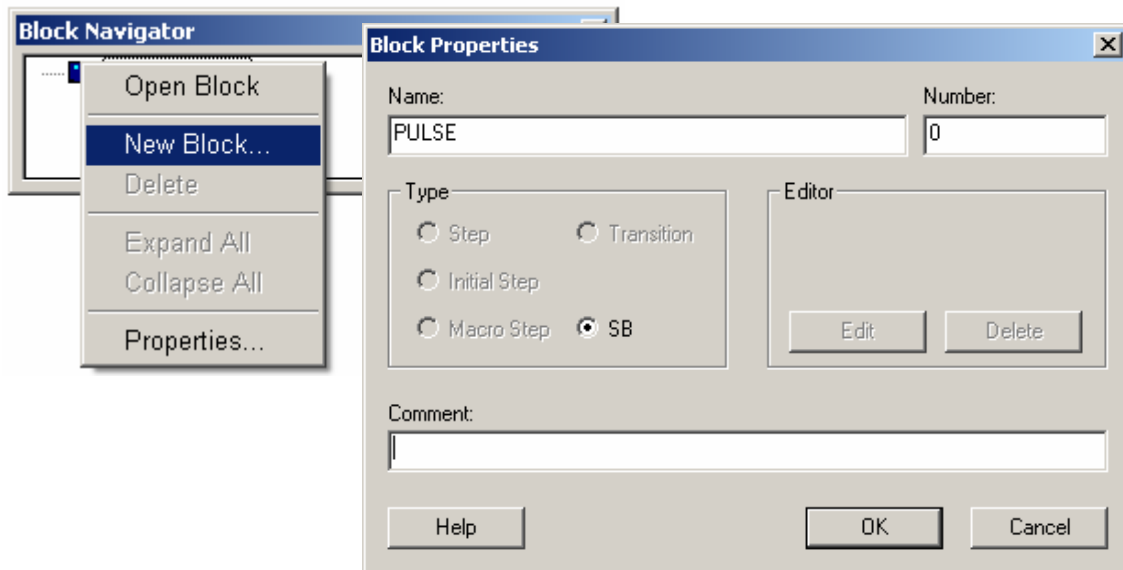
6.5 Die Organisation des SB

6.5.1 Block Navigator

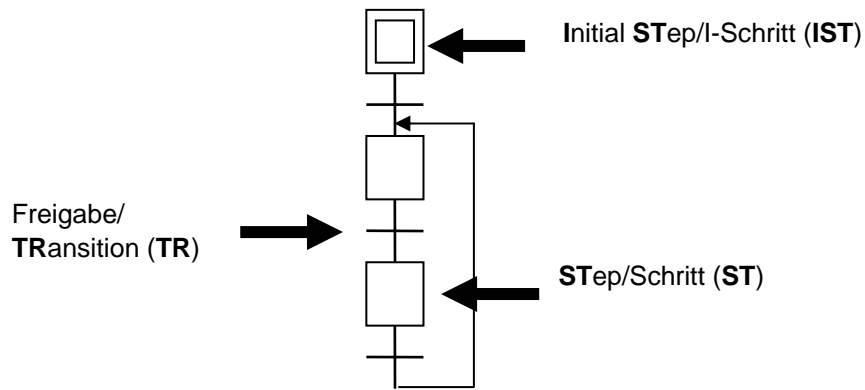
Beim Eröffnen einer neuen Graftec-Datei öffnet der Graftec -Editor den SB 0 mit einem Initialstep (IST). Der SB 0 wird automatisch in die SB-Liste eingetragen. Um zu dieser SB-Liste zu gelangen, ist die Graftec-Datei zu öffnen. Danach ist entweder das Schaltsymbol *SB Block List* oder es ist *Block, SB Block List* zu klicken. Es wird die SB-Liste angezeigt. Pro CPU kann eine Datei bis zu 32 SB umfassen. Besser ist allerdings das Anlegen einer eigenen Datei für jeden SB. Klicken auf *Edit* zeigt die SB-Nummer an. (Diese Nummer ist nicht mit der Anzahl der individuellen Elemente zu verwechseln). Die Angabe eines Namen ist sehr empfehlenswert und wird bei der Navigation durch die SB sehr hilfreich sein. Ohne Angabe eines Namens enthält die Liste nur die Nummer des SB.



Block
Navigator



6.5.2 Generelle Struktur eines SB



Die Programmierung in Graftec ist sehr einfach und bleibt auch bei komplexen Aufgaben noch übersichtlich. Das Anwenderprogramm wird in Ausführungsschritte **ST** und die dazu notwendigen Freigaben **TR** unterteilt, in welche der Anwender sein Anweisungen in Anweisungslisten oder graphisch mit Fupla einschreibt.

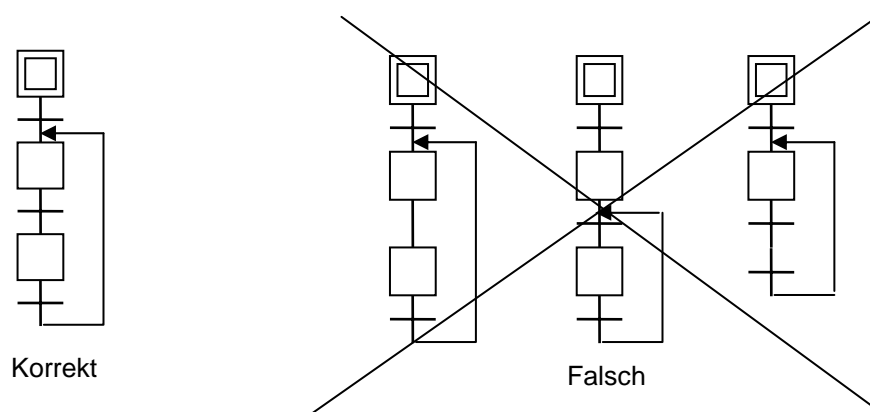
Der SB beginnt immer mit einem **IST** (Initial Step), dessen Doppelquadrat markiert den Beginn eines Programms. Hier wird die Bearbeitung einer Sequenz beim ersten Aufruf des SB gestartet (Kaltstart).

6.5.3 Grundregel für die Reihenfolge

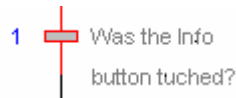
STEPS und TRANSITIONEN müssen immer abwechseln.

SB haben eine genau definierte Syntax.

- Ein SB startet immer mit einem IST.
- Danach müssen sich ST und TR immer abwechseln.
- Es dürfen sich nie 2 TR oder 2 ST direkt folgen



6.5.4 Transitionen (TR 0 to 1999¹⁾)



Eine TR enthält logische Verknüpfungen, welche solange abgearbeitet werden, bis die Freigabesituation für den nächsten Schritt erfolgt. Es kann z.B.:

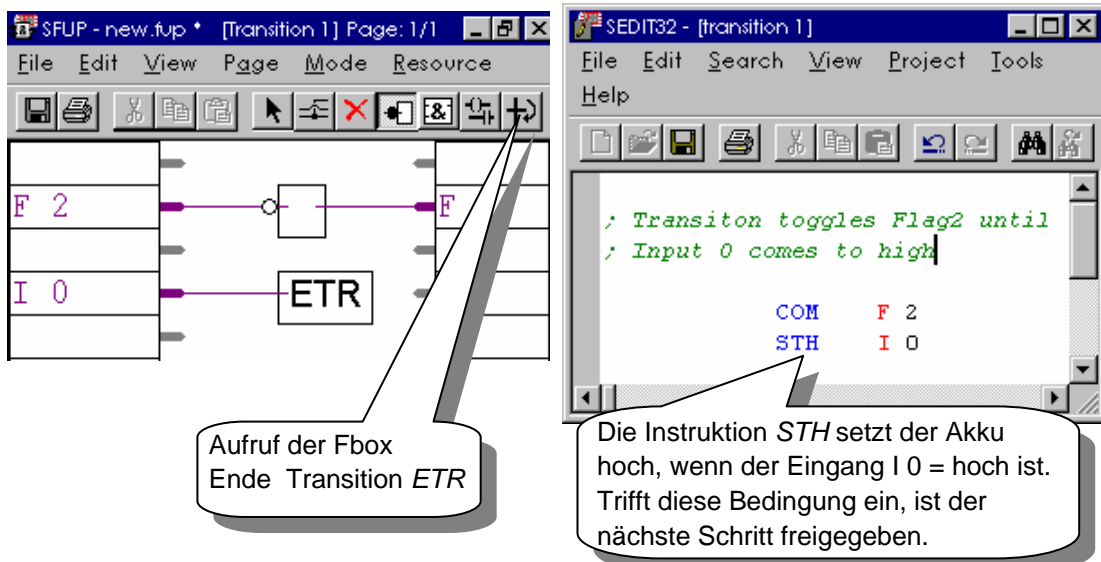
- der Empfang eines bestimmten Charakters, welcher über eine serielle Schnittstelle eintrifft, abgewartet werden.
- der Ablauf eines Zeitgliedes abgewartet werden.
- die Betätigung eines Endschalters abgewartet werden.

Im Fupla wird die Verknüpfung in der TR mit der FBox *ETR* abgeschlossen. Die TR wird solange wiederholt, bis die Freigabebedingung erfüllt ist, und somit *ETR* aktiviert wird. Ist die Bedingung nicht erfüllt, dann wird beim nächsten COB-Zyklus die ganze TR wiederholt abgearbeitet.

Beispiel: Flag 2 wird solange komplementiert wie der Eingang I 0 = H ist.

Mit Fupla

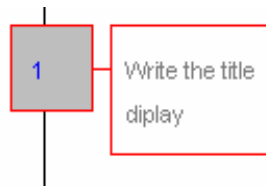
In Anweisungslisten



- TR müssen kein Programm enthalten. Eine leere TR wird als erfüllt (*true*) behandelt und es wird zum nächsten ST weiter geschaltet.
- TR in IL-Programmen:
Der ACCU ist zu Beginn einer TR oder eines ST immer = H.
- Es stehen 2000¹⁾ ST und 2000¹⁾ TR zum Bau von 32 SB zur Verfügung.

1) PCD2/4.M170, PCD2.M480 und PCD3 bearbeiten bis zu 6000 ST/TR und 96 SB

6.5.5 Steps (ST 0 to 1999 ¹⁾)

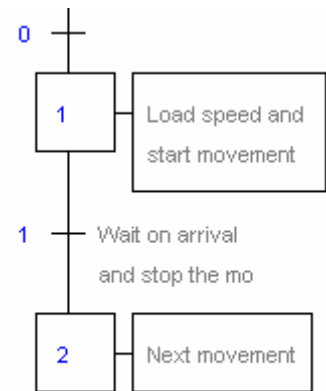


Die Schritte enthalten meistens Programmteile, die Aktionen im automatisierten Prozess generieren. Die Freigabe dieser Aktionen befinden sich in den unmittelbar vorher durchlaufenen Transitionen.

Ein Schlitten soll von A nach B bewegt werden.

Es wird zuerst die Geschwindigkeit und die Richtung der Bewegung definiert. Danach wird die Bewegung gestartet. Beides sind einmalige Aufgaben, welche, wie der Step selbst, nur ein einziges Mal ausgeführt werden.

Ist die Bewegung gestartet, ist die Ankunft beim Ziel B dauernd zu überwachen und die Bewegung beim Erreichen des Ziels auszuschalten. Diese wiederholte Überwachung (z.B. dauerndes Abfragen des Endschalters) wird in eine Transition gelegt. Da diese zyklisch bearbeitet wird, erfolgt die Erkennung des Ziels sofort.



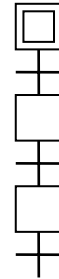
- Ein leerer Step geht direkt zur nächsten Transition.
- Ein Step wird nur ein einziges Mal abgearbeitet! Steps sind nicht zyklisch!

¹⁾ PCD2/4.M170, PCD2.M480 und PCD3 bearbeiten bis zu 6000 ST/TR und 96 SB

6.6 Typische SB-Strukturen

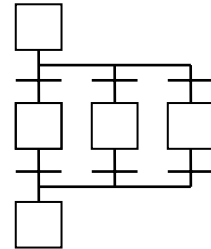
6.6.1 Einfache Sequenz (Schrittfolge)

Abwechselnde Folge von ST und TR. Es können nicht 2 ST oder 2 TR aufeinander folgen.



6.6.2 Alternativ-Verzweigung (ODER)

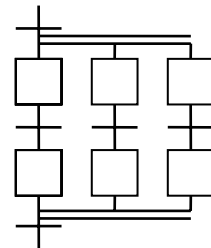
Eine Alternativ-Verzweigung erlaubt die Auswahl einer Sequenz unter mehreren. Alternativ-Verzweigungen beginnen immer mit parallelen Transitionen. Die TR werden im Programmablauf nacheinander (von links nach rechts) abgearbeitet, bis die erste TR als erfüllt erkannt wird. Danach wird nur dieser Zweig weiter behandelt. Jede Alternativ-Verzweigung endet mit einer TR welche auf ein- und denselben ST geht. Es können max. 32 alternative Zweige programmiert werden. Sind zu viele editiert, wird der XOB 9 aufgerufen (siehe Kapitel 5). Ist im XOB 9 keine Massnahme programmiert, leuchtet die Error-LED auf.



6.6.3 Simultan-Verzweigung (UND)

Es werden mehrere Sequenzen gleichzeitig gestartet und auf einer End-Transition wieder vereint.

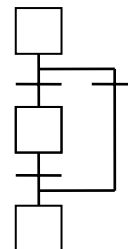
- Der gemeinsame Start folgt nach einer Transition mit mehreren parallelen Steps.
- Beendet wird eine Simultan-Sequenz mit der Synchronisation von 2 oder mehr Sequenzen auf eine Transition
- Es können max. 32 simultane Zweige programmiert werden. Sind zu viele programmiert erfolgt die Fehlermeldung wie oben beschrieben.



6.6.4 Überspringen von Sequenzen

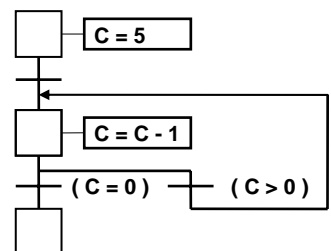
Es wird zuerst die linke und dann die rechte Transition bearbeitet.

Ist die rechte TR zuerst erfüllt, wird die linke Sequenz übersprungen.



6.6.5 Wiederholen einer Sequenz

Dieses Beispiel baut auf den vorangehenden auf. Die alternative Verzweigung ist abhängig vom Inhalt eines Zählers, der bei jedem Durchgang um eins dekrementiert wird. Solange der Zähler nicht bei 0 ankommt, wird der mittlere Schritt wiederholt (in diesem Fall 5 Mal). Bei 0 wird die Schrittfolge fortgesetzt. Die konsequente Abfolge von Schritten und Transitionen ist auch bei Verzweigungen zu beachten!

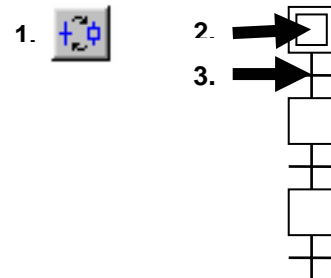


6.7 Editieren einer Sequenz

Wird ein neuer SB eröffnet, ist bereits der Initial-Step (IST) eingezeichnet. Die Programmausführung beginnt immer hier. Beim Anfügen weiterer Elemente kann entweder die Werkzeugleiste oder über das Tastenfeld das Menü verwendet werden.

6.7.6 Editieren einer einfachen Sequenz

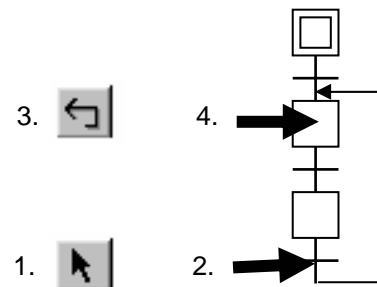
1. *Mixed Mode* anklicken
2. Cursor unter den IST bewegen und anklicken (linke Maustaste)
3. Cursor unter die neue TR bewegen und wieder anklicken
4. und so weiter



6.7.7 Editieren einer Verbindung

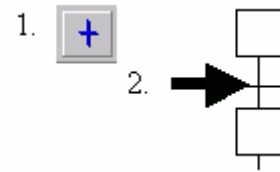
Ist die Sequenz fertig gezeichnet, so ist die Programmierung der abzulaufende Struktur fast beendet. Soll diese Sequenz nach Ablauf wieder neu starten, muss die Schleife (Link) geschlossen werden. Es können nur Verbindungen zwischen dem Ausgang einer TR und dem Beginn eines ST gezeichnet werden. Vorgehen:

1. *Select Mode* wählen
2. Markieren der Start-TR
3. *Link Mode* wählen
4. ST auswählen, zu welchem der Link führen soll.



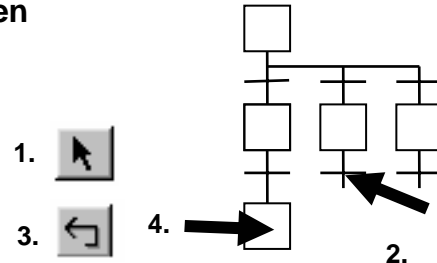
6.7.8 Editieren einer Alternativ-Verzweigung (ODER)

1. *Transition-Mode* wählen
2. Anklicken einer TR, welche bereits von einem ST gefolgt wird.
3. Mit jedem Klick wird eine neue parallele TR gezeichnet.



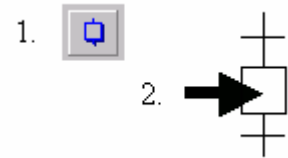
6.7.9 Schliessen von alternativen Sequenzen

1. *Select Mode* wählen
2. TR der zu schliessenden Sequenz anklicken
3. *Link Mode* wählen
4. ST, zu welchem verbunden werden soll, anklicken.



6.7.10 Editieren von Simultan-Verzweigungen (UND)

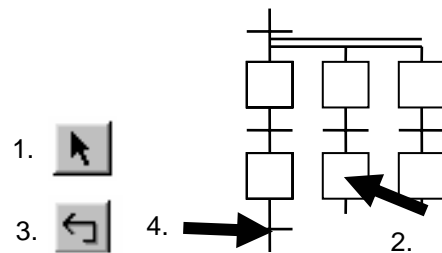
1. *Step-Mode* wählen
2. Anklicken eines ST, welcher bereits von einer TR gefolgt wird.
3. Mit jedem Klick wird ein weiterer paralleler ST gezeichnet.



6.7.11 Synchronisieren einer Simultan-Verzweigung

Synchronisieren (Schliessen) einer Simultan-Verzweigung:

1. *Select-Mode* wählen
2. Zu synchronisierenden ST markieren
3. *Link-Mode* wählen
4. TR anklicken, zu welcher die Verzweigung synchronisiert werden soll.

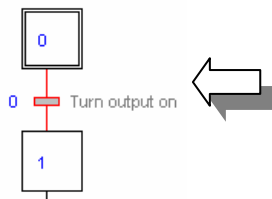


6.7.12 Zufügen eines Kommentars

1. *Select-Mode* wählen
2. Klicken Sie mit der rechten Maustaste auf ein Element. Wählen Sie das Menü *Properties...*
3. Geben Sie einen Kommentar in das Feld *Comment* ein.

Anmerkung:

Um einen Kommentar über zwei Zeilen einzugeben, fügen Sie die Zeichen `\n` ein.



Edit Code	Enter
Cut	Ctrl+X
Copy	Ctrl+C
Paste	Ctrl+V
Delete	Del
Properties...	Alt+Enter

Element Properties

Name: PULSE_ST_1 Number:

Type: Step Transition
 Initial Step
 Macro Step SB

Editor: Function Block Diagram

Comment: \n Turn the output on

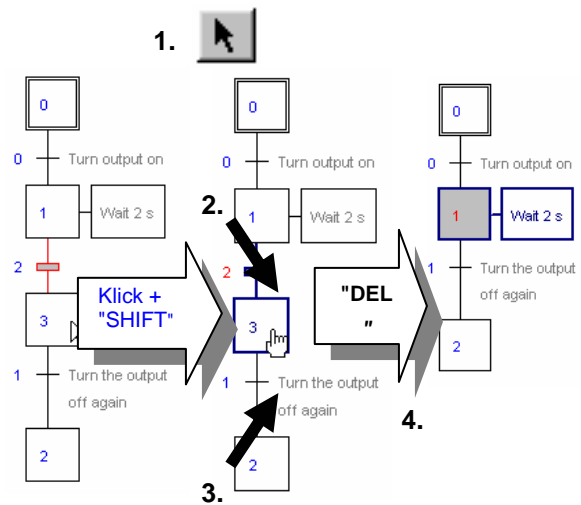
6.7.13 Sequenz einfügen

1. *Transition-Mode* wählen
2. ST, welcher von einer TR gefolgt wird, anklicken.

Der Editor wird eine neue TR und einen neuen ST einfügen.

6.7.14 Sequenz löschen

1. *Select Mode* wählen
2. Erste TR der Sequenz anklicken.
3. Letzten ST der zu löschenden Sequenz bei gedrückter <Shift> Taste anklicken.
4. Taste drücken.



6.7.15 Kopieren/Einfügen einer Sequenz

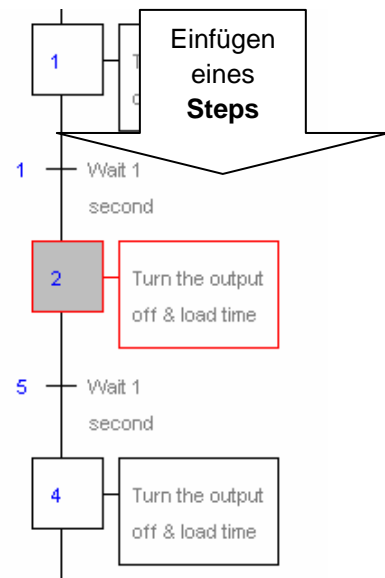
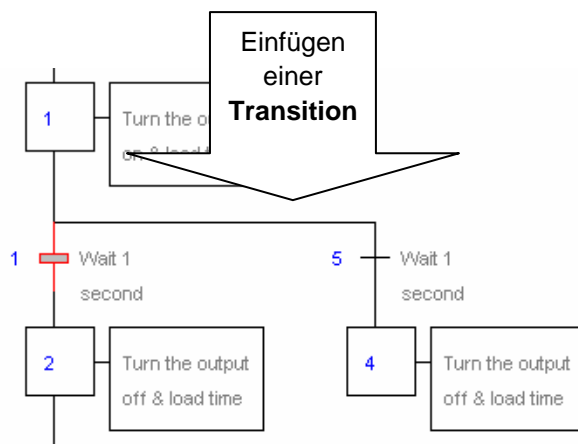
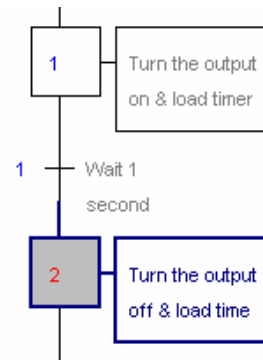
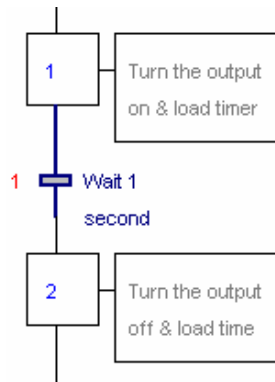
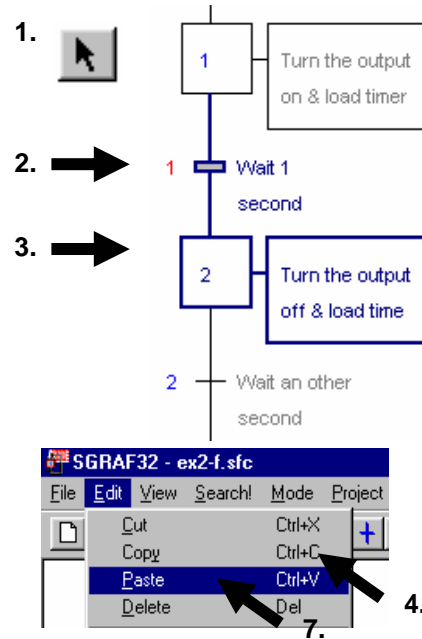
Sequenz kopieren:

1. *Select-mode* wählen
2. Beginn der Sequenz markieren
3. Letzten ST der zu kopierenden Sequenz bei gedrückter *Shift* Taste anklicken.
4. Menü *Edit* und *Copy*
5. *Select-mode* wieder aktivieren
6. Punkt anklicken, an welchen die Sequenz kopiert werden soll
7. Menü *Edit* und *Paste*

Sequenz einfügen:

Bemerkung:

Abhängig von der Art des Elements (TR oder ST) ist die Sequenz unterhalb oder neben dem gewählten Element einzufügen.



6.8 Editieren des ersten sequentiellen Blocks

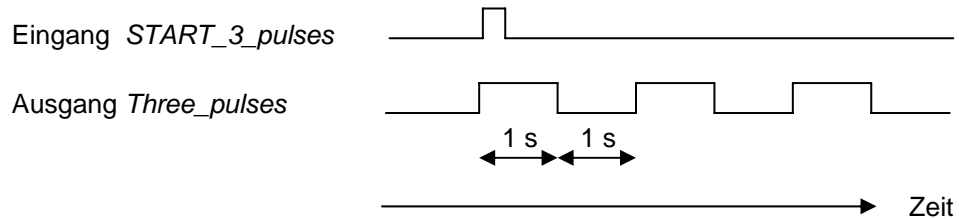
6.8.1 Öffnen einer Datei

Öffnen von *PULSE.sfc*. Zur SB-Liste wechseln und dort SB *PULSE* laden.

Ziel:

Es soll ein Programm editiert werden, welches einen digitalen Ausgang (*Three_pulses*, O 33) 3-mal blinken lässt und zwar jedesmal, wenn ein digitaler Eingang (*Start_3_pulses*, I 2) = H wird.

Diagramm:



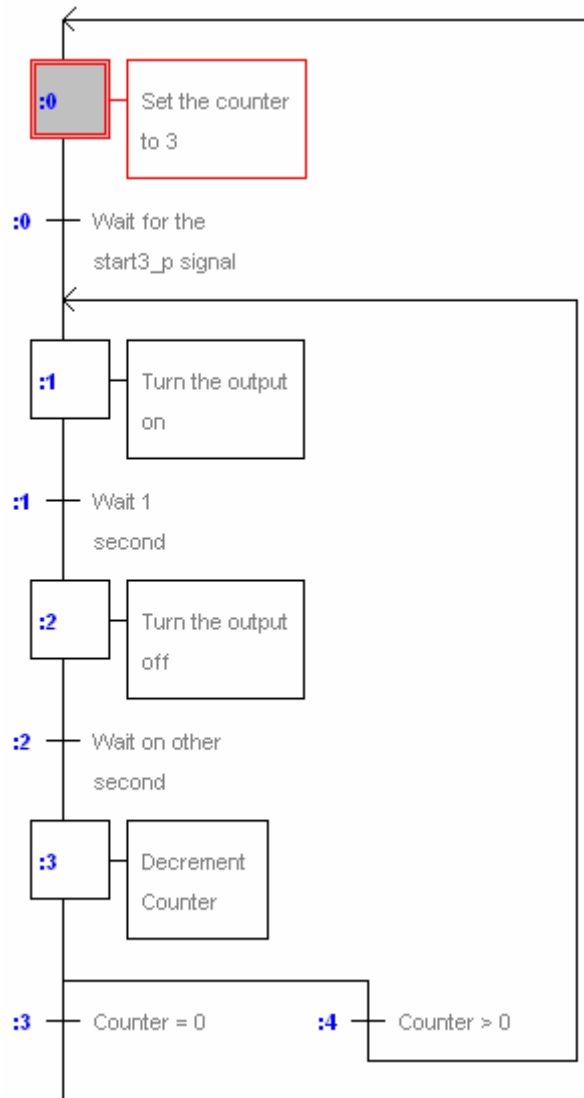
6.8.2 Zeichnen der Grundstruktur

Es wird, wie gewohnt, mit dem IST begonnen, mit welchem auch die Initialisierung des Zählers erfolgt. Ist die Initialisierung abgeschlossen, wird das Startsignal *Start_3_pulses* abgewartet. Elemente zeichnen und kommentieren, so wie sie nachfolgend dargestellt sind.

The screenshot shows the 'Element Properties' dialog box. The 'Name' field contains 'PULSE_TR_0'. The 'Number' field is empty. Under the 'Type' section, 'Transition' is selected with a radio button. Other options include 'Step', 'Initial Step', 'Macro Step', and 'SB'. The 'Editor' section shows 'Function Block Diagram' with 'Edit' and 'Delete' buttons. The 'Comment' field contains the text 'Wait for the \nstart3_p signal'. The dialog has 'Help', 'OK', and 'Cancel' buttons at the bottom.

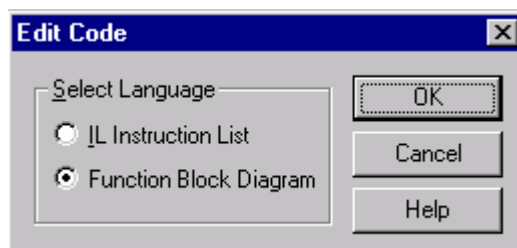
In the background, a ladder logic diagram is visible. It shows a counter block with a value of ':0' and a transition block with a value of ':0'. The transition block is connected to the counter block. The transition block has a comment 'Wait for the start3_p signal'.

Nach dem Start der Sequenz wird der Ausgang *Three_pulses* für 1 Sekunde ein und danach für 1 Sekunde ausgeschaltet. Dies geschieht 3 Mal, danach wird die Sequenz neu gestartet.



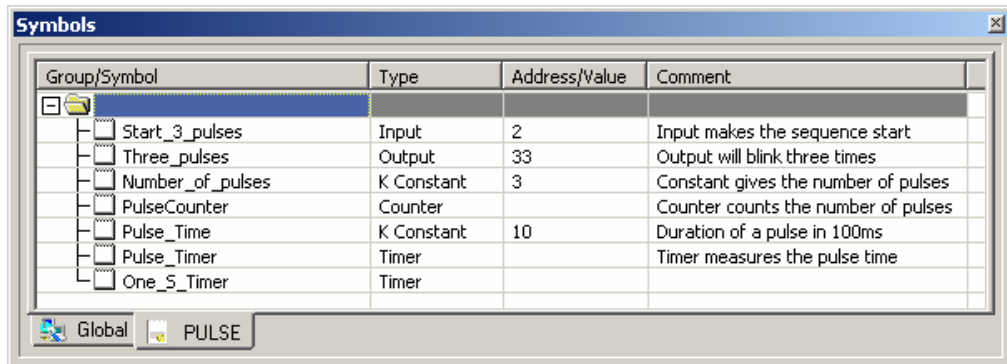
6.8.3 Wahl von S-Edit oder Fupla zum editieren des Programm-Code

Die Sequenz ist nun fertig gezeichnet und kommentiert. Es ist jetzt noch der Programm-Code entweder in IL (*Instruction List*) oder Fupla (*Function Block Diagram*) in jede TR und jeden ST einzutragen. Der IST wird in Fupla programmiert. Dies geschieht mit einem Doppelklick auf den IST und der Wahl des Fupla-Editors:



6.8.4 Vorbereiten der Symbole

Im Symbol-Editor wird zuerst eine Liste mit allen zu verwendenden Elementen erstellt. Die Elemente sind gemäss der folgenden Abbildung einzutragen.



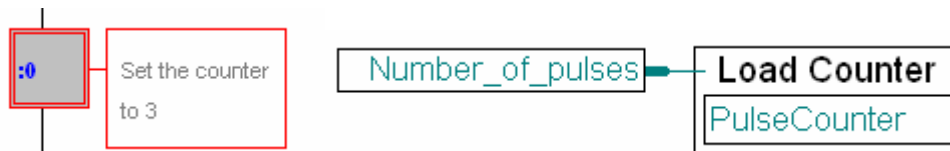
6.8.5 Editieren des Programm-Code

Als nächstes wird der Zähler *PulseCounter* mit der Konstante *Number_of_pulses* gleich 3 geladen.

Fupla- Programm:

Dazu wird die FBox *Graftec, Load counter* aus der Werkzeugkiste *Add Fbox* des Fupla (Function Block Diagram) verwendet.

Anmerkung: Counter und Timer sind nicht aus anderen FBox-Familien zu nehmen, da die anderen den zyklischen Programmen vorbehalten sind.



Als Anweisungsliste:

```
LD PulseCounter ;Initialisierung des Counters
Number_of_pulses
```

6.8.6 Das Programmieren einer Transition:

Eine TR wird solange wiederholt, bis die eingetragene Bedingung in der Anweisungsliste erfüllt oder das *ETR* im Fupla aktiviert wird.

In TR 0 wird gewartet, bis der Eingang *Start_3_pulses* = H wird. TR 0 öffnen und das folgende Fupla-Programm eingeben:

Fupla- Programm:



Add FBox: *Graftec, Ende TR*

Mit Anweisungsliste:

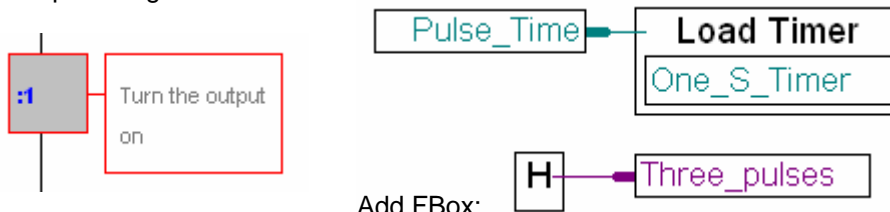
```
STH Start_3_pulses ;Den Zustand Start_3_pulses in den Akku laden
```

6.8.7 Benutzen von Timer in sequentiellen Blöcke

Vorgehensweise: Der Ausgang *Three_pulses* wird H gesetzt und der Timer *One_S_Timer* mit der Zeit *Pulse_time* geladen.

Bemerkung: Timer und Counter aus der Standard-Fupla-Bibliothek können nicht in Graftec-Programmen verwendet werden. Diese sind für den Einsatz in zyklischen Programmen vorgesehen. Innerhalb eines SB sind die Funktionen aus der Graftec-Familie zu verwenden. Diese sind etwas anders gestaltet, so dass sie in einem ST gestartet und in einer nachfolgenden TR abgefragt werden können. Laden des Timer und Setzen des Ausganges im ST

Fupla- Programm:



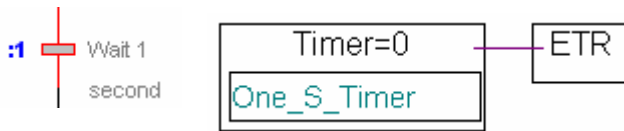
Add FBox:
Graftec, Lade Timer
Binäre Funktion, Setze H

Als Anweisungsliste:

```
SET      Three_pulses      ;Ausgang wird gesetzt
LD       One_S_Timer       ;Timer wird geladen
                Pulse_Time
```

6.8.8 Warten bis die Zeit abgelaufen ist.

Fupla- Programm:



Add FBox:
Graftec, Timer abgelaufen
Graftec, Ende TR

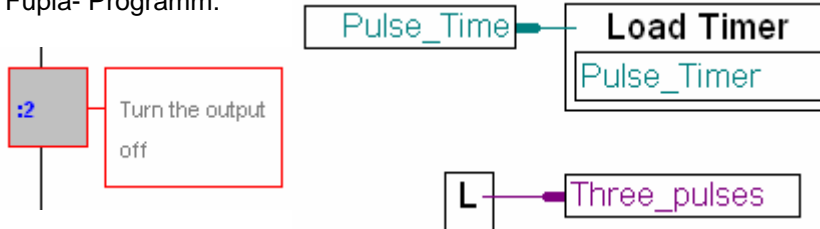
Als Anweisungsliste:

```
STL      One_S_Timer      ;Wenn die Zeit abgelaufen ist,
                        ;dann wird mit STL der Akku H gesetzt
```


6.8.9 Wiederholen der Sequenz für die Pause des Blinkers

Die soeben programmierten ST und TR werden wiederholt, mit dem Unterschied, dass im ST der Ausgang *Three_pulses* L gesetzt und ein anderer Timer benutzt wird.

Fupla- Programm:



FBox:
 Graftec, Lade Timer
 Binäre Funktion,
 Setzer L

Als Anweisungsliste:

```
RES      Three_pulses      ;Ausgang wird rückgesetzt
LD       Pulse_Timer       ;Timer wird geladen
                Pulse_Time
```

TR:

Fupla- Programm:



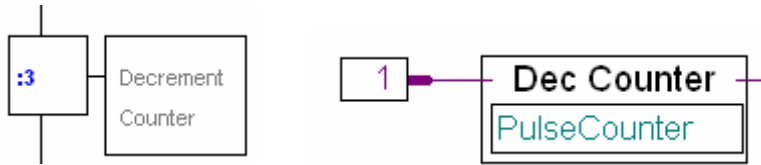
Als Anweisungsliste:

```
STL      Pulse_Timer      ;Akku = H wenn Zeit abgelaufen ist
```

Bemerkung: Zwei unterschiedliche Timer sind in diesem Programm benutzt worden (*One_S_Timer* und *Pulse_Timer*). Diese sind aber nie zur gleichen Zeit aktiv. Es ist deshalb möglich zweimal denselben Timer zu benutzen, um nicht Timer unnötig zu vergeben.

6.8.10 Dekrementieren eines Zählers

Fupla- Programm:



Add Fbox: Graftec, Counter - 1

Als Anweisungsliste:

```
DEC          PulseCounter      ;Der Zähler PulseCounter wird um 1
                                   ;dekrementiert, sofern der Akku = H
ist
```

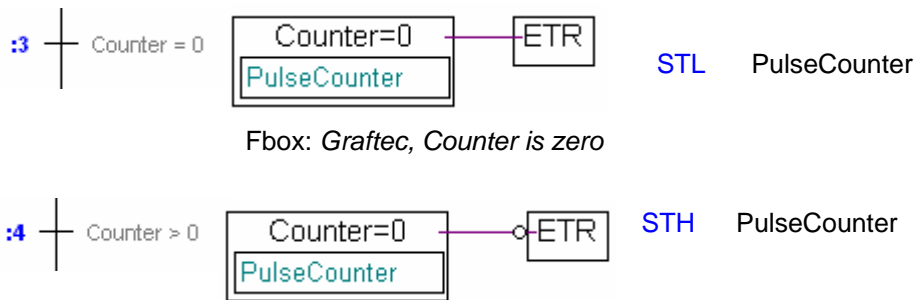
(N.B.: Der Akku ist immer H am Anfang eines ST/TR)

6.8.11 Alternative Verzweigung

Die folgenden zwei TR werden für die alternative Verzweigung programmiert:

Fupla Programm:

Als Anweisungsliste:



Fbox: Graftec, Counter is zero

Bei der oberen TR wird ETR aktiv wenn der Zähler 0 ist.
Die untere TR aktiviert ETR wenn der Zähler nicht 0 ist.



Invert Binary connector

Die Inversion am Eingang des ETR wird mit dem Schalter invert binary connector gewählt.

6.9 Kompilation (build) und Debuggen eines Programms



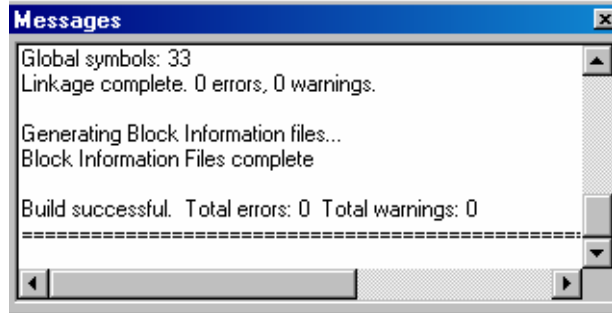
Build
All

Das fertig editierte Beispiel kann kompiliert werden, indem die *Build*-Funktion gestartet wird. (vorher ist sicherzustellen, dass die COB- und die *PULSE.sfc*-Datei die einzigen Dateien sind, welche gelinkt werden. Auch der SB *OpenMainGate* kann gelöscht werden).

6.9.1 Messages-Fenster

Wurde das Programm korrekt editiert, sollte die folgende Meldung erscheinen:

Build successful.
Total errors: 0 Total warnings: 0



6.9.2 Online-Werkzeuge

Enthält das Programm Fehler, so werden sie in roter Schrift angezeigt. Ein Doppelklick auf eine Fehlermeldung führt zur Fehlerstelle. Das Programm wird nun in die SPS geladen. Der Ablauf des SB kann online verfolgt werden. Der rote Punkt, bzw. die roten Punkte zeigen die aktiven TR an.



Download
Program

Run Stop Schritt für Schritt

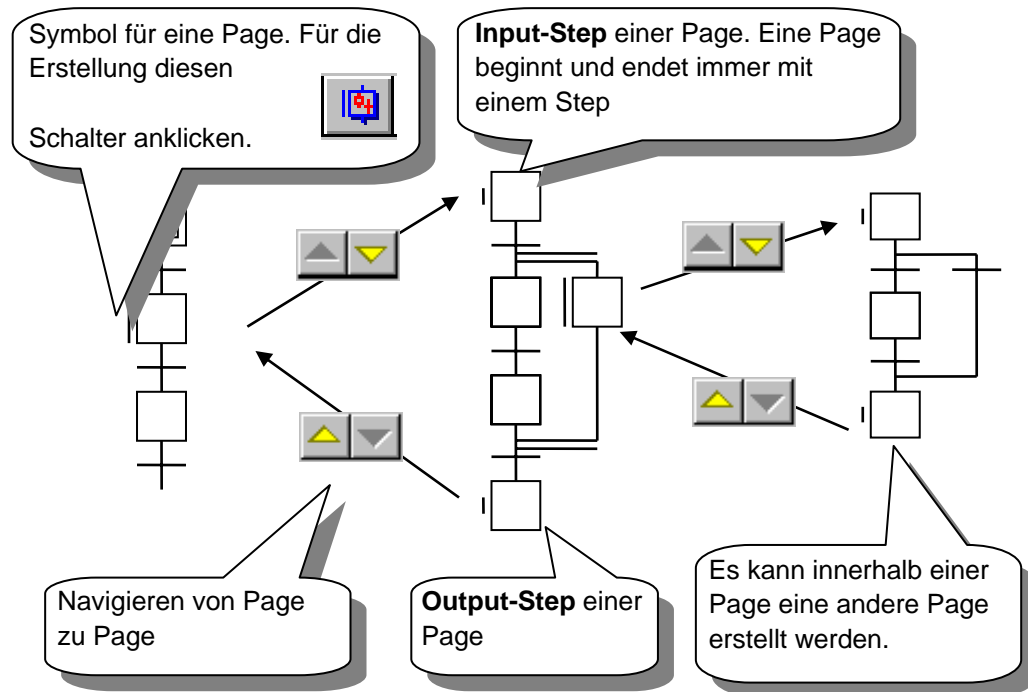
Der rote Punkt markiert eine aktive Transition

Der Programmablauf kann zu jeder Zeit unterbrochen und Schritt für Schritt fortgesetzt werden.

6.10 Struktur eines Graftec-Programms mit Pages

6.10.1 Was sind Pages?

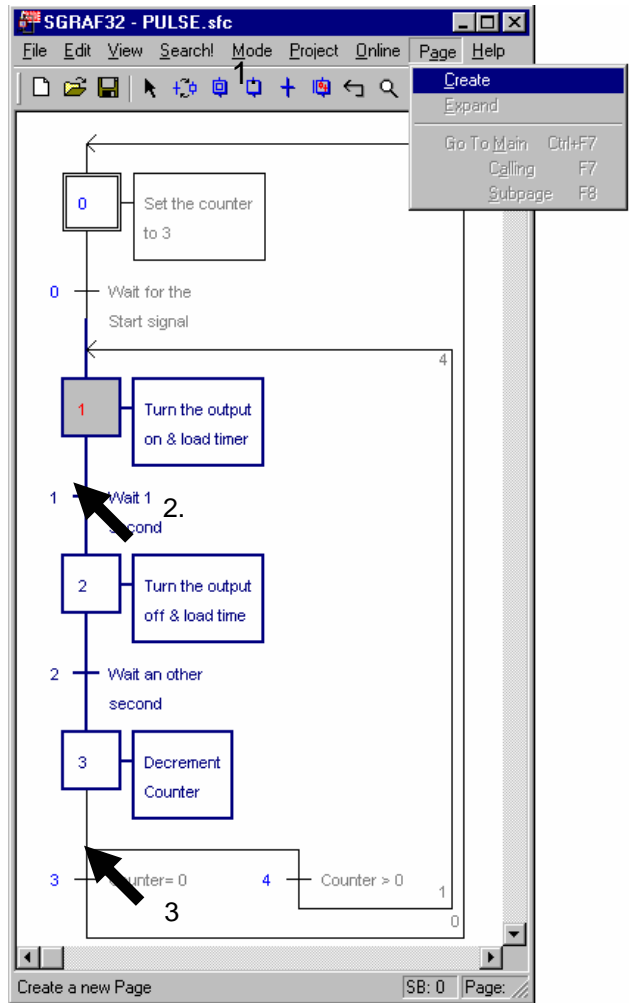
Mit PG5 kann ein grosses Programm in einzelne aufeinander zugreifende Sub-Pages unterteilt werden.



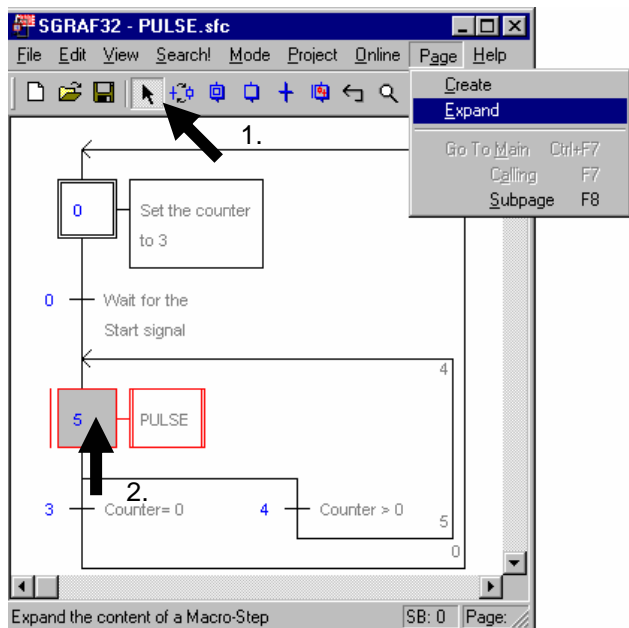
- Eine Page beginnt und endet immer mit einem Step.
- Eine Page kann nur einen einzigen **Input-Step** und einen einzigen **Output-Step** enthalten.
- Es können beliebig viele Pages innerhalb einer Page editiert werden.
- Der **Input-Step** und der **Output-Step** können nicht verschoben oder gelöscht werden.

6.10.2 Erstellen einer Page

1. *Select-Mode* wählen
2. Ersten ST der Sequenz anklicken
3. Bei gedrückter *Shift*-Taste letzten ST der Sequenz anklicken
4. *Page* und dann *Create* anwählen

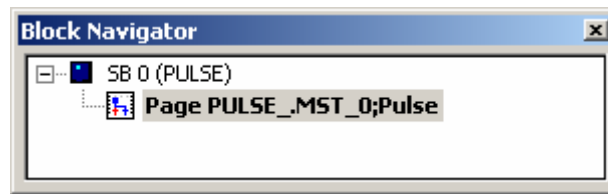


1. *Select-Mode* wählen
2. *Page* anklicken.
3. *Page* ausführen und *Calling* oder *Subpage* anwählen



1. *Select-Mode* wählen
2. *Page* anklicken
3. *Page*, *Expand* ausführen

Anmerkung: Der *Block Navigator* vereinfacht die Navigation zwischen den Seiten eines beliebigen SB.



Inhalt

7	IL-PROGRAMMIERUNG (INSTRUCTION LIST)	3
7.1	Zusammenfassung	3
7.2	Vorbereitung eines IL Projektes	4
7.2.1	Eröffnen eines neuen Projekts	4
7.2.2	Eröffnen einer neuen IL Programm-Datei	4
7.3	Aufbau eines IL Editor-Fensters	5
7.3.1	Schreiben von Programm-Zeilen	6
7.3.2	Format der Programm-Zeilen	7
7.3.3	Aufbau eines Organisationsblocks	7
7.3.4	Abarbeiten der Befehle und Blöcke	7
7.3.5	Regeln beim Schreiben von Blöcken	8
7.4	Symbols-Fenster	9
7.4.1	Neue Symbole der <i>Symbols</i> Liste hinzufügen	10
7.4.2	Operanden Adressierung	11
7.4.3	Benutzen der Symbole aus der <i>Symbols</i> -Liste im IL Programm	12
7.4.4	Lokale und Globale Symbole	13
7.5	Einführung in den PCD-Befehlssatz	14
7.5.1	Der Akkumulator	14
7.5.2	Binäre Befehle	15
7.5.3	Dynamisierung	19
7.5.4	Status Flags	20
7.5.5	Wort-Befehle für Timer	21
7.5.6	Befehle für Counter	23
7.5.7	Akkumulator-abhängige Befehle	24
7.5.8	Wort-Befehle für ganzzahlige Arithmetik	25
7.5.9	Wort-Befehle für Fliesskomma Arithmetik	26
7.5.10	Umwandlung von Ganzzahl- und Fliesskomma Registern	26
7.5.11	Index Register	27
7.5.12	Programmsprünge	28
7.6	Editieren eines ersten Anwender Programms	30
7.6.1	Verarbeiten (build) des Programms	32
7.7	Laden des Programms in die PCD	33
7.8	On-line testen (debuggen) des Programms	33
7.8.1	Anzeigen des Programm-Kodes	33
7.8.1	On-/Off-line, Run und Stop	34
7.8.2	Step-by-step Modus	35
7.8.3	Anhalte-Punkte (Breakpoints)	36
7.8.4	On-line Änderung am Programm	37
7.8.5	Betrachten und Ändern von Symbol-Zuständen mit dem <i>Watch Window</i>	38
7.9	Inbetriebnahme eines Analogmoduls	39
7.9.1	Beispiel für PCD2.W340 Analog-Eingangsmodule	39

7 IL-Programmierung (Instruction List)

7.1 Zusammenfassung

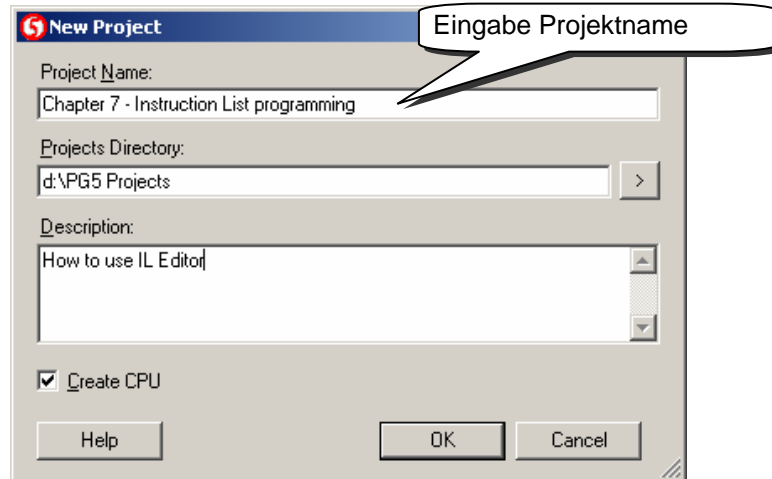
Der IL Editor ist das flexibelste und mächtigste Werkzeug zum Programmieren von PCD Steuerungen. IL steht für Instruction List: eine nicht-graphische Programmierumgebung zum Schreiben von Programmen mit Hilfe des mächtigen PCD Befehlssatzes. Alle PCD Steuerungen benutzen diesen Befehlssatz, dies garantiert die Portabilität 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.

7.2 Vorbereitung eines IL Projektes

Vor dem Schreiben eines Beispiels, empfehlen wir ein neues Projekt und eine neue Programm-Datei anzulegen.

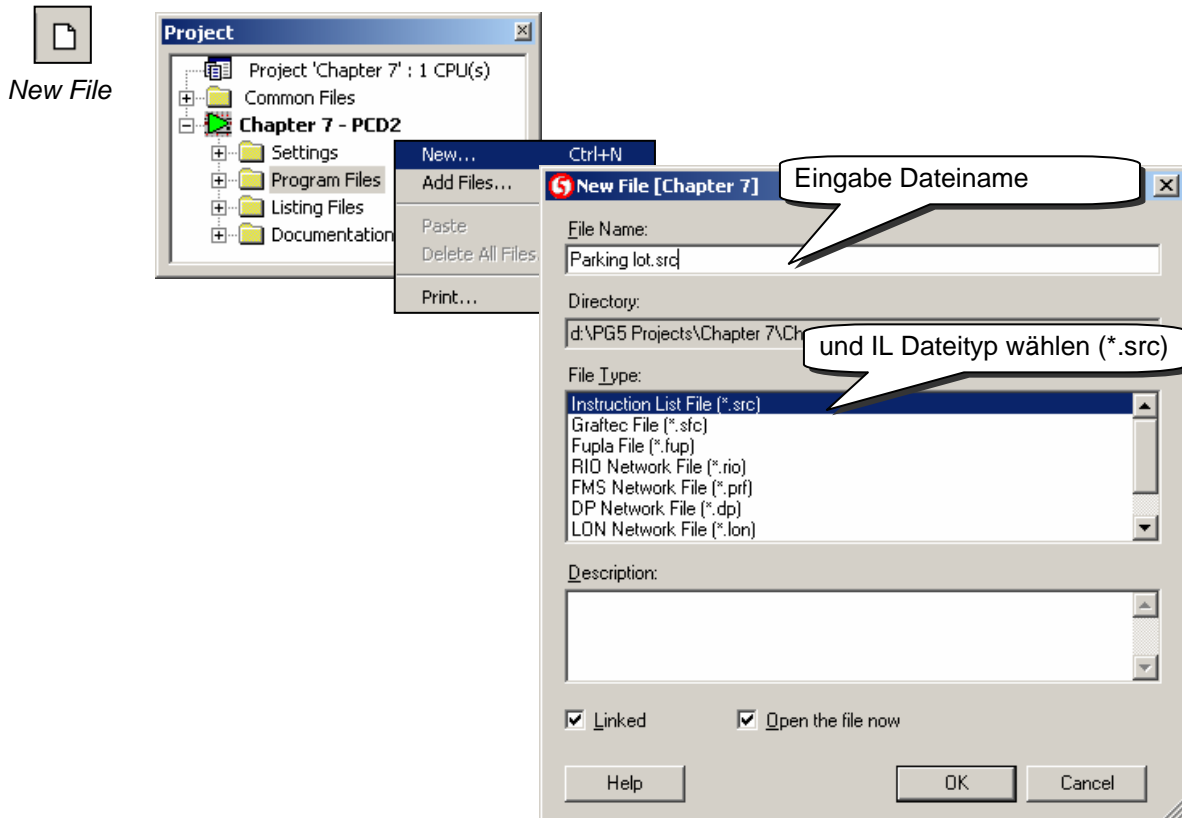
7.2.1 Eröffnen eines neuen Projekts

Im *SAIA Project Manager* Fenster, Menü *Project, New...* auswählen und ein neues Projekt eröffnen.



7.2.2 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):



7.3 Aufbau eines IL Editor-Fensters

Mnemo-Kode

Labels	Operanden	Kommentare
; Cyclical Organisation Block		
COB	0	; Cyclical program
	0	; No supervision time
STH	Car_incoming	; A car comes into the parking:
DYN	Dynamise_incoming_car_signal	; On the positiv flank of inco
DEC	Number_of_free_slots	; Decrement the number of fre
;-----		
STH	Car_outgoing	; A car leaves into the parking
DYN	Dynamise_leaving_car_signal	; On the positiv flank of outgo
INC	Number_of_free_slots	; Increment the number of fre
;-----		
STL	Number_of_free_slots	; If no more free parking slots
OUT	Red_light	; Set the red light
ECOB		; End of Cyclical program

Group/Symbol	Type	Address...	Comment
Car_incoming	Input	0	Gets high when a car comes into the parking
Car_outgoing	Input	1	Gets high when a car leaves the parking
Red_light	Output	32	Stops new cars at the entry
Number_of_free_slots	Counter		Counts the number of free parking slots
Dynamise_incoming_car_signal	F		Flag detects the rising edge of the car incoming
Dynamise_leaving_car_signal	F		Flag detects the rising edge on the car leaving

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 getestet werden

7.3.1 Schreiben von Programm-Zeilen

Label	Mnemo.	Operand	Kommentar
			<i>;Increment a register</i>
	STH	Flag	<i>;Copy the Flag state into the accu</i>
	DYN	DFlag	<i>;On a positiv flank of the Flag , set the accu eigh</i>
	JR	L Next	<i>;If the accu is low, jump to the label Next</i>
	INC	Register	<i>; Increment the register</i>
Next:	NOP		<i>;No instruction</i>

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.

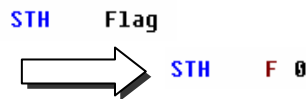
Operand

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.



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

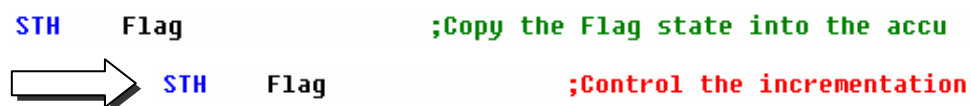
```

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.

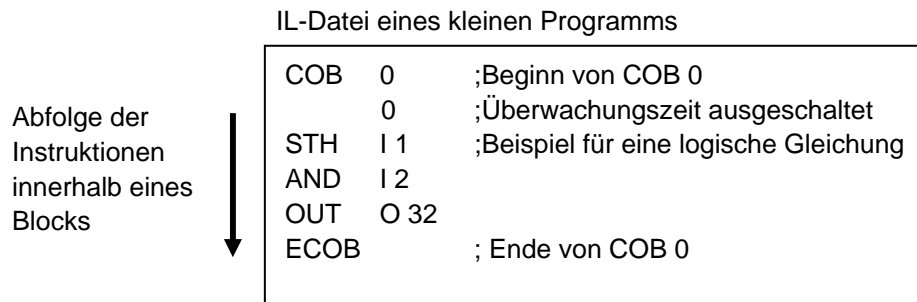


7.3.2 Format der Programm-Zeilen

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.

7.3.3 Aufbau eines Organisationsblocks



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.

7.3.4 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.

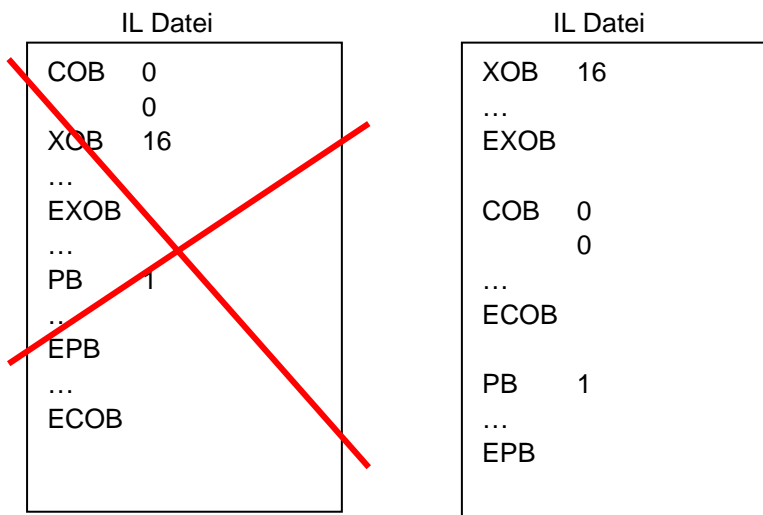
Dann folgen die COBs im Programm. Diese werden in numerischer Reihenfolge abgearbeitet: COB 0, COB 1, ... COB 15, gleichgültig in welcher Reihenfolge sie in der Datei auftreten. Nach dem letzten COB beginnt das Programm wieder mit COB 0.

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.

7.3.5 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.

7.4 Symbols-Fenster



Show Hide Symbols Editor

Group/Symbol	Type	Address...	Comment
Car_incoming	Input	0	Gets high when a car comes into the parking
Car_outgoing	Input	1	Gets high when a car leaves the parking
Red_light	Output	32	Stops new cars at the entry
Number_of_free_slots	Counter		Counts the number of free parking slots
Dynamise_incoming_car_signal	F		Flag detects the rising edge of the car incoming
Dynamise_leaving_car_signal	F		Flag detects the rising edge on the car leaving

Das Symbols 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

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 in jeder Programm-Zeile durchzuführen, muss nur im Symbols-Fenster 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 8191
 Register: R 0, ..., R 4095¹, 16383²
 Timer/Counter: T/C 0...T/C 1599

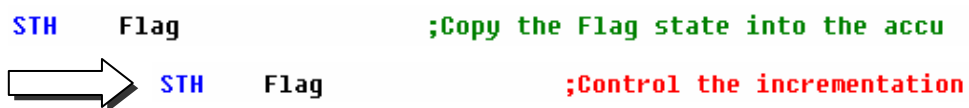
Comment

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



¹ Alle PCD




² PCD2.M480,PCD3

7.4.1 Neue Symbole der *Symbols* Liste hinzufügen

Einfache Methode:

Zum hinzufügen von neuen Symbolen zur Liste das *Symbols*-Fenster öffnen, die Maus in der Mitte des Fensters positionieren und mit rechtem Maus-Klick das Kontext-Menü *Insert Symbol* auswählen. Dann die Felder *Group/Symbol*, *Type*, *Address/Value* und *Comment* ausfüllen.

Schnelle Methode 1:

Group/Symbol	Type	Address/Value	Comment
			
<ul style="list-style-type: none">  Red_light o 32 ;Stop new cars 			
<ul style="list-style-type: none">  Red_light 	Output	32	Stop new cars

Enter →


Eine weitere Möglichkeit ist die Eingabe von Variablen für die verschiedenen Informationsfelder vom *Group/Symbol* 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:

Group/Symbol	Type	Address/Value	Comment
			

OUT Red_light = 0 32 ;Stop new cars|

Enter →

Group/Symbol	Type	Address/Value	Comment
			
<ul style="list-style-type: none">  Red_light 	Output	32	Stop new cars

OUT Red_light



Neue Symbole könne auch während des Programmierens hinzugefügt werden. Dazu eine Programm-Zeile mit Mnemo-Code 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, 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).

7.4.2 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 *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.

7.4.3 Benutzen der Symbole aus der *Symbols*-Liste im IL Programm

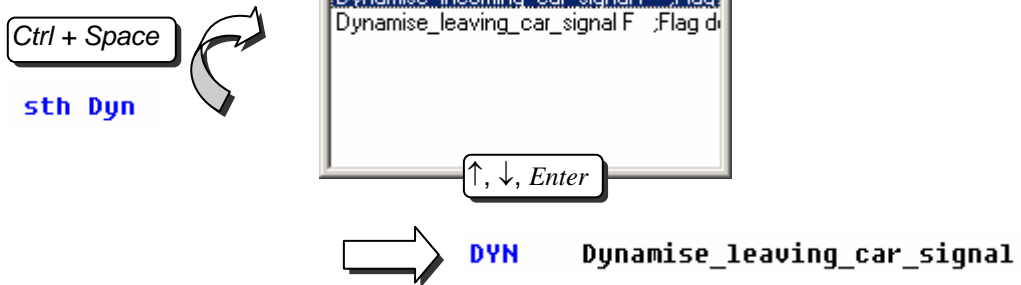
Nach erfolgter Programmierung können die im *Symbols*-Fenster festgelegten Symbole auf verschiedene Arten benutzt werden:

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.

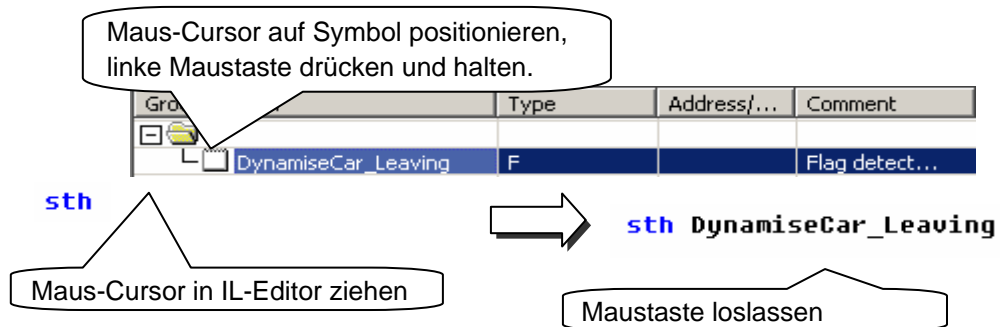
Symbol eingeben durch selektive Suche

Group/Symbol	Type	Address...	Comment
Car_incoming	Input	0	Gets high when a
Car_outgoing	Input	1	Gets high when a
Red_light	Output	32	Stops new cars a
Number_of_free_slots	Counter		Counts the numb
Dynamise_incoming_car_signal	F		Flag detects the
Dynamise_leaving_car_...			detects the



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.

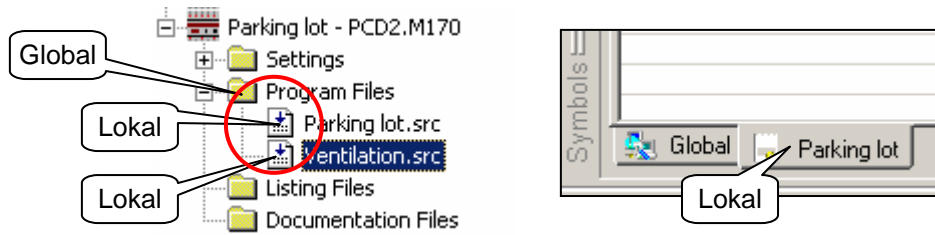
Symbol eingeben mit Drag-and-Drop



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. Maus-Cursor in den IL-Editor ziehen und Maustaste loslassen. Das ausgewählte Symbol wird automatisch an den vom Maus-Cursor angezeigten Platz eingefügt.

7.4.4 Lokale und Globale Symbole

Das Symbols-Fenster besteht aus zwei Ordnern: *Global* und *Lokal*



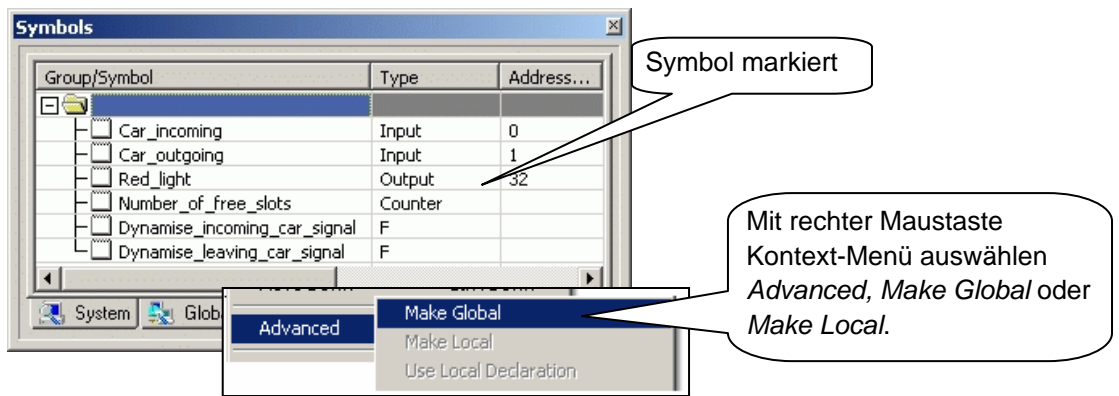
Definition

Lokale Symbole erscheinen in dem Ordner, der den Namen der Datei trägt die diese Symbole benutzt (*Parking lot.src*).

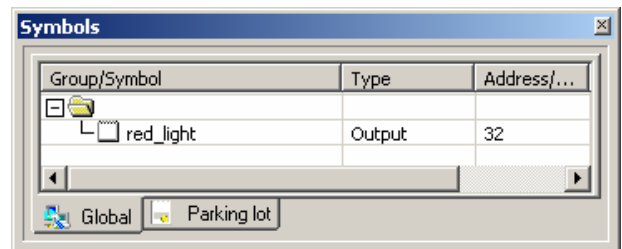
Globale Symbole im *Global* Ordner können von allen Dateien der CPU benutzt werden (*Parking lot.src* und *Ventilation.src*).

Lokal-Ordner in den Global-Ordner verschieben

Symbole im *Symbols*-Fenster können, wenn gewünscht, vom Lokal-Ordner in den Global-Ordner verschoben werden und umgekehrt.



Das Symbol wird in den *Global*- oder *Lokal*-Ordner verschoben



Jedes neue im IL Editor definierte Symbol, wird entweder im globalen oder im lokalen Ordner abgelegt. Wo, hängt von den Einstellungen in der *Global symbols* Option ab. Siehe Kontext-Menü *Advanced, Options* des *Symbols*-Fenster.

7.5 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*.

7.5.1 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

7.5.2 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.

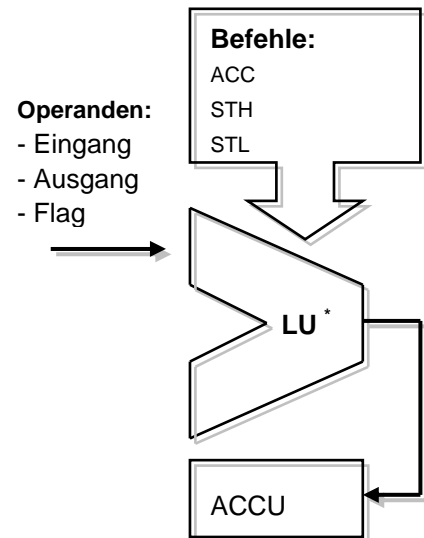
Binäre Befehle betreffen immer den Akkumulator. Einige binäre Befehle wirken auf den Status des Akkumulators:

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..



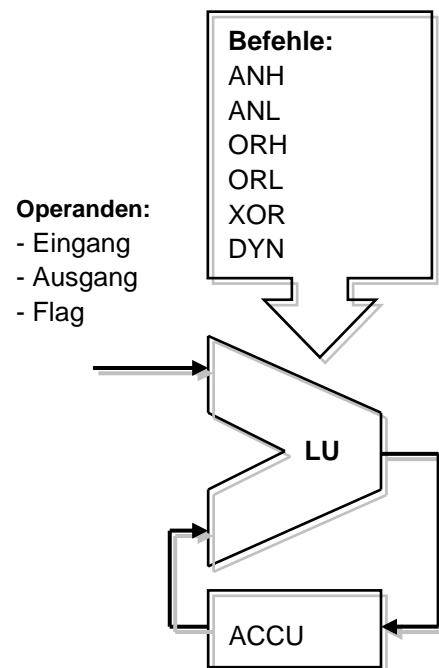
Weitere Befehle lesen den Status des Akkumulators, um binäre Funktionen auszuführen und das Ergebnis an den Akkumulator zurück zu schicken:

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.

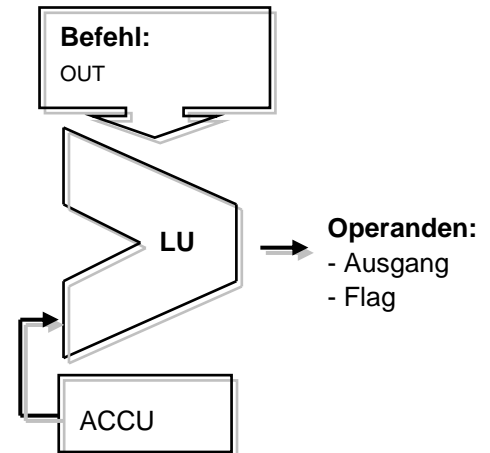
ORH F 100
Liest Akkumulator Status und führt eine logische ODER Funktion mit dem Status des Flags 100 aus. Der Akkumulator wird auf den Wert des Ergebnisses gesetzt.

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.



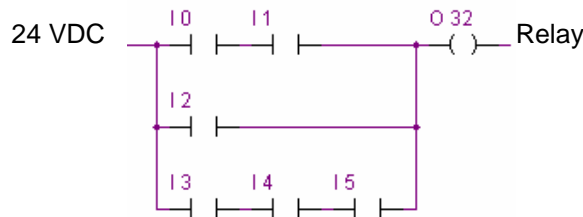
Das Ergebnis aus einer binären Gleichung wird immer im Akkumulator gespeichert. Der *OUT* Befehl kopiert den Inhalt des Akkumulators an einen Ausgang oder ein Flag:

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: $O32 = I0 \cdot I1 + I2 + I3 \cdot I4 \cdot I5$
 Die Gleichung wird auch durch diese Schaltung veranschaulicht:



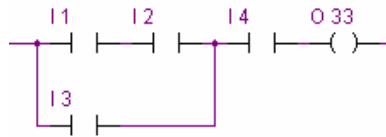
Eine binäre Gleichung beginnt immer mit einem *STH* oder *STL* Befehl, dem dann die notwendigen *ANH* (*), *ORH* (+), *XOR* Funktionen folgen. Beachten, dass der *ORH* Befehl Priorität vor dem *ANH* Befehl hat. Jeder *ORH* Befehl kennzeichnet den Beginn einer neuen Kontaktreihe in der obigen Schaltung. 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  I0          ;Status von Eingang I 0 wird an Akkumulator
kopierte: Accu = I0
ANH  I1          ;AND-Funktion zwischen Akkustatus und
Status von Eingang 1:Accu = I0*I1
ORH  I2          ;OR-Funktion zwischen Akkustatus und Status
von Eingang 2:Accu= I0*I1+I2
ORH  I3          ; Accu = I0*I1+I2+I3
ANH  I4          ; Accu = I0*I1+I2+I3*I4
ANH  I5          ; 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  I1           ;Erste Gleichung
ANH  I2
ORH  I4
OUT  F0           ;Ergebnis der Funktion in Klammern: F0
      =(I1*I2+I4)

STH  F0           ;Zweite Gleichung
ANH  I3
OUT  O33          ;Endergebnis: O33 = F0*I3
ECOB
    
```

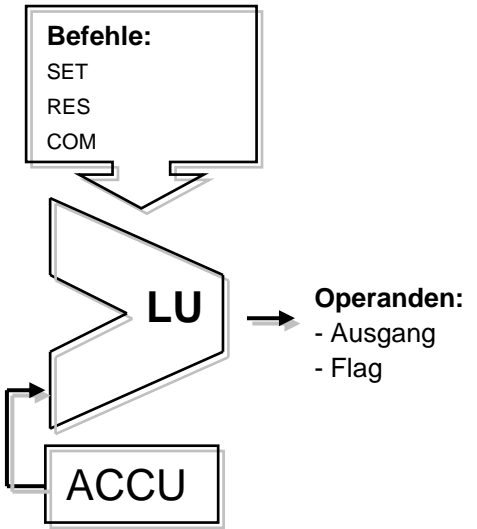
Mit weiteren binären Befehlen kann der Akkumulator dazu benutzt werden, den Zustand von Ausgängen oder Flags zu ändern. Jeder Befehl unterstützt eine andere Funktion.

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.

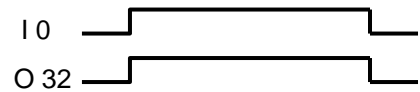
COM O 33
 Ist der Akkumulator Status high, wird Ausgang 33 auf high invertiert. Andernfalls verharrt der Ausgang im gegenwärtigen Zustand.



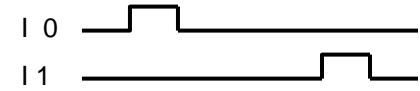
Beispiel:

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.

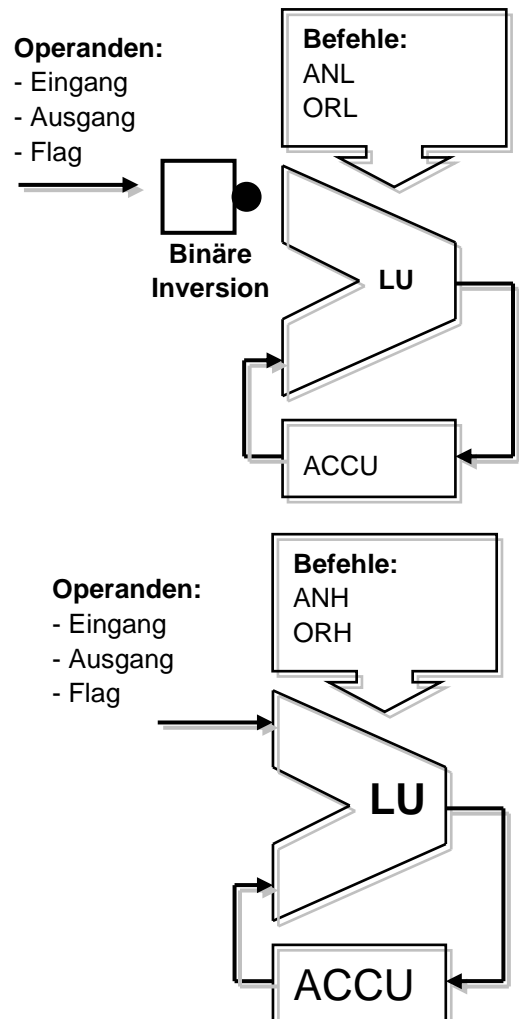
Beispiele:

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.



7.5.3 Dynamisierung

Binäre Befehle benutzen in der Regel den binären low oder high Zustand zur Ausführung einer binären Funktion oder Statusänderung eines Ausgangs oder Flags.

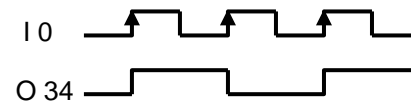
Manchmal interessiert aber nicht der binäre low oder high Zustand, sondern der Übergang von low zu high (z. B. um einen Counter hochzuzählen).

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

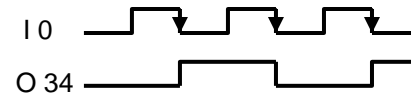
```
STH I 0
DYN F 3
COM O 34
```



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.

7.5.4 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

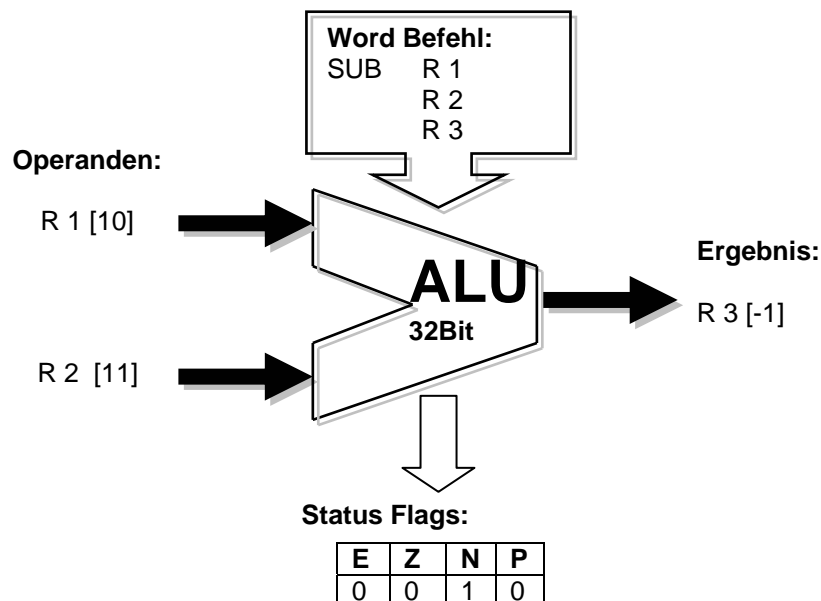
Quadratwurzel aus einer negativen Zahl

Fehler an der Kommunikations- Schnittstelle (SASI Befehl)

Beispiel:

Status Flags nach einer Subtraktion

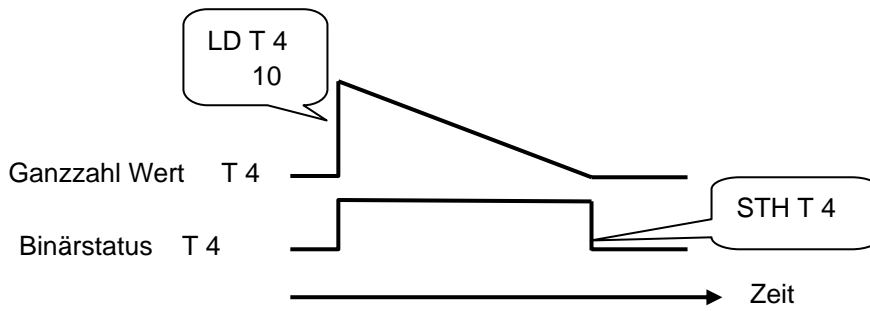
Status Flags werden gesetzt, abhängig vom Ergebnis einer Subtraktion ($R 3 = R 1 - R 2$). Register Werte stehen in eckigen Klammern []. Das Ergebnis der Subtraktion ist negativ: einzig Flag N ist gesetzt.



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

7.5.5 Wort-Befehle für Timer



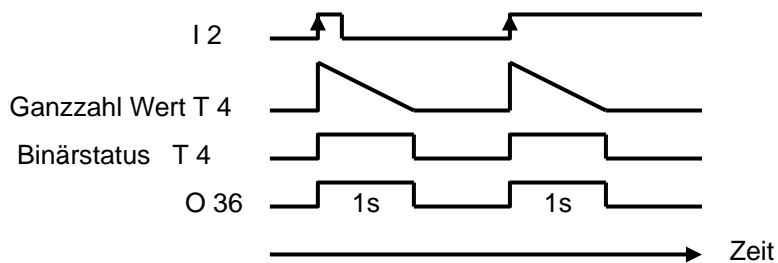
Timer enthalten zwei Werte: den Wert für die feste Verzugszeit und den Binärstatus. Zum Einfügen einer Verzugszeit, die Zeit als positiven ganzzahligen Wert laden entsprechend der Länge der Verzugszeit in Zehntelsekunden. Die Steuerung dekrementiert diesen Wert automatisch, bis Null erreicht ist. Der Binärstatus des Timers während des Dekrementierens ist high und geht auf low, wenn die Zeit den Wert Null erreicht.

<p>Laden einer Verzugszeit</p> <p>LD T 4</p> <p>Ist der Akkumulator Status high, wird Timer T 4 mit einer Konstante 10 geladen. Andernfalls behält der Timer den gegenwärtigen Wert bei.</p>	<p>Lesen des Timer-Zustands</p> <p>Use a binary instruction, such as:</p> <p>STH T 4 , ANH T 4, ORH T 4, ...</p>
---	---

Beispiel:

Sende bei jeder aufsteigenden Flanke an Eingang 2 einen Ein-Sekunden-Impuls an Ausgang 36.

Statusdiagramm:



Programmierung:

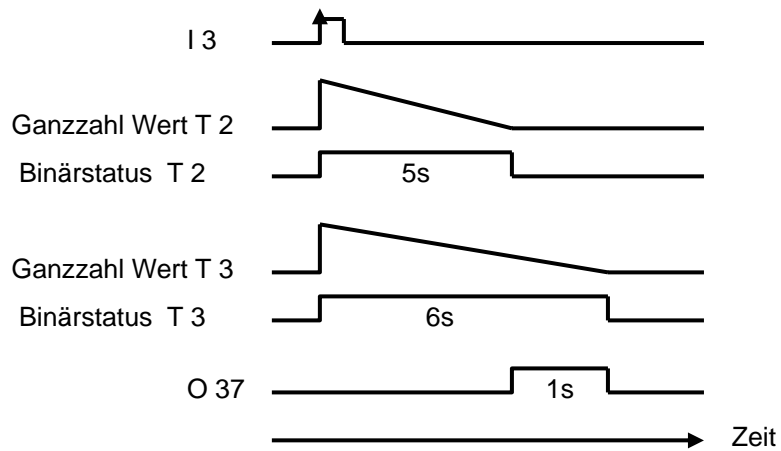
```

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 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-Diagramm:

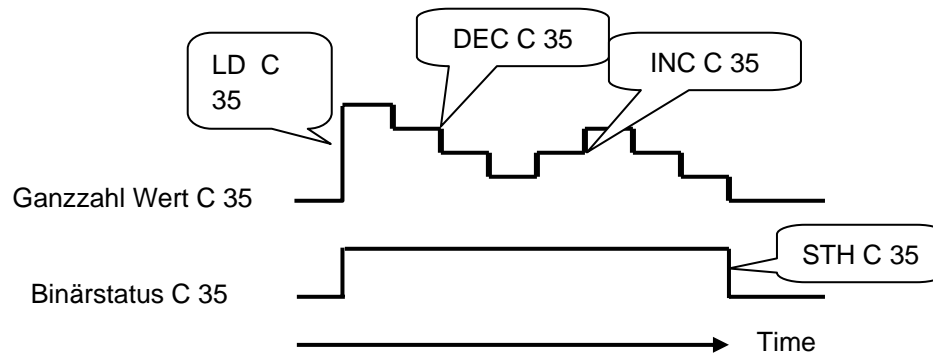
**Programmierung:**

```

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

```

7.5.6 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.

<p>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.</p>	<p>Lesen des Counter-Zustands Binär-Befehle einsetzen, wie: STH C 35, ANH C 35, ORH C 35, ...</p>
<p>Counter inkrementieren INC C 35 Ist der Akkumulator Status high, wird Counter 35 um eine Einheit inkrementiert. Andernfalls behält er den gegenwärtigen Wert bei.</p>	<p>Counter dekrementieren DEC C 35 Ist der Akkumulator Status high, wird Counter 35 um eine Einheit dekrementiert. Andernfalls behält er den gegenwärtigen Wert bei.</p>

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.

7.5.7 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
```


7.5.8 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 (**Positive**, **Negativ**, **Zero**, **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 O 33 ; R 0 = R 1
 ACC N
 OUT O 34 ; R 0 < R 1

7.5.9 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

7.5.10 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-flt point	Wandeln fltg point-integer
IFP R 0 ; integer -> float 0 ; exponent	FPI R 0 ;float ->integer 0 ; exponent

7.5.11 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	SEt Index register	Lädt das Index Register mit der Konstante 10
INI K 99	IN crement Index register	Inkrementiert das Index Register und setzt Akkumulator Status high, so lang das Index Register <= K 99 ist
DEI K 5	DE crement Index register	Dekrementiert das Index Register und setzt Akkumulator Status high, so lang das Index Register >= K 5
STI R 0	ST ore Index register	Kopiert Index Register nach Register 0
RSI R 0	Re Store Index 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

7.5.12 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
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

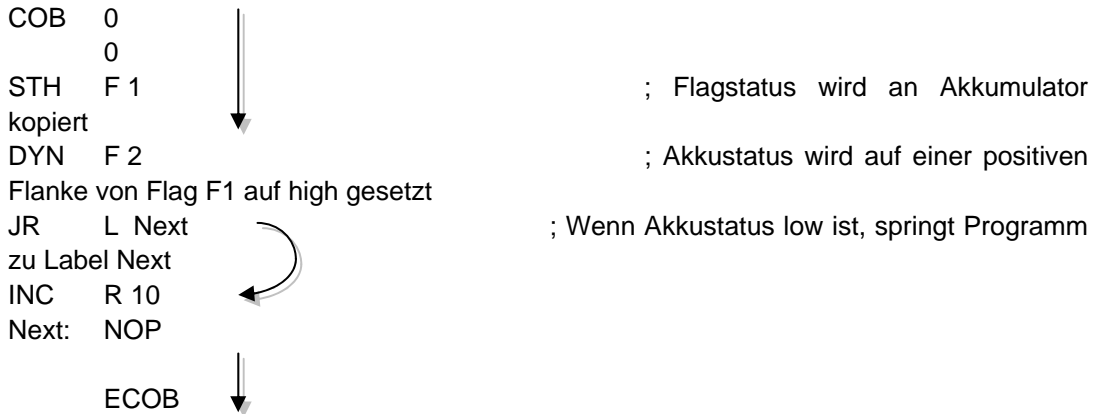
Mnemo-Kode	Label	Beschreibung
JR		Sprung immer auf Zeile in Übereinstimmung zum Label
JPD		
JPI		

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

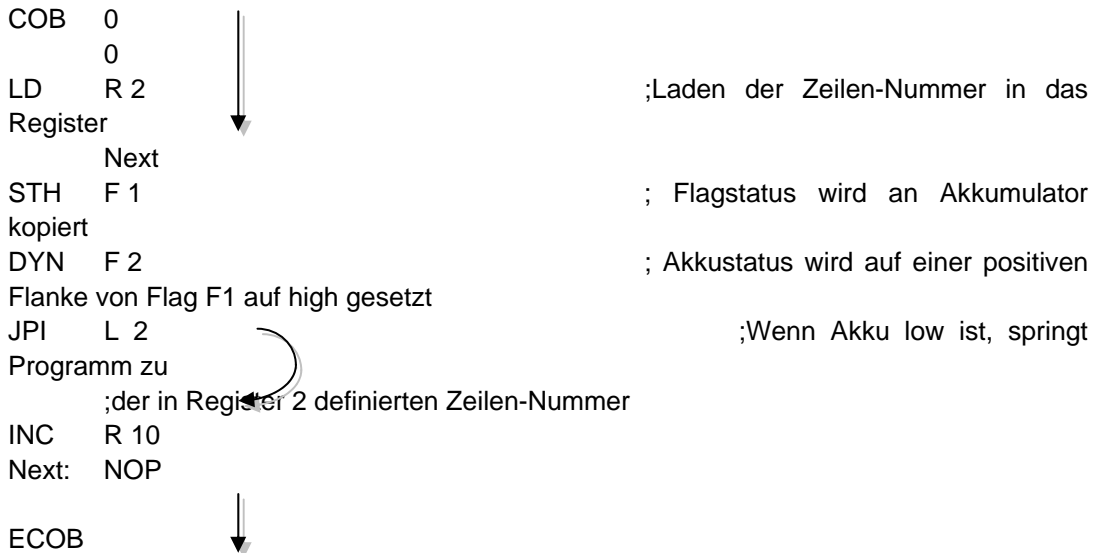
Beispiel: Impulse eines binären Eingangs binär zählen mit einem Register (relativer Sprung)

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.



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).

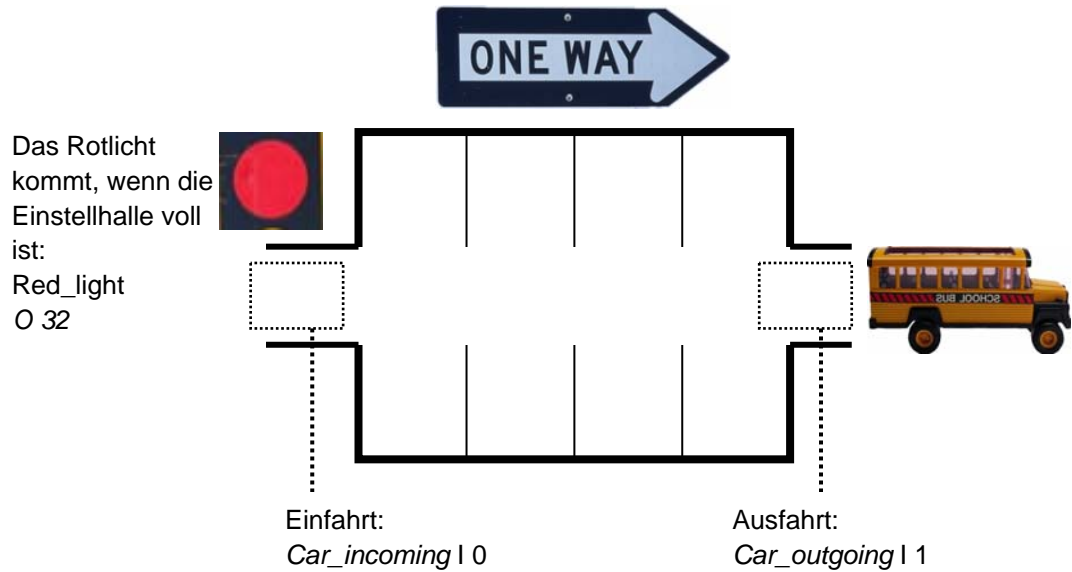
Beispiel: Lösung mit indirektem Sprung



Der indirekte Sprung ist sehr flexibel. Das Programm passt die Zeilen-Nummer, zu der gesprungen werden soll, selbständig an.

7.6 Editieren eines ersten Anwender Programms

Zählen der verbleibenden Parkplätze in einer Einstellhalle mit 8-Parkplätzen und einschalten eines Rotlichts, wenn die Einstellhalle voll ist.

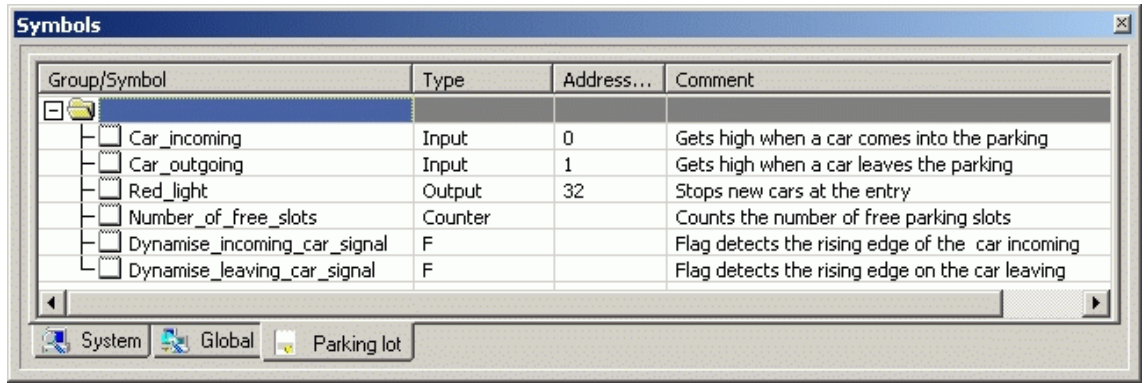


Beim Einschalten der Spannungsversorgung wird angenommen, dass alle Parkplätze frei sind. Deswegen wird beim initialisieren der Zähler für die freien Parkplätze mit dem Wert 8 geladen. Dies wird nur einmal beim Aufstarten der PCD durchgeführt und daher im Kaltstart-Block XOB 16 programmiert. Die verbleibenden 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.



```

;-----
Kaltstart-Organisationsblock
;-----

```

```

XOB 16 ; Programm wird beim Start ausgeführt
ACC H
LD Number_of_free_slots ; Counter für freie Slots wird initialisiert
8 ; mit dem Wert 8 (unbedingt)
EXOB ; Ende des Start Programms

```

```

;-----
; Zyklischer Organisationsblock
;-----

```

```

COB 0 ; Zyklisches Programm
0 ; Keine Überwachungszeit

STH Car_incoming ; Ein Auto fährt in den Parkbereich
DYN Dynamise_incoming_car_signal ; Auf der positiven Flanke des Eingangssignals
DEC Number_of_free_slots ; Anzahl an freien Slots wird dekrementiert

```

```

;-----

```

```

STH Car_outgoing ; Ein Auto verlässt den Parkbereich
DYN Dynamise_leaving_car_signal ; Auf der positiven Flanke des Ausgangssignals
INC Number_of_free_slots ; Anzahl an freien Slots wird inkrementiert

```

```

;-----

```

```

STL Number_of_free_slots ; Wenn keine Slots frei sind (Counter-Status =
Low)
OUT Red_light ; Licht wird auf rot gestellt

ECOB ; Ende des zyklischen Programms

```

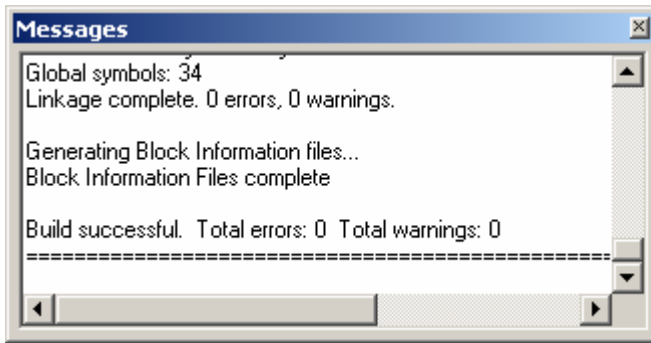
7.6.1 Verarbeiten (build) des Programms



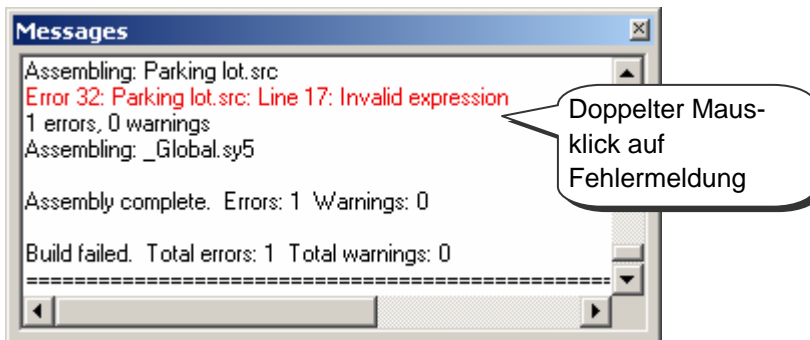
Build All

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 *CPU Build* Menü aktiviert oder den *Build* 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 successful. 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.



Fehler ist rot markiert

```
STH Car_incomingZ
DYN Dynamise_incoming_car_signal
DEC Number_of_free_slots
```

Fehlerkorrektur

```
STH Car_incoming
DYN Dynamise_incoming_car_signal
DEC Number_of_free_slots
```


7.7 Laden des Programms in die PCD



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.

Download Program

Wenn Kommunikationsprobleme auftreten, die Konfiguration nochmals in *Settings Online* und *Settings Hardware* sowie die Verbindungskabel zwischen PC und der PCD (PCD8.K111, USB) überprüfen.

7.8 On-line testen (debuggen) des Programms

Die Programme sind in der ersten Version oft noch mangelhaft und sollten daher sorgfältig ausgetestet werden. Dazu ist derselbe Editor zu verwenden wie für die Programmierung.

7.8.1 Anzeigen des Programm-Kodes



Show Hide Code

Mit dem *View Code* Menü oder dem *Show/Hide Code* Knopf kann der verarbeitete Programm-Kode auf einer einzigen Seite angezeigt werden.

Die weissen Zeilen zeigt den Original-Quell Code, mit Symbolen und Kommentare.

Die grauen Zeilen zeigen den assemblierten Code, mit den Adressen der Operanden und den Programm Zeilennummern.

```

Parking lot.src
; Cyclical Organisation Block
;-----
          COB      0          ; Cyclical program
          0          ; No supervision time
000007 COB      0
000008 COB      0
000010 NOP

          STH      Car_incoming |          ; A car comes into the parking:
000011 STH      I|0 0
          DYN      Dynamise_incoming_car_signal ; On the positiv flank of incoming si
000012 DYN      F 7502
          DEC      Number_of_free_slots      ; Decrement the number of free park:
000013 DEC      C 1400

;-----

          STH      Car_outgoing          ; A car leaves into the parking:
000014 STH      I|0 1
          DYN      Dynamise_leaving_car_signal ; On the positiv flank of outgoing si
000015 DYN      F 7503

          INC      Number_of_free_slots      ; Increment the number of free park:
000016 INC      C 1400

```

7.8.1 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.

000015	DYN	Dynamise_leaving_car_signal	; On the positiv flank of outgoing signal
	DYN	F 7503	
000016	INC	Number_of_free_slots	; Increment the number of free parking slots
	INC	C 1400 [8]	A0 Z0 N0 P1 E0 IX0000

7.8.2 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:



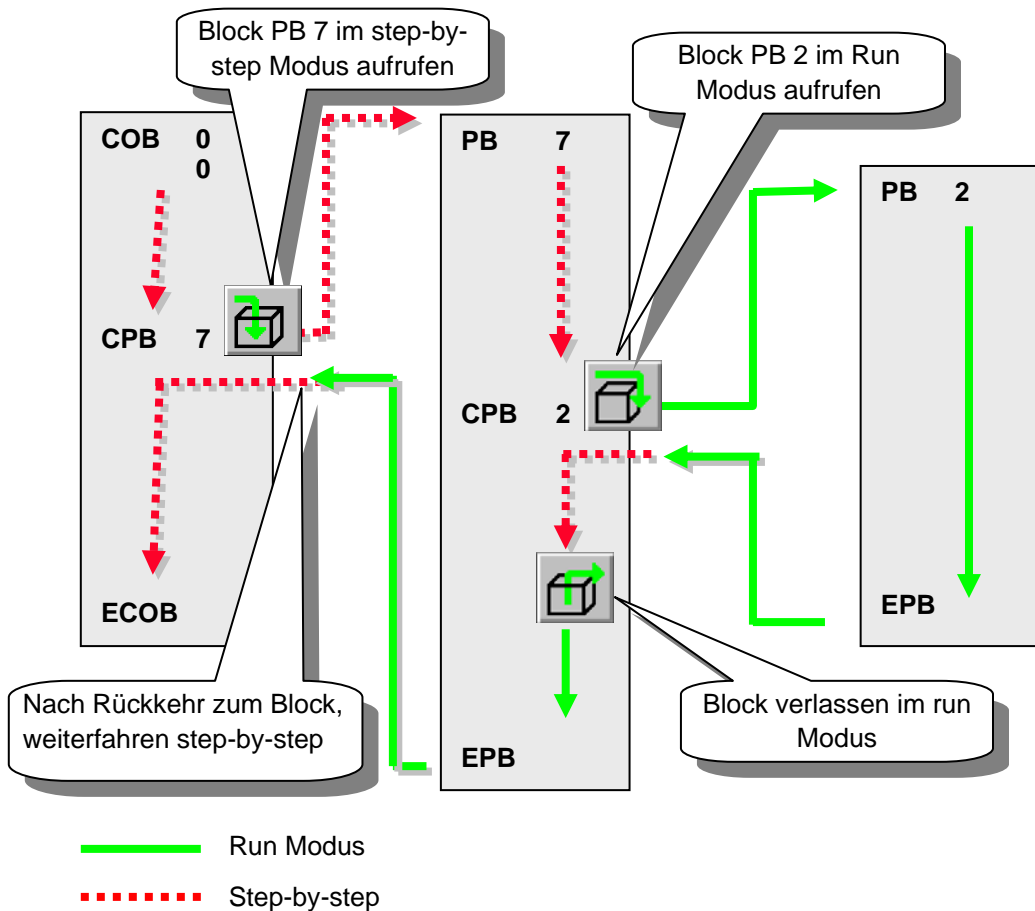
In den Block eintreten und durchgehen



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 parking:
000014 STH   I|0 1           [0]           A0 Z0 N0 P1 E0 IX0000
    
```

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.

7.8.3 Anhalte-Punkte (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.



Set/Clear
Breakpoint

Type:	Address:	Condition:	Value:
Counter	1400	>	4

History:			
Counter	1400	>	4
Output	32	=	0

Im Fenster oben den Symboltyp und Adresse/Nummer 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.

Anhalte-Punkt auf einer Programmzeile

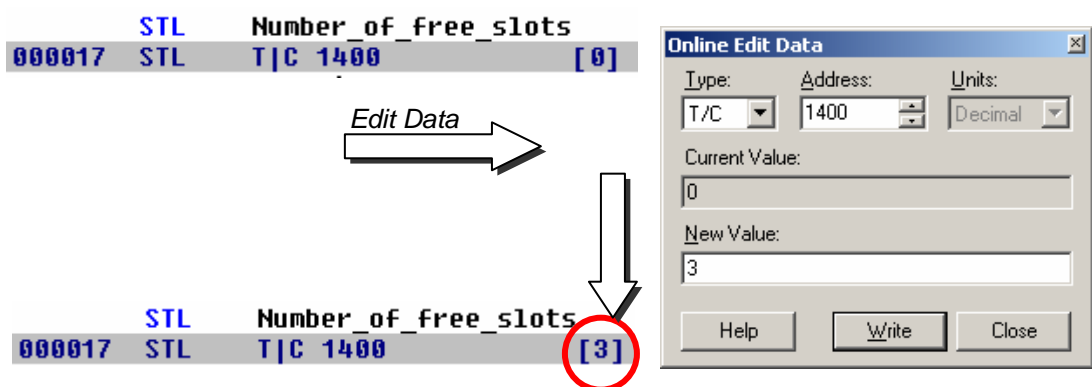
Wählen Sie eine Programmzeile aus und führen Sie das Menü *Online, Run to, Cursor* aus, um das Programm an der gewählten Zeile anzuhalten. Setzen Sie Ihre Arbeit im Step-by-Step-Modus fort.

7.8.4 On-line Änderung am Programm

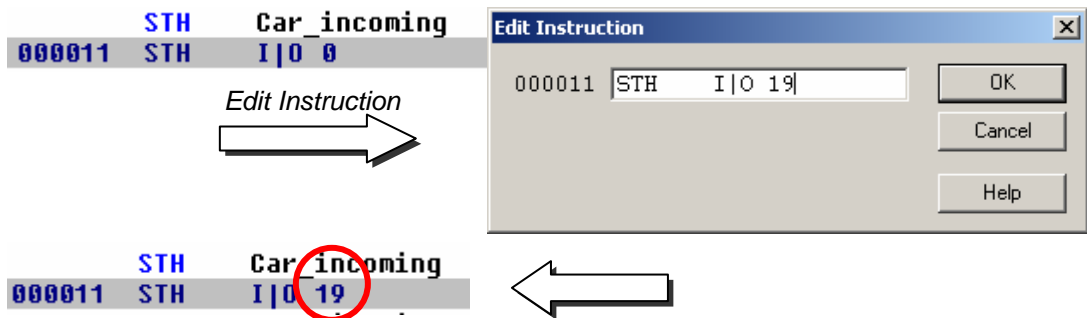
Beim Testen eines Programm step-by-step, ist es oft hilfreich Zustände/Werte bestimmter Operanden/Symbole zu verändern, um damit das Programmverhalten unter verschiedenen Bedingungen zu prüfen.

Eine der aktiven Zeilen (grau) mit der Maus auswählen und mit rechtem Mausklick das Kontext-Menü anzeigen.

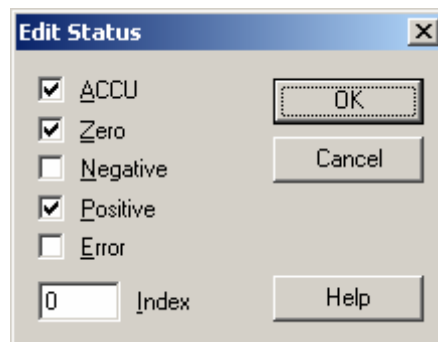
Im *Edit Data* Kontext-Menü kann der Zustand/Wert des ausgewählten Operanden geändert werden.



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.



7.8.5 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.

The screenshot shows the 'Watch Window' icon on the left and a 'Group/Symbol' tree on the right. A blue arrow points from the tree to the 'Watch Window' table. The table has columns: Symbol, Address, Value, Modify Value, and Symbol Comment. The 'Number_of_free_slots' row is highlighted. A blue arrow points from the 'Value' column to the number '8'. A speech bubble points to the 'Watch Window' icon with the text '4. Start/Stop Monitoring'.

1. Maus-Cursor im Symbol Icon positionieren und linke Maustaste drücken

2. Linke Maustaste gedrückt halten und Symbol ins Watch Window

3. Symbol mit Zustand/Wert und Kommentar

4. Start/Stop Monitoring

Zum ändern des Zustands/Wertes eines Symbols ist folgendermassen zu verfahren:

The screenshot shows the 'Watch Window' icon on the left and a 'Parking lot.5ww * [Parking lot]' window on the right. The table has columns: Symbol, Address, Value, Modify Value, and Symbol Comment. The 'Number_of_free_slots' row is highlighted, and the 'Value' column contains the number '5'. A speech bubble points to the 'Value' column with the text '2. Platzieren Sie den Mauszeiger auf dem zu editierenden Wert. Führen Sie mit der linken Maustaste einen Doppelklick durch und geben Sie den neuen Wert ein.' A blue arrow points from the 'Value' column to the 'Modify Value' column. A speech bubble points to the 'Watch Window' icon with the text '3. Download Values'.

1. Start/Stop Monitoring

2. Platzieren Sie den Mauszeiger auf dem zu editierenden Wert.
Führen Sie mit der linken Maustaste einen Doppelklick durch und geben Sie den neuen Wert ein.

3. Download Values

7.9 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

Mit analogen Ein- oder Ausgängen muss jedoch eine Erfassungsroutine für den Analogwert benutzt werden. Es gibt verschiedene Routinen für verschiedene Analogmodularten. Beschreibungen hierzu finden Sie im Hardwarehandbuch Ihres PCD.

7.9.1 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.

7.9.2 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 Register 12	Ausgangsspannung [V]	Ausgangsstrom [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.ch>

Inhalt

8	ZUSÄTZLICHE WERKZEUGE	3
8.1	Zusammenfassung	3
8.2	Datenübertragungsprogramm	4
8.2.1	Einsatz der Datenübertragung	4
8.2.2	Datenübertragung starten	4
8.2.3	Datensicherung mit Quick Data Upload	4
8.2.4	Daten zurückspeichern	5
8.2.5	Datensicherung mit Hilfe einer Script-Datei	5
8.2.6	Zurückspeichern mit Hilfe einer Script-Datei	6
8.2.7	Hochlade-Optionen	6
8.2.8	Datensicherung mit Befehlszeilen	7
8.3	Watch window	8
8.3.1	Öffnen des <i>Watch Windows</i>	8
8.3.2	Daten zum <i>Watch Window</i> hinzufügen	9
8.3.3	On-line Anzeige der Daten	9
8.3.4	On-line Änderung von Daten	10
8.3.5	Anzeigeformat	10
8.4	On-line Konfigurator	11
8.4.1	Off-line Konfigurator	11
8.4.2	On-line Konfigurator	11
8.4.3	On-line Konfigurator-Fenster	11
8.4.4	PCD-Zeit einstellen	12
8.4.5	Das History	12
8.5	EPROM Programmierung	13
8.6	Aktualisieren der Firmware. (<i>Firmware Downloader</i>)	14
8.7	Anwender Menüs	15

8 Zusätzliche Werkzeuge

8.1 Zusammenfassung

PG5 stellt ihnen mehrere zusätzliche Dienstprogramme für eine Vielfalt von Anwendungen zur Verfügung.

8.2 Datenübertragungsprogramm

8.2.1 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:

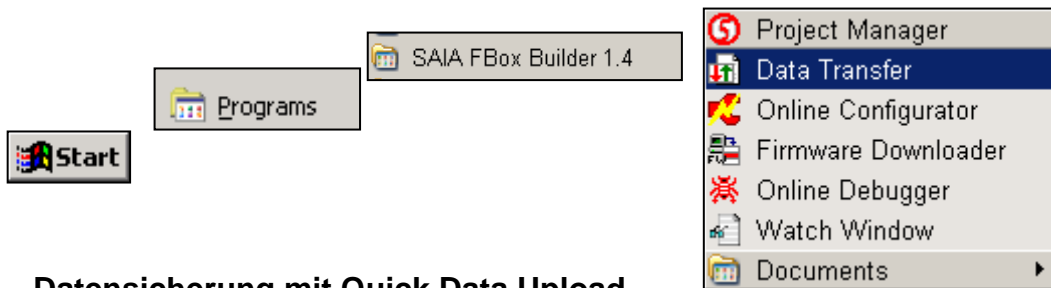
Eingänge, Ausgänge, Flags, Timer, Counter, Registers, Daten und Textblöcke.

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.

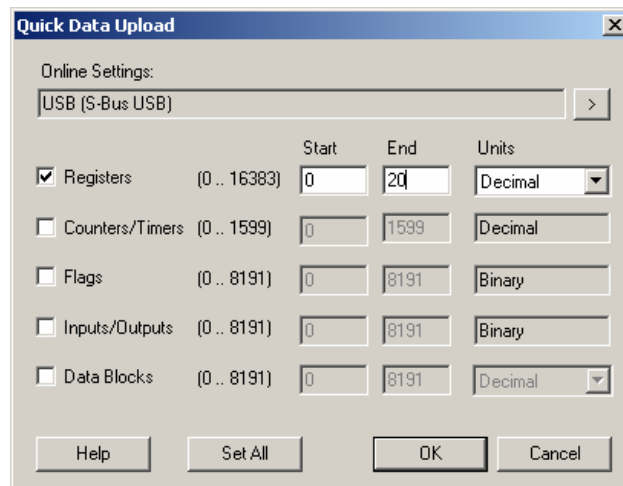
8.2.2 Datenübertragung starten

Aufstarten des Programms mit:

Start --> Programs --> SAIA FBox Builder 1.4 --> Data Transfer



8.2.3 Datensicherung mit Quick Data Upload

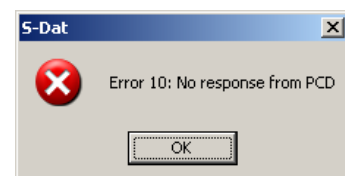


Im Menü wählen *Online, Quick Data Upload ...* oder den *Quick Data Upload* Knopf drücken. Es erscheint das Fenster oben.

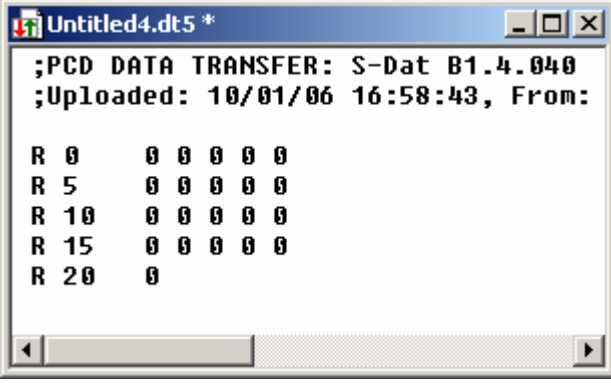
Datentypen auswählen, die gesichert werden sollen, deren Adressbereich und Zahlenformat eingeben.

Mit OK werden die Daten geladen.

Kommt die nebenstehende Meldung, sind die Kommunikationsparameter mit *Online, Settings Online* zu prüfen und die korrekte Kabelverbindung zwischen PC und PCD sicherzustellen.



Das Laden der Daten dauert einen Moment, dann kommt folgende Anzeige:



```

;PCD DATA TRANSFER: S-Dat B1.4.040
;Uploaded: 10/01/06 16:58:43, From:

R 0      0 0 0 0 0
R 5      0 0 0 0 0
R 10     0 0 0 0 0
R 15     0 0 0 0 0
R 20     0

```

Die Datei kann mit neuen Werten überschrieben und dann im Menü *File, Save* oder mit dem *Save* Knopf gesichert werden.

8.2.4 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.

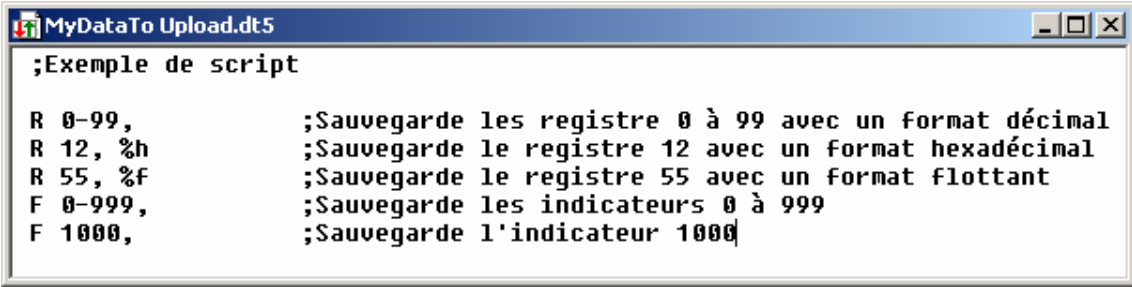


*Download
To PCD*

Zurückspeichern zum PCD Speicher mit *Online, Download Data to the PCD* oder dem *Download* Knopf.

8.2.5 Datensicherung mit Hilfe einer Script-Datei

Falls nötig, kann die Liste der zu sichernden Daten auch in einer Script-Datei geändert werden. Beispiel:



```

;Exemple de script

R 0-99,           ;Sauvegarde les registre 0 à 99 avec un format décimal
R 12, %h          ;Sauvegarde le registre 12 avec un format hexadécimal
R 55, %f          ;Sauvegarde le registre 55 avec un format flottant
F 0-999,         ;Sauvegarde les indicateurs 0 à 999
F 1000,          ;Sauvegarde l'indicateur 1000

```



*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*.

8.2.6 Zurückspeichern mit Hilfe einer Script-Datei

Mit einer Script-Datei können auch Daten die bereits gespeichert waren, geändert werden. Beispiel:



```

MyDataToDownload.dt5 *
;Exemple de script
R 0-99, 0 ;Charge les registre 0 à 99 avec zéro
R 12, 32h ;Charge le registre 12 avec 32 hexadécimal
R 55, 64.3 ;Charge le registre 55 avec 64.3 flottant
F 0-999, 0 ;Charge les indicateurs 0 à l'état bas
F 1000, 1 ;Charge l'indicateur 1000 à l'état haut

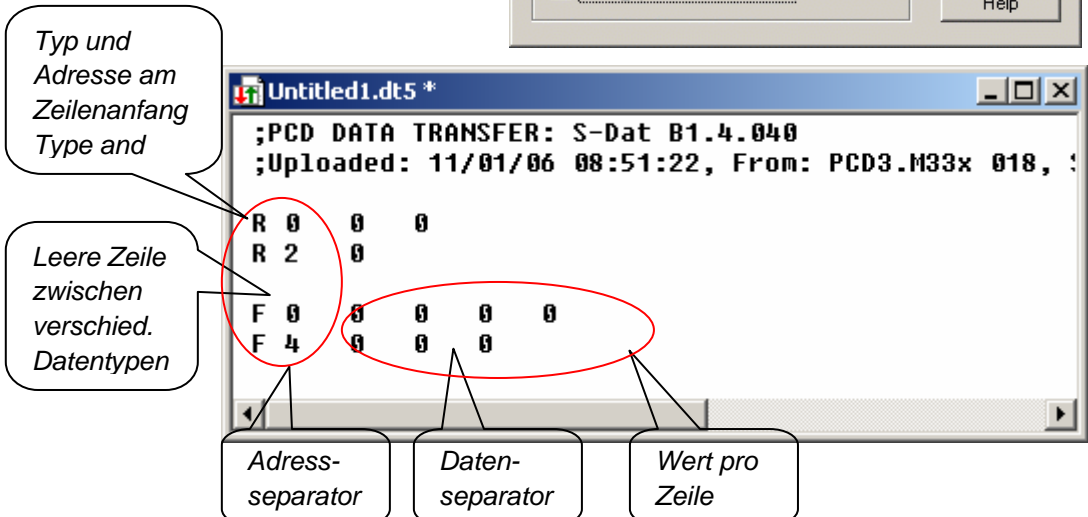
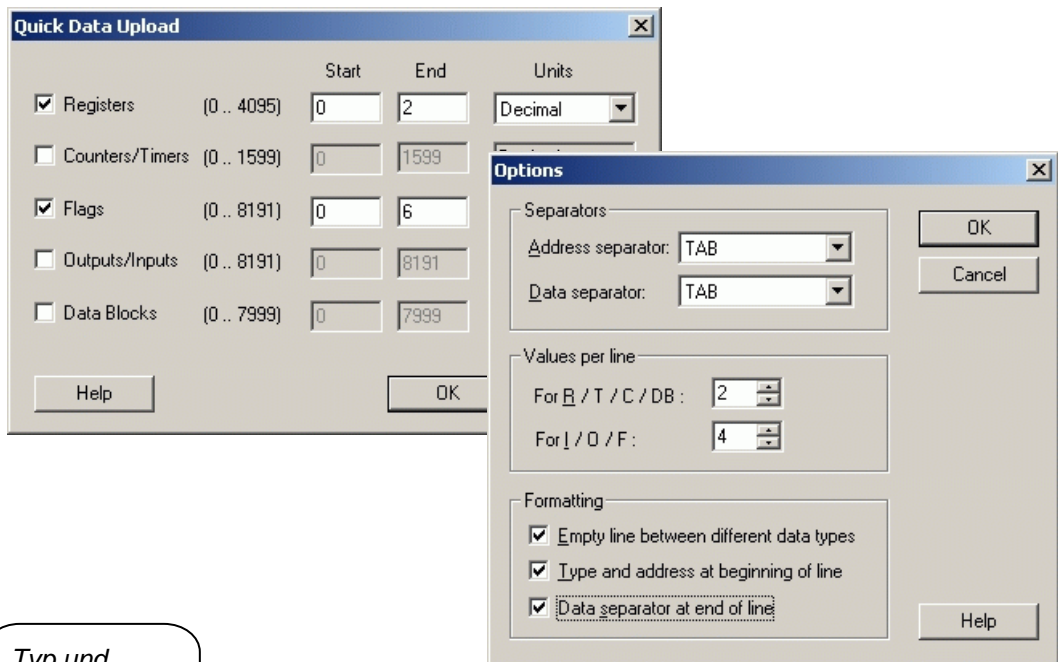
```

Zurückspeichern der Script-Datei in die PCD mit *Online, Download Data to PCD* im Menü oder den *Download* Knopf drücken.

8.2.7 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.



8.2.8 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 Name der Datei zum speichern/rückspeichern

_of_file

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=nn nnn = Wert pro Zeile für R,T,C,DB (1..256, default = 5)

n nnn = Wert pro Zeile für I,O,F (1..256, default = 10)

/I nnn = Adressseparator (TAB,SPACE,COMMA,COLON, default= TAB)

=nnn nnn = Datenseparator (TAB,SPACE,COMMA,COLON, default= TAB)

/A=nn

n

/D=nn

n

Beispiel:

sdat5 MyDatas.dt5 R0-99 R12H R55F F0-999 F1000 /R005 /I010

8.3 Watch window

Das *Watch Window* wird ein hilfreiches Werkzeug zum testen, ändern und betrachten von Symbol-Zuständen und Anwendungen.

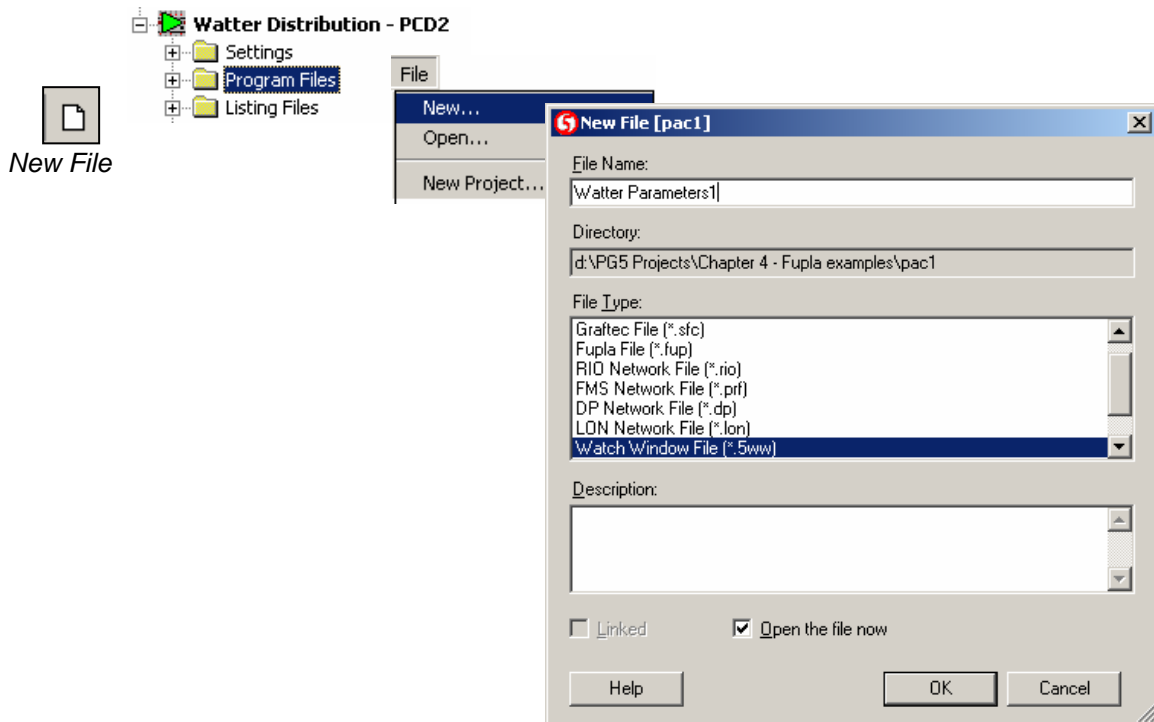
8.3.1 Öffnen des *Watch Windows*



Watch Window

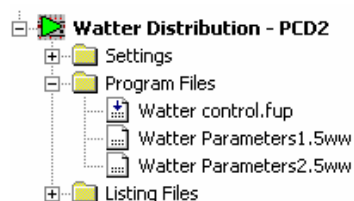
Das *Watch Window* wird durch Auswahl im Menü *View, Watch Window* oder durch drücken des *Watch Window* Knopfes angezeigt.

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.



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.



8.3.2 Daten zum Watch Window hinzufügen

Symbole aus dem Programm oder aus dem Symbol Editor ins Watch Window ziehen.

1. Maus-Cursor im Symbol Icon positionieren und linke Maustaste drücken

2. Linke Maustaste gedrückt halten und Symbol ins Watch Window

3. Symbol mit Zustand/Wert und Kommentar

4. Start/Stop Monitoring

Symbol	Address	Value	Modify Value	Symbol Comment
Car_incoming	I 0	0		Gets high when a car comes i...
Car_outgoing	I 1	0		Gets high when a car leaves t...
Red_light	O 32	0		Stops new cars at the entry
Number_of_free_slots	C 1400	8		Counts the number of free p...

Die Symbole können direkt im Fenster editiert werden

1. Start/Stop Monitoring

2. Platzieren Sie den Mauszeiger auf dem zu editierenden Wert. Führen Sie mit der linken Maustaste einen Doppelklick durch und geben Sie den neuen Wert ein.

3. Download Values

Symbol	Address	Value	Modify Value	Symbol Comment
Car_incoming	I 0	0		Gets high when a car comes i...
Car_outgoing	I 1	0		Gets high when a car leaves t...
Red_light	O 32	0		Stops new cars at the entry
Number_of_free_slots	C 1400	5		Counts the number of free p...

8.3.3 On-line Anzeige der Daten

Zum betrachten der Werte/Zustände eines Symbols den Go On/Offline Knopf drücken.

Go On/Offline

Symbol	Address	Value	Module	Comment
DailyTimer	O 32	0	Daily Timer	Daily Timer
HMS	R 2003	113245	Daily Timer	PCD Clock with curr
OFFTIME	R 2004	182000	Daily Timer	Switch off time
ONTIME	R 2005	53000	Daily Timer	Switch on time

8.3.4 On-line Änderung von Daten

Die Werte der Symbole können On-line geändert werden.

1. Maus-Cursor auf den Wert positionieren. Rechte Maustaste drücken.

2. Wert ändern

Symbol	Address	Value	Module	Comment
DailyTimer	O 32	1	Daily Timer	Daily Timer
HMS	R 2003	113245	Daily Timer	PCD Clock with curr
OFFTIME	R 2004	182000	Daily Timer	Switch off time
ONTIME	R 2005	53000		

Type:	Address:	Units:
R	2005	Decimal

Current Value: 53000
New Value: 91000

Buttons: Help, Write, Close

8.3.5 Anzeigeformat

Das Anzeigeformat der Werte kann den jeweiligen Erfordernissen angepasst werden.

Beispiel: Anzeige des Registers R 2004 hexadezimal

Address	Value
O 32	0
R 2003	113245
R 2004	182000
R 2005	53000

- Edit Data... Ctrl+E
- Units
 - Decimal Ctrl+D
 - Floating Point Ctrl+F
 - Hex Ctrl+H**
 - ASCII
 - Binary

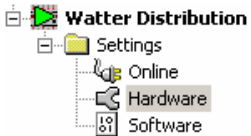
Address	Value
O 32	0
R 2003	113245
R 2004	0002C6F0H
R 2005	53000

8.4 On-line Konfigurator

PG5 stellt zwei Konfigurations-Werkzeuge zur Verfügung:

- Den Off-line Konfigurator, auf den in den *Hardware Settings* im Projekt Manager zugegriffen werden kann.
- Den On-line Konfigurator, auf den im Menü *Tools, Online Configurator*, oder mit dem *Online Configurator* Knopf zugegriffen werden kann.

8.4.1 Off-line Konfigurator



Konfiguriert den Speicher, die Kommunikations-Parameter und das PCD Passwort. Diese Informationen werden in einer PG5 Projekt Datei gespeichert. Der Anwender muss den *Download* Knopf benutzen, um die Konfiguration in den PCD Speicher zu laden.

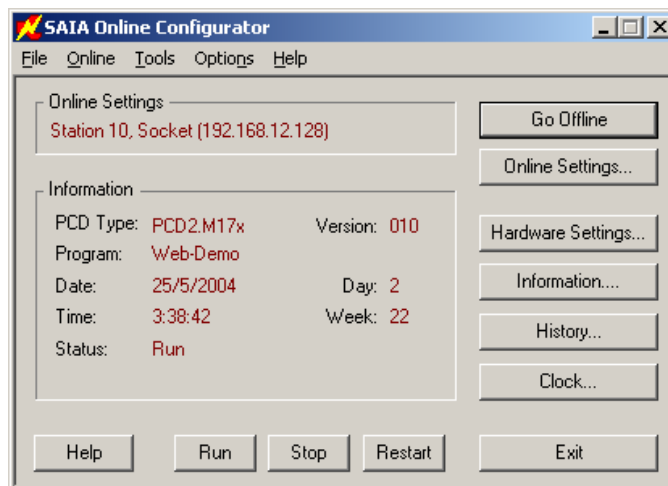
8.4.2 On-line Konfigurator



Konfiguriert den Speicher, die Kommunikations-Parameter und das PCD Passwort. Die Informationen werden jedoch direkt in den PCD Speicher geschrieben. Es bleibt keine Kopie der Information im PG5 Projekt. Ohne Steuerung ist über diese Information nichts bekannt.

Daher ist der On-line Konfigurator für die Überprüfung von PCD Daten einzusetzen, die Konfiguration jedoch mit *Hardware Settings* durchzuführen.

8.4.3 On-line Konfigurator-Fenster



<i>PCD type</i>	PCD Typ Referenz-Nummer
<i>Version</i>	Version der PCD Firmware
<i>Program Name</i>	Anwender-Programmname
<i>Date</i>	PCD Uhrdatum (wenn keine Uhr: 1/1/92)
<i>Time</i>	PCD Uhrzeit
<i>Day</i>	Wochentag: 1 = Montag, ... 7 = Sonntag
<i>Week</i>	Wochennummer
<i>Status</i>	Betriebszustand: Run, Stop, Halt, Conditional Run
<i>Onlinesettings</i>	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?

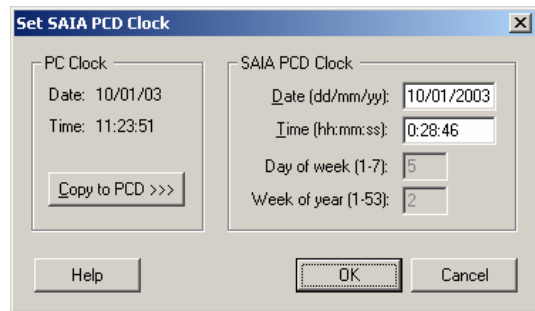
Hardware Settings...

Konfiguration des PCD Speichers. Dies sind dieselben Parameter, wie die bereits in *Hardware Settings* des Project Manager vergebenen.

8.4.4 PCD-Zeit einstellen

Clock...

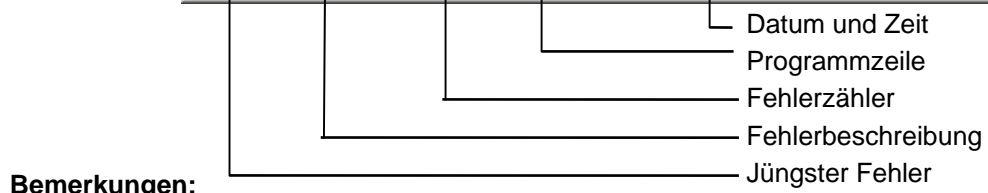
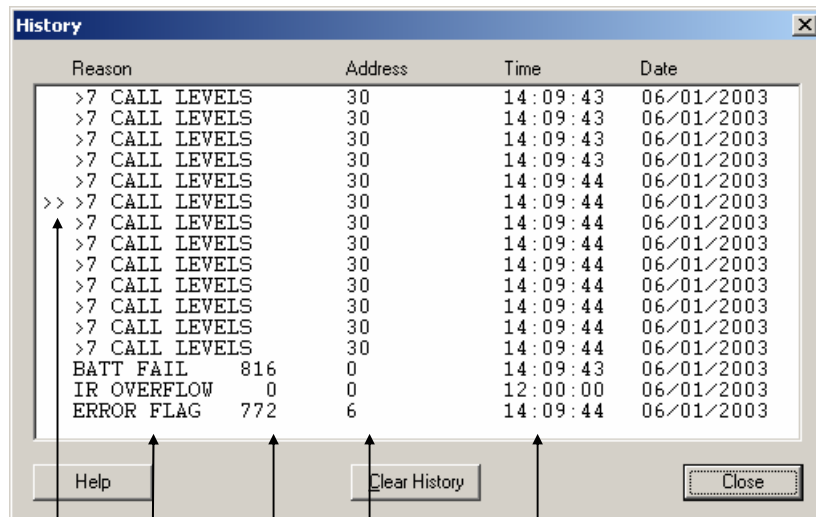
1. *Online configurator* Knopf im *Projekt Manager* Fenster drücken. Dann *Clock* Knopf drücken.
2. Mit dem *Copy to PCD >>>* Knopf PC-Zeit in die Steuerung kopieren, oder die Daten direkt in den Feldern des *PCD Clock* Fensters anpassen.



8.4.5 Das History

Hjstory...

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:

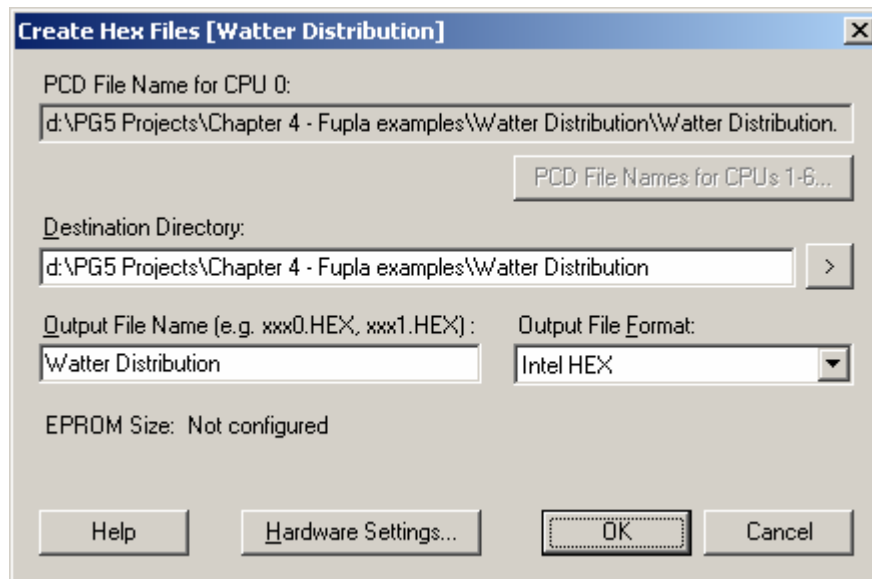
- Jede CPU hat ihre eigene History.
- Der *BATT FAIL* Fehler kommt nur auf der CPU 0 vor.

- 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.

8.5 EPROM Programmierung

PG5 unterstützt die Programmierung binärer oder hexadezimaler Dateien für alle Standardtypen von EPROM Programmier-Geräten auf dem Markt.

Im Projekt Manager, im Menü *CPU, Create Hex Files ...* auswählen



EPROM Datei herstellen:

- Konfiguration der *Hardware Settings*
- Verarbeiten (Build) des Programms
- *Output File Format* auswählen
- Definition des *Destination Directory* und *Output File Name*
- Mit *OK* abschliessen

Technische Details:

Das EPROM enthält nicht nur das PCD Programm, sondern auch die *Hardware Settings*. Beim Einschalten der Versorgungsspannung werden die *Hardware Settings* automatisch vom EPROM in die PCD kopiert, aber nur, wenn die PCD die Informationen verloren hat (wegen Spannungsausfall aufgrund einer defekten oder fehlenden Batterie).

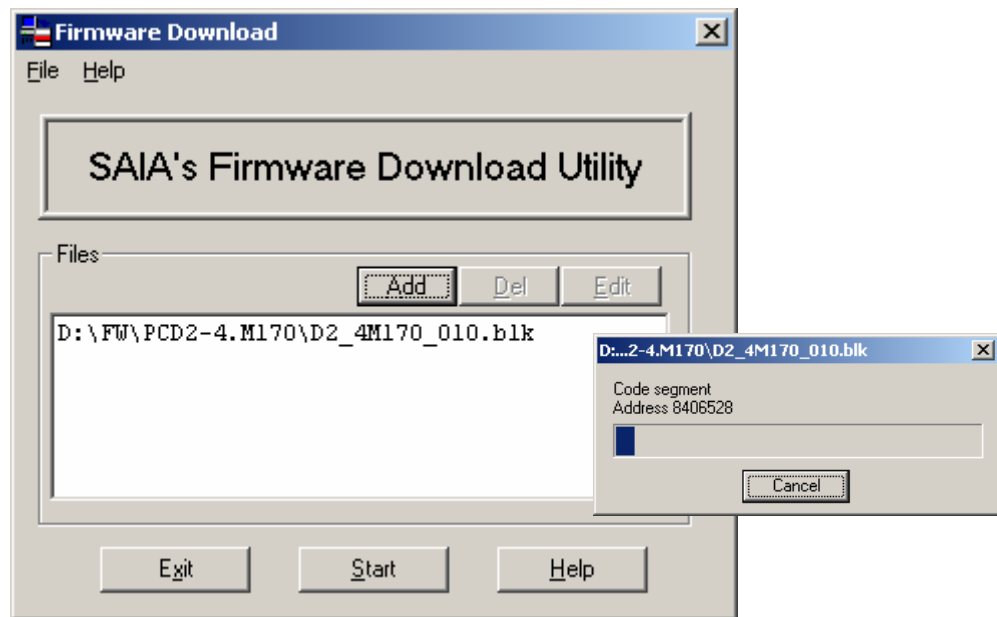
Beim Einsetzen des EPROM auf die korrekte Position der PCD Speicher-Jumper achten.

8.6 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 neuesten¹⁾ 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.



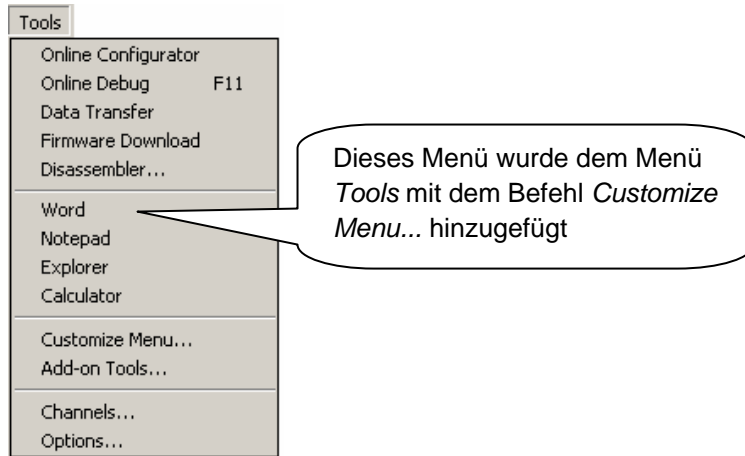
Download Instruktionen:

- 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.

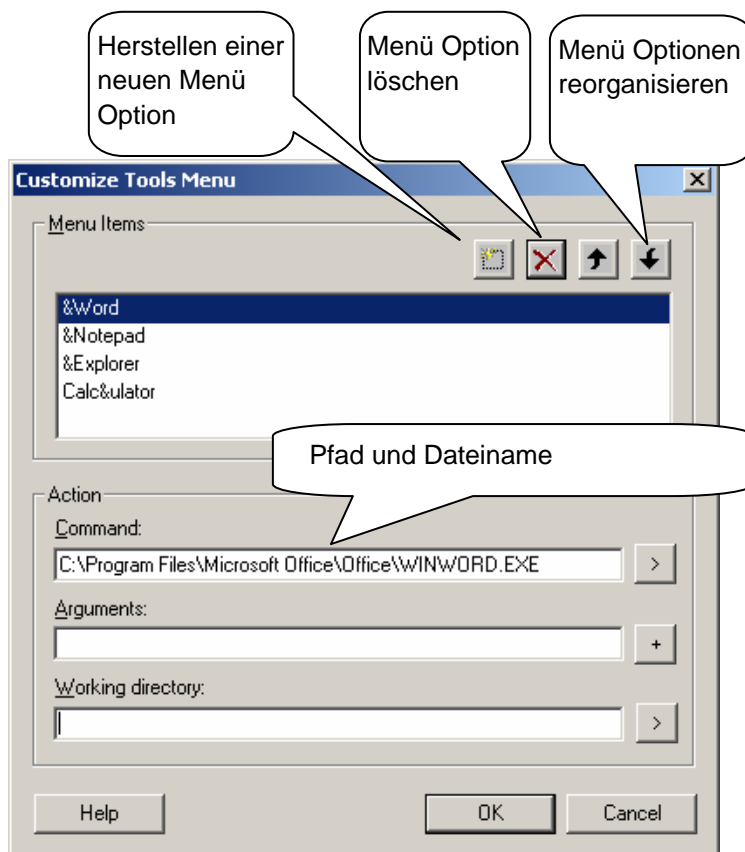
¹⁾ PCD2.M170, PCD4.M170, PCD2.M480

8.7 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

INHALT	1
9 SAIA-NETZWERKE (S-NET)	2
9.1 Zusammenfassung	2
9.2 Wahl des Netzwerks	2
9.2.1 Unterstützte Services	2
9.2.2 Funktionen	3

9 Saia-Netzwerke (S-Net)

9.1 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.

9.2 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.

9.2.1 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.

9.2.2 Funktionen

9.2.2.1 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.

9.2.2.2 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.

9.2.2.3 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.

9.2.2.4 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.

9.2.2.5 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. Übertragungsgeschwindigkeit	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	Saia	Saia	Saia	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

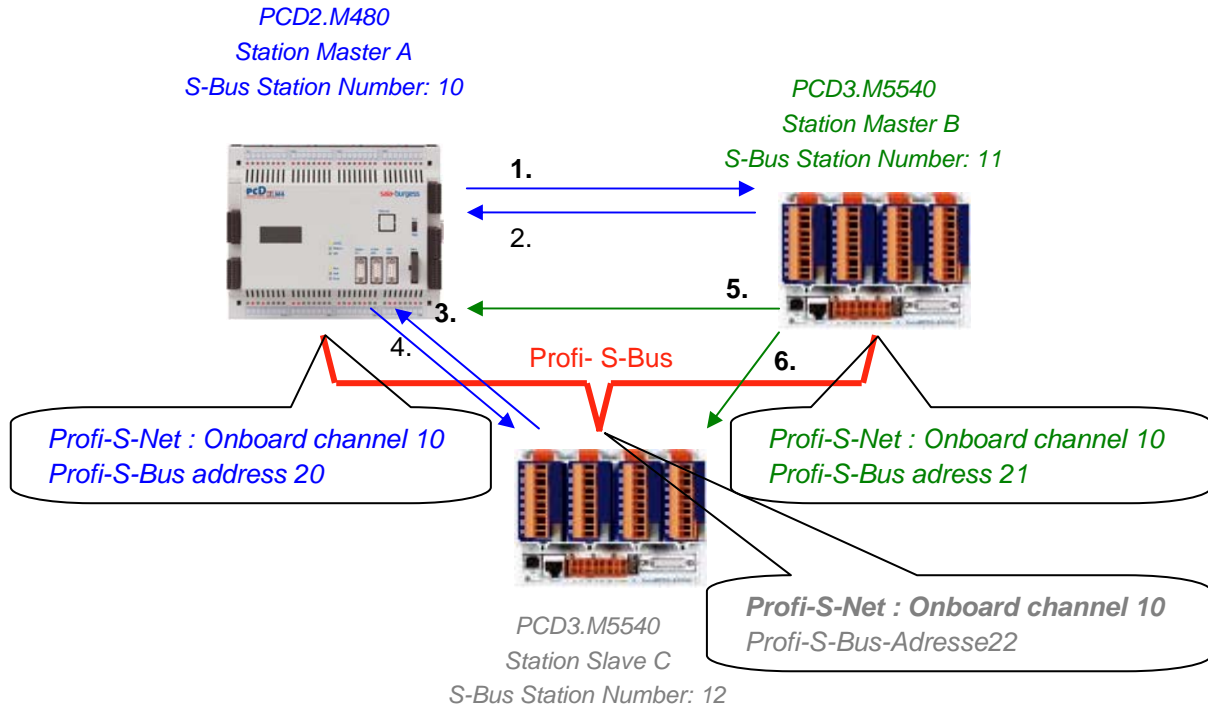
Inhaltsverzeichnis

INHALTSVERZEICHNIS	1
10 PROFI-S-BUS	2
10.1 Beispiel eines Profi-S-Bus-Netzes.	2
10.2 Beispiele eines Profi-S-Bus-Datenaustausch.	2
10.3 Projekt PG5	3
10.4 Hardwares Settings Masters, Slaves	3
10.4.1 Festlegen der PCD-Parameter	3
10.4.2 Festlegen der S-Bus-Stationsnummer auf dem Netzwerk	4
10.4.3 Festlegen des Profi-S-Bus-Kommunikationskanals	4
10.4.4 Laden der Hardwares Settings auf der CPU	6
10.5 Fupla-Programm	6
10.5.1 Zuweisung des Kanals mithilfe einer Fbox SASI	6
10.5.2 Zuweisung des Masterkanals	7
10.5.3 Zuweisung des Slavekanals	7
10.5.4 Prinzip eines Datenaustauschs auf einem Multimasternetzwerk.	7
10.5.5 Datenaustausch zwischen Master- und Slavestationen.	8
10.5.6 Diagnosen	9
10.6 IL-Programm	12
10.6.1 Zuweisung des Masterkanals mithilfe einer SASI-Anweisung	12
10.6.2 Zuweisung des Slave-Kanals	12
10.6.3 Prinzip eines Datenaustauschs auf einem Multimasternetzwerk.	12
10.6.4 Datenaustausch zwischen Master- und Slavestationen.	13
10.6.5 Diagnosen	14
10.7 Gateway-Funktion	17
10.7.1 Anwendung	17
10.7.2 Konfiguration der Gateway PGU-Funktion	18
10.7.3 Konfiguration eines zusätzlichen Slave-Gateway-Slave-Port	20
10.7.4 Kommunikationstimings	21
10.8 Andere Referenzen	22

10 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.

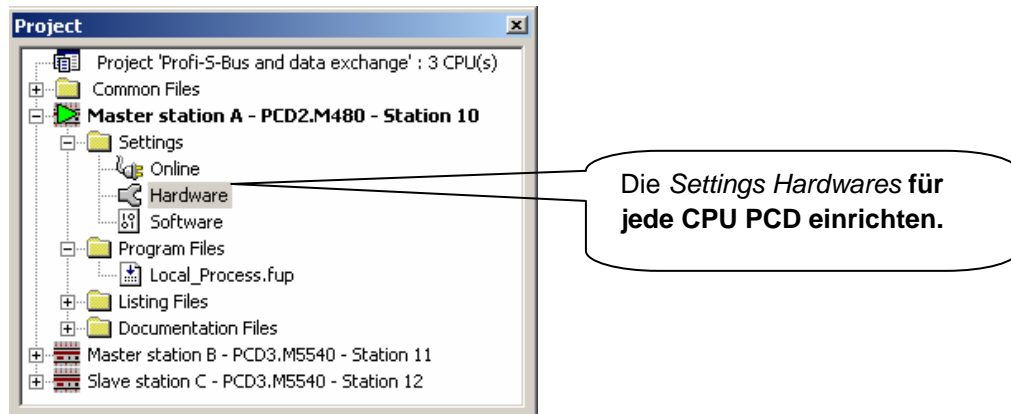
10.1 Beispiel eines Profi-S-Bus-Netzes.



10.2 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

10.3 Projekt PG5

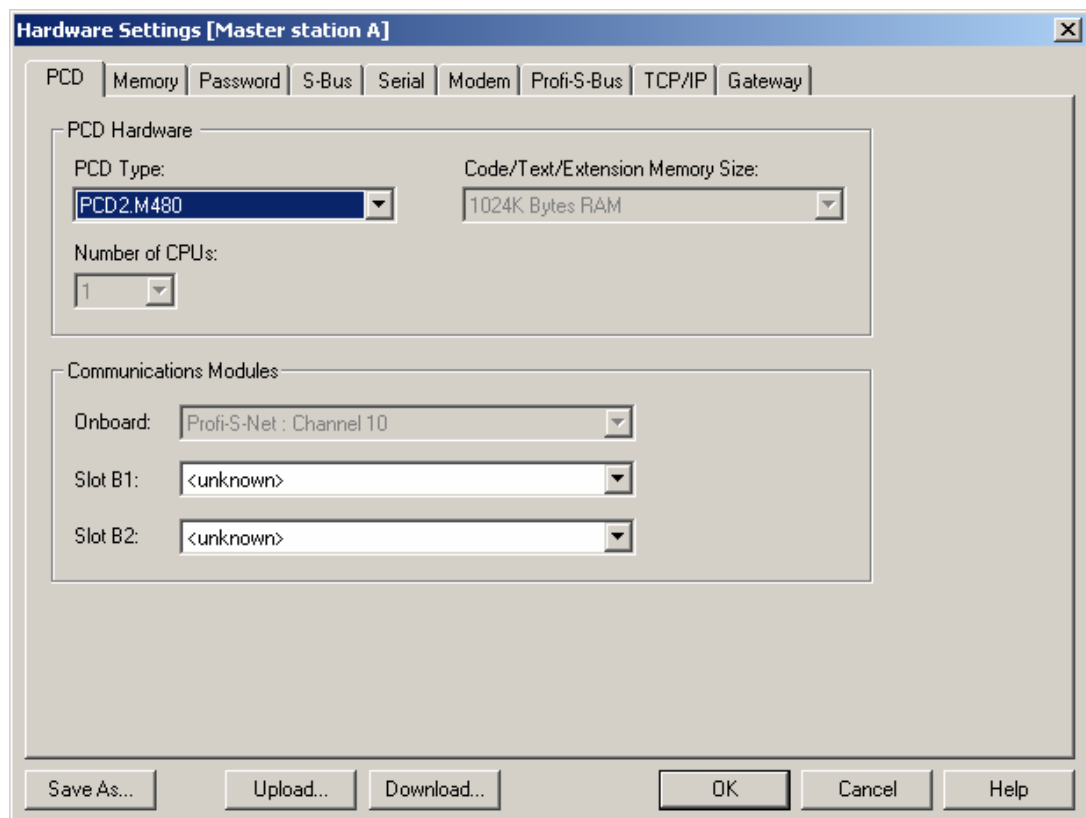


Der *Saia Project Manager* gibt einen Überblick über die PCD eines Applikationsprojektes sowie die Parameter für die Kommunikationsnetze. Beginnen wir mit dem Hinzufügen einer CPU in das Projekt für jede auf dem Netzwerk verfügbare Station.

10.4 Hardwares Settings Masters, Slaves

Die Konfigurationen der *Hardwares Settings* einer Masterstation und der Slaves sind fast gleich.

10.4.1 Festlegen der PCD-Parameter

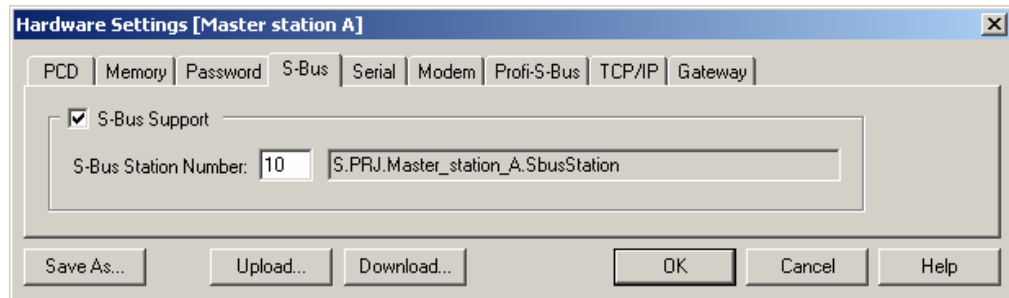


PCD-Type

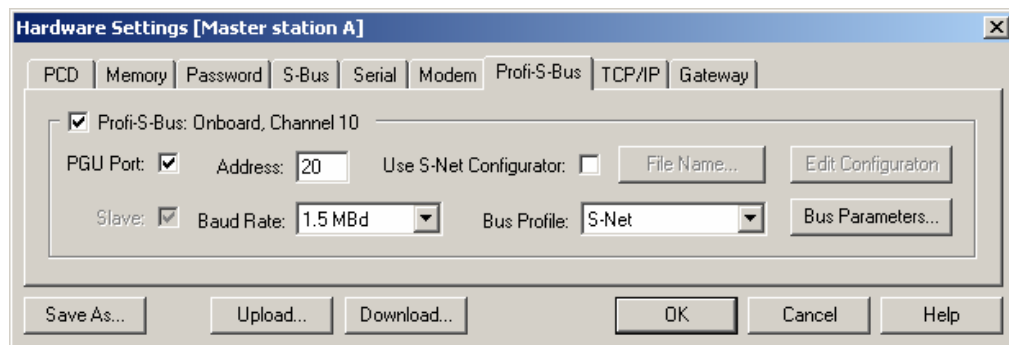
Festlegen des CPU-Typs

Communication Modules

Falls erforderlich, den Typ der in den Slots B1 und B2 des PCD2.M480 eingesetzten Kommunikationsmodulen angeben.

10.4.2 Festlegen der S-Bus-Stationsnummer auf dem Netzwerk**S-Bus-Station Number**

Die S-Bus-Stationsnummer ist für alle Kommunikationskanäle der PCD gleich.

10.4.3 Festlegen des Profi-S-Bus-Kommunikationskanals**Address**

Sie mit dem Kanal verbundene Profi-S-Bus-Stationsnummer

PGU Port oder Slave

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 Hilffsystem zur Programmierung und Inbetriebnahme von PG5.

Slave

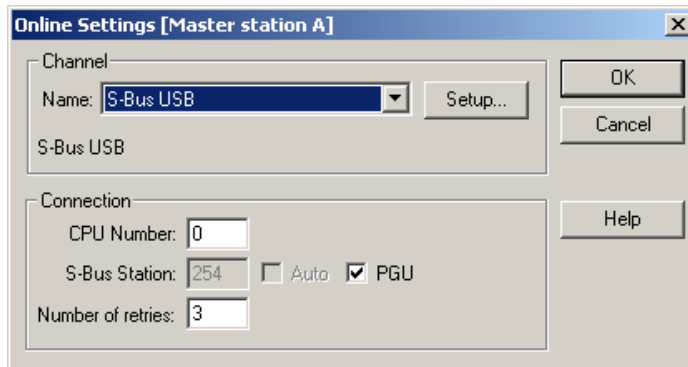
Unterstützt nur den Datenaustausch mit den Masterstationen, den Überwachungssystemen und Terminals.

Baud Rate

Kommunikationsgeschwindigkeit, muss auf allen Stationen des Netzwerks identisch sein.

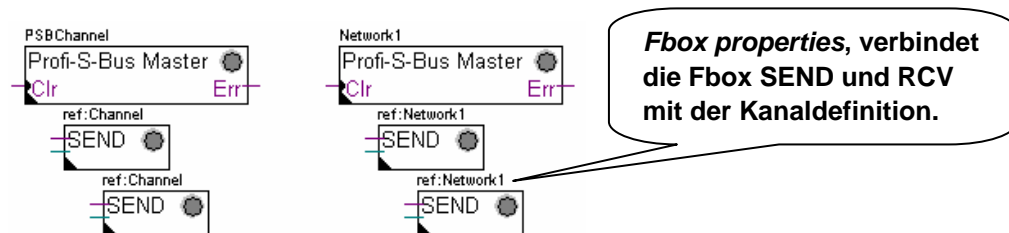
S-Bus Profile

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.

10.4.4 Laden der Hardware Settings auf der CPU

Mit den neuen Systemen PCD2.M480 und PCD3, können die *Hardware Settings* auf den PCD2.M480 und PCD3 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 *Hardware Settings*, in die PCD herunterladen.

10.5 Fupla-Programm**10.5.1 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.

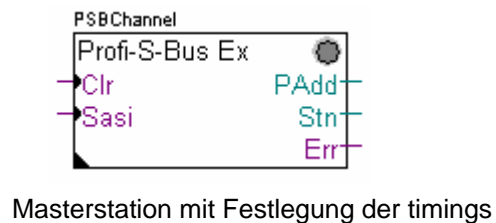
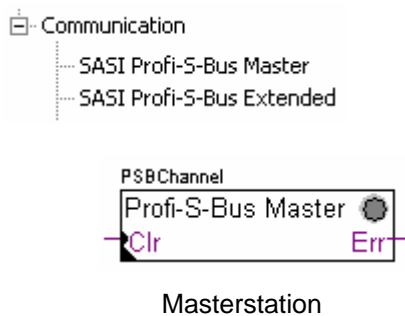
Wenn die PCD mehrere Kommunikationskanäle benutzt, sollte jeder einzelne Kanal über die entsprechende Fbox SASI definiert werden. Anschließend den Cursor mit der Maus auf die Fbox SASI setzen, das Kontextmenü markieren und für jede Fbox,

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 *Hardware Settings* 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, ...

10.5.2 Zuweisung des Masterkanals



Die Zuweisung des Masterkanals erfolgt über die Vervollständigung der *Hardware Settings* mit einer der obigen Fboxen.

Nur der Kommunikationskanal und die Masterkanaltimings können über die Fbox eingestellt werden. Die anderen Parameter werden in den Hardware Settings festgelegt.

Parameter 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.

10.5.3 Zuweisung des Slavekanals

Für die Slavestation eines Profi-S-Bus-Netzes ist keine FBox SASI erforderlich. Alle erforderlichen Festlegungen wurden bereits in den *Hardware Settings* gemacht.

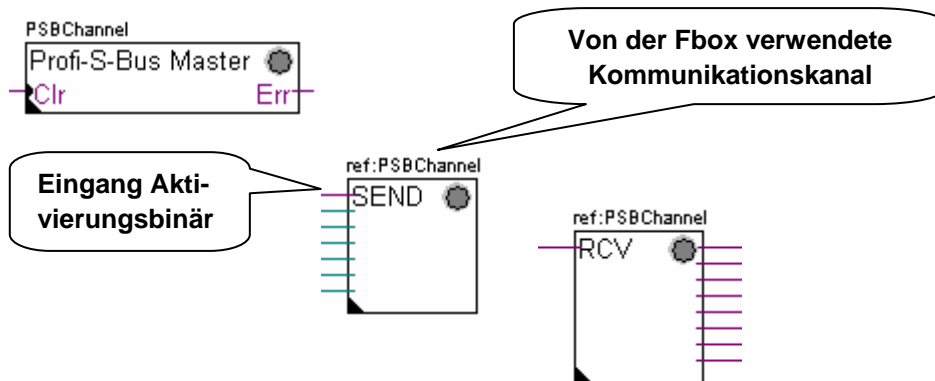
10.5.4 Prinzip eines Datenaustauschs auf einem Multimasternetzwerk.

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.

10.5.5 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 *Fbox properties Name* 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*)

Parameter 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.

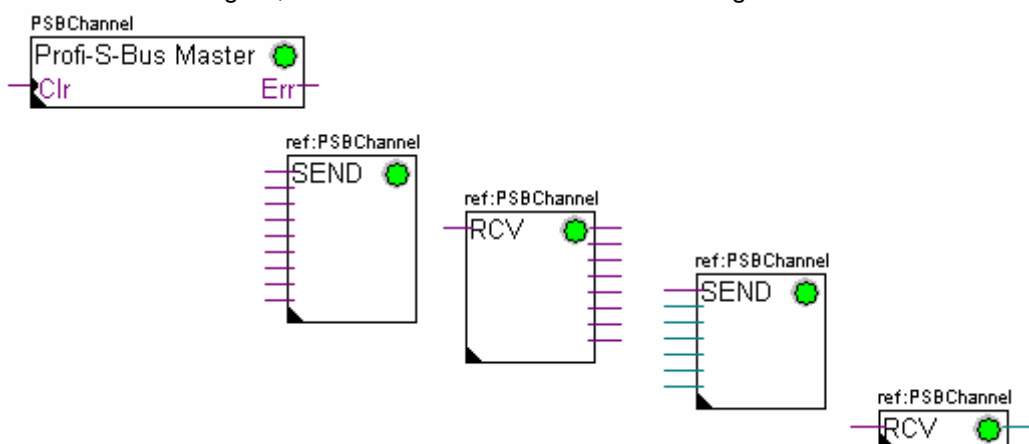
10.5.6 Diagnosen



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.

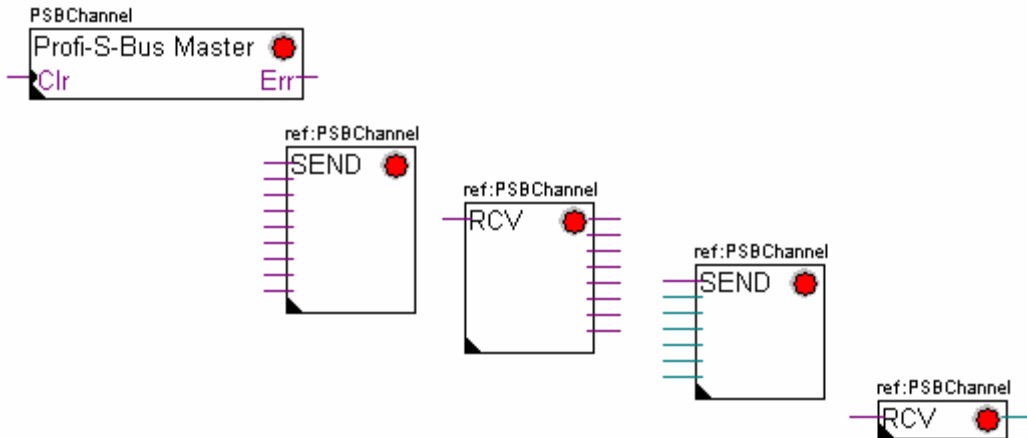
Einwandfreie Funktion

Alle Fboxen sind grün, der Datenaustausch funktioniert richtig.



Es können keine Daten über das Netzwerk ausgetauscht werden.

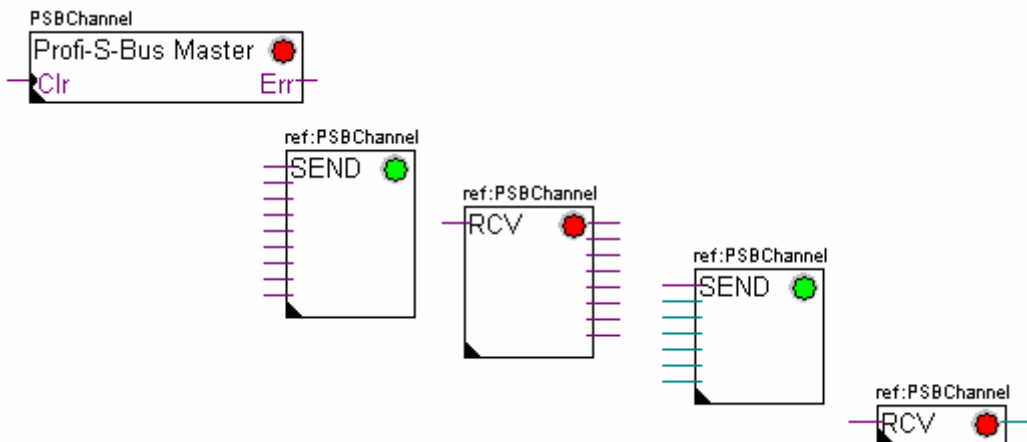
Die Fboxen *SASI*, *SEND* und *RCV* sind rot, es können keine Daten über das Netzwerk ausgetauscht werden.



- 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.

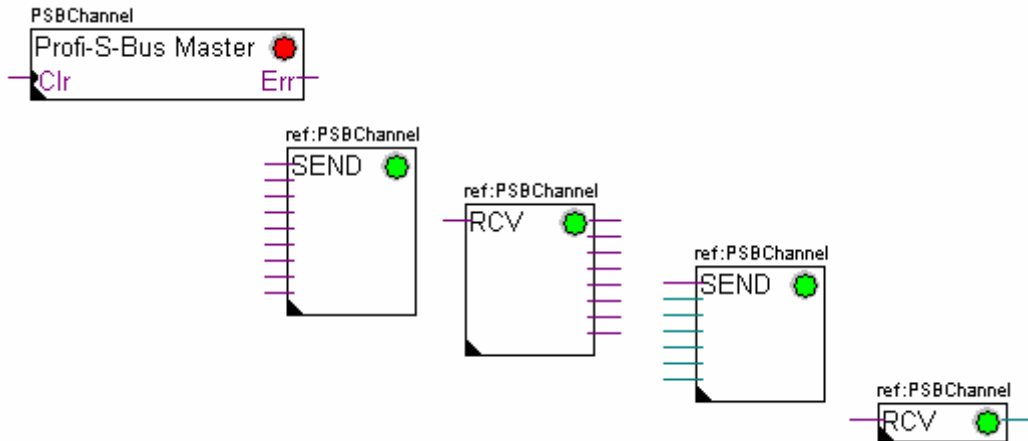
Mögliche Korrekturmaßnahmen auf der Slavestation

Für jede fehlerhafte Fbox *SEND* und *RCV* die Slavestationsnummer notieren und die entsprechenden Stationen prüfen.

- Prüfen, ob die *Hardware Settings* 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.

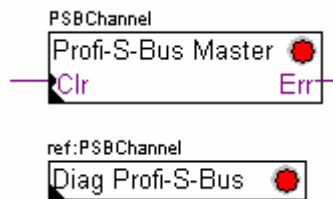
Nur die Fbox SASI leuchtet rot

Einstellungsfenster der Fbox SASI öffnen und den letzten Fehler über die Schaltfläche *Clear* auf Null zurückstellen.



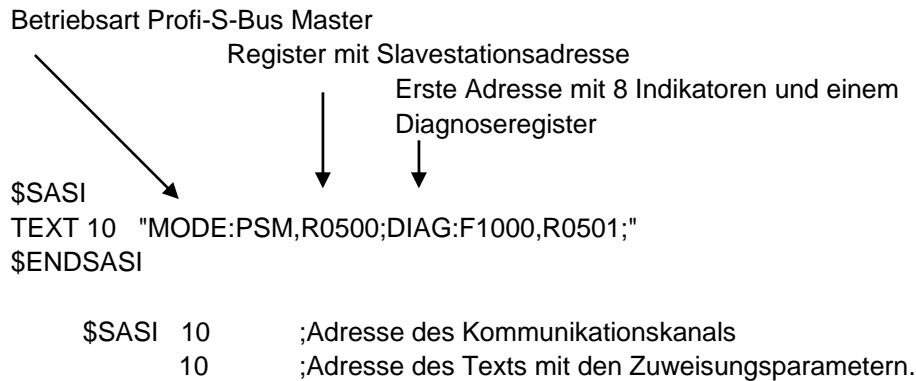
Diagnose-Fbox

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.



10.6 IL-Programm

10.6.1 Zuweisung des Masterkanals mithilfe einer SASI-Anweisung



Die Zuweisung eines Kanals erfolgt über eine zu Beginn des Programms platzierte SASI-Anweisung: Graftec-Initialisierungssequenz oder XOB 16-Initialisierungsblock.

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 *Hardware Settings* vervollständigt werden.

10.6.2 Zuweisung des Slave-Kanals

Für die Slavestation eines Profi-S-Bus-Netzes ist keine SASI-Anweisung erforderlich. Alle erforderlichen Festlegungen wurden bereits in den *Hardware Settings* gemacht.

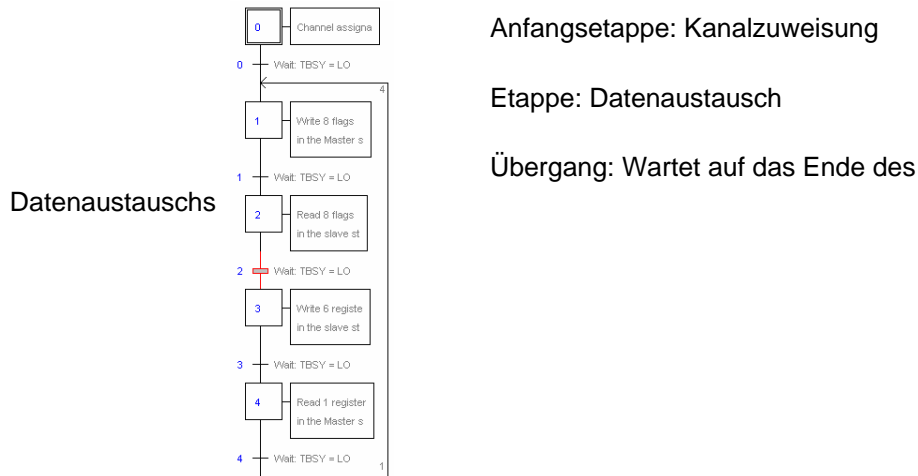
10.6.3 Prinzip eines Datenaustauschs auf einem Multimasternetzwerk.

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.

10.6.4 Datenaustausch zwischen Master- und Slavestationen.



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 (ISlavestation)
      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.

10.6.5 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:

- *Hardwares Settings* prüfen.
- Prüfen, ob die *Hardwares Settings* 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 *Hardwares Settings* 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 *Hardware Settings* 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.

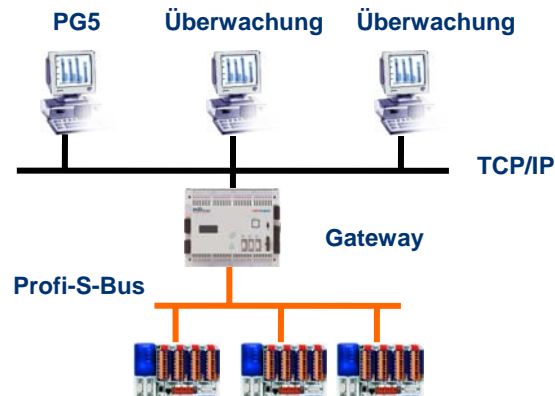
Diagnoseregister

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.

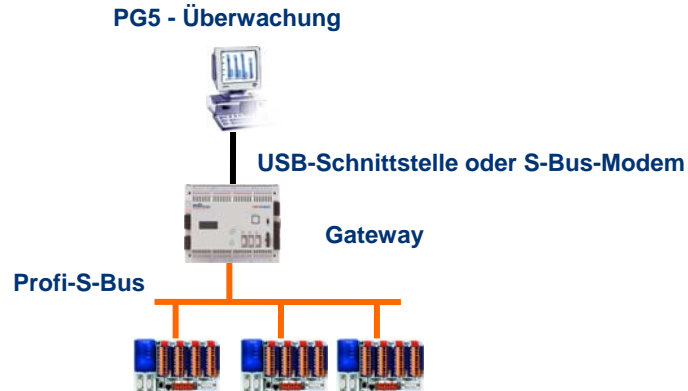
10.7 Gateway-Funktion

Die *Gateway*-Funktion wird häufig verwendet, damit zwei verschiedene Kommunikationssysteme zusammen kommunizieren können oder zum Anpassen eines PG5-Programmierungssystems, eine Saia®Visi.Plus-Überwachung auf einem anderen als das direkt von diesen Geräten unterstützten Netzwerks.

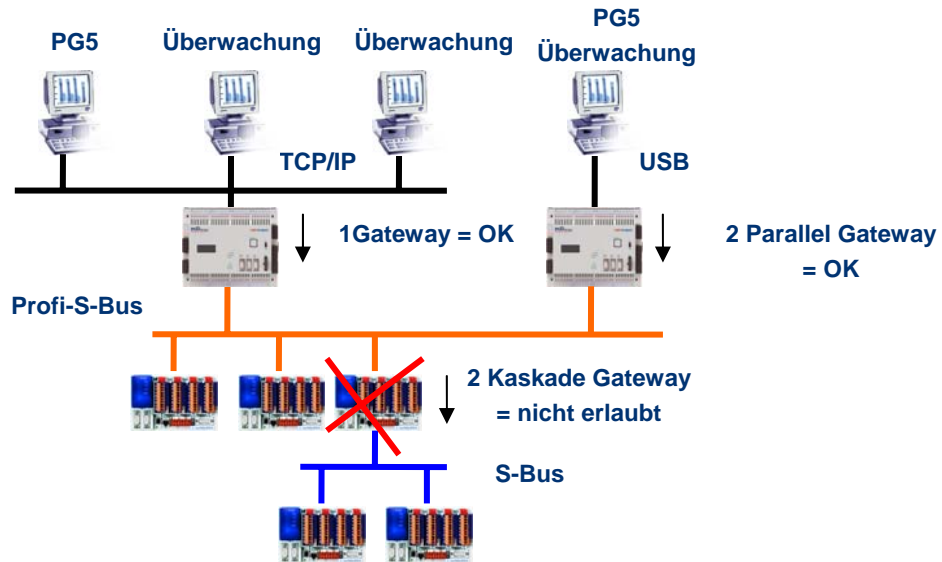
10.7.1 Anwendung



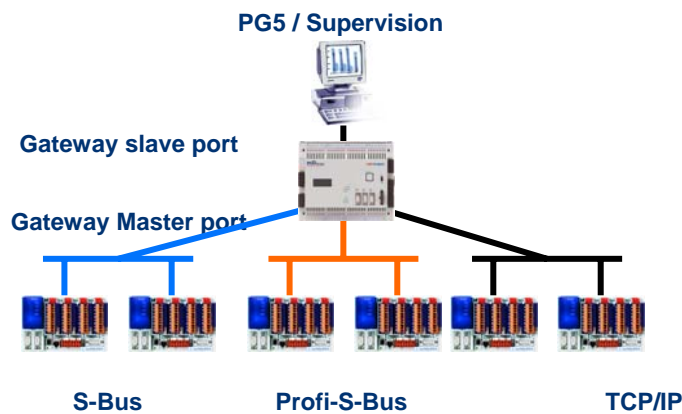
Die *Gateway*-Funktion ermöglicht die Durchführung einer Brücke zwischen zwei Netzwerken. Beispielsweise ein Ethernet-Netzwerk mit dem Profi-S-Bus-Netzwerk verbinden. Auf diese Weise tauschen die PCD-Systeme die Daten auf einem Bus aus dem Bereich Automatisierung aus, der vom EDV-Netz des Unternehmens getrennt ist. Aber die mit der PG5—Software oder einem Saia®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.

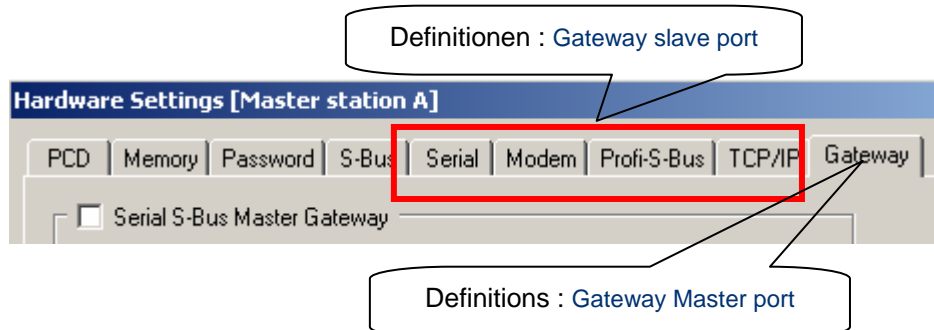


Um die Timingvorgaben der Kommunikation zu respektieren, können nicht zwei kaskadische Gateways definiert werden. Aber es ist möglich, zwei parallele Gateways im gleichen Netz zu definieren.



Falls erforderlich, kann die Funktion Gateway eine Verbindung zu verschiedenen Unterkommunikationsnetzen herstellen.

10.7.2 Konfiguration der Gateway PGU-Funktion

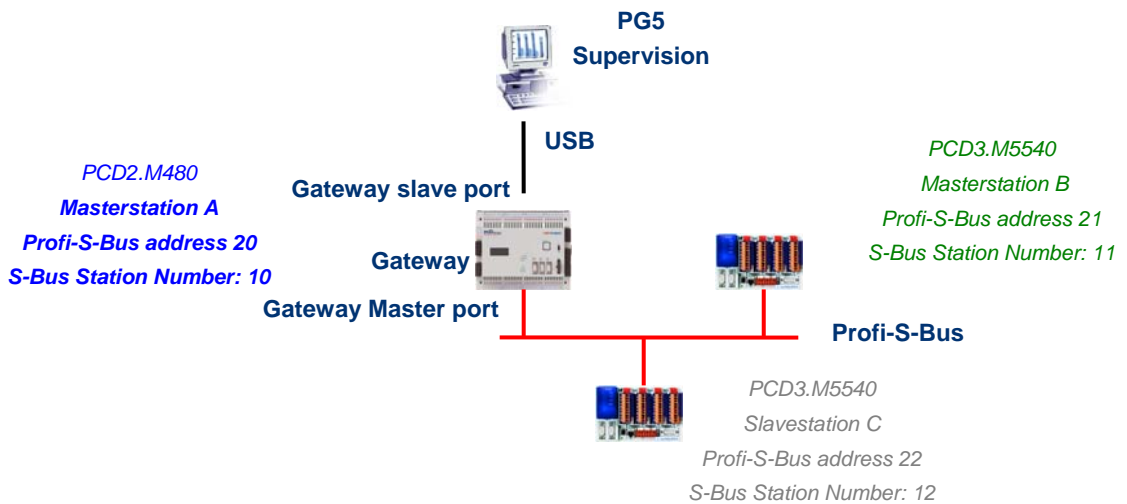


Die *Gateway*-Funktion lässt sich sehr leicht einrichten, sie benötigt kein Programm, sondern nur ein paar Parameter in den *Hardware Settings* der PCD.

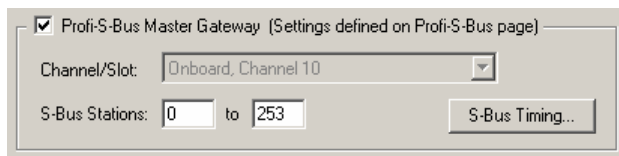
Im Allgemeinen muss man nur einen *Gateway Slave Port* und einen *Gateway Master Port* konfigurieren, anschließend wird alles automatisch von der *Gateway-Funktion* unterstützt.

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

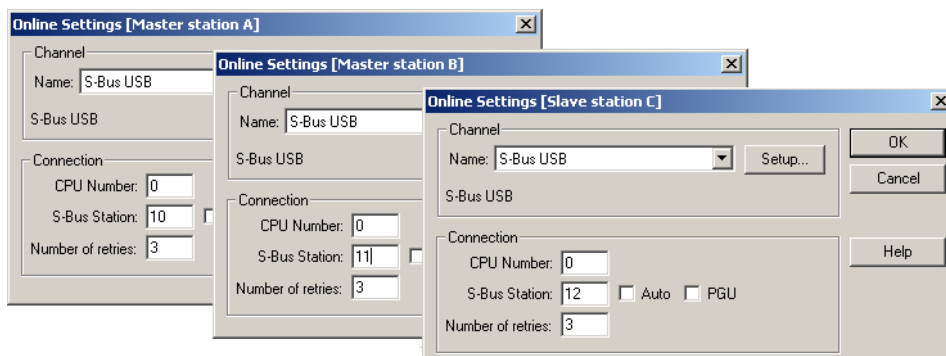


Hardware Settings der Masterstation A



Der Gateway USB ist eine Ausnahme und erfordert keinen Parameter für den *Gateway Slave port*, nur der *Gateway Master port* muss parametrisiert werden. (Nicht vergessen, die neue Konfiguration in den Master A herunterzuladen!)

Online Settings der CPU des Projekts



Zum Erstellen der USB-Kommunikation mit jeder PCD muss man noch die *Online Settings* jeder CPU 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 CPU, *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 CPU aufladen und testen und gleichzeitig mit dem USB-Kabel mit der Station *Master A* verbunden bleiben.

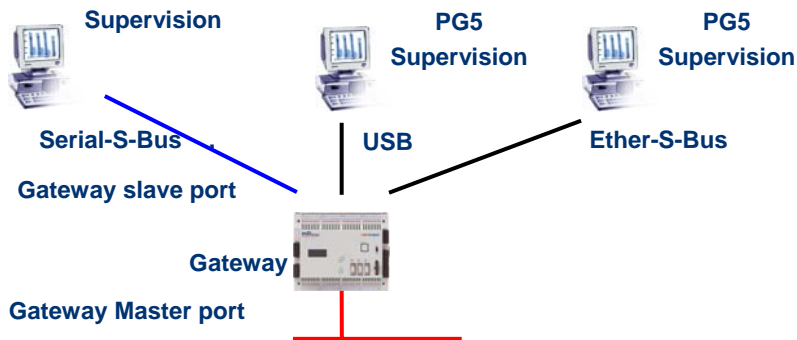
Master station B - PCD3.M5540 - Station 11

Um mit einer anderen Netzwerkstation zu kommunizieren, die CPU 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.

10.7.3 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.

Hardware Settings

Im Allgemeinen unterstützt die PCD nur einen einzigen PGU-Slavekanal. Aber die neuen Automaten PCD2.M480 und PCD3.Mxxxx können mehrere auf demselben PCD unterstützen. Die Konfiguration des zweiten Gateway Slave PGU wird vollständig von den *Hardware Settings* unterstützt.

Beispiel: Hinzufügen eines zweiten Gateway Ether-S-Bus, Profi-S-Bus

TCP/IP

TCP/IP : Slot B2, Channel 8

IP Node: S.PRJ.Master_station_A.IPNode

IP Address:

Subnet Mask:

Default Router:

PGU Port:

Slave:

Der zweite *Gateway Slave port PGU* wird durch Konfigurieren der *Hardware Settings* mit dem Knoten und der TCP/IP-Adresse hinzugefügt. Wenn die SPS eine PCD2.M480 ist, muss man auch das PCD-Kommunikationsmodul auf Slot B2 mit dem PCD7.F65x (Ethernetmodul) konfigurieren.

Fupla- oder IL-Programm

Mit den alten PCD aber auch mit den neuen PCD2M.480 und PCD3.Mxxxx, ist es 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



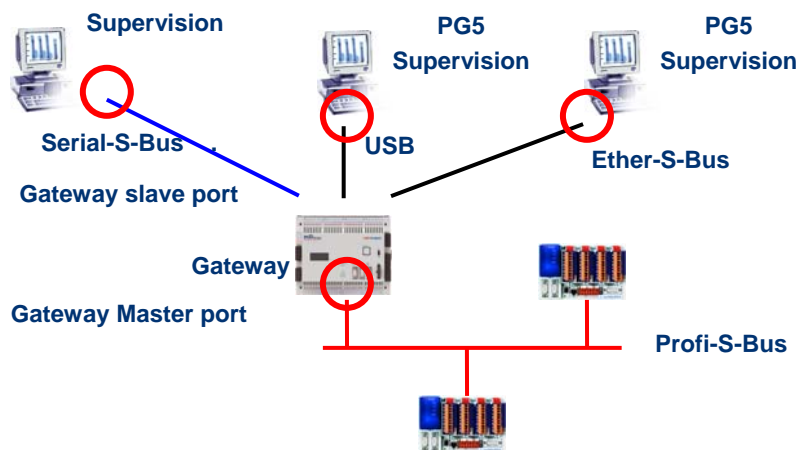
Der *Gateway*-Einstellungsparameter muss daher mit der Option *Yes* definiert werden. Je nach Kanaltyp müssen die anderen Parameter im Einstellungsfenster ebenso korrekt definiert werden.

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

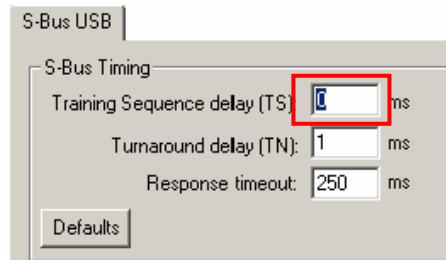
Indikator und Diagnoseregister
 Mode S-Bus Gateway Slave Data mode
 Übertragungsgeschwindigkeit

10.7.4 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*



10.8 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.

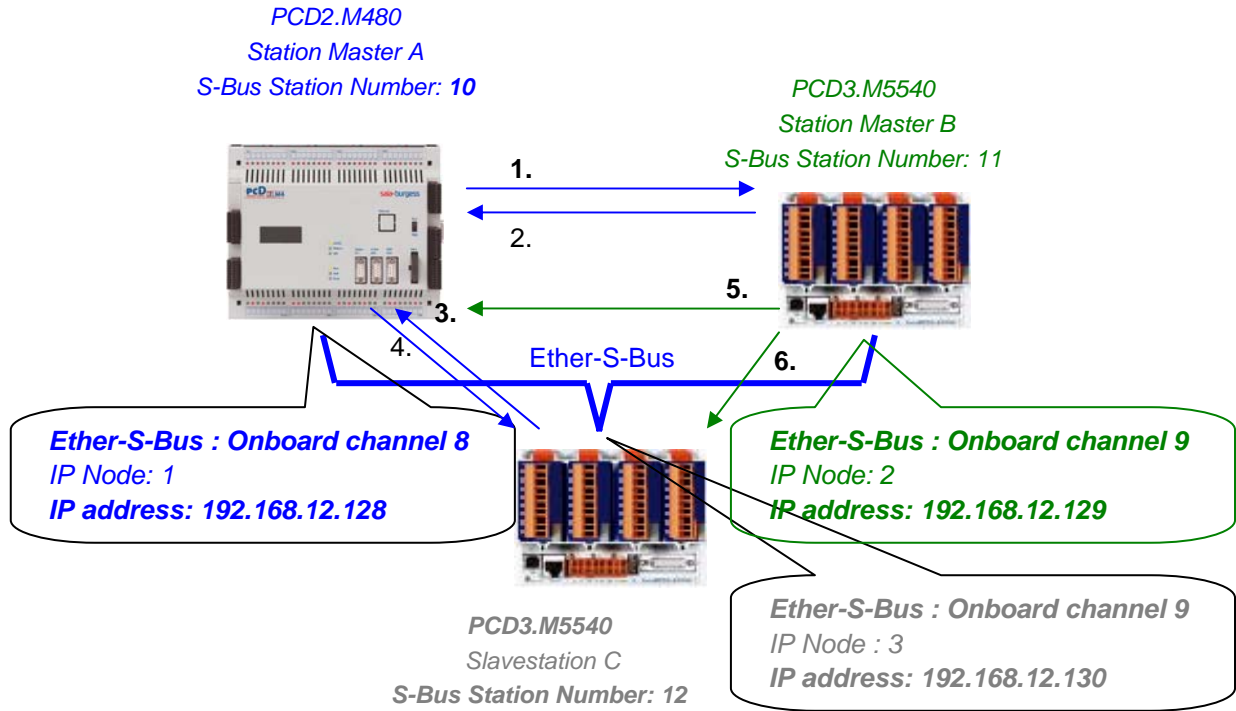
Inhaltsverzeichnis

INHALTSVERZEICHNIS	1
11 ETHER-S-BUS	2
11.1 Beispiel eines Ether-S-Bus-Netzes.	2
11.2 Beispiele eines Ether-S-Bus-Datenaustausch.	2
11.3 Projekt PG5	3
11.4 Hardwares Settings Masters, Slaves	3
11.4.1 Festlegen der PCD-Parameter	3
11.4.2 Festlegen der S-Bus-Stationsnummer auf dem Netzwerk	4
11.4.3 Festlegen des Ether-S-Bus-Kommunikationskanals	4
11.4.4 Aufladen der Hardwares Settings auf der CPU	5
11.5 Fupla-Programm	5
11.5.1 Zuweisung des Kanals mithilfe einer Fbox SASI	5
11.5.2 Zuweisung des Masterkanals	6
11.5.3 Zuweisung des Slavekanals	6
11.5.4 Prinzip eines Datenaustauschs auf einem Multimasternetzwerk.	6
11.5.5 Datenaustausch zwischen Master- und Slavestationen.	7
11.5.6 Diagnosen	8
11.6 IL-Programm	11
11.6.1 Zuweisung des Masterkanals mithilfe einer SASI-Anweisung	11
11.6.2 Zuweisung des Slavekanals	11
11.6.3 Prinzip eines Datenaustauschs auf einem Multimasternetzwerk.	11
11.6.4 Datenaustausch zwischen Master- und Slavestationen.	12
11.6.5 Diagnosen	13
11.7 Gateway-Funktion	16
11.7.1 Anwendung	16
11.7.2 Konfiguration der Gateway PGU-Funktion	17
11.7.3 Konfiguration eines zusätzlichen Slave-Gateway-Slave-Port	19
11.7.4 Kommunikationstimings	21
11.8 Andere Referenzen	21

11 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.

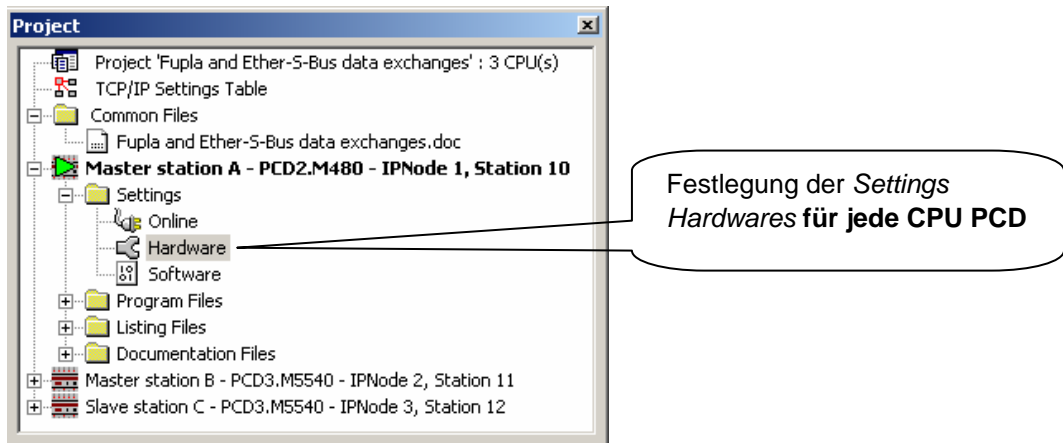
11.1 Beispiel eines Ether-S-Bus-Netzes.



11.2 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

11.3 Projekt PG5

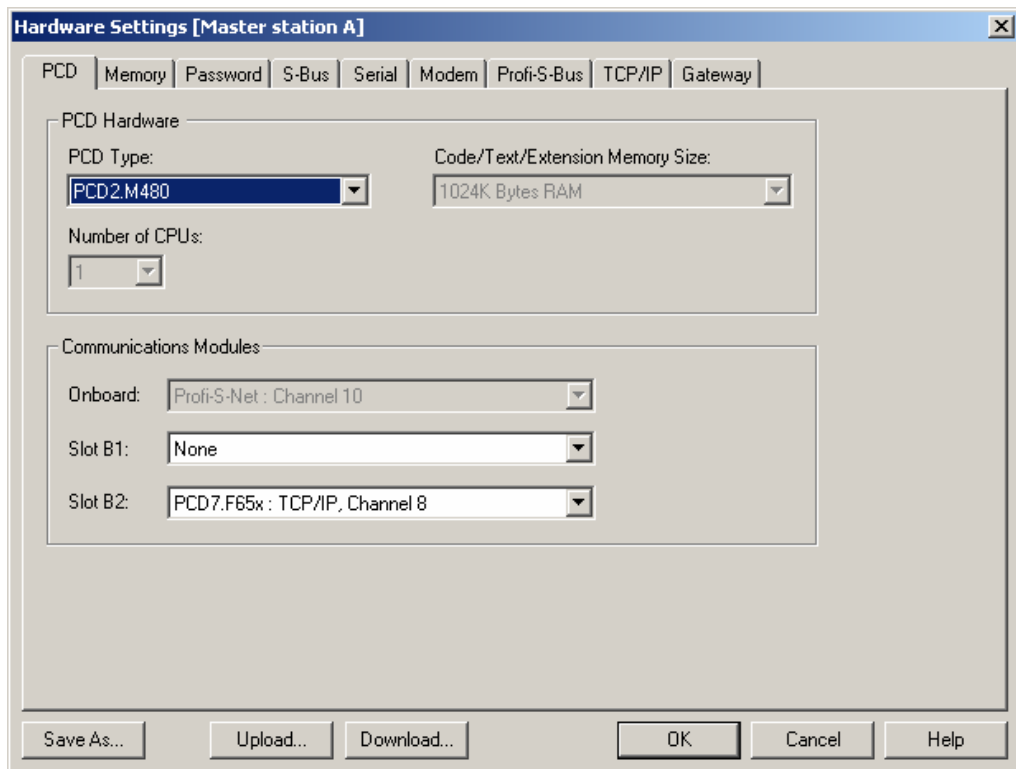


Der Saia Project Manager gibt einen Überblick über die PCD eines Applikationsprojektes sowie die Parameter für die Kommunikationsnetze. Beginnen wir mit dem Hinzufügen einer CPU in das Projekt für jede auf dem Netzwerk verfügbare Station.

11.4 Hardwares Settings Masters, Slaves

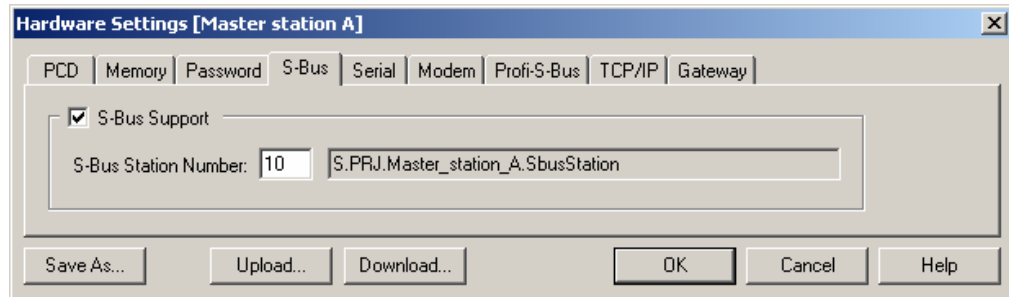
Die Konfigurationen der *Hardwares Settings* einer Masterstation und der Slaves sind fast gleich.

11.4.1 Festlegen der PCD-Parameter

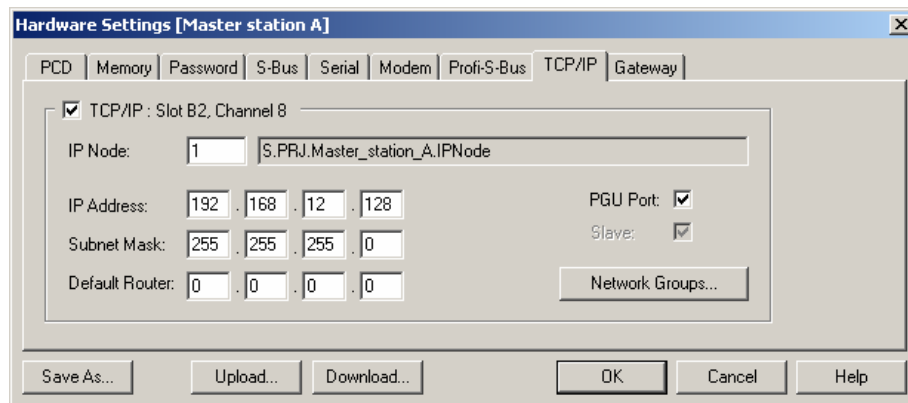


PCD-Typ**Festlegen des CPU-Typs****Communication Modules**

Falls erforderlich, den Typ der in den Slots B1 und B2 des PCD2.M480 eingesetzten Kommunikationsmodule angeben.

11.4.2 Festlegen der S-Bus-Stationsnummer auf dem Netzwerk**S-Bus-Stationsnummer**

Die S-Bus-Stationsnummer ist für alle Kommunikationskanäle des PCD gleich.

11.4.3 Festlegen des Ether-S-Bus-Kommunikationskanals**IP Address**

Die mit dem Kanal verbundene Ether-S-Bus-Stationsnummer

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.

PGU Port oder Slave

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.

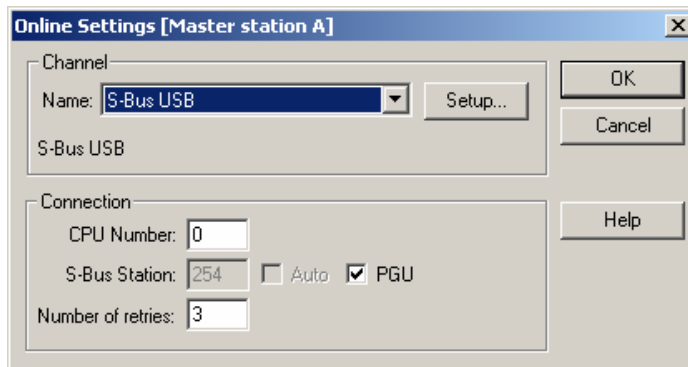
PGU-Slave

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.

11.4.4 Aufladen der Hardwares Settings auf der CPU

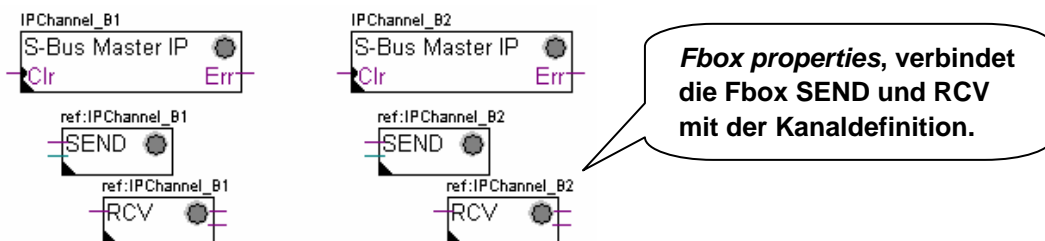


Mit den neuen Systemen PCD2.M480 und PCD3, können die *Hardwares Settings* auf den PCD2.M480 und PCD3 über die USB-Schnittstelle heruntergeladen werden. Man muss nur prüfen, ob die *Online Settings* mit einem *Ether-S-Bus PGU*-Kanal definiert worden sind.

Die Parameter in die PCD über die Schaltfläche *Download* im Fenster *Hardware Settings* herunterladen.

11.5 Fupla-Programm

11.5.1 Zuweisung des Kanals mithilfe einer Fbox SASI



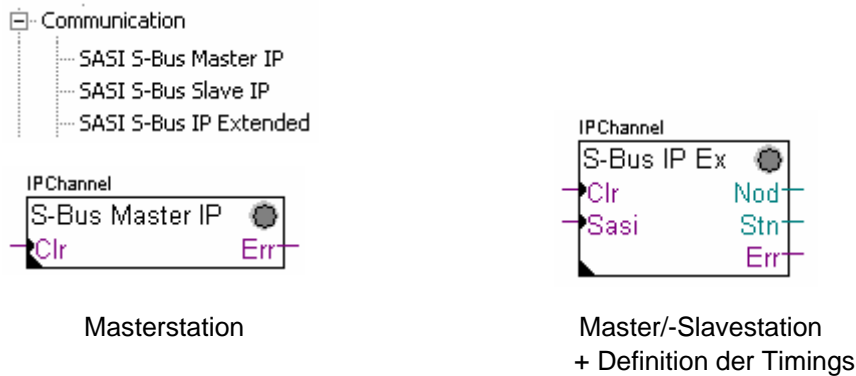
Die Zuweisung eines Kanals erfolgt über eine zu Beginn der Fupla-Datei platzierte FBox SASI. Jedes Kommunikationsnetz verfügt über seine eigene Fbox SASI, weil sich die Zuweisungsparameter von einem zum anderen Netz voneinander unterscheiden. Dasselbe gilt für eine Slave- oder Masterstation.

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 *Hardwares Settings* 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, ...

11.5.2 Zuweisung des Masterkanals



Die Zuweisung des Masterkanals erfolgt über die Vervollständigung der *Hardware's Settings* mit einer der obigen Fboxen.

Parameter 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.

11.5.3 Zuweisung des Slavekanals

Für die Slavestation eines Ether-S-Bus-Netztes ist keine FBox SASI erforderlich. Alle erforderlichen Festlegungen wurden bereits in den *Hardware's Settings* gemacht.

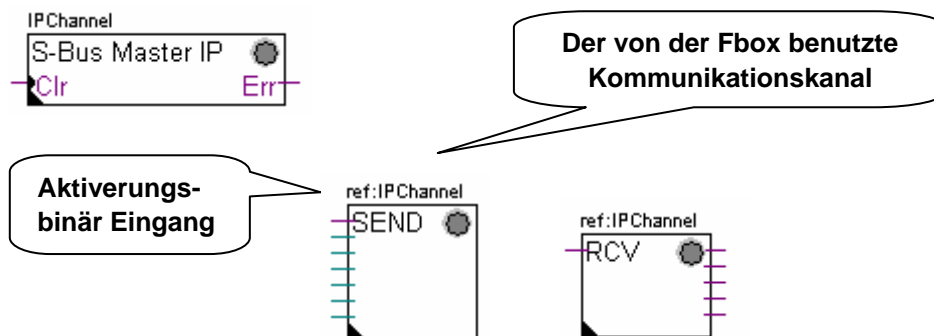
11.5.4 Prinzip eines Datenaustauschs auf einem Multimasternetzwerk.

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.

11.5.5 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 *Fbox properties Name* 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*)

Parameter 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.

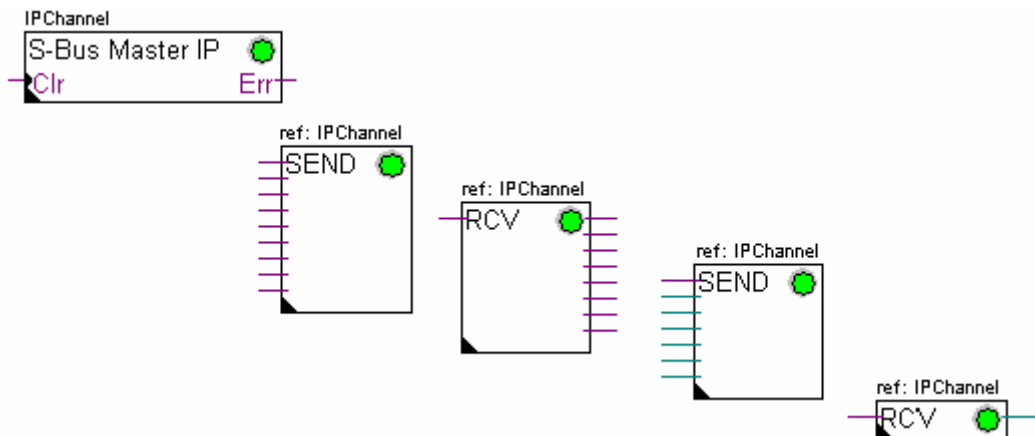
11.5.6 Diagnosen



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.

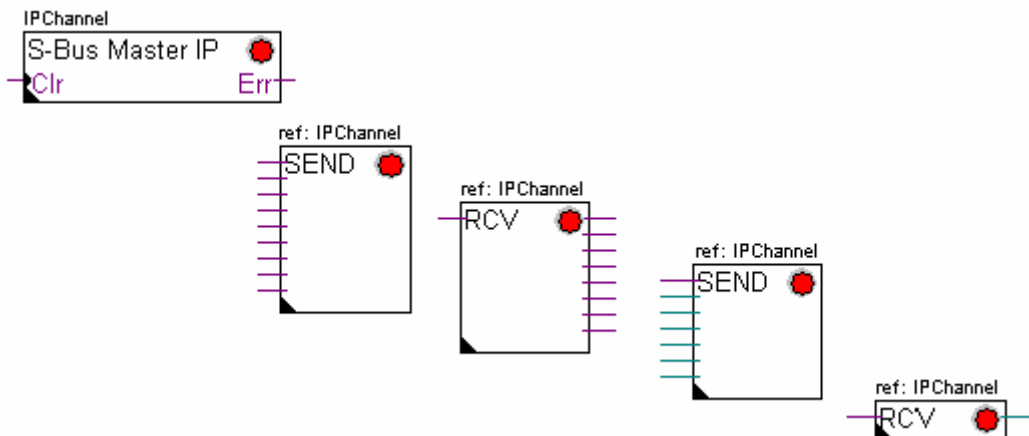
Einwandfreie Funktion

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.

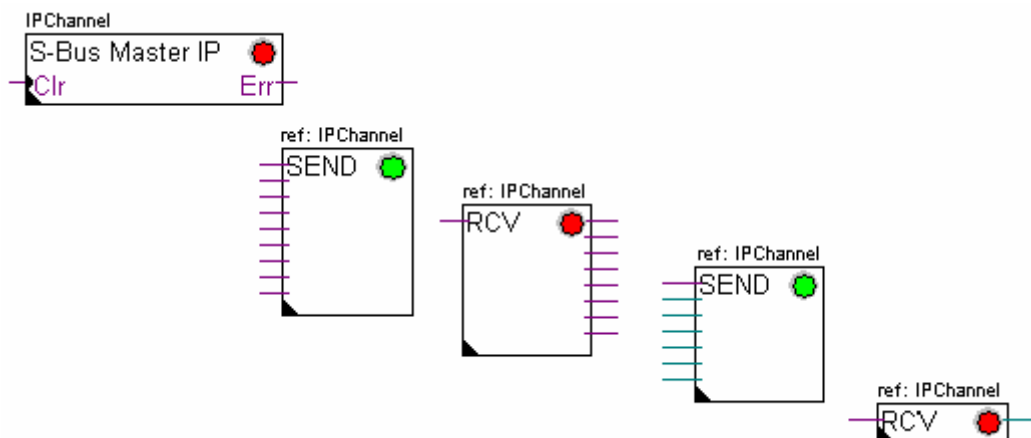


Mögliche Korrekturmaßnahmen auf der Master- oder Slavestation:

- *Hardware Settings* prüfen.
- Prüfen, ob die *Hardware Settings* in die PCD heruntergeladen worden sind.
- Prüfen, ob der mit den *Hardware Settings* 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.

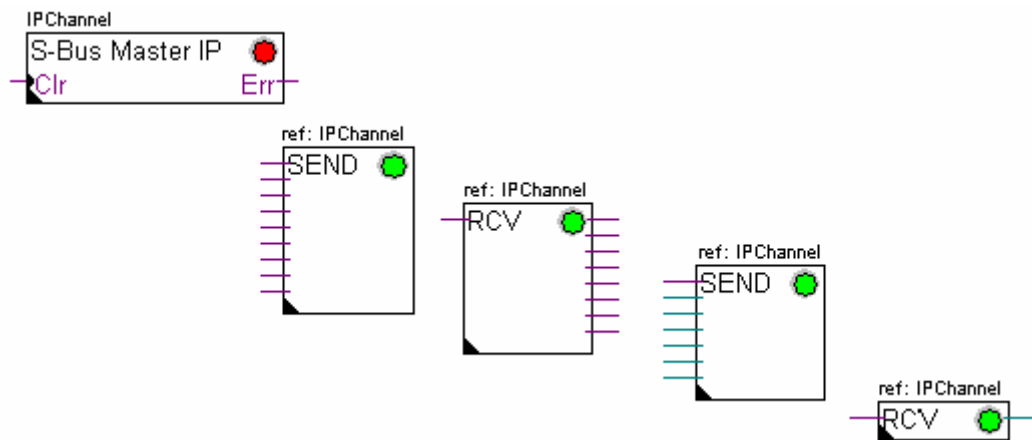
Mögliche Korrekturmaßnahmen auf der Slavestation

Für jede fehlerhafte Fbox *SEND* und *RCV* die Slavestationsnummer notieren und die entsprechenden Stationen prüfen.

- Prüfen, ob die *Hardware Settings* 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.



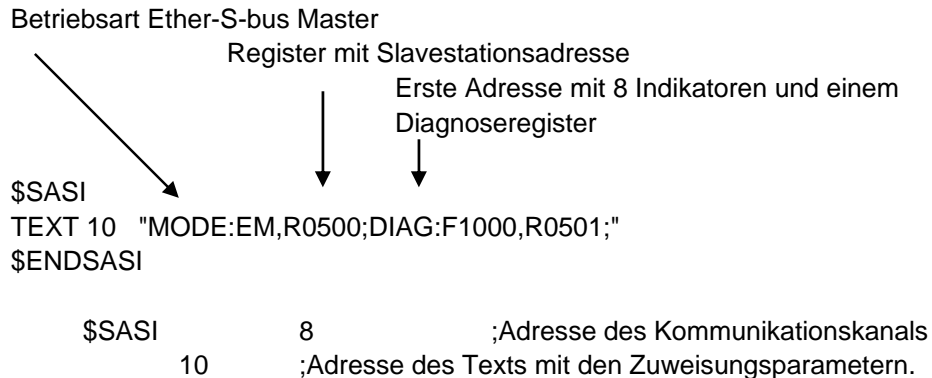
Diagnose-Fbox

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.



11.6 IL-Programm

11.6.1 Zuweisung des Masterkanals mithilfe einer SASI-Anweisung



Die Zuweisung eines Kanals erfolgt über eine zu Beginn des Programms platzierte SASI-Anweisung: Grafterc-Initialisierungssequenz oder XOB 16-Initialisierungsblock.

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 *Hardware Settings* vervollständigt werden.

11.6.2 Zuweisung des Slavekanals

Für die Slavestation eines Ether-S-Bus-Netzes ist keine SASI-Anweisung erforderlich. Alle erforderlichen Festlegungen wurden bereits in den *Hardware Settings* gemacht.

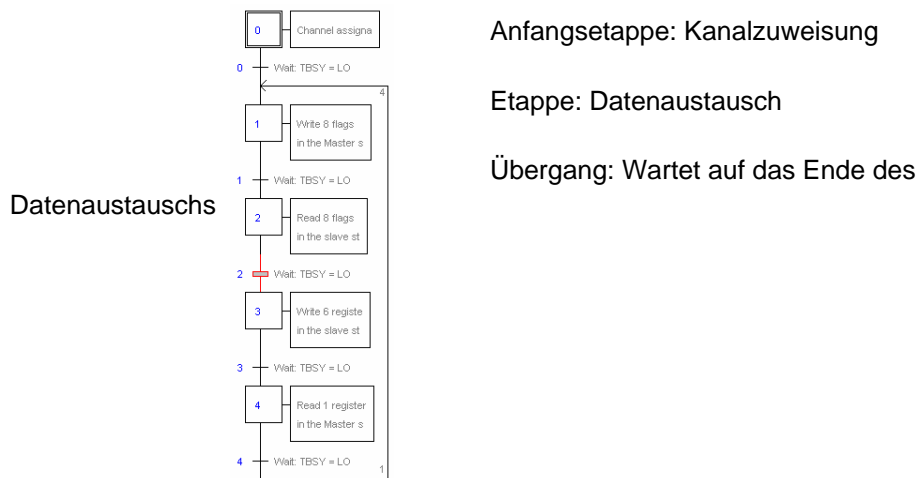
11.6.3 Prinzip eines Datenaustauschs auf einem Multimasternetzwerk.

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.

11.6.4 Datenaustausch zwischen Master- und Slavestationen.



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 (ISlavestation)
      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.

11.6.5 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:

- *Hardwares Settings* prüfen.
- Prüfen, ob die *Hardwares Settings* 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 *Hardwares Settings* 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 *Hardware Settings* 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.

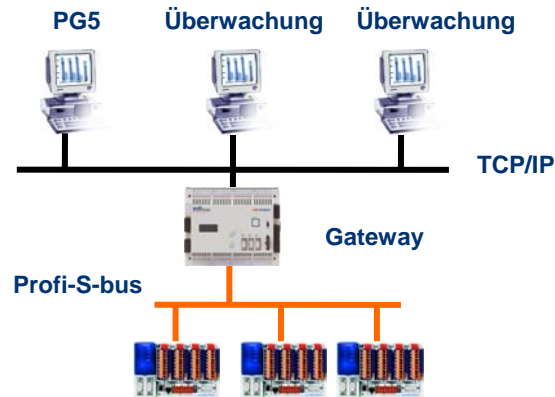
Diagnoseregister

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.

11.7 Gateway-Funktion

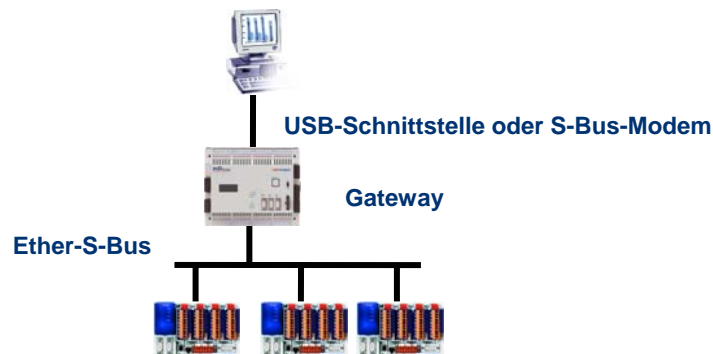
Die *Gateway*-Funktion wird häufig verwendet, damit zwei verschiedene Kommunikationssysteme zusammen kommunizieren können und zum Anpassen eines PG5-Programmierungssystems, eine Saia@Visi.Plus-Überwachung auf einem anderen als das direkt von diesen Geräten unterstützten Netzwerks.

11.7.1 Anwendung

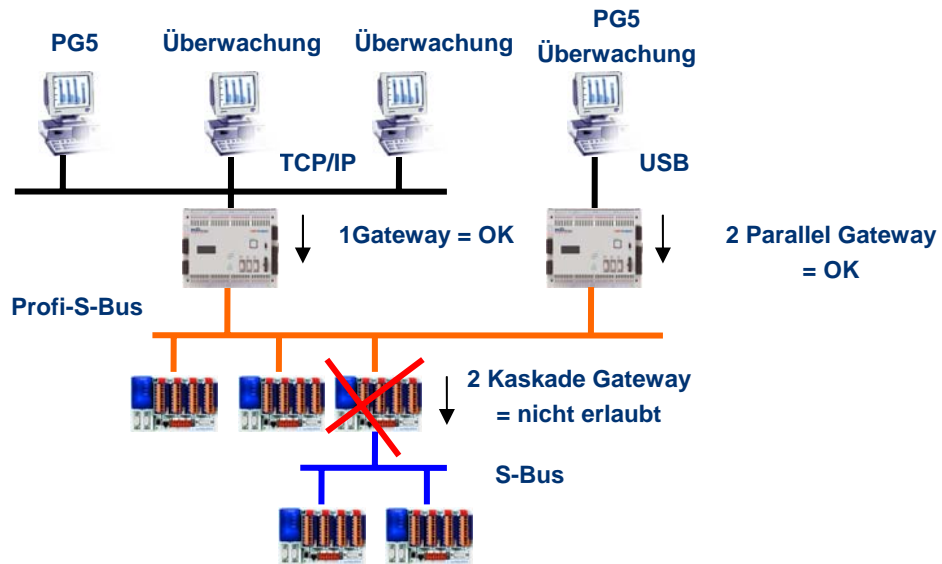


Die *Gateway*-Funktion ermöglicht die Durchführung einer Brücke zwischen zwei Netzwerken. Beispielsweise ein Ethernet-Netzwerk mit dem Profi-S-Bus-Netzwerk verbinden. Auf diese Weise tauschen die PCD-Systeme die Daten auf einem Bus aus dem Bereich Automatisierung aus, der vom EDV-Netz des Unternehmens getrennt ist. Aber die mit der PG5—Software oder einem Saia@Visi.Plus Überwachungssystem bestückten Computer können Daten mit verschiedenen PCD austauschen.

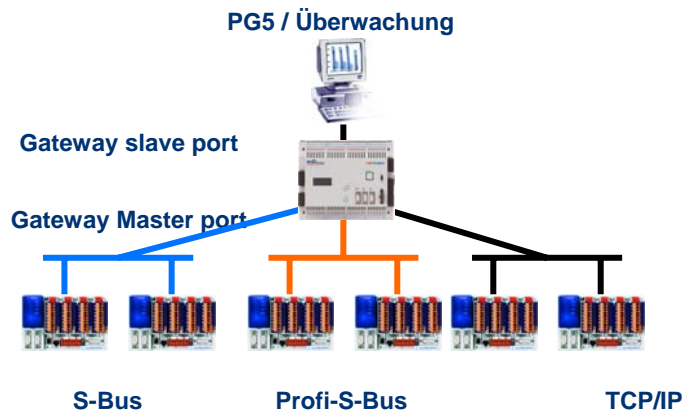
PG5 - Supervision



Die Funktion *Gateway* kann als Schnittstelle zwischen dem Kommunikationsnetzwerk und der Außenwelt dienen. Beispielsweise, eine Kommunikation über Modem herstellen oder eine USB-Kommunikationsschnittstelle.

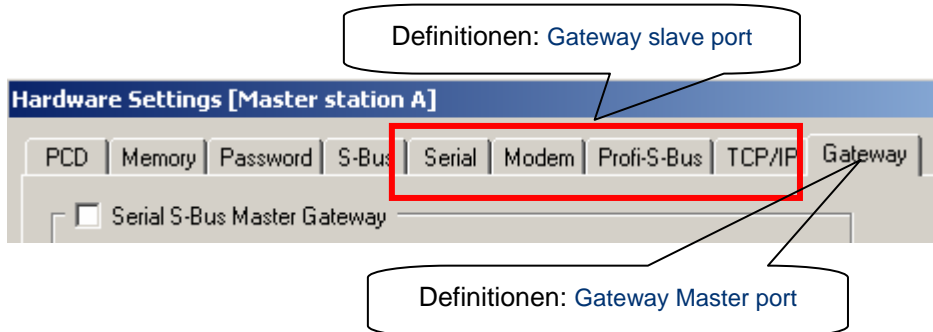


Um die Timingvorgaben der Kommunikation zu respektieren, können nicht zwei kaskadische Gateways definiert werden. Aber es ist möglich, zwei parallele Gateways im gleichen Netz zu definieren.



Falls erforderlich, kann die Funktion Gateway eine Verbindung zu verschiedenen Unterkommunikationsnetzen herstellen.

11.7.2 Konfiguration der Gateway PGU-Funktion

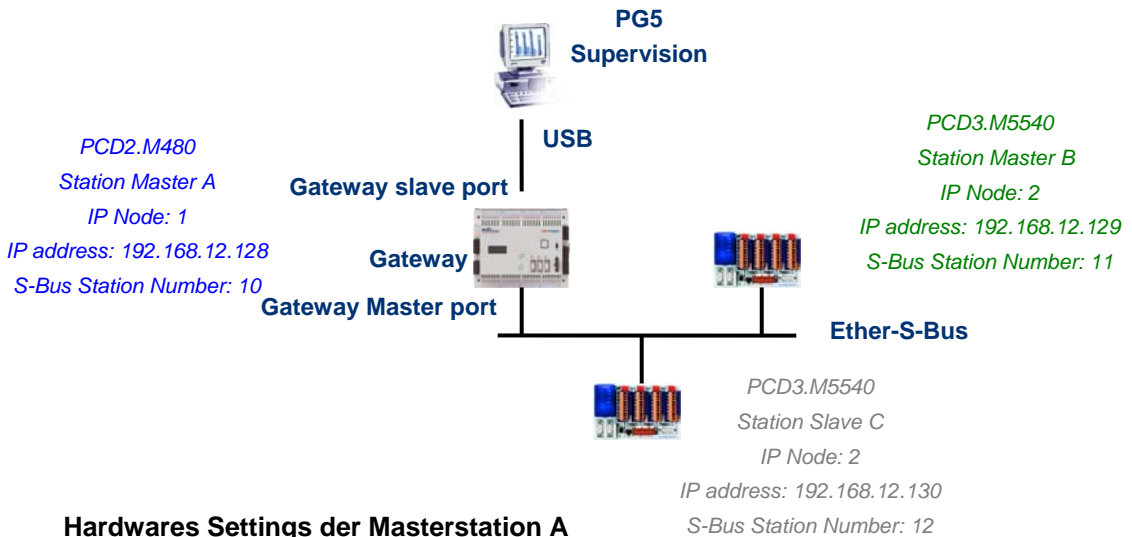


Die Gateway-Funktion lässt sich sehr leicht einrichten, sie benötigt kein Programm, sondern nur ein paar Parameter in den *Hardware Settings* der PCD.

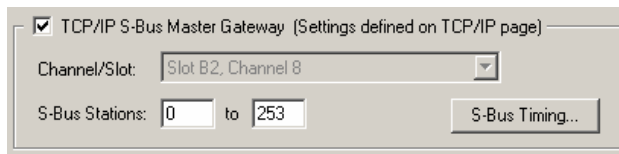
Im Allgemeinen muss man nur einen *Gateway Slave Port* und einen *Gateway Master Port* konfigurieren, anschließend wird alles automatisch von der *Gateway*-Funktion unterstützt.

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

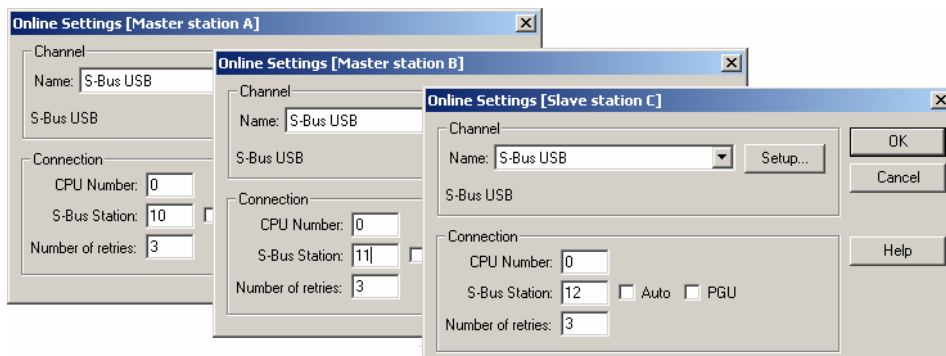


Hardware Settings der Masterstation A




Der Gateway USB ist eine Ausnahme und erfordert keinen Parameter für den *Gateway Slave port*, nur der *Gateway Master port* muss parametrisiert werden. (Nicht vergessen, die neue Konfiguration in den Master A herunterzuladen!)

Online Settings der CPU des Projekts



Zum Erstellen der USB-Kommunikation mit jeder der PCD muss man noch die *Online Settings* jeder CPU 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 - IPNode 3, Station 12

Aktivieren eines der CPU, *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 CPU aufladen und testen und gleichzeitig mit dem USB-Kabel mit der Station *Master A* verbunden bleiben.

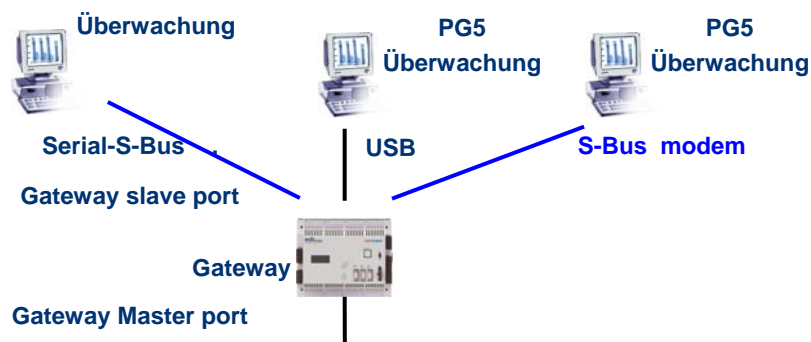
 Master station B - PCD3.M5540 - IPNode 2, Station 11

Um mit einer anderen Netzwerkstation zu kommunizieren, die CPU 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.

11.7.3 Konfiguration eines zusätzlichen Slave-Gateway-Slave-Port

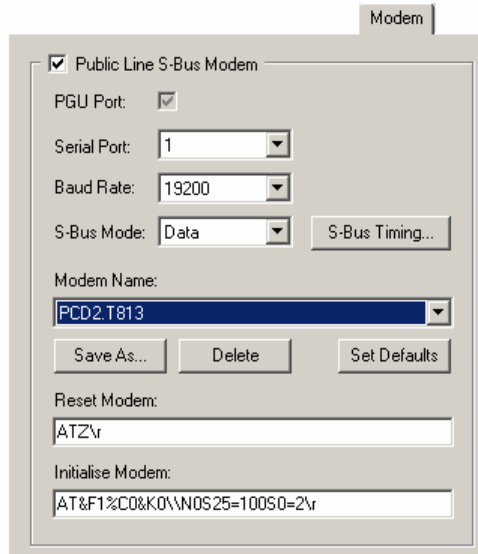


Der Gateway Slave port ist ein Mittel, um von außen in das Netzwerk zu gelangen. Falls erforderlich, kann ein zweiter oder ein dritter *Gateway Slave port* definiert werden.

Hardware Settings

Im Allgemeinen unterstützt die PCD nur einen einzigen PGU-Slavekanal. Aber die neuen SPS PCD2.M480 und PCD.3Mxxxx können mehrere auf demselben PCD unterstützen. Die Konfiguration des zweiten Gateway Slave PGU wird vollständig von den *Hardware Settings* unterstützt.

Beispiel: Hinzufügen eines zweiten Gateway Ether-S-Bus



Der zweite *Gateway Slave port PGU* wird durch Konfigurieren der *Hardware Settings* mit den Parametern für das Modem hinzugefügt.

Fupla- oder IL-Programm

Mit den alten PCD aber auch mit den neuen PCD2.M480 und PCD3.Mxxxx, 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



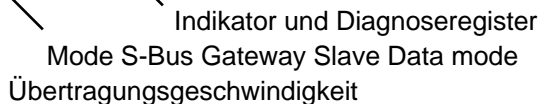
Der *Gateway*-Einstellungsparameter muss daher mit der Option *Yes* definiert werden. Je nach Kanaltyp müssen die anderen Parameter im Einstellungsfenster ebenso korrekt definiert werden.

Beispiel I: Hinzufügen eines dritten seriellen S-Bus, Ether-S-Bus

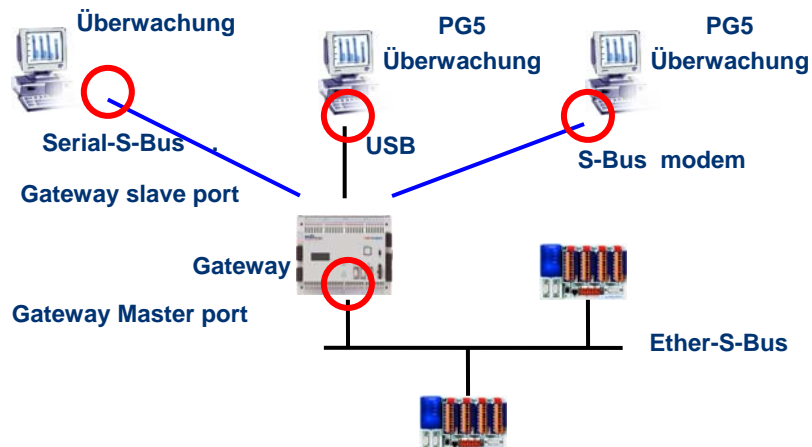
Benutzen Sie nachfolgenden Text zur Zuweisung des Kanals:

```

$SASI
TEXT 11 "UART:9600;MODE:GS2;DIAG:F1110,R0501;"
$ENDSASI
    
```

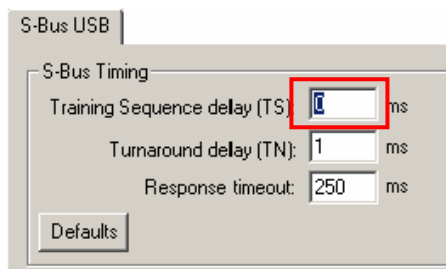


11.7.4 Kommunikationstimings



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*



11.8 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.

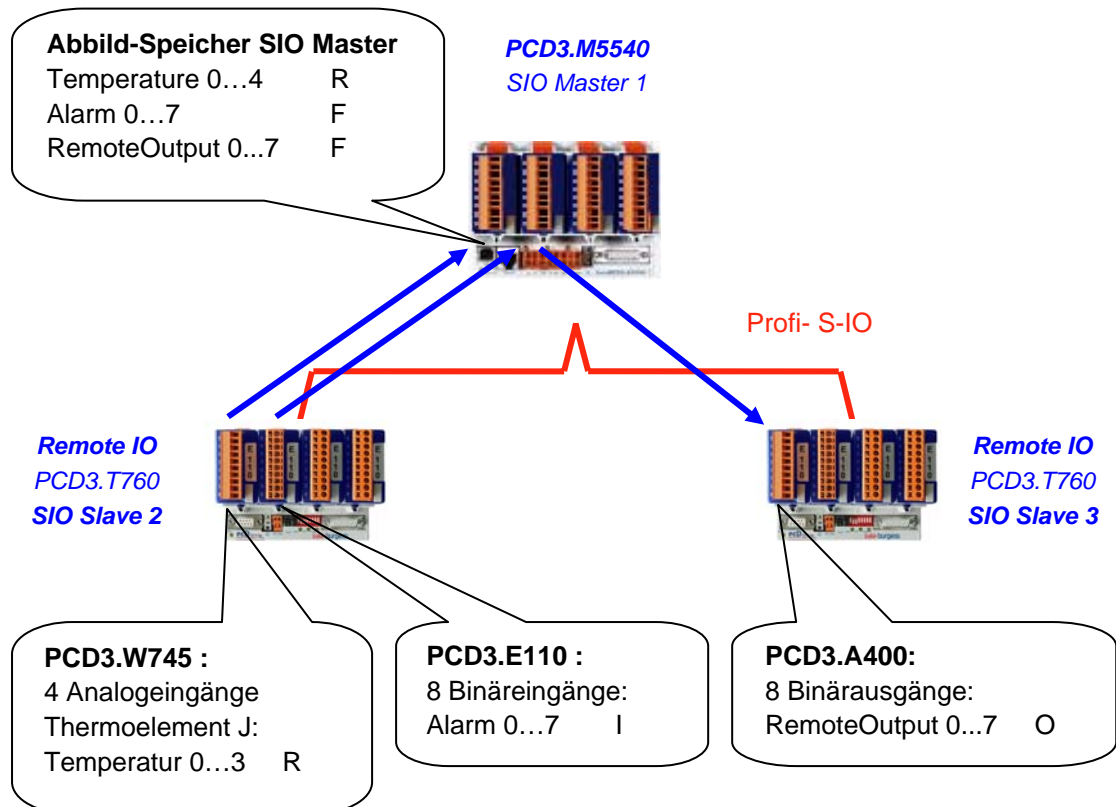
Inhaltsverzeichnis

INHALTSVERZEICHNIS	1
12 PROFI-S-IO	2
12.1 Beispiel eines Profi-S-IO.	2
12.2 Allgemeines Funktionsprinzip	2
12.3 Projekt PG5	3
12.4 Festlegung der Stationen auf dem Netzwerk	3
12.5 Konfigurieren der Masterstation	4
12.6 Konfigurieren der Slavestationen	4
12.6.1 Konfigurieren der Ein-/Ausgangsmodule.	4
12.6.2 Konfigurieren der Symbole der verlagerten Daten	5
12.6.3 Konfigurieren der Einstellungsparameter	5
12.7 Konfigurieren der Netzwerkparameter.	6
12.8 Bearbeitung der Netzwerksymbole in einem Fupla- oder IL-Programm.	6
12.9 Andere Referenzen	7

12 Profi-S-IO

Dieses Beispiel zeigt, wie man verlagerte binäre oder analogische Ein- und Ausgänge auf den RIO-Klemmen, Typ PCD3.T7xx benutzt.

12.1 Beispiel eines Profi-S-IO.



12.2 Allgemeines Funktionsprinzip

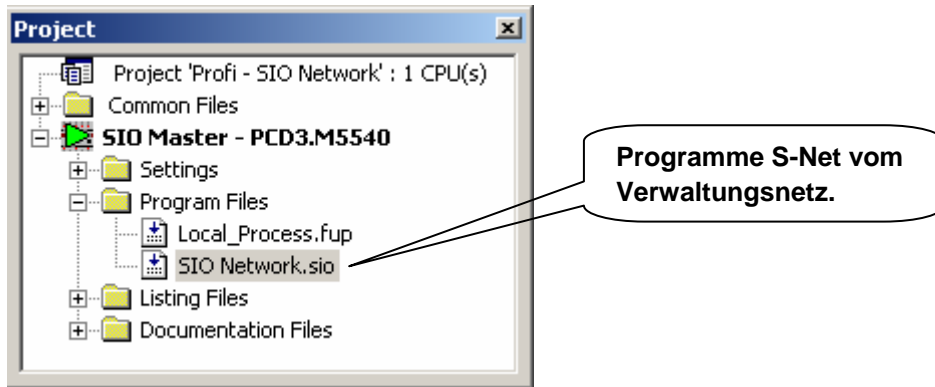
Mit den Netzwerken Profibus DP und Profi-S-IO werden die Daten auf dem Netzwerk weder mit Hilfe des Editors Fupla oder IL definiert, noch mit Hilfe der *Hardware Settings* PG5, sondern nur über den Konfigurator S-Net.

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.

12.3 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.

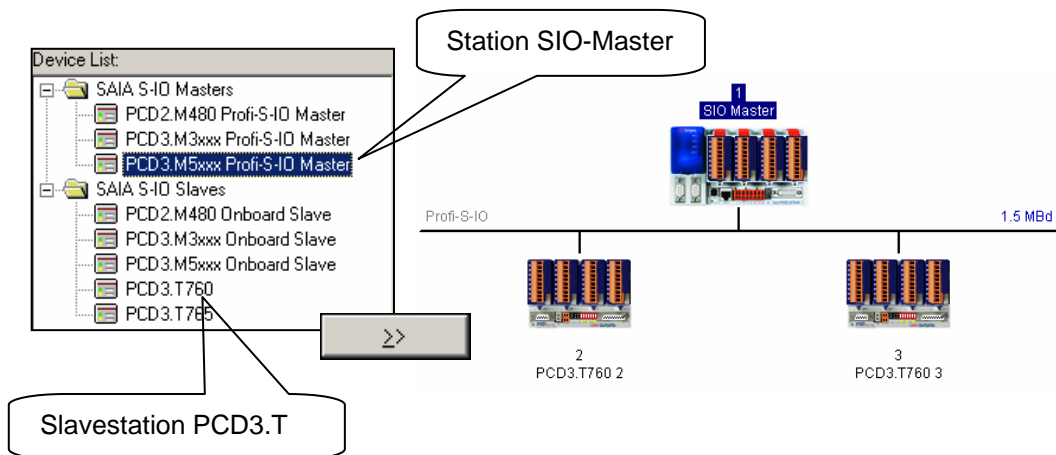
Tatsächlich ist die Benutzung des S-Net- Konfigurators für den Datenaustausch über ein Profi-S-IO und Profibus DP-Netzwerk identisch, mit Ausnahme von:

- 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:

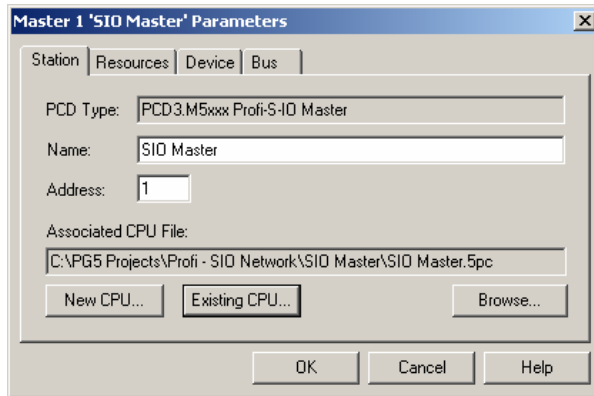
Wenn PCD7.T7xx auf dem Netzwerk vorhanden sind, immer das S-Net-Profil wählen.

12.4 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.

12.5 Konfigurieren der Masterstation

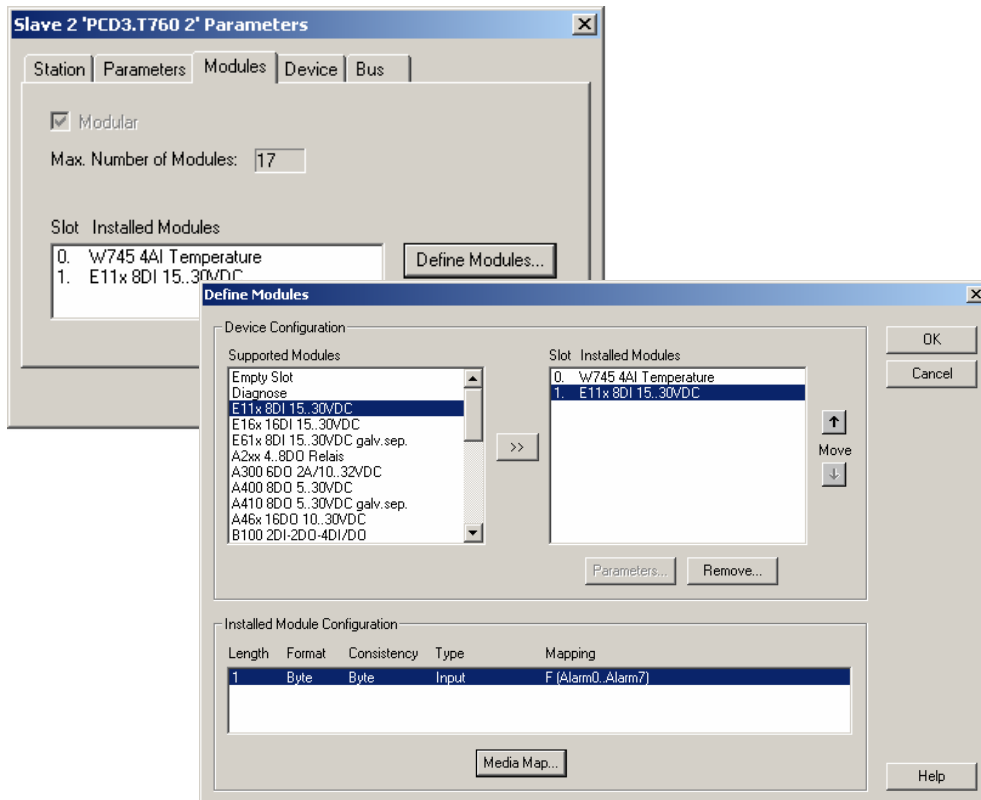


Eine einzige Information muss auf der Masterstation festgelegt werden. Dies ist der Zugangspfad zur Master-CPU in dem das S-Net das erforderliche Programm zur Ausarbeitung des Abbild-Speichers für die Masterstation einspeichern muss.

Wahlweise unterstützt dieser Dialog auch die Änderung von Namen und Adresse der CPU.

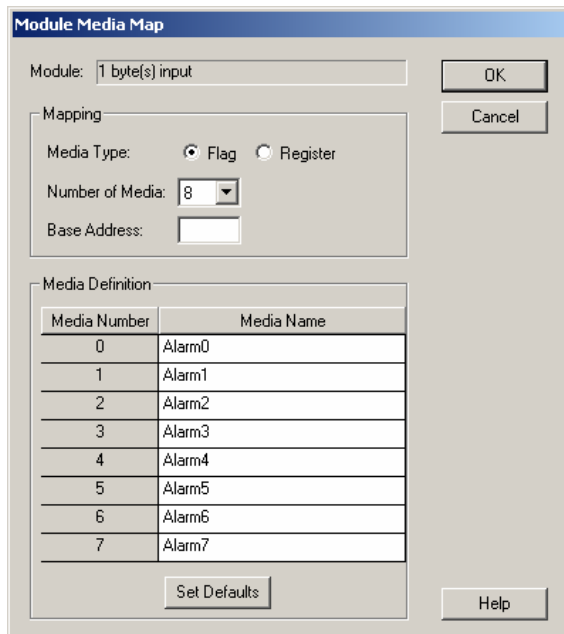
12.6 Konfigurieren der Slavestationen

12.6.1 Konfigurieren der Ein-/Ausgangsmodule.



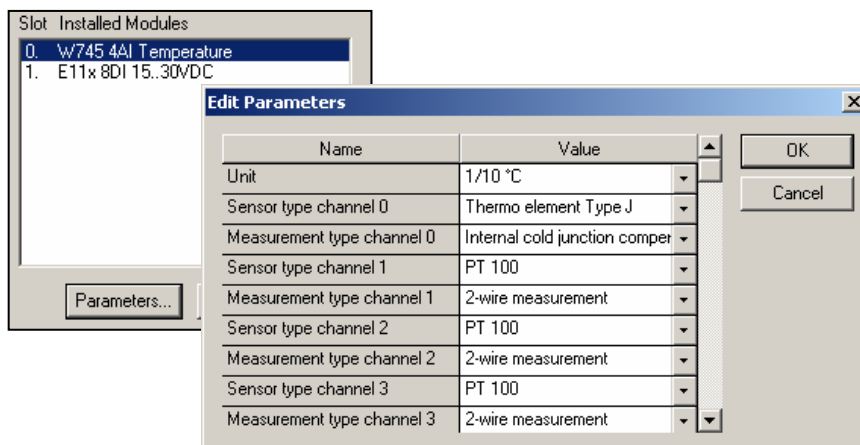
Für jedes Ein-/Ausgangsmodul auf der Slavestation, den Modultyp in der Liste *Supported modules* wählen und diesen im Fenster *Slot Installed Modules*, durch Klicken auf die entsprechende Schaltfläche hinzufügen. Achtung, die Nummer des *Slot* und der Typ des in die Slavestation eingefügten Moduls müssen immer der Nummer des *Slot* und der im Fenster *Slot Installed Modules* entsprechen!

12.6.2 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.

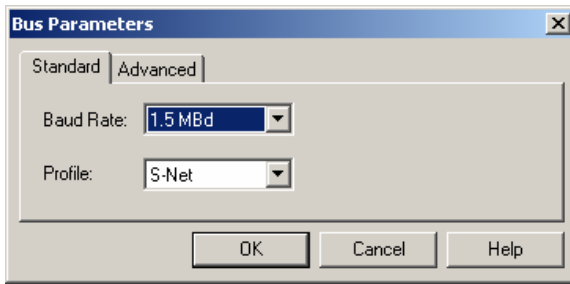
12.6.3 Konfigurieren der Einstellungsparameter



Mit einigen Modulen, wie beispielsweise Analogmessmodulen, kann es erforderlich sein, ein paar zusätzliche Parameter für die Maßumwandlung festzulegen: Maßeinheiten, Temperatursondentypen,

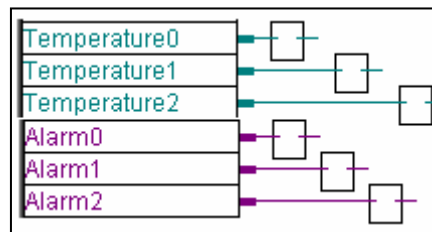
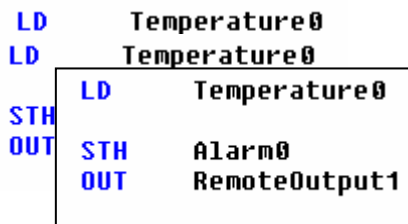
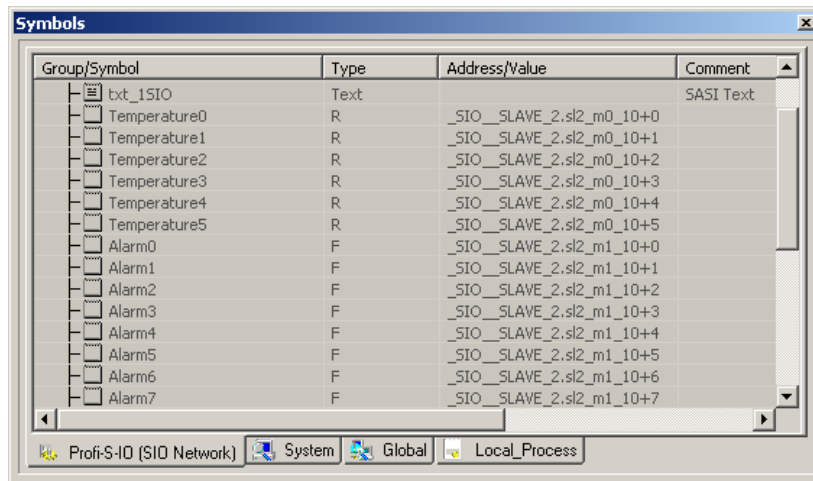
Diese Informationen können über das angezeigte Einstellungsfenster eingestellt werden, indem man das Modul markiert und auf *Parameters* klickt.

12.7 Konfigurieren der Netzwerkparameter.



Mit dem Menü *Edit Bus Parameter* kann man die Geschwindigkeit und das Profil des Kommunikationsnetzes festlegen.

12.8 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.

12.9 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 und Bestellungen

saia-burgess
Smart solutions for comfort and safety

Technische Daten

Betriebssystem	Windows 95 B Windows 98 second edition Windows NT 4.0 SP5 Windows 2000 Windows XP TCP/IP muss installiert sein TAPI 2.0 muss installiert sein
IBM kompatibler PC	Pentium 150 oder besser; 32 MB RAM oder mehr; 30 MB freie Harddisk; CD-ROM Laufwerk
PCD-Befehlssatz	alle 150 Instruktionen der PCD werden unterstützt
Standard FBoxen	die Standard-FBoxen des PG5 umfassen mehr als 250 FBoxen
Modem	Basis Modem-Konfiguration und Kommunikation sind im PG5 implementiert, Bibliotheken mit umfangreicheren Modemfunktionen wie SMS oder Pager sind verfügbar
Programmiersprachen	PG5 enthält Editoren für Instruktionsliste, FUPLA und GRAFTECProgramme
Unterstützte CPUs	sämtliche SAIA®PCD CPU werden unterstützt (gilt nicht für die Serie xx7)
Kompatibilität	PG3- und PG4-Dateien können im PG5 weiter verwendet werden
Kommunikation	TCP/IP-, SAIA®S-Bus-, PROFIBUS DP-, PROFIBUS FMS- und LONWORKS®-Kommunikation sind im PG5 vorhanden

Bestellangaben

Typ	Beschreibung
PCD8.P59 000 M9	Komplettes PG5-Paket Das Paket enthält eine Lizenzdiskette, die Dokumentation und das Programm auf CD-ROM
PCD8.P59 000 M1	PG5-Demopak Das Paket enthält die Vollversion von PG5, wobei das Drucken der Programmdateien ausgeschaltet ist und nur Programme bis zu einer Grösse von 2000 Zeilen bearbeitet werden können.

Saia-Burgess Controls AG

Bahnhofstrasse 18
CH-3280 Murten / Schweiz
Telefon 026 / 672 72 72
Telefax 026 / 672 74 99
E-mail: pcd@saia-burgess.com
Homepage: www.saia-burgess.com
Support: www.sbc-support.ch
Saia-Burgess Dreieich GmbH & Co. KG

(Zweigniederlassung der Saia-Burgess

Oldenburg GmbH & Co. KG)

Otto-Hahn-Strasse 31-33

D-63303 Dreieich

Telefon 06 103 / 89 06-0

Telefax 06 103 / 89 06 66

E-mail: sbc-info@saia-burgess.com

Homepage: www.saia-burgess-controls.de

Saia-Burgess Österreich GmbH

Linzer Bundesstrasse 101

A-5023 Salzburg

Telefon 0662 / 88 49 10

Telefax 0662 / 88 49 10 11

Niederlassung:

Zieglerstrasse 56

A-1070 Wien

Telefon 01 / 522 19 74

Telefax 01 / 522 19 74 11

E-mail: office@saia-burgess.at

Homepage: www.saia-burgess.at

Saia-Burgess Benelux B.V.

Hanzeweg 12 C,

NL-2803 MC Gouda

Telefon 0182 / 54 31 54

Telefax 0182 / 54 31 51

E-mail: office@saia-burgess.nl

Homepage: www.saia-burgess.com