SIEMENS

SIMATIC

Prozessleitsystem PCS 7 Programmieranleitung Bausteine

Handbuch

Ausgabe 04/2005
A5E00391612-02

Sicherheitstechnische Hinweise

Dieses Handbuch enthält Hinweise, die Sie zu Ihrer persönlichen Sicherheit sowie zur Vermeidung von Sachschäden beachten müssen. Die Hinweise sind durch ein Warndreieck hervorgehoben und je nach Gefährdungsgrad folgendermaßen dargestellt:

Gefahr
bedeutet, dass Tod, schwere Körperverletzung oder erheblicher Sachschaden eintreten **werden**, wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.

Warnung
bedeutet, dass Tod, schwere Körperverletzung oder erheblicher Sachschaden eintreten **können**, wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.

Vorsicht
bedeutet, dass eine leichte Körperverletzung oder ein Sachschaden eintreten können, wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.

Vorsicht
bedeutet, dass ein Sachschaden eintreten kann, wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.

Achtung
ist eine wichtige Information über das Produkt, die Handhabung des Produktes oder den jeweiligen Teil der Dokumentation, auf den besonders aufmerksam gemacht werden soll.

Qualifiziertes Personal

Inbetriebsetzung und Betrieb eines Gerätes dürfen nur von **qualifiziertem Personal** vorgenommen werden. Qualifiziertes Personal im Sinne der sicherheitstechnischen Hinweise dieses Handbuchs sind Personen, die Berechtigung haben, Geräte, Systeme und Stromkreise gemäß den Standards der Sicherheitstechnik in Betrieb zu nehmen, zu erden und zu kennzeichnen.

Bestimmungsgemäßer Gebrauch

Beachten Sie Folgendes:

Warnung

Das Gerät darf nur für die im Katalog und in der technischen Beschreibung vorgesehenen Einsatzfälle und nur in Verbindung mit von Siemens empfohlenen bzw. zugelassenen Fremdgeräten und -komponenten verwendet werden.

Der einwandfreie und sichere Betrieb des Produktes setzt sachgemäßen Transport, sachgemäße Lagerung, Aufstellung und Montage sowie sorgfältige Bedienung und Instandhaltung voraus.

Marken

SIMATIC®, SIMATIC HMI® und SIMATIC NET® sind Marken der Siemens AG.

Die übrigen Bezeichnungen in dieser Schrift können Marken sein, deren Benutzung durch Dritte für deren Zwecke die Rechte der Inhaber verletzen kann.

Haftungsausschluss
Wir haben den Inhalt der Druckschrift auf Übereinstimmung mit der beschriebenen Hard- und Software geprüft. Dennoch können Abweichungen nicht ausgeschlossen werden, so dass wir für die vollständige Übereinstimmung keine Gewähr übernehmen. Die Angaben in dieser Druckschrift werden regelmäßig überprüft, und notwendige Korrekturen sind in den nachfolgenden Auflagen enthalten. Für Verbesserungsvorschläge sind wir dankbar.

Siemens AG
Bereich Automation and Drives
Geschäftsgebiet Industrial Automation Systems
Postfach 4848, D- 90327 Nürnberg

Siemens Aktiengesellschaft

A5E00391612-02

Vorwort

Zweck der Programmieranleitung

Diese Programmieranleitung beschreibt, wie Sie PCS 7-konforme AS-Bausteine oder Bildbausteine erstellen.

Die wesentlichen Punkte, die einen PCS 7-konformen AS-Baustein von einem reinen S7-Baustein unterscheiden, sind:

- Die Möglichkeit, Parameterwerte über einen Bildbaustein zu **beobachten.**
- Die Möglichkeit, Parameterwerte und damit das Verhalten des Bausteins über einen Bildbaustein zu **bedienen**.
- Die Möglichkeit, asynchron auftretende Ereignisse und Bausteinzustände an die OS zu **melden** und dort über einen Bildbaustein oder eine WinCC-Meldeliste anzuzeigen.

Leserkreis

Diese Programmieranleitung wendet sich an Entwickler von Automatisierungsbausteinen (AS-Bausteine) und/oder Bildbausteinen, die zusammen mit den von Siemens gelieferten leittechnischen PCS 7-Bausteinen in derselben Anlage durchgängig und aufeinander abgestimmt verwendet werden sollen.

Voraussetzungen

Voraussetzungen sind daher Vorkenntnisse in der Entwicklung und der Anwendung von AS- und Bildbausteinen und der hierfür verwendeten Hard- und Software. Im Weiteren werden nur die Punkte beschrieben, die für die Konformität eines Bausteins mit den PCS 7-Bausteinen notwendig sind.

Allgemeine Informationen über die Verwendung von PCS 7-Komponenten können Sie dem PCS 7-Projektierungshandbuch entnehmen.

Vorgehensweise

Die Programmieranleitung bietet Ihnen einen Überblick über die einzelnen Bestandteile eines PCS 7-konformen Bausteins. Sie orientiert sich dabei an der Reihenfolge, in der Sie vorgehen, wenn Sie Funktions- und Bildbausteine entwickeln.

- Schritt für Schritt entwickeln Sie den AS-Baustein ”CONTROL”, einen einfachen Reglerbaustein. Dazu definieren Sie nacheinander den Bausteinkopf, die Parameter des Bausteins und seine lokalen Variablen. Dann erstellen Sie den Quellcode.
- Als Nächstes entwickeln Sie einen Bildbaustein. Diesen erstellen Sie mit dem WinCC Graphic Designer und den Elementen des Faceplate Designers.
- Abschließend entwerfen Sie für den Baustein eine Online-Hilfe und erstellen aus allen Komponenten eine lieferfähige Bibliothek MYLIB.

In jedem Kapitel werden nur die für das Verständnis der jeweils beschriebenen Punkte benötigten Teile des Beispiels angegeben. Das gesamte Beispiel des AS-Bausteins ist im Kapitel 1.11 abgedruckt.

Im Anhang sind als Beispiele für PCS 7-konforme Bausteine der Quellcode von den in der PCS 7 Library enthaltenen Bausteinen MEAS_MON, MOTOR und VALVE dargestellt. Diesen Quellcode - oder Teile daraus - können Sie als Vorlage für eigene Bausteine verwenden.

Hinweis

Die Verwendung des im Anhang enthaltenen Beispiel-Quellcodes liegt in Anwenderverantwortung. Eine Gewähr auf fehlerfreie Darstellung besteht nicht.

Weitere Unterstützung

Bei Fragen zur Nutzung der im Handbuch beschriebenen Produkte, die Sie hier nicht beantwortet finden, wenden Sie sich bitte an Ihren Siemens-Ansprechpartner in den für Sie zuständigen Vertretungen und Geschäftsstellen.

Ihren Ansprechpartner finden Sie unter:

http://www.siemens.com/automation/partner

Den Wegweiser zum Angebot an technischen Dokumentationen für die einzelnen SIMATIC Produkte und Systeme finden Sie unter:

http://www.siemens.de/simatic-tech-doku-portal

Den Online-Katalog und das Online-Bestellsystem finden Sie unter:

http://mall.ad.siemens.com/

Trainingscenter

Um Ihnen den Einstieg in das Prozessleitsystem SIMATIC PCS 7 und das Automatisierungssystem S7 zu erleichtern, bieten wir entsprechende Kurse an. Wenden Sie sich bitte an Ihr regionales Trainingscenter oder an das zentrale Trainingscenter in D 90327 Nürnberg.

Telefon: +49 (911) 895-3200.

Internet: http://www.sitrain.com

Technical Support

Sie erreichen den Technical Support für alle A&D-Produkte

- Über das Web-Formular für den Support Request http://www.siemens.de/automation/support-request
- Telefon: + 49 180 5050 222
- Fax: + 49 180 5050 223

Weitere Informationen zu unserem Technical Support finden Sie im Internet unter http://www.siemens.com/automation/service

Service & Support im Internet

Zusätzlich zu unserem Dokumentations-Angebot bieten wir Ihnen im Internet unser komplettes Wissen online an.

http://www.siemens.com/automation/service&support

Dort finden Sie:

- den Newsletter, der Sie ständig mit den aktuellsten Informationen zu Ihren Produkten versorgt.
- die für Sie richtigen Dokumente über unsere Suche in Service & Support.
- ein Forum in welchem Anwender und Spezialisten weltweit Erfahrungen austauschen.
- Ihren Ansprechpartner für Automation & Drives vor Ort.
- Informationen über Vor-Ort Service, Reparaturen, Ersatzteile. Vieles mehr steht für Sie unter dem Begriff "Leistungen“ bereit.

Inhaltsverzeichnis

1 AS-Bausteine erstellen

1.1 Voraussetzungen und Vorkenntnisse

Die hier beschriebenen Bausteine sind für die Verwendung mit PCS 7 ab V6.1 auf einer CPU S7-4xx gedacht. Für die Erstellung der Bausteine benötigen Sie die folgenden Softwarepakete:

- STEP 7 Basis ab V5.2
- SCL-Compiler ab V5.1 SP4
- CFC ab V6.0

AS-Bausteine für PCS 7 werden mit der Programmiersprache SCL erstellt. Daher wird im Folgenden nur dieser Weg beschrieben. Weitere Informationen zu SCL entnehmen Sie bitte:

- der Online-Hilfe im SIMATIC Manager
  (Aufruf von Hilfen zu Optionspaketen > Bausteine programmieren mit S7-SCL).
- dem Handbuch "S7-SCL Erste Schritte"
- dem Handbuch "S7-SCL für S7-300 und S7-400"

Die Handbücher finden Sie unter "Start > Simatic > Dokumentation".

Hinweise zu Anwender-Datenbausteine

Sollen außer den Funktionsbausteinen (FB) auch eigene Datenbausteine (DB) erstellt werden (nicht Bestandteil dieser Anleitung), beachten Sie bitte Folgendes:

Verwenden Sie eine DB-Nummer im Bereich von 1 bis 60. Dieser Nummernbereich wird im CFC für andere Applikationen reserviert. Damit wird sichergestellt, dass die vom CFC beim Übersetzen erzeugten Instanz-DBs nicht mit den Nummern der Anwender-DBs kollidieren.

Ein späteres Verändern des voreingestellten Nummernbereichs im CFC (Extras > Einstellungen für Übersetzen/Laden) führt dazu, dass anschließend nur noch ein Gesamtladen im Stoppzustand der CPU durchgeführt werden kann.

1.1.1 Mitgelieferter Beispiel-Baustein

Der hier beschriebene Baustein "CONTROL" wird als SCL-Quelle CONTx:SCL auf der PCS 7 Toolset-CD mitgeliefert und im Pfad
...\STEP7\EXAMPLES\zdt25_01\S7_CONTA.SCL (Deutsch)
...\STEP7\EXAMPLES\zen25_01\S7_CONTB.SCL (Englisch)
...\STEP7\EXAMPLES\zfr25_01\S7_CONTC.SCL. (Englisch, bei französischer Installation) installiert.

Um sie in Ihr Projekt zu übernehmen, gehen Sie folgendermaßen vor:

1. Selektieren Sie in Ihrem Projekt den Quellordner und wählen Sie **Einfügen > Externe Quelle...** . Im Dialogfeld "Externe Quelle einfügen" gehen Sie in der Verzeichnisstruktur bis zum Ablageort der SCL-Quelle (S7_CONTA.SCL bzw. S7_CONTB.SCL), selektieren sie und klicken auf "Öffnen".

   Die SCL-Quelle befindet sich nun im Quellordner und muss mit dem SCL-Compiler übersetzt werden.

2. Stellen Sie vor dem Übersetzen sicher, dass sich der Baustein "OP_A_LIM" (FB 46) im Bausteinordner Ihres Projekts befindet. Ist dies nicht der Fall, kopieren Sie ihn aus der Bibliothek "PCS 7 Library V61" in Ihr Projekt.

3. Überprüfen Sie die Voreinstellung des SCL-Compilers (siehe Kapitel 1.2.1).

4. Tragen Sie in der Symboltabelle den symbolischen Namen "CONTROL" und als Adresse FB 501 ein und speichern Sie den Eintrag.
   (Empfohlen werden die Nummern ab 500 aufwärts, um Kollisionen mit den Nummern der PCS 7-Standardbausteine zu vermeiden).
   Siehe auch Abschnitt "Eintrag in die Symboltabelle" in Kapitel 1.2.2

5. Öffnen Sie mit Doppelklick die SCL-Quelle "S7_CONTA" bzw. "S7_CONTB", starten Sie die Übersetzung und beenden Sie den SCL-Compiler nach fehlerfreiem Durchlauf.

Der Beispiel-Baustein **FB 501** befindet sich nun im Bausteinordner Ihres Projekts.

1.2 Aufbau eines AS-Bausteins

Damit ein AS-Baustein im PCS 7-Umfeld lauffähig ist, muss er bestimmte formale und inhaltliche Kriterien erfüllen. Die folgenden Kapitel beschreiben, was Sie zur Erfüllung dieser Kriterien beachten bzw. unternehmen müssen.

Das untenstehende Blockbild zeigt den prinzipiellen Aufbau des Bausteins "CONTROL" (FB 501). Die einzelnen Elemente des Bausteins werden in den angegebenen Kapiteln genauer erklärt.

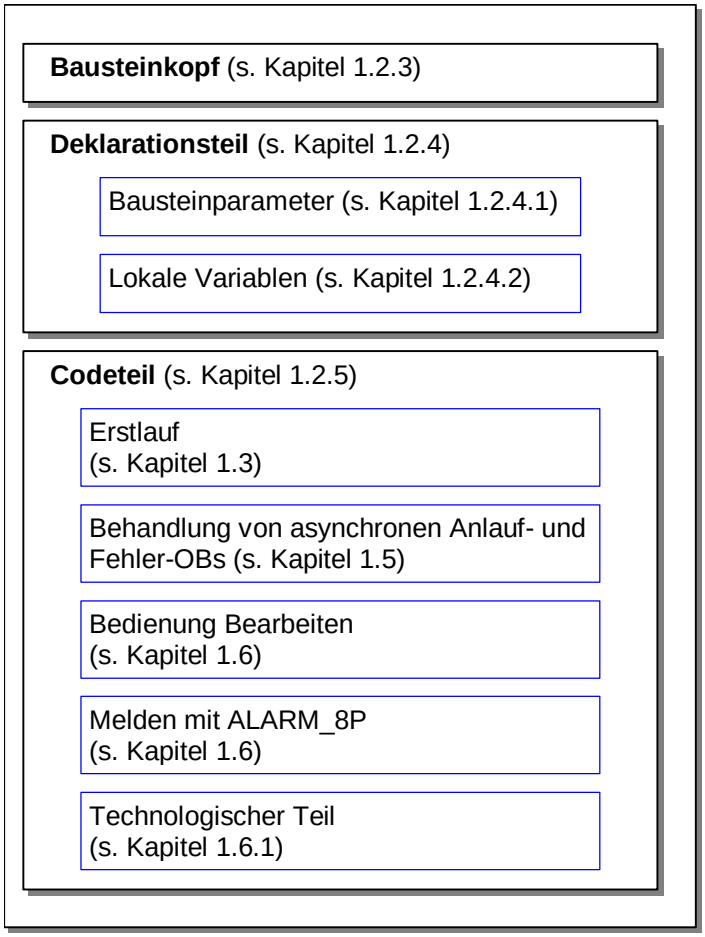


Bild 1-1: Aufbau des Beispiel-Bausteins "CONTROL" (FB 501)

Funktionsbaustein oder Funktion

Falls Ihr Baustein Werte speichern, melden oder bedien- und beobachtbar sein soll, müssen Sie ihn als Function block (FB) realisieren. Ein FB hat ein Gedächtnis in Form eines Datenbausteins (DB), auch Instanzdaten genannt.

Falls Sie dies nicht benötigen, können Sie Ihren Baustein auch als Function (FC) realisieren.

1.2.1 Voreinstellungen im SCL-Compiler

Voreinstellungen für "Bausteine erzeugen"

Im SCL-Compiler sind unter **Extras > Einstellungen > Bausteine erzeugen** folgende Optionen einstellbar:

- **Bausteine überschreiben**

  Überschreibt bereits existierende Bausteine im Ordner "Bausteine" eines S7-Programms, falls beim Übersetzungsvorgang Bausteine mit der gleichen Bezeichnung erzeugt werden.

  Ebenso werden beim Laden Bausteine mit gleichem Namen, die bereits im Zielsystem vorhanden sind, überschrieben.

  Wenn Sie diese Option nicht gewählt haben, müssen Sie eine Meldung bestätigen, bevor der Baustein überschrieben wird.

- **Warnungen anzeigen**

  Legt fest, ob nach einem Übersetzungslauf zusätzlich zu den Fehlern auch Warnungen gemeldet werden sollen.

- **Fehler vor Warnungen**

  Legt fest, ob in der Meldung die Fehler vor den Warnungen aufgelistet werden.

- **Referenzdaten erzeugen**

  Wählen Sie diese Option, wenn bei der Erzeugung eines Bausteins automatisch Referenzdaten erzeugt werden sollen.

  Über den Menübefehl **Extras > Referenzdaten** haben Sie jedoch die Möglichkeit, die Referenzdaten später zu erzeugen oder zu aktualisieren.

- **Systemattribut "S7_server" berücksichtigen**

  Wählen Sie diese Option, wenn bei der Erzeugung eines Bausteins das Systemattribut für Parameter "S7_server“ berücksichtigt werden soll. Dieses Attribut vergeben Sie, wenn der Parameter für die Verbindungs- oder Meldungsprojektierung relevant ist. Der Parameter enthält die Verbindungs- bzw. die Meldungsnummern.

Hinweis für dieses Beispiel:

Das Optionskästchen **Systemattribut "S7_server" berücksichtigen** müssen Sie unbedingt setzen, da dieser Baustein Meldungen enthält. Beim Importieren bzw. beim Einfügen des Bausteins in einen CFC-Plan würde dieser Vorgang sonst mit Fehlermeldung abgebrochen.

Voreinstellungen für "Compiler"

Im SCL-Compiler können Sie unter **Extras > Einstellungen > Compiler** die drei Optionskästchen

- "Feldgrenzen überwachen"
- "Debug Info erstellen"
- "OK Flag setzen"

an- oder abwählen. Die restlichen Optionskästchen sollten Sie immer angewählt lassen. Nähere Informationen zu den einzelnen Optionen können Sie dem SCL-Handbuch entnehmen.

Bei der Entscheidung, ob Sie diese Optionen anwählen oder nicht, müssen Sie die folgenden Punkte beachten:

- **Feldgrenzen überwachen**

  Bei Einsatz von Arrays im Programm wird zur Laufzeit überprüft, ob Laufindex und Feldlängen-Vereinbarung im zulässigen Bereich liegen. Im Fehlerfall wird das OK-Flag beeinflusst und der ENO-Ausgang zurückgesetzt. Diese Überprüfung ist eine sehr laufzeitintensive Aktion.

  Falls Sie Arrays verwenden, sollten Sie die Option nur solange angewählt lassen, bis Ihr Baustein ausreichend getestet ist und sicher ist, dass Index und Feldlänge zusammenpassen.

- **Debug Info erstellen**

  Diese Option ermöglicht es einen Testlauf mit dem Debugger durchzuführen, nachdem das Programm übersetzt und in die CPU geladen ist. Der Speicherbedarf des Programmes und die Ablaufzeiten im AS erhöhen sich jedoch durch diese Option. Sie sollten sie daher nur während der Testphase des Bausteins anwählen, nicht aber in der Lieferversion.

- **OK Flag setzen**

  Das OK-Flag ist eine systeminterne Variable. Tritt während der Ausführung einer Operation ein Fehler auf, z.B. Überlauf bei arithmetischen Operationen, so wird das OK-Flag vom System beeinflusst und an den Ausgang ENO durchgereicht. Diese Überprüfung ist sehr laufzeitintensiv. Aus diesem Grund ist es sinnvoll, das automatische Setzen des OK-Flags abzuschalten und stattdessen unzulässige Operationen / Bereichsüberschreitungen im Bausteinalgorithmus selbst abzufangen. Im Fehlerfall können Sie das OK-Flag dann explizit setzen, wenn Sie den ENO-Ausgang zur Weiterverschaltung benutzen möchten. (Dies wird vom System übernommen und kostet keine Performance, da der Zustand des OK-Flags stets vom System an den Ausgang durchgereicht wird.)

1.2.2 Voreinstellungen im SIMATIC Manager

Bei PCS 7-konformen AS-Bausteinen sollte die Schnittstelle zum Anwender (Parameternamen, Kommentare usw.) in englischer Sprache sein. Die Bausteine selbst können Sie in jeder beliebigen Landessprache entwickeln.

Auswahl der Landessprache

Falls Sie Ihre Bausteine in einer Bibliothek zusammenfassen wollen (vgl. Kapitel 4.1), muss auch die "Landessprache" auf Englisch eingestellt sein, damit die einzelnen Kataloge Ihrer Bibliothek PCS 7-konforme Namen erhalten (**Sources**, **Symbols** und **Blocks**). Hierzu müssen Sie im SIMATIC Manager über **Extras > Einstellungen > Sprache** als Landessprache und als Mnemonic "English" einstellen.

Eintrag in der Symboltabelle

Der Name des Bausteins, der im Bausteinkopf eingetragen wird (wie nachfolgend beschrieben), muss in der Symboltabelle als symbolischer Name eingetragen werden.

1. Zum Eintragen öffnen Sie im S7-Programm mit Doppelklick auf "Symbole" die Symboltabelle.
2. Geben Sie in der Spalte "Symbol" den symbolischen Namen (hier: "CONTROL") ein.
3. Tragen Sie in der Spalte "Adresse" eine FB-Nummer ein (hier: FB 501).
4. Als Kommentar tragen Sie in der Spalte "Kommentar" einen Text ein, der die Funktion des Bausteins näher bezeichnet (max. 80 Zeichen).

   Der Kommentar ist eine Ergänzung zum symbolischen Bausteinnamen, da dieser allein in der Regel nicht genügend über den Verwendungszweck bzw. über die Funktionalität des Bausteins aussagt.
   Dieser Bausteinkommentar ist identisch mit dem Symbolkommentar, der im SIMATIC Manager angezeigt wird (in der Detailsicht oder den Objekteigenschaften des Bausteins).

   Beim Einfügen des Bausteins in einen CFC-Plan wird der Text als Bausteinkommentar im Kopf des Instanzbausteins dargestellt und kann, unabhängig vom Eintrag in der Symboltabelle, instanzspezifisch geändert werden.
   Hinweis: Je nach Darstellungsbreite der CFC-Bausteine wird nur eine entsprechend begrenzte Anzahl Zeichen dargestellt. Die gesamte Kommentarlänge wird aber temporär in der Kurzinformation (Mauszeiger auf Bausteinkopf positioniert) angezeigt.
5. Sichern und schließen Sie die Symboltabelle.

Siehe auch Abschnitt 1.10, Namenskonventionen und Nummernbereich.

1.2.3 Bausteinkopf

Der Bausteinkopf enthält die Verwaltungsinformationen (im weiteren Baustein-Attribute genannt) des Bausteins. Diese Attribute werden von den einzelnen PCS 7-Tools für verschiedene Zwecke genutzt. Sie werden im SIMATIC Manager in den Objekteigenschaften des Bausteins angezeigt und können dort auch geändert werden (vgl. dazu Attribut KNOW_HOW_PROTECT).

Ausschnitt des Beispiel-Bausteins:

```
//Copyright (C) Siemens AG 1999. All Rights Reserved. Confidential
(*******************************************************************************
  KURZBESCHREIBUNG:

  Dieser Baustein gibt Ihnen ein Beispiel f¸r die Entwicklung eines PCS 7-konformen
  AS-Bausteins.

  Er realisiert einen einfachen Regelalgorithmus nach der Formel:
  Stellgr^fle = Verst‰rkungsfaktor * (Sollwert - Istwert)

  ‹berschreitet der Prozesswert die obere Alarmgrenze, wird der Fehlerausgang QH_ALM
  gesetzt. Zudem wird eine Meldung an die OS mit ALARM_8P generiert. Mit der Variablen
  M_SUP_AH kann die Meldung unterdr¸ckt werden.

  Unterschreitet der Prozesswert die untere Alarmgrenze, wird der Fehlerausgang QL_ALM
  gesetzt. Zudem wird eine Meldung an die OS mit ALARM_8P generiert. Mit der Variablen
  M_SUP_AL kann die Meldung unterdr¸ckt werden.

  Der Baustein unterst¸tzt SIMATIC BATCH und besitzt daher die daf¸r n^tigen Parameter
  BA_EN, BA_NA, BA_ID, OCCUPIED und STEP_NO.

  Zur Verdeutlichung einer Zeitverz^gerung enth‰lt der Baustein zus‰tzliche Eing‰nge:
  Der Ausgang SUPP_OUT folgt dem Eingang SUPP_IN nach einer parametrierbaren Wartezeit
  SUPPTIME.

  *******************************************************************************)

//Ersteller: ABC       Datum:  13.08.00       Vers.:1.00
//Ge‰ndert:            Datum:  18.11.03       Vers.:
//fnderung:

//*******************************************************************************
// Bausteinkopf
//*******************************************************************************

FUNCTION_BLOCK "CONTROL"
TITLE =        'CONTROL'

{  // Liste der Systemattribute
S7_tasklist:= 'OB80,OB100';  // Baustein wird bei Zeitfehler u. Neustart aufgerufen
S7_m_c:=    'true';          // Bausteins ist bedien- und beobachtbar
S7_alarm_ui:= '1'            // Einstellung PCS7-Meldedialog ('0'=Standard-Meldedialog)
}
AUTHOR:   ABC
NAME:       CONTROL
VERSION:         '0.02'
FAMILY:   XYZ
KNOW_HOW_PROTECT
```

Die folgenden beiden Bilder zeigen die Objekteigenschaften des übersetzten Beispiel-Bausteins mit Verweisen auf die jeweils betroffenen Attribute des Bausteinkopfs.

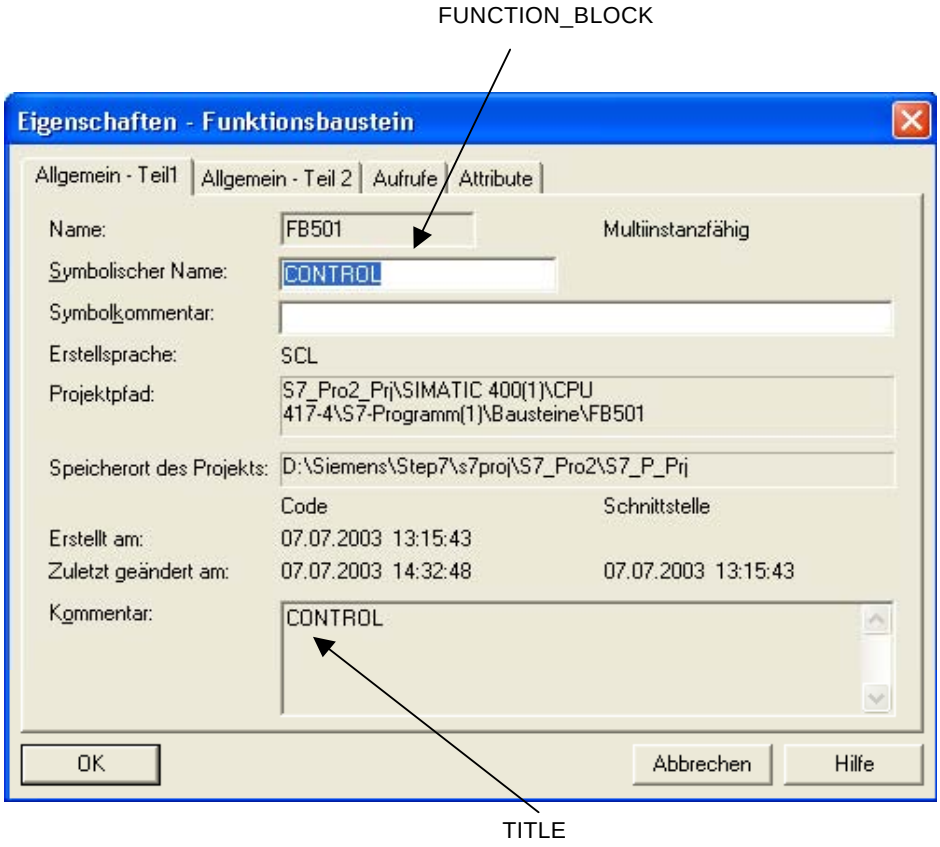


Bild 1-2: Objekteigenschaften des Bausteins (Allgemein - Teil 1)

- **FUNCTION_BLOCK**

  Hier legen Sie den Namen des Bausteins mit maximal 8 Zeichen fest. Dieser Name wird in den Objekteigenschaften des Bausteins sowie in der Detaildarstellung des SIMATIC Managers und im Katalog des CFC angezeigt. Vor dem Übersetzen des Bausteins muss diesem Namen in der Symboltabelle eine Bausteinnummer zugeordnet werden.

- **TITLE**

  Diese Information wird unter PCS 7 nicht ausgewertet, sie wird jedoch im SIMATIC Manager in den Objekteigenschaften des Bausteins im Kommentarfeld angezeigt. Direkt unterhalb dieses Attributs angegebene Kommentare werden ebenfalls in den Objekteigenschaften des Bausteins im Kommentarfeld angezeigt. Alle anderen Kommentare im Bausteinkopf können nur mit dem SCL-Editor eingesehen werden.

  Es wird empfohlen, hier eine Kurzbeschreibung des Bausteins einzutragen

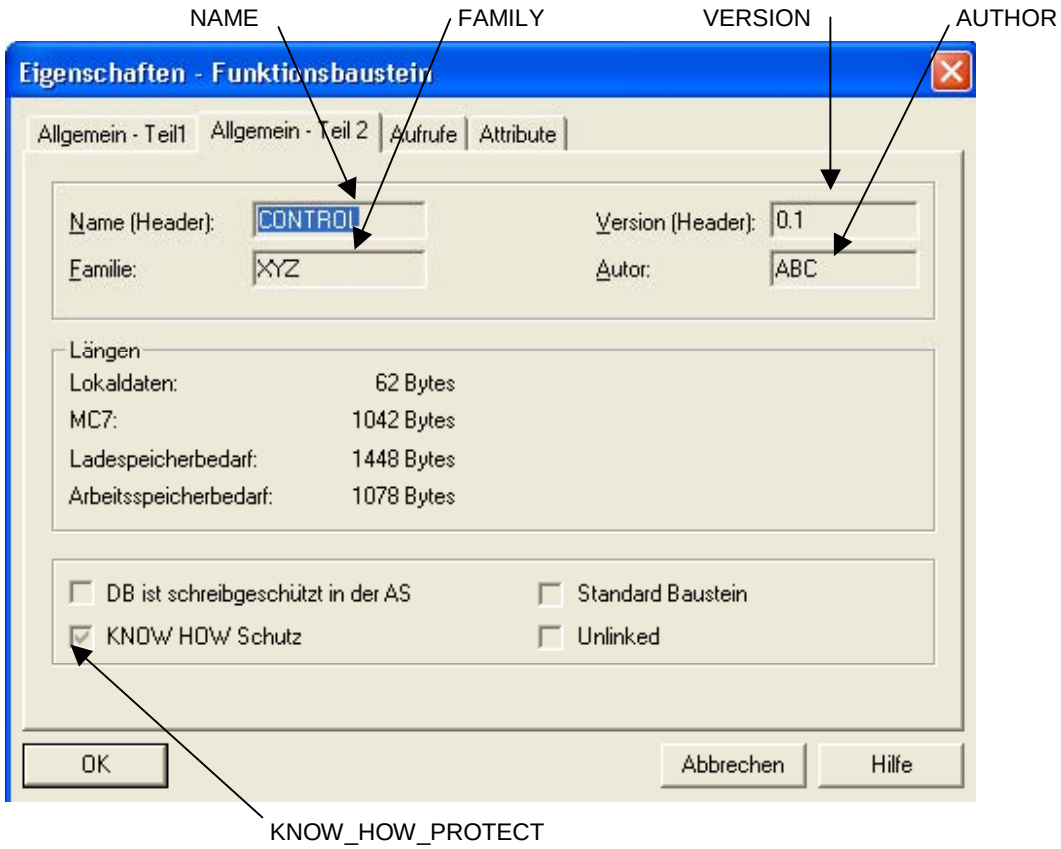


Bild 1-3: Objekteigenschaften des Bausteins (Allgemein - Teil 2)

- **NAME**

  Tragen Sie hier denselben Namen wie bei FUNCTION_BLOCK ein. Bei Verwendung einer Online-Hilfe ist dieser Name (und FAMILY) ein Teilschlüssel zur Suche des Hilfetexts dieses Bausteins in der Hilfedatei.

- **VERSION**

  Tragen Sie hier eine Versionskennung von 0.00 bis 15.15 ein.

- **FAMILY**

  Falls Sie Ihre Bausteine zu einer eigenen Bibliothek zusammenfassen und in dieser Bibliothek in verschiedene Gruppen einordnen wollen, geben Sie hier einen maximal 8 Zeichen langen Gruppennamen für diesen Baustein an.

  Bei Verwendung einer Online-Hilfe sind FAMILY und NAME Teilschlüssel zur Suche des Hilfetexts dieses Bausteins in der Hilfedatei (siehe auch Kap. 3.2).

- **AUTHOR**

  Dieses Attribut enthält im Normalfall den Namen oder die Abteilung des Bausteinerstellers. Bei PCS 7-konformen Bausteinen wird es noch für zwei weitere Punkte genutzt:

  Falls Sie Ihre Bausteine zu einer Bibliothek zusammenfassen wollen, geben Sie hier einen maximal 8 Zeichen langen gemeinsamen Namen für alle Bausteine dieser Bibliothek an.

  Bei Verwendung einer Online-Hilfe wird über diesen Namen die passende Hilfedatei gesucht.

- **KNOW_HOW_PROTECT**

  Mit Hilfe dieses Attributs können Sie den Algorithmus und die Attribute des Bausteins gegen Einsichtnahme und Änderung schützen. Ist es gesetzt, werden die Attribute des Bausteins im SIMATIC Manager in den Objekteigenschaften des Bausteins nur angezeigt, können aber nicht geändert werden. Der Baustein selbst lässt sich außerhalb Ihres Projekts ohne die entsprechende Quelle nur noch per AWL-Editor und nicht mehr per SCL öffnen. Es werden dabei jedoch nur die Bausteinparameter angezeigt. Im eigenen Projekt wird der SCL-Compiler gestartet.

- **Liste der Systemattribute für Bausteine**

  Mit Hilfe der Systemattribute bereiten sie einen Baustein für die Verbindung mit der OS vor. So definiert z.B: **S7_m_c**, ob der Baustein für eine OS relevant ist, d.h. ob dafür in der OS notwendige interne Datenstrukturen angelegt werden. Zudem können Sie mit den Systemattributen den Einbau des Bausteins in einen CFC-Plan steuern. So wird z.B. mit S7_tasklist festgelegt, in welche OBs der Baustein automatisch eingebaut werden soll.

Tabelle 1-1: Systemattribute für PCS 7-konforme Bausteine

Systemattribut	Bedeutung	Default
S7_tasklist	Enthält eine Liste der OBs (z.B. Fehler- oder Anlauf-OBs) in die der Baustein vom CFC eingebaut werden soll.	wird nicht mehrfach eingebaut
S7_m_c	Definiert, ob der Baustein von einer OS aus bedient oder beobachtet werden kann.	false
S7_alarm_ui	Kennung für Meldeserver: S7_alarm_ui:='0' Standard-Meldedialog S7_alarm_ui:='1' PCS7-Meldedialog	0
S7_tag	Hat dieses Systemattribut den Wert 'false', wird der Baustein nicht in die Messstellenliste der OS eingetragen, d.h. er erhält auf der OS keine Anwahl für "Loop in Alarm". Dies ist sinnvoll für Bausteine, die nur melden, aber keinen Bildbaustein besitzen. Ist das Systemattribut nicht vorhanden, wird der Baustein in die Messstellenliste eingetragen, wenn er auch das Systemattribut S7_m_c hat	true
S7_driver	Kennung für Signal vorverarbeitenden Treiberbaustein, der mit der CFC-Funktion im SIMATIC Manager "Baugruppentreiber erzeugen" automatisch mit dem zugehörenden Baustein verschaltet wird. Werte: 'chn', 'f'	- -

Systemattribut	Bedeutung	Default
S7_hardware	Kennung für baugruppenspezifischen Treiberbaustein, der mit der CFC-Funktion im SIMATIC Manager "Baugruppentreiber erzeugen" automatisch in einen CFC-Plan eingefügt, parametriert und verschaltet wird. Werte: 'subnet', 'rack', 'sm', 'im', 'fm'	- -
S7_read_back	Definiert, ob die Bausteininstanz für die Funktion "Plan > Rücklesen" im CFC vorgesehen wird. Hat dieses Systemattribut den Wert 'false', können die Parameter des Instanzbausteins nicht rückgelesen werden.	true

Die Systemattribute werden im SIMATIC Manager in den Objekteigenschaften des Bausteins im Register "Attribute" angezeigt und können dort auch geändert werden, falls der Baustein nicht schreibgeschützt ist (Attribut KNOW_HOW_PROTECT im Bausteinkopf).

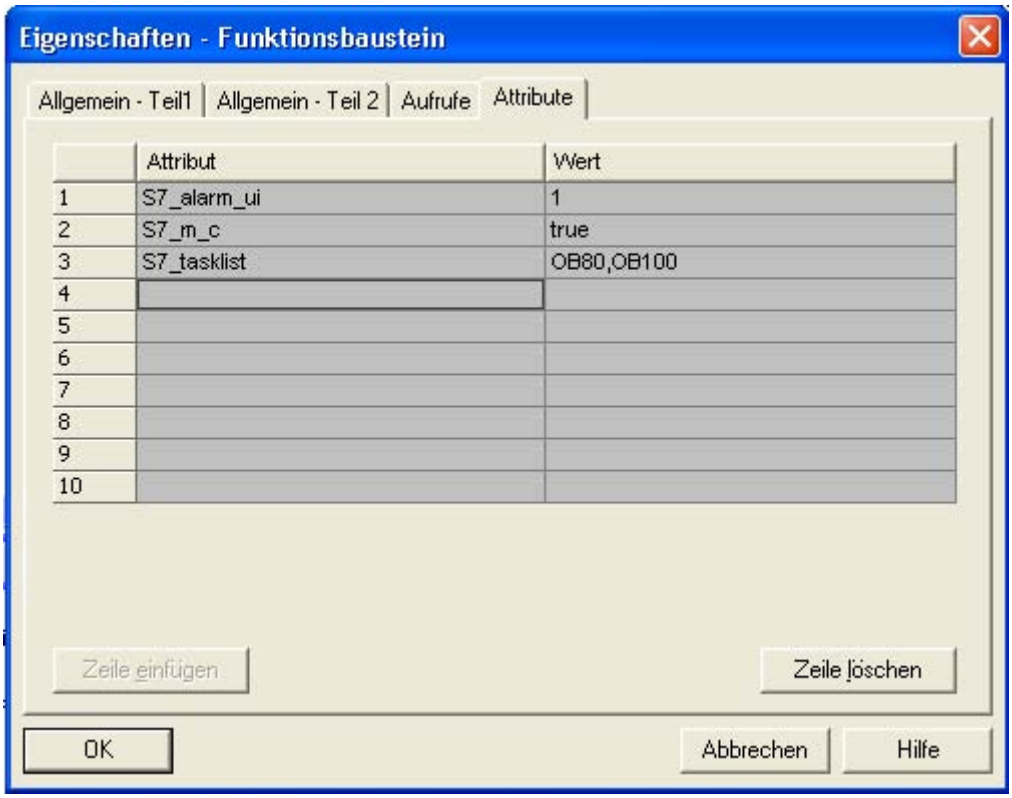


Bild 1-4: Systemattribute des Bausteins

Hinweis

Eine vollständige Liste der Systemattribute erhalten Sie über die Kontext-Hilfe und/oder über das Hilfethema "Attribute für Bausteine und Parameter".

1.2.4 Deklarationsteil

1.2.4.1 Bausteinparameter

Die Bausteinparameter definieren die Schnittstelle des Bausteins zu anderen Bausteinen sowie zu den Bedien- und Beobachtungs-Werkzeugen (CFC, WinCC ...).

Parametertypen

Es gibt die folgenden Parametertypen:

- **Eingangsparameter**

  Die Festlegung als Eingangsparameter ist für PCS 7-konforme Bausteine erforderlich, wenn Sie

  - Parameterwerte von einem anderen Baustein holen wollen oder
  - Parameter von der OS aus bedienen wollen
  - die Darstellung eines Bildbausteins auf der OS per Parameter beeinflussbar machen wollen (z.B. Grenzen für Darstellbereiche)
  - Parameter für Testzwecke vom CFC aus bedienbar machen wollen
  - Parameter zur Erzeugung von Meldungen (Message Event-ID des ALARM_8P-Bausteins)
  - Wenn Sie im Baustein eine stoßfreie Umschaltung zwischen Eingangswerten vom Programm (AS) und Bedienwerten (OS) benötigen, können die Eingangsparameter vom Bausteinalgorithmus sowohl gelesen als auch zurückgeschrieben werden (siehe Durchgangsparameter). Im Gegensatz zu den Durchgangsparametern werden die Eingangsparameter nicht auf den verschalteten Ausgangsparameter zurückgeschrieben.

- **Ausgangsparameter**

  Die Festlegung als Ausgangsparameter ist für PCS 7-konforme Bausteine erforderlich, wenn Sie

  - Parameterwerte an einen anderen Baustein weitergeben wollen
  - Parameter von der OS aus beobachten wollen
  - Parameter für Testzwecke vom CFC aus beobachten wollen

- **Durchgangsparameter**

  Durchgangsparameter können vom Bausteinalgorithmus sowohl gelesen als auch zurückgeschrieben werden. Die Festlegung als Durchgangsparameter ist für PCS 7-konforme Bausteine erforderlich, wenn Sie im Baustein eine stoßfreie Umschaltung zwischen Eingangswerten vom Programm (AS) und Bedienwerten (OS) benötigen. Für die Umsetzung dieser Funktionalität benötigen Sie 3 Parameter:

  - einen Eingangsparameter zum Umschalten
  - einen Eingangsparameter für den verschalteten Wert
  - einen Durchgangsparameter für den bedienten Wert. Dieser Parameter muss ein Durchgangsparameter sein, da der verschaltete Wert immer auf den bedienten Wert zurückgeschrieben werden muss. Auf diese Weise ist gewährleistet, dass bei der Umschaltung vom verschalteten auf den bedienten Wert diese Umschaltung stoßfrei erfolgt.

Kommentare für Parameter

Falls Sie die Bausteinparameter mit Kommentaren versehen wollen, tragen Sie diese durch "//" getrennt hinter der jeweiligen Parameterdefinition ein.

Die Kommentare werden im CFC in den Objekteigenschaften des jeweiligen Anschlusses sowie in den Objekteigenschaften des Bausteins im Register Anschlüsse angezeigt. Dort können Sie auch geändert werden, unabhängig vom Know-How-Schutz (Attribut KNOW_HOW_PROTECT im Bausteinkopf).

Systemattribute für Parameter

Bausteinparameter können ebenfalls (wie der Baustein selbst) mit Hilfe von Systemattributen weiter spezifiziert werden.

Sie können damit definieren,

- wie der Parameter auf der OS dargestellt werden soll.
  Beispiel: **S7_unit** definiert die Einheit des Parameters (z.B. Liter). Den an diesem Attribut angegebenen Text können Sie in Ihren Bildbaustein einblenden.
- ob und wie der Parameter im CFC behandelt werden soll.
  Beispiel: **S7_visible** definiert, ob der Parameter im CFC-Plan angezeigt wird oder nicht.

Tabelle 1-2: Systemattribute für Parameter für PCS 7-konforme Bausteine

Systemattribut	Betrifft	Bedeutung	Default
S7_sampletime	Zeitver-halten	Besitzt ein Parameter dieses Systemattribut, wird er automatisch mit der Zykluszeit des aufrufenden Weck-OBs parametriert. Dazu muss bei der Übersetzung des CFC-Plans das Optionskästchen "Aktualisierung der Abtastzeit" angewählt sein (vgl. Kapitel 1.4).	false
S7_dynamic	CFC	Besitzt ein Parameter dieses Systemattribut, wird er im Testbetrieb des CFC automatisch zum Testen angemeldet.	false
S7_edit	CFC	Definiert, ob der Parameter für die spätere Bearbeitung im SIMATIC Manager in der Tabelle "Parameter/Signale bearbeiten" vorgesehen werden soll (ohne den CFC-Plan zu öffnen).	false
S7_link	CFC	Definiert, ob der Parameter im CFC verschaltbar ist	true
S7_param	CFC	Definiert, ob der Parameter im CFC parametrierbar ist	true
S7_visible	CFC	Ein Parameter, bei dem dieses Systemattribut auf 'false' gesetzt ist, wird im CFC-Plan nicht dargestellt.	true
S7_qc	CFC, B&B	Dieses Attribut kennzeichnet Parameter, die neben dem Prozesswert über Quality Code verfügen. Der Quality Code muss dabei unmittelbar als nächster Parameter nach dem Prozesswert im Interface deklariert sein. Der Datentyp des Prozesswertes ist beliebig, der Datentyp des Quality Codes muss vom Typ "BYTE" sein.	false
S7_contact	SFC	Parameter gehört zu einer Anschlussgruppe	false
S7_m_c	B&B	Definiert, ob der Parameter von einer OS aus bedient oder beobachtet werden kann.	false
S7_shortcut	B&B	Enthält eine maximal 16 Zeichen lange Bezeichnung des Parameters. Diese Bezeichnung (z.B. "Sollwert") kann auf der OS in einem Bildbaustein ausgegeben werden.	- -
S7_string_0	B&B	Dieses Systemattribut ist nur für Eingangsparameter (oder Durchgangsparameter) vom Datentyp BOOL sinnvoll. Es enthält einen maximal 16 Zeichen langen Text, der in einem Bildbaustein als Bedientext ausgegeben werden kann (z.B. 'Ventil öffnen'). Durch die Ausführung dieser Bedienung erhält der Parameter den Wert 0.	- -
S7_string_1	B&B	Dieses Systemattribut ist nur für Eingangsparameter (oder Durchgangsparameter) vom Datentyp BOOL sinnvoll. Es enthält einen max. 16 Zeichen langen Text, der in einem Bildbaustein als Bedientext ausgegeben werden kann (z.B. 'Ventil schließen'). Durch die Ausführung dieser Bedienung erhält der Parameter den Wert 1.	- -
S7_unit	B&B	Enthält eine max. 16 Zeichen lange Einheit des Parameters. Die Einheit (z.B. "mbar") kann im CFC am Bausteinanschluss angezeigt werden.	- -
S7_server	Server	Interface-Parameter ist einem Server zugeordnet. Meldeserver: S7_server:='alarm_archiv'	kein Server-aufruf
S7_a_type	Meldeserver	Interface-Parameter ist Meldenummerneingang von der Meldeart x oder Archivnummerneingang	- -

Verwendung und Änderung der Systemattribute

Bei Verwendung der Systemattribute "S7_string_0" und "S7_string_1" müssen Sie noch Folgendes beachten:

Der angegebene Wert kann im Bildbaustein als Bedientext ausgeben werden. Wird die Bedienung ausgeführt, wird der Wert 0 bzw. 1 an das AS übertragen. Im CFC wird der aktuelle Wert des Parameters ausgegeben. Diese Wertausgabe können Sie ebenfalls mit Hilfe des Systemattributs anpassen.

Dazu müssen Sie den Wert des Systemattributs in zwei Teile teilen und diese beiden durch ein Gleichheitszeichen trennen, z.B. S7_string_1 := 'Suppress HH =YES'. Der CFC erkennt das Gleichheitszeichen und ersetzt die Werteausgabe am Parameter durch den Teil hinter dem Gleichheitszeichen; d.h. in diesem Fall wird statt dem Wert 1 das Wort "YES" ausgegeben.

Es werden im CFC maximal 8 Zeichen ausgegeben, auch wenn Sie mehr angeben. Im Bedienprotokoll wird immer der ganze Text ausgegeben, d.h. hier "Suppress HH =YES".

Die Systemattribute werden im CFC in den Objekteigenschaften des jeweiligen Anschlusses angezeigt und können dort auch geändert werden. Die folgenden Bilder zeigen die Objekteigenschaften von Parametern vom Datentyp BOOL und von Parametern, die nicht vom Datentyp BOOL sind (Bild 1-6). Zudem enthalten die Bilder Verweise auf die betroffenen Systemattribute.

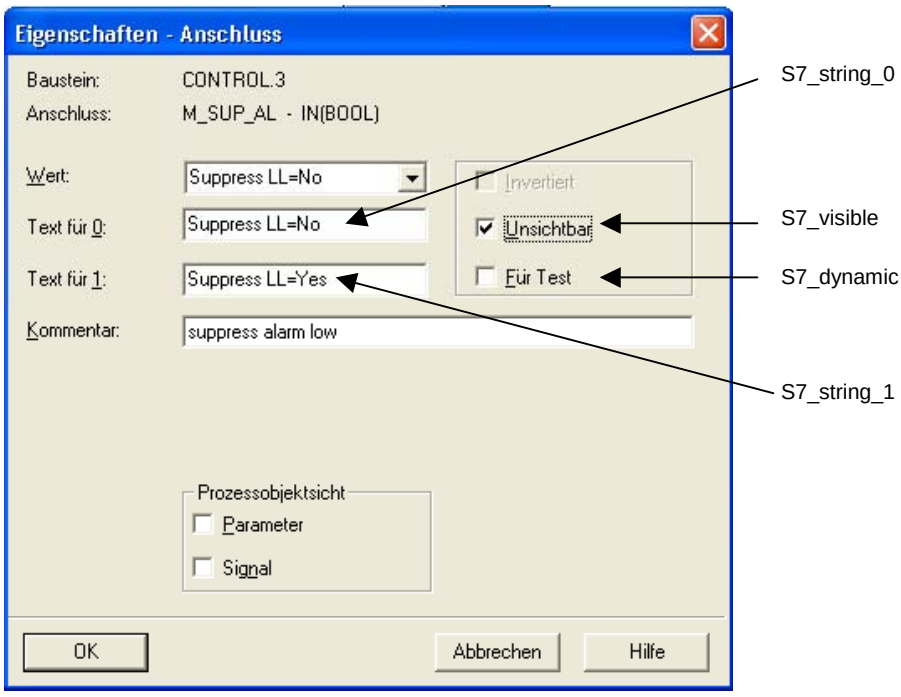


Bild 1-5: Objekteigenschaften von Parametern vom Datentyp BOOL

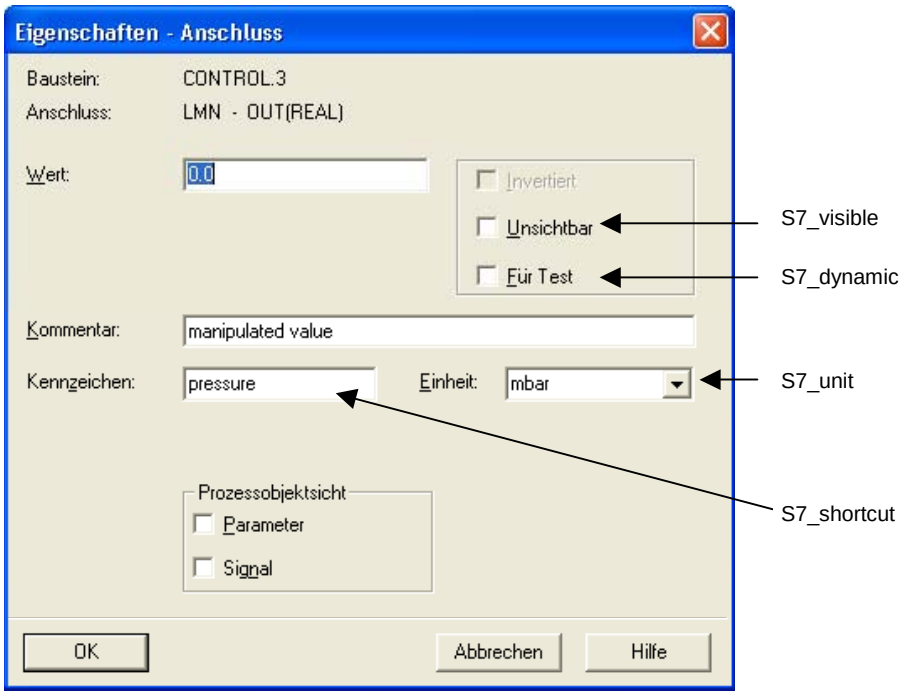


Bild 1-6: Objekteigenschaften von Parametern, die nicht vom Datentyp BOOL sind

Der folgende Auszug des Beispiel-Bausteins zeigt die Codierung der Bausteinparameter:

```
//*******************************************************************************
// Deklarationsteil: Bausteinparameter
//*******************************************************************************
VAR_INPUT
SAMPLE_T {S7_sampletime:= 'true';  // Param. der Baustein-Abtastzeit (Zyklus der Task)
       S7_visible:='false';        // Parameter ist unsichtbar
       S7_link:= 'false'           // Parameter nicht verschaltbar
      }          :REAL := 1;       // Verz^gerungszeit [s] (Vorbesetzung 1 Sek)

  H_ALM {S7_m_c := 'true';
     S7_visible:='false';
     S7_link := 'false'}  :REAL :=100;  // oberer Grenzwert Alarm (Vorbesetzung 100)

  L_ALM {S7_m_c := 'true';         // Parameter ist B&B-f%hig
     S7_visible:='false';          // Parameter ist unsichtbar
     S7_link := 'false'            // und nicht verschaltbar
     }           :REAL := 0;       // unterer Grenzwert Alarm (Vorbesetzung 0)

  M_SUP_AL {S7_visible:='false';
    S7_link:='false';
    S7_m_c:='true';
    S7_string_0:= 'Suppress LL=No';    // Bedientext f¸r Wert (M_SUP_AL)= 0
    S7_string_1:= 'Suppress LL=Yes'    // Bedientext f¸r Wert (M_SUP_AL)= 1
      }           :BOOL;               // Meldungsunterdr¸ckung unterer Alarm

  M_SUP_AH {S7_visible:='false';
    S7_link:='false';
    S7_m_c:='true';
    S7_string_0:= 'Suppress HH=No';
    S7_string_1:= 'Suppress HH=Yes'
      }           :BOOL;            // Meldungsunterdr¸ckung oberer Alarm

  SP_OP_ON {S7_visible:='false';
      S7_dynamic:='true'            // CFC in Test/IBS: Anzeige aktueller Wert in AS)
      }           :BOOL := 1;       // 1 = Bedienfreigabe f¸r Sollwert-Eingabe

  SPBUMPON {S7_visible:='false';
       S7_link:='false';
       S7_m_c:='true';
       S7_string_0:='SP bumpless=Off';
       S7_string_1:='SP bumpless=On'
      }
                :BOOL := 1;       // 1 = Stoflfreiheit f¸r Sollwert
  SP_EXTON {S7_visible:='false';
      S7_dynamic:='true'          // CFC in Test/IBS: Anzeige aktueller Wert in AS)
      }
                :BOOL := 1;          // 1: Externer Sollwert ein
  SP_EXT   {S7_dynamic:='true'}
                 :REAL := 0;        // Externer Sollwert

  SP_HLM   {S7_visible:='false';
    S7_link:='false';
    S7_m_c:='true';
    S7_shortcut:='SP high limit';     // Text(max 16 Zeichen) zur Anzeige auf OS
    S7_unit:=''}                      // Einheit (max 16 Zeichen)
              :REAL := 100;           // oberer Bediengrenzwert f¸r Sollwert

  SP_LLM    {S7_visible:='false';
    S7_link:='false';
    S7_m_c:='true';
    S7_shortcut:='SP low limit';
    S7_unit:=''}
              :REAL := 0;             // unterer Bediengrenzwert f¸r Sollwert
```

```
PV_IN {S7_dynamic:='true';
    S7_m_c:='true';
    S7_unit:='%'} :   REAL := 0;      // Prozesswert (zu Begleitwert_PR04)

GAIN   {S7_link:='false';
    S7_edit:='para';                  // Parametrierung im IEA
    S7_m_c:='true';
    S7_shortcut:='Gain';
    S7_unit:=''}        :REAL := 1;   // Proportionalbeiwert

  EV_ID   {S7_visible:='false';
    S7_link:='false';
    S7_param :='false';               // Parameter im CFC nicht parametrierbar
    S7_server:='alarm_archiv';        // Vergabe der Meldenummer durch Server
    S7_a_type:='alarm_8p'             // Baustein meldet mit ALARM_8P
       }           :DWORD := 0;       // Meldungsnummer

                                      // Parameter f¸r SIMATIC BATCH
  STEP_NO {S7_visible := 'false';
      S7_m_c   := 'true'}  :DWORD;    // Batch Schrittnummer
  BA_ID {S7_visible  := 'false';
     S7_m_c    := 'true'}  :DWORD;    // Batch laufende Chargennummer
  BA_EN {S7_visible := 'false';       // Parameter im CFC-Plan unsichtbar
     S7_m_c := 'true'                 // Parameter ist B&B-f‰hig
     }            :BOOL := 0;         // Batch Belegt-Freigabe
  BA_NA {S7_visible := 'false';
     S7_m_c   := 'true'} :STRING[32] := '';  // Batch Chargenbezeichnung

  OCCUPIED {S7_visible := 'false';
       S7_m_c   := 'true'} :BOOL := 0;       // Batch Belegt-Kennung

  RUNUPCYC   {S7_visible:='false';
      S7_link:='false'} :INT := 3;  // Anzahl Erstlaufzyklen
  SUPPTIME       :REAL := 0;        // Verz^gerungszeit
  SUPP_IN        :REAL := 0;        // Eingangswert f¸r Verz^gerungszeit
END_VAR

VAR_OUTPUT
  LMN {S7_shortcut:='pressure';     // Bezeichnung des Parameters auf OS
    S7_unit := '%';                 // Einheit des Parameters
    S7_m_c := 'true'                // beobachtbar
    } :REAL;                        // Stellwertausgang

  QH_ALM  :BOOL := false;           // 1 = Oberer Grenzwert Alarm hat angesprochen

  QL_ALM  :BOOL := false;           // 1 = Unterer Grenzwert Alarm hat angesprochen

  QSP_HLM   {S7_visible:='false';
    S7_dynamic:='true'} :   BOOL := 0;   // 1= Sollwertausgang nach oben begrenzt

  QSP_LLM   {S7_visible:='false';
    S7_dynamic:='true'} :   BOOL := 0;   // 1 = Sollwertausgang nach unten begrenzt

  Q_SP_OP   {S7_visible:='false';
    S7_dynamic:='true';
    S7_m_c:='true'} :   BOOL := 0;       // Status: 1 = Bedienfreigabe f¸r Sollwert

  QOP_ERR   {S7_visible:='false';
    S7_dynamic:='true'} :   BOOL := 0;   // 1 = Bedienfehler

  QMSG_ERR   {S7_visible:='false';
    S7_dynamic:='true'} :   BOOL := 0;   // ALARM_8P: Meldungsfehler

  MSG_STAT   {S7_visible:='false';
    S7_dynamic:='true'} :   WORD := 0;   // Meldungsfehler-Informationen

  MSG_ACK   {S7_visible:='false';
    S7_dynamic:='true'} :   WORD := 0;   // Meldungen quittieren

  SUPP_OUT           :REAL := 0;         // Ausgangswert f¸r Verz^gerungszeit
  SP   {S7_dynamic:='true';
    S7_m_c:='true'} :    REAL := 0;      // Aktiver Sollwert

END_VAR
```

```
VAR_IN_OUT
 SP_OP {S7_visible:='false';
    S7_link:='false';
    S7_m_c:='true';
    S7_shortcut:='Setpoint';
    S7_unit:='%'} :   REAL := 0;           // Bedieneingabe Sollwert

 // freibelegbare Meldebegleitwerte des ALARM_8P

  AUX_PR05 {S7_visible := 'false'} : ANY;  // Begleitwert 5
  AUX_PR06 {S7_visible := 'false'} : ANY;  // Begleitwert 6
  AUX_PR07 {S7_visible := 'false'} : ANY;  // Begleitwert 7
  AUX_PR08 {S7_visible := 'false'} : ANY;  // Begleitwert 8
  AUX_PR09 {S7_visible := 'false'} : ANY;  // Begleitwert 9
  AUX_PR10 {S7_visible := 'false'} : ANY;  // Begleitwert 10

END_VAR
```

1.2.4.2 Lokale Variablen

Zusätzliche Variablen, d.h. solche, die nicht als Bausteinparameter nach außen gegeben werden, müssen Sie als lokale Variablen definieren.

Es gibt zwei Sorten von lokalen Variablen:

- Statische Variablen
- Temporäre Variablen

Statische Variablen

Statische Variablen behalten im Gegensatz zu temporären Variablen ihren Wert über mehrere Aufrufe des Bausteins hinweg, solange, bis Sie ihn im Bausteinalgorithmus ändern.

Für PCS 7-konforme Bausteine sind diese Variablen vor allem dann wichtig, wenn Sie in Ihrem Baustein bereits vorhandene eigene oder Standard-Bausteine aufrufen wollen. In diesem Fall müssen Sie einen **Multiinstanz**-Baustein implementieren. Dazu definieren Sie innerhalb der statischen Variablen eine Instanz des aufgerufenen Bausteins.

Die aufgerufenen Bausteine müssen zum fehlerfreien Übersetzen des aufrufenden Bausteins im Bausteinordner des S7-Programms vorhanden sein.

Falls Sie Parameter des aufgerufenen Bausteins nach außen sichtbar und verschaltbar machen wollen, müssen Sie diese im Bausteinalgorithmus von bzw. in Parameter Ihres Bausteins kopieren. Die Parameter des aufgerufenen Bausteins selbst sind nach außen nicht sichtbar.

Multiinstanzen

Beispiele zu Multiinstanzanwendungen können Sie dem Kapitel zu den CFC-Bausteintypen bzw. dem entsprechenden SCL-Code im Beispielprojekt entnehmen.

Hinweis

Aufgerufene SFBs und SFCs, wie z.B. SFC 6 (RD_SINFO) oder SFB 0 (CTU) werden beim Compilieren des aufrufenden Bausteins automatisch in der Standard Library gesucht und in Ihr S7-Programm eingefügt.

Aufgerufene FBs werden beim Einfügen des aufrufenden Bausteins in einen CFC-Plan in den Bausteinordner kopiert, falls Sie sich in der selben Bibliothek befinden wie der aufrufende Baustein. Ansonsten müssen Sie sie selbst kopieren.

Temporäre Variablen

Temporäre Variablen haben nur während **eines** Bausteinaufrufs Gültigkeit, d.h. sie müssen bei jedem Bausteinaufruf neu berechnet werden.

Für PCS 7-konforme Bausteine müssen Sie hierbei keine Besonderheiten beachten.

Ausschnitt des Beispiel-Bausteins:

```
//*******************************************************************************
// Deklarationsteil: tempor‰re Variablen
//*******************************************************************************

VAR_TEMP
  // Startinfo: Struktur mit Info f¸r den OB, der den Baustein gerade aufgerufen hat
  TOP_SI:    STRUCT
   EV_CLASS :BYTE;
   EV_NUM   :BYTE;
   PRIORITY :BYTE;
   NUM      :BYTE;
   TYP2_3   :BYTE;
   TYP1     :BYTE;
   ZI1      :WORD;
   ZI2_3    :DWORD;
  END_STRUCT;

// Startinfo: Struktur mit Info f¸r den letzten aufgerufenen Anlauf-OB
  START_UP_SI:  STRUCT
   EV_CLASS :BYTE;
   EV_NUM   :BYTE;
   PRIORITY :BYTE;
   NUM      :BYTE;
   TYP2_3   :BYTE;
   TYP1     :BYTE;
   ZI1      :WORD;
   ZI2_3    :DWORD;
  END_STRUCT;

  S7DT   :DATE_AND_TIME;           // Lokale Zeitvariable
  DUMMY  :INT;                     // Hilfsvariable
END_VAR
```

1.2.5 Codeteil

Der Codeteil enthält den eigentlichen Algorithmus des Bausteins. Für PCS 7-konforme Bausteine bedeutet dies, dass Sie hier neben den reinen technologischen Funktionen des Bausteins noch die Eigenschaften realisieren können, um z.B. asynchron auftretende Ereignisse und Bausteinzustände an die OS zu melden und dort über einen Bildbaustein oder eine WinCC-Meldeliste anzuzeigen.

1.3 Erstlauf

Beim ersten Aufruf Ihres Bausteins müssen Sie im Regelfall verschiedene Parameter initialisieren. Zudem kann es je nach technologischer Funktion Ihres Bausteins weitere Tätigkeiten geben, die Ihr Baustein nur einmalig durchführen muss. Falls dies bei Ihrem Baustein so ist, müssen Sie einen Erstlaufteil implementieren.

Dazu definieren Sie einen Variable vom Datentyp BOOL (z.B. sbRESTART). Sie können die Variable als statische Variable realisieren.

Da nicht gewährleistet werden kann, dass Ihr Baustein nur bei Neustart das erste Mal durchlaufen wird (z.B. beim Neuladen des Bausteins im Betriebszustand RUN der CPU), müssen Sie den Erstlaufteil typischerweise in den zyklischen Teil Ihres Bausteins integrieren. Dadurch können Sie die Bearbeitung des Erstlaufs, falls notwendig, auch auf mehrere Aufrufzyklen des Bausteins ausdehnen.

```
//*****************************************************************************
// Abhängigkeiten vom aufrufenden OB
//*****************************************************************************

// Auslesen der Startinfo mittels SFC6 (RD_SINFO)
  DUMMY := RD_SINFO (TOP_SI := TOP_SI, START_UP_SI := START_UP_SI);

  IF sbRESTART THEN
  // Erstlauf
  TOP_SI.NUM := 100;     // Erstlauf als Neustart ausführen
  sbRESTART := FALSE;    // Rücksetzen Erstlauf
  END_IF;
// Aus welchem OB wurde der Baustein aufgerufen ?

 CASE WORD_TO_INT(BYTE_TO_WORD(TOP_SI.NUM)) OF

//*****************************************************************************
// Behandlung von Fehler-OBs
//*****************************************************************************

    // OB80: Zeitfehler
    80:
    QH_ALM  := 0;           // Fehlerausgänge zurücksetzen
    QL_ALM  := 0;

//*****************************************************************************
// Anlauf
//*****************************************************************************

    // OB100: Neustart
    100:
    QH_ALM   := 0;          // Fehlerausgänge zurücksetzen
    QL_ALM   := 0;
    siRUNUPCNT := RUNUPCYC; // RUNUPCYC-Wert abspeichern
 ELSE
....
```

1.4 Zeitabhängigkeit

Falls Ihr Baustein in einer äquidistanten Zeitalarmebene bearbeitet wird, und er zur Durchführung zeitabhängiger Tätigkeiten die Länge des Zeitintervalls auswerten muss (z.B. Reglerbausteine), definieren sie einen Eingangsparameter (z.B. SAMPLE_T) vom Datentyp REAL, an dem die Länge des Zeitintervalls angegeben werden kann.

Diesen Parameter müssen Sie je nach Weckalarm-OB, in dem Ihr Baustein aufgerufen wird, umparametrieren. Damit ist gewährleistet, dass Ihr Bausteinalgorithmus immer mit der richtigen Zeit arbeitet.

Wenn Sie diesen Parameter mit dem Systemattribut **S7_sampletime** versehen und dieses auf 'true' setzen, wird er vom CFC automatisch auf den für den aufrufenden OB passenden Wert gesetzt. Eine eventuelle Untersetzung wird dabei ebenfalls berücksichtigt. Sie sollten den Parameter dann auch mit den Systemattributen **S7_visible** und **S7_link** versehen und diese auf 'false' setzen. Damit wird der Parameter im CFC unsichtbar bzw. nicht verschaltbar und damit verhindert, dass sein Wert vom Anwender versehentlich geändert wird.

Die automatische Belegung des Parameters durch den CFC funktioniert jedoch nur dann, wenn beim Übersetzen des Programms das Optionskästchen "Update sampling time" aktiviert ist.

Der folgende Ausschnitt des Beispiel-Bausteins zeigt die Realisierung einer solchen Zeitabhängigkeit: Mit Hilfe des Parameters SUPPTIME kann am Baustein eine Wartezeit parametriert werden. Änderungen am Eingang SUPP_IN werden nach Ablauf dieser Wartezeit auf den Ausgang SUPP_OUT durchgeschaltet.

```
//*****************************************************************************************
// Deklarationsteil: Bausteinparameter
//*****************************************************************************************
VAR_INPUT

  SAMPLE_T {S7_sampletime:= 'true' // Parameter Baustein-Abtastzeit (=Zyklus der Task)
       S7_visible:='false';        // Parameter ist unsichtbar
       S7_link:= 'false'           // Parameter nicht verschaltbar
      }        :REAL := 1;         // Verzögerungszeit [s] (Vorbesetzung 1 Sekunde)
....
END_VAR

//*****************************************************************************************
// Deklarationsteil: statische Variablen
//*****************************************************************************************
VAR
....
  sSUPP_IN    :REAL := 0;          // Altwert des Delay-Beispieleingangs
  ACT_TIME    :REAL := 0;          // Zeitzähler
....
END_VAR

VAR_OUTPUT
....
SUPP_OUT           :REAL := 0;     // Ausgangswert für Verzögerungszeit
....
END_VAR

//*****************************************************************************
// Technologischer Teil
//*****************************************************************************

  IF (SUPP_IN <> sSUPP_IN) THEN
   ACT_TIME := SUPPTIME;           // Zeitzähler initialisieren
   sSUPP_IN := SUPP_IN;
  END_IF;

  IF (ACT_TIME > 0) THEN           // Wenn Wartezeit noch nicht abgelaufen ist
   ACT_TIME := ACT_TIME-SAMPLE_T;  // Wartezeit herunterzählen
  ELSE
   SUPP_OUT := SUPP_IN;            // Eingang durchschalten
  END_IF;
....
```

1.5 Behandlung von asynchronen Anlauf- und Fehler-OBs

Beim Auftreten eines asynchronen Ereignisses wie Neustart oder auch Ziehen/Stecken, Rack-Ausfall und dergleichen, wird vom AS ein asynchroner OB aufgerufen. Falls Ihr Baustein auf ein solches Ereignis reagieren soll, müssen Sie Ihren Baustein in den betreffenden OB einbauen und im Bausteinalgorithmus feststellen, ob ein derartiges Ereignis aufgetreten ist.

Einbau in asynchrone OBs

Damit Ihr Baustein in bestimmte OBs eingebaut wird, verwenden Sie das Systemattribut "S7_tasklist". Als Wert tragen Sie alle OBs ein, die Sie benötigen (z.B. S7_tasklist := 'OB80,OB100') . Beim Einfügen des Bausteins in einen CFC-Plan wird der Baustein somit vom CFC automatisch neben dem aktuellen Weckalarm-OB auch in alle mit S7_tasklist angegebenen OBs eingebaut.

Prüfen des aufrufenden OBs

Um zu überprüfen, in welchem OB Ihr Baustein gerade läuft, müssen Sie im Bausteinalgorithmus den SFC 6 (RD_SINFO) aufrufen. Dieser liest die Startinfo Ihres Bausteins aus und liefert damit Informationen über den gerade aktiven OB (Parameter TOP_SI) und den zuletzt aufgerufenen Anlauf-OB (Parameter START_UP_SI).

Die beiden Parameter sind zwei identisch aufgebaute Strukturen, die Sie beide in Ihren temporären Variablen definieren müssen. Die einzelnen Strukturelemente haben dabei folgende Bedeutung:

Tabelle 1-3: Parameter TOP_SI und START_UP_SI

Strukturelement	Datentyp	Bedeutung
EV_CLASS	BYTE	Bit 0 bis 3: Ereigniskennung Bit 4 bis 7: Ereignisklasse
EV_NUM	BYTE	Ereignisnummer
PRIORITY	BYTE	Nummer der Ablaufebene
NUM	BYTE	Nummer des aufrufenden OBs
TYP2_3	BYTE	Datenkennung für ZI2_3
TYP1	BYTE	Datenkennung für ZI1
ZI1	WORD	Zusatzinfo 1
ZI2_3	DWORD	Zusatzinfo 2_3

Die Strukturelemente entsprechen inhaltlich den temporären Variablen des aufrufenden OBs. Diese können aber je nach OB andere Namen und Datentypen haben. Das bedeutet, dass Sie anhand der jeweiligen OB-Beschreibung (siehe Handbuch "STEP 7 - System- und Standardfunktionen) die einzelnen Strukturelemente einander zuordnen und entsprechend auswerten müssen. Die folgende Tabelle bzw. der Ausschnitt des Beispiel-Bausteins zeigen dies am Beispiel des OB 80 (Zeitfehler).

Tabelle 1-4: Zuordnung der Elemente der Startinfo TOP_SI zu den temporären Variablen des OB80

TOP_SI / STARTUP_SI		**OB80**	
Strukturelement	**Datentyp**	**Temporäre Variable**	**Datentyp**
EV_CLASS	BYTE	OB80_EV_CLASS	BYTE
EV_NUM	BYTE	OB80_FLT_ID	BYTE
PRIORITY	BYTE	OB80_PRIORITY	BYTE
NUM	BYTE	OB80_OB_NUMBR	BYTE
TYP2_3	BYTE	OB80_RESERVED_1	BYTE
TYP1	BYTE	OB80_RESERVED_2	BYTE
ZI1	WORD	OB80_ERROR_INFO	WORD
ZI2_3	DWORD	OB80_ERR_EV_CLASS	BYTE
		OB80_ERR_EV_NUM	BYTE
		OB80_OB_PRIORITY	BYTE
		OB80_OB_NUM	BYTE

Hinweis

Jeder OB enthält in seinen temporären Variablen noch Datum und Uhrzeit des Aufrufs. Diese sind jedoch nicht in der mit dem SFC 6 gelesenen Startinfo enthalten.

PCS 7-konforme Bausteine werden nicht in den Wiederanlauf (OB 101) oder Kaltstart (OB 102) eingebaut.

Der folgende Ausschnitt des Beispiel-Bausteins zeigt die jeweilige Behandlung des OBs:

```
//*****************************************************************************
// Codeteil
//*****************************************************************************

 CASE WORD_TO_INT(BYTE_TO_WORD(TOP_SI.NUM)) OF

//*****************************************************************************
// Behandlung von Fehler-OBs
//*****************************************************************************

    // OB80: Zeitfehler
    80:
    QH_ALM  := 0;           // Fehlerausgänge zurücksetzen
    QL_ALM  := 0;

//*****************************************************************************
// Anlauf
//*****************************************************************************

    // OB100: Neustart
    100:
    QH_ALM   := 0;          // Fehlerausgänge zurücksetzen
    QL_ALM   := 0;
    siRUNUPCNT := RUNUPCYC; // RUNUPCYC-Wert abspeichern
 ELSE
```

1.6 Bedienen, Beobachten und Melden

Ein Baustein, dessen Parameter von der OS aus **bedient** und **beobachtet** werden sollen, muss für diese Anbindung an die OS entsprechend vorbereitet werden. Das betrifft sowohl die gewünschten Parameter als auch den Baustein selbst.

Bedienungen

Falls Sie einen Parameterwert nur von der OS aus bedienen wollen, benötigen Sie dazu einen Durchgangs- oder Eingangsparameter für den bedienten Wert (mit dem Systemattribut **S7_m_c**).

Falls Sie dagegen einen Parameterwert wahlweise von einem anderen Baustein holen oder von der OS bedienen wollen und der Umschaltvorgang vom verschalteten auf den bedienten Wert stoßfrei erfolgen soll, benötigen Sie dazu insgesamt drei Parameter:

- einen Eingangsparameter zum Umschalten zwischen Bedienung und Verschaltung.
- einen Eingangsparameter für den verschalteten Wert
- einen Durchgangsparameter für den bedienten Wert (mit dem Systemattribut **S7_m_c**). Dieser Parameter muss ein Durchgangsparameter sein, weil der verschaltete Wert für das stoßfreie Umschalten vom Bausteinalgorithmus auf den bedienten Wert zurückgeschrieben werden muss, solange der verschaltete Wert angewählt ist.

Alle Bedienfunktionen sollten über die Bedienbausteine der Bibliothek "PCS 7 Library V61" und deren korrespondierende Bedienmethode auf der OS abgewickelt werden. Damit sind alle notwendigen Verriegelungen und die (wahlweise stoßfreie) Umschaltung zwischen bedientem Wert und verschaltetem Wert vorhanden (z. B. für Hand- / Automatikumschaltung). Die Bedienbausteine können Sie mittels der Multiinstanztechnik in Ihren Baustein einbauen.

Bei PCS 7 wird u.a. der **OP_A_LIM** (**o**peration **a**nalog **lim**ited) eingesetzt.

Mit dem Baustein OP_A_LIM haben Sie ein begrenzendes Bedienverhalten gewählt. Alternativ können Sie den Baustein OP_A_RJC einsetzen. Dieser weist bei Grenzwertverletzung die Bedieneingabe ab. Wenn Sie keine Grenzwertprüfung benötigen, setzen Sie den Baustein OP_A ein.

Hinweis

AS-Baustein und Bildbaustein laufen asynchron zueinander, d.h. bei einer Bedienung durch den Bildbaustein wird der Bedienwert in den Instanz-DB des AS-Bausteins geschrieben und zu einem späteren Zeitpunkt vom AS-Baustein ausgewertet. Da sich zu diesem Zeitpunkt bereits die maßgeblichen Grenzen geändert haben können, sollten Sie den Bedienwert sowohl auf dem AS als auch auf der OS auf Fehler prüfen.

Für binäre Bedieneingaben stehen die Bausteine OP_D (FB 48), OP_D3 (FB 49) und OP_TRIG (FB 50) der PCS 7 Library V61 zur Verfügung (weitere Informationen: siehe Online-Hilfe).

Der folgende Ausschnitt zeigt die Definition einer Bedieneingabe:

```
//*****************************************************************************
// Bedieneingabe Sollwert SP_OP (Real-Wert) oder verschalteter Sollwert SP_EXT
//*****************************************************************************

// Multiinstanz-Aufruf OP_A_LIM (Bedeutung der Parameter siehe Online Hilfe OP_A_LIM)

    OP_A_LIM_1(U := SP_OP, U_HL:= SP_HLM, U_LL:= SP_LLM, OP_EN:= SP_OP_ON, BTRACK:=
SPBUMPON, LINK_ON:= SP_EXTON, LINK_U:= SP_EXT);

    OK := OK AND ENO; //Enable Out des OP_A_LIM in OK-Flag des Bausteins übernehmen

    Q_SP_OP := OP_A_LIM_1.QOP_EN;    // 1: Freigabe Bedieneingabe SP

    QOP_ERR := OP_A_LIM_1.QOP_ERR;   // 1: Bedienfehler
    QSP_HLM := OP_A_LIM_1.QVHL;      // 1: Begrenzung Obergrenze
    QSP_LLM := OP_A_LIM_1.QVLL;      // 1: Begrenzung Untergrenze
    SP   := OP_A_LIM_1.V;            // wirksamer Sollwert
```

Meldungen

Wenn Ihr Baustein Meldungen und/oder Ereignisse an die OS senden soll, können Sie in den statischen Variablen eine Multiinstanz eines Alarm-Bausteins definieren. Das Meldungs- und Quittierverhalten sowie die Übergabe von Begleitwerten wird durch die Eigenschaften der CPU (wählbar "quittiergetriggertes Melden“) und des eingebauten Alarm-Bausteins bestimmt.

Fertige Alarm-Bausteine sind als SFBs in der "Standard Library" enthalten.
Das sind z.B.:

ALARM	SFB 33	Überwachung eines Signals mit 1 bis 10 Begleitwerten **mit** Quittierungsanzeige
ALARM_8	SFB 34	Überwachung von bis zu 8 Signalen
ALARM_8P	SFB 35	Überwachung von bis zu 8 Signalen mit 1 bis 10 Begleitwerten
NOTIFY	SFB 36	Überwachung eines Signals mit 1 bis 10 Begleitwerten **ohne** Quittierungsanzeige
NOTIFY_8P	SFB 31	Überwachung von bis zu 8 Signalen mit 1 bis 10 Begleitwerten **ohne** Quittierungsanzeige

Einträge im Bausteinkopf

Damit der Baustein von der OS aus bedient und/oder beobachtet werden kann, setzen Sie zunächst in der Liste der Systemattribute im Bausteinkopf das Systemattribut "S7_m_c" auf 'true'.

Damit der PCS 7-Meldedialog im SIMATIC Manager aufgerufen wird, tragen Sie im Bausteinkopf das Attribut S7_alarm_ui := '1' ein (bei Wert '0' erhalten Sie den STEP 7-konformen Dialog).

Einträge im Deklarationsteil

Damit die Parameter Ihres Bausteins von der OS aus bedient und beobachtet werden können, setzen Sie für jeden einzelnen Parameter Ihres Bausteins, den Sie bedienen und beobachten wollen das Systemattribut "S7_m_c" auf 'true'.

Wenn Ihr Baustein Meldungen und/oder Ereignisse an die OS senden soll, definieren Sie einen Eingang vom Datentyp DWORD (hier: EV_ID). Dieser Eingang nimmt beim Instanz-DB die Meldenummer auf, die automatisch vom System (Meldeserver) vergeben wird.

Die Meldenummer ist im gesamten S7-Projekt eindeutig, damit es in Projekten mit mehreren AS und OS nicht zu Kollisionen kommt. Aus dieser Meldenummer werden beim Datentransfer (OS übersetzen) die für WinCC notwendigen Nummern für die Einzelmeldungen abgeleitet.

Geben Sie an diesem Eingang das Systemattribut "S7_server" mit dem Wert 'alarm_archiv' und das Systemattribut "S7_a_type" mit dem Wert 'alarm_8p' (bzw. gemäß dem eingebauten Meldungsbaustein) ein.
Der Eingang soll im CFC nicht sichtbar, nicht verschaltbar und nicht parametrierbar sein, um die vom System vergebenen Daten nicht versehentlich zu verändern.

Der folgende Ausschnitt des Beispiel-Bausteins zeigt die Verwendung der Systemattribute im **Bausteinkopf** und für den **Eingang** EV_ID, der die Meldenummer aufnehmen soll.

```
//*******************************************************************************************
// Bausteinkopf
//*******************************************************************************************

FUNCTION_BLOCK    "CONTROL"
TITLE=                              'CONTROL'

{  // Liste der Systemattribute
S7_tasklist:=     'OB80,OB100'; // Baustein wird bei Zeitfehler und Neustart aufgerufen
S7_m_c:=                         'true';       // Baustein ist bedien- und beobachtbar
S7_alarm_ui:=     '1'           // Einstellung PCS7-Meldedialog ('0' = Standard-Dialg
}
AUTHOR:                          ABC
NAME:                              CONTROL
VERSION:                         '0.02'
FAMILY:                          XYZ
KNOW_HOW_PROTECT

//*******************************************************************************************
// Deklarationsteil: Bausteinparameter
//*******************************************************************************************

VAR_INPUT
....
                                   // Parameter EVENT ID f¸r Meldenummer
  EV_ID   {S7_visible:='false';    // Parameter im CFC nicht sichtbar
    S7_link:='false';              // Parameter im CFC nicht verschaltbar
    S7_param :='false';            // Parameter im CFC nicht parametrierbar
    S7_server:='alarm_archiv';     // Vergabe der Meldenummer durch Server
    S7_a_type:='alarm_8p'          // Baustein meldet mit ALARM_8P
       }        :DWORD := 0;       // Meldungsnummer
...
END_VAR
```

Die nicht im Baustein benötigten Eingänge des ALARM-Bausteins können auf das Baustein-Interface gelegt werden, um dem späteren Anwender weitere Möglichkeiten für Meldungen zu geben. Durch die Definition von zusätzlichen Eingängen und der Verknüpfung im Algorithmus Ihres Bausteins, können Sie diese Meldungen sperren oder freigeben (siehe 1.6.2.). Andernfalls werden diese Meldungen ohne weitere Vorkehrungen behandelt und können nur vom Meldesystem der OS gesperrt werden. Das gilt auch für die nicht verwendeten Begleitwerte. Diese können Sie dann in den Meldungen verwenden wie in Abschnitt 1.7 beschrieben.

Im folgenden Beispiel sehen Sie die Definition des ALARM_8P:

```
//*******************************************************************************************
// Deklarationsteil: Bausteinparameter
//*******************************************************************************************
....
                  // freibelegbare Meldebegleitwerte des ALARM_8P
  AUX_PR05 {S7_visible := 'false'} : ANY;  // Begleitwert 5
  AUX_PR06 {S7_visible := 'false'} : ANY;  // Begleitwert 6
  AUX_PR07 {S7_visible := 'false'} : ANY;  // Begleitwert 7
  AUX_PR08 {S7_visible := 'false'} : ANY;  // Begleitwert 8
  AUX_PR09 {S7_visible := 'false'} : ANY;  // Begleitwert 9
  AUX_PR10 {S7_visible := 'false'} : ANY;  // Begleitwert 10
....
//*******************************************************************************************
// Deklarationsteil: statische Variablen
//*******************************************************************************************
....
//*******************************************************************************************
// Deklarationsteil Multiinstanzen
//*******************************************************************************************
  OP_A_LIM_1:  OP_A_LIM;   // Bedienbaustein 1

  ALARM_8P_1:  ALARM_8P;   // Erzeugung max. 8 Meldungen mit max. 10 Begleitwerten
...
//*******************************************************************************************
// Melden mit ALARM_8P
//*******************************************************************************************

   // STRING-Variablen d¸rfen nicht als Begleitwert auf ALARM8_P verschaltet
   // werden, deshalb in array of byte ¸bertragen

    FOR DUMMY := 1 TO 16
    DO
      sbyBA_NA[DUMMY] := 0;  //array l^schen als Vorbesetzung
    END_FOR;

    DUMMY := BLKMOV (SRCBLK:= BA_NA,DSTBLK:=sbyBA_NA);
    swSTEP_NO  := STEP_NO;     // Batch Schrittnummer (wegen I/O Begleitwert ALARM_8P)
    sdBA_ID    := BA_ID;       // Batch ID            (wegen I/O Begleitwert ALARM_8P)
    srPV_IN    := PV_IN;      // Begleitwert darf kein Input sein

 ALARM_8P_1(EN_R := TRUE,      // Aktualisierung des Ausgangs ACK_STATE
      ID := 16#EEEE,           // Datenkanal f¸r Meldungen (immer 16#EEEE)
      EV_ID:= EV_ID,           // Meldungsnummer > 0
      SIG_1:= NOT M_SUP_AH AND QH_ALM,  // zu ¸berw. Signal 1 -> Meldung Alarm oben
      SIG_2:= NOT M_SUP_AL AND QL_ALM,  // zu ¸berw. Signal 2 -> Meldung Alarm unten
      SIG_3:= 0,                 // zu ¸berwachendes Signal 3 -> keine Meldung
      SIG_4:= 0,                 // zu ¸berwachendes Signal 4
      SIG_5:= 0,                 // zu ¸berwachendes Signal 5
      SIG_6:= 0,                 // zu ¸berwachendes Signal 6
      SIG_7:= 0,                 // zu ¸berwachendes Signal 7
      SIG_8:= 0,                 // zu ¸berwachendes Signal 8
      SD_1 := sbyBA_NA,      // Begleitwert 1
      SD_2 := swSTEP_NO,     // Begleitwert 2
      SD_3 := sdBA_ID,       // Begleitwert 3
      SD_4 := srPV_IN,       // Begleitwert 4
      SD_5 := AUX_PR05,      // Begleitwert 5
      SD_6 := AUX_PR06,      // Begleitwert 6
      SD_7 := AUX_PR07,      // Begleitwert 7
      SD_8 := AUX_PR08,      // Begleitwert 8
      SD_9 := AUX_PR09,      // Begleitwert 9
      SD_10:= AUX_PR10);     // Begleitwert 10

    QMSG_ERR := ALARM_8P_1.ERROR;     // Zustandsparameter ERROR
    MSG_STAT := ALARM_8P_1.STATUS;    // Zustandsparameter STATUS
    MSG_ACK := ALARM_8P_1.ACK_STATE;  // aktueller OS Quittierzustand
....
```

1.6.1 Meldungsunterdrückung im Anlauf

Falls Sie die Belastung des AS im Anlauf durch das gleichzeitige Generieren mehrerer Meldungen (von verschiedenen Bausteinen) verringern wollen, definieren Sie dafür einen Eingangsparameter RUNUPCYC vom Datentyp INT. An diesem Parameter können Sie die Anzahl der Anlaufzyklen angeben, während derer keine Meldung generiert werden soll. Sie müssen dann im Bausteinalgorithmus die Anzahl der Aufrufe zählen und erst nach Ablauf der parametrierten Zyklen freigeben. Der folgende Ausschnitt des Beispiel-Bausteins zeigt dieses Verfahren.

```
//*****************************************************************************
// Deklarationsteil: Bausteinparameter
//*****************************************************************************
VAR_INPUT
...
  H_ALM {S7_m_c := 'true';             // Parameter ist B&B-f‰hig
     S7_visible:='false';              // Parameter ist unsichtbar
     S7_link := 'false'                // und nicht verschaltbar
     }  :REAL :=100;                   // oberer Grenzwert Alarm (Vorbesetzung 100)

  L_ALM {S7_m_c := 'true';             // Parameter ist B&B-f‰hig
     S7_visible:='false';              // Parameter ist unsichtbar
     S7_link := 'false'                // und nicht verschaltbar
     }  :REAL := 0;                    // oberer Grenzwert Alarm (Vorbesetzung 0)
...
  RUNUPCYC   {S7_visible:='false';
      S7_link:='false'} :INT := 3;     // Anzahl Erstlaufzyklen
END_VAR
//*****************************************************************************
// Deklarationsteil: statische Variablen
//*****************************************************************************
VAR
...
siRUNUPCNT   :INT := 0;          // Z‰hler f¸r RUNUPCYC-Bearbeitung
...
END_VAR
//*****************************************************************************
// Anlauf
//*****************************************************************************
    // OB100: Neustart
    100:
...
    siRUNUPCNT := RUNUPCYC;    // RUNUPCYC-Wert abspeichern
...
//*****************************************************************************
// Technologischer Teil
//*****************************************************************************
  IF siRUNUPCNT = 0            // RUNUPCYC-Zyklus bereits abgelaufen ?
  THEN
   IF (PV_IN > H_ALM) THEN     // Wenn der Prozesswert die obere Alarmgrenze verletzt
     QH_ALM  := 1;             // Fehlerausgang setzen
     QL_ALM  := 0;             // Fehlerausgang r¸cksetzen

   ELSIF (PV_IN < L_ALM) THEN  // Wenn der Prozesswert die untere Alarmgrenze verletzt
     QL_ALM  := 1;             // Fehlerausgang setzen,
     QH_ALM  := 0;             // Fehlerausgang r¸cksetzen
   ELSE
     QH_ALM  := 0;             // Fehlerausg‰nge r¸cksetzen
     QL_ALM  := 0;

   END_IF;
  ELSE
   siRUNUPCNT := siRUNUPCNT - 1;
  END_IF;
 END_CASE;
```

1.6.2 Meldungsunterdrückung für bestimmte Meldungen

Sollen im Bedarfsfall, z.B. bei vorhersehbaren Meldungen, bestimmte Meldungen unterdrückt werden, können Sie wie folgt vorgehen:

Sie definieren an Ihrem Baustein einen Eingangsparameter vom Datentyp BOOL, den Sie in Ihrem Bausteinalgorithmus auswerten, sodass bei unterdrückter Meldung dieses Ereignis nicht zum SIG-Eingang des ALARM-Bausteins weitergeleitet wird.

Im folgenden Beispiel werden die Eingänge M_SUP_AL und M_SUL_AH für die Unterdrückung eines einzelnen Alarms verwendet:

```
//*****************************************************************************
// Deklarationsteil: Bausteinparameter
//*****************************************************************************

VAR_INPUT
.....
                      // Unterdr¸ckung des ALARM LOW
  M_SUP_AL {S7_visible:='false';
    S7_link:='false';
    S7_m_c:='true';
    S7_string_0:= 'Suppress LL=No';     // Bedientext f¸r Wert (M_SUP_AL)= 0
    S7_string_1:= 'Suppress LL=Yes'     // Bedientext f¸r Wert (M_SUP_AL)= 1
      }           :BOOL;                // Meldungsunterdr¸ckung unterer Alarm Istwert

                      // Unterdr¸ckung des ALARM HIGH
  M_SUP_AH {S7_visible:='false';
    S7_link:='false';
    S7_m_c:='true';
    S7_string_0:= 'Suppress HH=No';      // Bedientext f¸r Wert (M_SUP_AH)= 0
    S7_string_1:= 'Suppress HH=Yes'      // Bedientext f¸r Wert (M_SUP_AH)= 1
      }           :BOOL;                 // Meldungsunterdr¸ckung oberer Alarm Istwert

END_VAR

//*****************************************************************************
// Melden mit ALARM_8P
//*****************************************************************************
.....
 ALARM_8P_1(EN_R := TRUE,               // Aktualisierung des Ausgangs ACK_STATE
      ID := 16#EEEE,                    // Datenkanal f¸r Meldungen (immer 16#EEEE)
      EV_ID:= EV_ID,                    // Meldungsnummer > 0
      SIG_1:= NOT M_SUP_AH AND QH_ALM,  // zu ¸berw. Signal 1 -> Meldung Alarm oben
      SIG_2:= NOT M_SUP_AL AND QL_ALM,  // zu ¸berw. Signal 2 -> Meldung Alarm unten
      SIG_3:= 0,                        // zu ¸berw. Signal 3 -> keine Meldung
      SIG_4:= 0,                        // zu ¸berw. Signal 4
.....
.....
```

1.6.3 Quelle übersetzen

Nach der Programmierung müssen Sie die Quelle mit dem SCL-Compiler übersetzen. Wählen Sie "Datei > Übersetzen" oder das Symbol für Übersetzen in der Funktionsleiste. Nach fehlerfreier Übersetzung ist der Baustein FB 501 im Bausteinordner des S7-Programms enthalten.

Weitere Informationen erhalten Sie im Handbuch "S7-SCL für S7-300 und S7-400".

1.7 Meldungsprojektierung

Allgemeines

Für einen Baustein, der Meldungen an die OS schicken soll, müssen Sie in den statischen Variablen die Multiinstanz eines Alarm-Bausteins definieren.

Sie können mit dem ALARM_8 / ALARM_8P-Baustein bis zu 8 Signale überwachen, die Sie als Parameter am Alarm-Baustein angeben. Der Baustein merkt sich bei jedem Aufruf den aktuellen Zustand der Signale und sendet beim nächsten Aufruf eine Meldung an die OS, falls sich eines der Signale geändert hat.

Meldungsprojektierung im SIMATIC Manager

Den EV_ID können Sie im SIMATIC Manager für den selektierten Baustein über den Dialog **Bearbeiten > Spezielle Objekteigenschaften > Meldungen...** bearbeiten.

Dabei können Sie einzelne Bestandteile der Meldungen (z.B. Meldetext, Meldeklasse usw.) gegen Änderung an anderer Stelle verriegeln, d.h. Sie können verhindern, dass bei einem Einbau Ihres Bausteins in einen CFC-Plan diese Meldung an der Bausteininstanz veränderbar ist.

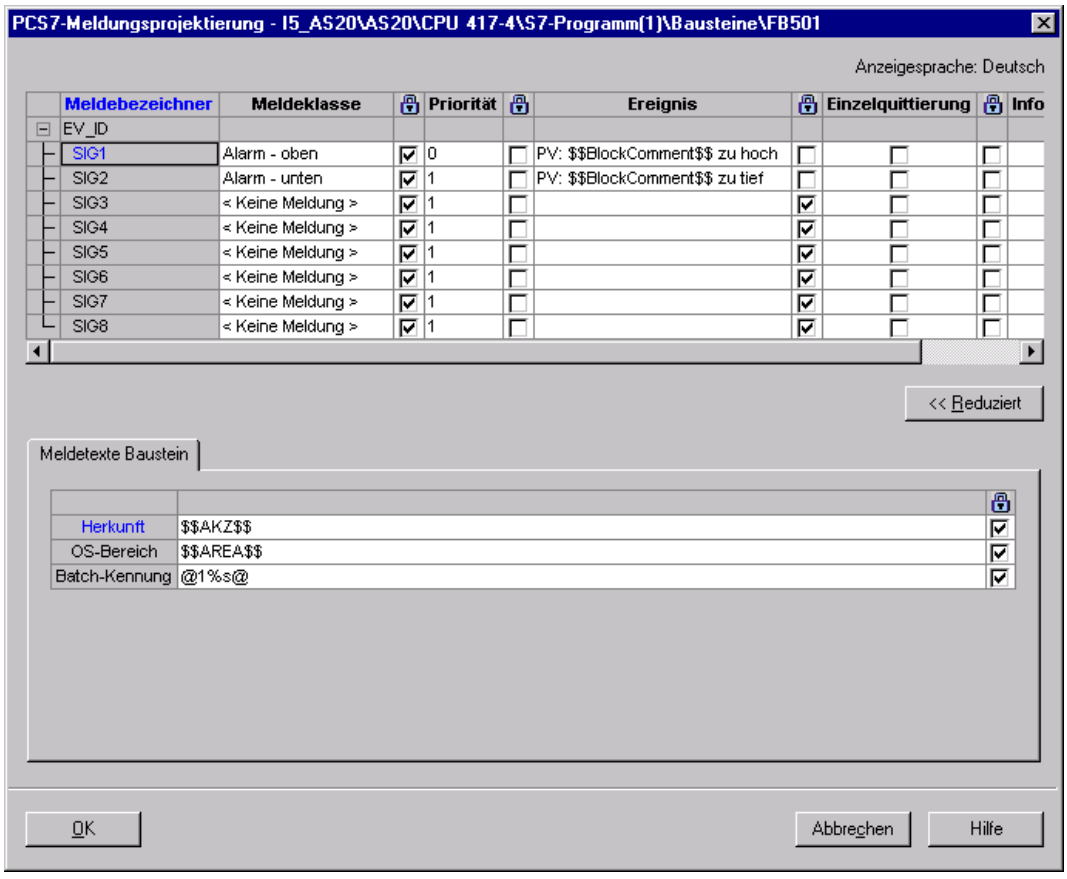


Bild 1-7 : Meldungsprojektierung im SIMATIC Manager

Zuerst müssen Sie die Texte eingeben, die für alle Meldungen dieses Bausteins gelten. Die einzelnen Texte entsprechen dabei den Anwendertextblöcken im AlarmLogging von WinCC.

Herkunft:

Hier können Sie die Herkunft der Meldung angeben.

Geben Sie die Herkunft als Schlüsselwort $$AKZ$$ ein, wird beim Transferieren der Daten beim Übersetzen der OS der Pfad des Hierarchieordners, der Planname und der Bausteinname ermittelt und in den OS-Meldetexten abgelegt.
Hinweis: Der TH-Pfad wird nur eingetragen, wenn die entsprechenden Hierarchieordner namensbildend sind (Eigenschaften THO bzw. Einstellungen der TH).

OS-Bereich:

Hier können Sie die Bereichszuordnung der Meldung angeben.

Geben Sie den Bereich als Schlüsselwort $$AREA$$ ein oder tragen Sie nichts ein, wird beim Transferieren der Daten beim Übersetzen der OS das entsprechende Attribut des Hierarchieordners ausgewertet und in den OS-Meldetexten abgelegt.

Batch-Kennung:

Hier können Sie eine Batch-Kennung für die Meldung angeben.

Geben Sie die Batch-Kennung ein, wird beim Transferieren der Daten beim Übersetzen der OS das entsprechende Attribut ausgewertet und in der Meldeliste von WinCC in der Spalte "Charge Name" abgelegt. Es handelt sich hierbei aber nicht um die Batch-ID sondern um den Batch-Namen. Falls Ihr Baustein für das S7-Optionspaket SIMATIC BATCH geeignet sein soll, müssen Sie hier @1%s@ eintragen. Dadurch wird die Meldung mit der BATCH-Chargenbezeichnung als ersten Meldebegleitwert versehen (vgl. Kapitel 1.8). Falls Sie SIMATIC BATCH nicht verwenden, müssen Sie hier nichts angeben.

Meldeklassen

Anschließend legen Sie für jede Meldung die Meldeklasse fest. Sobald Sie in der jeweiligen Meldungszeile die Spalte "Meldeklasse" anklicken, wird diese zum Kombinationsfeld und Sie können die Meldeklasse auswählen. Nicht belegte Meldungen müssen die Meldeklasse "< keine Meldung >" erhalten. Genaueres über die Behandlung von Meldungen können Sie der Dokumentation zu WinCC entnehmen.

Tragen Sie in der Spalte "Ereignis" eine Beschreibung der Fehlerursache ein (maximal 40 Zeichen inklusive möglicher Begleitwerte) und in der Spalte "Einzelquittierung", ob die Meldung einzeln quittiert werden muss (falls das Auswahlkästchen angewählt ist) oder ob sie per Sammelquittierung quittiert werden kann.

In der Spalte "Locked" (blaues Schloss-Symbol) definieren Sie, ob die Meldeklasse vom Anwender des Bausteins instanzspezifisch im CFC geändert werden darf (Auswahlkästchen nicht angewählt) oder gesperrt ist (Auswahlkästchen angewählt).

Priorität

Hier können unterschiedliche Prioritäten einer Meldung vergeben werden.

In der Spalte "Locked" (blaues Schloss-Symbol) definieren Sie, ob die Priorität vom Anwender des Bausteins instanzspezifisch im CFC geändert werden darf (Auswahlkästchen nicht angewählt) oder gesperrt ist (Auswahlkästchen angewählt).

Ereignis

Geben Sie in diesem Feld den Meldetext ein.

Begleitwerte für Meldungen

Soll eine Meldung zusätzliche Informationen (z.B. Messwerte) an die OS übertragen, müssen Sie zum Melden einen ALARM-Baustein verwenden, der Ihnen die Angabe von Begleitwerten erlaubt (ALARM_8P = 10 Begleitwerte). Die an den Parametern SD_1 bis SD_10 des ALARM-Bausteins übergebenen Werte können Sie in der folgenden Form in die Meldungstexte einblenden:

@ Parameternummer Formatanweisung @

Im folgenden Beispiel wird der am Parameter SD_4 angegebene Wert in dezimaler Form ausgegeben. Die angebbaren Formatanweisungen entsprechen der C-Syntax.

@4%d@

Mit Hilfe des Schlüsselwortes **$$BlockComment$$** können Sie den Kommentar der Bausteininstanz in den Meldetext einblenden.

In der Spalte "Locked" (blaues Schloss-Symbol) definieren Sie, ob der Meldetext vom Anwender des Bausteins instanzspezifisch im CFC geändert werden darf (Auswahlkästchen nicht angewählt) oder gesperrt ist (Auswahlkästchen angewählt).

Hinweis: Die Begleitwerte eines ALARM_8P sind vom Datentyp ANY, Anschlusstyp I/O. In der Online-Hilfe des ALARM_8P sind die zulässigen Datentypen der Begleitwerte aufgezählt. Wenn Sie im Baustein-Interface die Begleitwerte nicht als I/O sondern als Input definieren wollen, müssen Sie diese auf eine Variable in der VAR-Sektion speichern und diese Variable als Begleitwert mit ALARM_8P übertragen.

Der Datentyp STRING darf nicht als Begleitwert übertragen werden. Sie müssen diesen, wie im Beispiel-Baustein implemetiert, in ein ARRAY OF BYTE kopieren und dieses ARRAY als Begleitwert übertragen.

Einzelquittierung

Klicken Sie das Optionskästchen an, wenn die Meldung als Einzelmeldung quittiert werden soll.

Infotext

Geben Sie in diesem Feld den Infotext ein.

1.8 Anbindung von SIMATIC BATCH

Falls Sie Ihren Bausteine mit dem S7-Optionspaket "SIMATIC BATCH" verwenden wollen, müssen Sie folgende Eingangs- bzw. Durchgangsparameter definieren:

Parametername	Bedeutung	Parametertyp	Datentyp
BA_EN	BATCH-Belegfreigabe	INPUT	BOOL
BA_NA	BATCH-Chargenbezeichnung	INPUT	STRING[32]
BA_ID	laufende Chargennummer	INPUT	DWORD
OCCUPIED	BATCH-Belegkennung	INPUT	BOOL
STEP_NO	BATCH-Schrittnummer	INPUT	DWORD

Ausschnitt des Beispiel-Bausteins:

```
//*****************************************************************************
// Deklarationsteil: Bausteinparameter
//*****************************************************************************
VAR_INPUT
....
                  // Parameter f¸r SIMATIC BATCH
  STEP_NO {S7_visible := 'false';
      S7_m_c   := 'true'}  :DWORD;             // Batch Schrittnummer
  BA_ID {S7_visible  := 'false';
     S7_m_c    := 'true'}  :DWORD;             // Batch laufende Chargennummer
  BA_EN {S7_visible := 'false';                // Parameter im CFC-Plan unsichtbar
     S7_m_c := 'true'                          // Parameter ist B&B-f‰hig
     }             :BOOL := 0;                 // Batch Belegt-Freigabe
  BA_NA {S7_visible := 'false';
     S7_m_c   := 'true'} :STRING[32] := '';  // Batch Chargenbezeichnung

  OCCUPIED {S7_visible := 'false';
       S7_m_c   := 'true'} :BOOL := 0;         // Batch Belegt-Kennung
....
END_VAR
```

Falls Sie in einem solchen Baustein Meldungen generieren wollen, müssen Sie dabei die Eingänge BA_NA, STEP_NO und BA_ID (in dieser Reihenfolge) als Meldebegleitwerte verwenden.

Die Begleitwerte sind wie folgt belegt:

Begleitwert	Bedeutung
1	BATCH-Chargenbezeichnung BA_NA
2	BATCH-Schrittnummer STEP_NO
3	BATCH: laufende Chargennummer BA_ID
4 bis 7	Bausteinspezifisch belegt oder frei für Anwender.

1.9 Erstellen von CFC-Bausteintypen

Im Unterschied zur Programmierung mit SCL, bei der Variablen deklariert und Zuweisungen ausprogrammiert werden, basiert CFC auf der Verschaltung von grafischen Objekten. D.h. Sie können damit durch Platzieren und Verschalten von bereits vorhandenen Bausteinen neue Bausteine entwickeln.

Die folgende Beschreibung soll Ihnen nur einen Überblick und die prinzipielle Vorgehensweise erläutern. Ein ausführliche Beschreibung der Bausteinerstellung im CFC finden Sie im Handbuch "CFC für S7" bzw. in der CFC-Online-Hilfe.

1.9.1 Beispiel: CONTROL2

Der fertiggestellte Beispiel-Baustein "CONTROL" soll um einen Multiplizierer erweitert werden. Mit diesem wird durch die Multiplikation zweier Eingangswerte (IN1 und IN2) der Istwert gebildet. Der erweiterte Baustein soll als CONTROL2 (FB 601) erzeugt werden.

Vorgehensweise:

- Öffnen Sie einen neuen CFC-Plan und platzieren Sie darin den Beispiel-Baustein **CONTROL**.
- Aus der CFC Library\ELEMENTA ziehen Sie mit Drag&Drop einen Multiplizierer **MUL_R** (FC 63) in den Plan.
- Verschalten Sie den Ausgang "OUT" des **MUL_R** mit dem Prozesswert (Parameter "PV_IN") des Beispiel-Bausteins.
- Öffnen Sie im Plan den Interface-Editor (Menü "Ansicht > Plananschlüsse" oder mit dem Symbol ) und selektieren Sie im linken Fenster der "Plananschlüsse" das Symbol "IN".
- Verschalten Sie die Eingänge "IN1" und "IN2" des **MUL_R** mit den Plananschlüssen: Mit Drag&Drop ziehen Sie den Bausteinanschluss auf den Plananschluss (rechtes Fenster).
- Schalten Sie alle unsichtbaren Anschlüsse - außer EN und ENO - des Bausteins CONTROL sichtbar (Objekteigenschaften > Register: Anschlüsse > Spalte: Unsichtbar > gesetzte Anschlüsse rücksetzen).
- Verschalten Sie alle sichtbaren Anschlüsse (außer dem bereits verschalteten Istwert) mit den Plananschlüssen des CFC-Plans.
- Übersetzen Sie über den Dialog "*Plan > Übersetzen > Plan als Bausteintyp*" den CFC-Plan als Baustein.
  - Im Register: "Allgemein" tragen Sie die FB-Nummer ein (hier 601). Tragen Sie danach in den entsprechenden Feldern die weiteren Eigenschaften ein: Symbolischer Name, Familie, Autor, Version. Im Feld Name (Header) ist bereits der Name des CFC-Plans vorbesetzt.
  - Im Register: "Attribute" geben Sie die gewünschten Baustein-Attribute und Systemattribute an. Das Systemattribut "S7_tasklist" müssen Sie hier nicht angeben (siehe nachfolgende Einbauregel).
  - Starten Sie die Übersetzung mit "OK".

Einbauregel:

Der CFC-Bausteintyp wird in den als Default vorgesehenen zyklischen OB (z.B. OB 35) und in jeden OB eingebaut, der in einer Taskliste eines unterlagerten Bausteins enthalten ist, d.h. seine Taskliste ist die Vereinigungsmenge der Tasklisten der unterlagerten Bausteine. Die unterlagerten Bausteine selbst werden jedoch nur in den OBs aufgerufen, die in ihrer eigenen Taskliste enthalten sind. Im hier angegeben Beispiel heißt dies:

- Der Beispiel-Bausteins **CONTROL** hat die Taskliste "S7_tasklist = 'OB80,OB100' ".
- Der Multiplizierer **MUL_R** hat keine Taskliste.
- Der CFC-Bausteintyp hat daher die Taskliste "S7_tasklist = 'OB80,OB100' ". Im OB80 und OB100 wird aber nur der **CONTROL** aufgerufen, nicht der **MUL_R**.

Wenn der CONTROL-Baustein nicht im OB 35 aufgerufen werden soll, müssen Sie Ihn aus dem OB 35 entfernen bzw. in einen anderen OB verschieben. Das Gleiche gilt für den Baustein MUL_R.

1.10 Namenskonventionen und Nummernbereich

Nummernbereich

Um Nummernkonflikte mit den von Siemens gelieferten leittechnischen PCS 7-Bausteinen zu vermeiden, sollten Sie die Nummerierung Ihrer Bausteine ab der Nummer 501 beginnen. Bitte berücksichtigen Sie bei der Festlegung Ihrer Bausteinnummern auch die Leistungsdaten der von Ihrer Bibliothek unterstützten CPU-Typen.

Namen

Bei der Benennung Ihrer Bausteinparameter sollten Sie folgende Regel beachten:

Binäre Ausgänge beginnen mit **Q**, z.B. QH_ALM oder Q_L_ALM

1.11 Quellcode des Beispiels

```
//Ersteller: ABC       Datum: 13.08.00              Vers.:1.00
//Ge‰ndert:            Datum: 18.11.03              Vers.:

//*****************************************************************************************
// Bausteinkopf
//*****************************************************************************************

FUNCTION_BLOCK "CONTROL"
TITLE =    'CONTROL'
{  // Liste der Systemattribute
S7_tasklist:= 'OB80,OB100';  // Baustein wird bei Zeitfehler u. Neustart aufgerufen
S7_m_c:=     'true';      // Bausteins ist bedien- und beobachtbar
S7_alarm_ui:= '1'            // Einstellung PCS7-Meldedialog ('0'=Standard-Meldedialog)
}
AUTHOR:           ABC
NAME:             CONTROL
VERSION:          '0.02'
FAMILY:           XYZ
KNOW_HOW_PROTECT

//*****************************************************************************************
// Deklarationsteil: Bausteinparameter
//*****************************************************************************************

VAR_INPUT
SAMPLE_T {S7_sampletime:= 'true';  // Param. der Baustein-Abtastzeit (Zyklus der Task)
       S7_visible:='false';        // Parameter ist unsichtbar
       S7_link:= 'false'           // Parameter nicht verschaltbar
      }          :REAL := 1;       // Verz^gerungszeit [s] (Vorbesetzung 1 Sek)

  H_ALM {S7_m_c := 'true';
     S7_visible:='false';
     S7_link := 'false'}  :REAL :=100;  // oberer Grenzwert Alarm (Vorbesetzung 100)

  L_ALM {S7_m_c := 'true';         // Parameter ist B&B-f‰hig
     S7_visible:='false';          // Parameter ist unsichtbar
     S7_link := 'false'            // und nicht verschaltbar
     }           :REAL := 0;       // unterer Grenzwert Alarm (Vorbesetzung 0)

  M_SUP_AL {S7_visible:='false';
    S7_link:='false';
    S7_m_c:='true';
    S7_string_0:= 'Suppress LL=No';    // Bedientext f¸r Wert (M_SUP_AL)= 0
    S7_string_1:= 'Suppress LL=Yes'    // Bedientext f¸r Wert (M_SUP_AL)= 1
      }           :BOOL;               // Meldungsunterdr¸ckung unterer Alarm

  M_SUP_AH {S7_visible:='false';
    S7_link:='false';
    S7_m_c:='true';
    S7_string_0:= 'Suppress HH=No';
    S7_string_1:= 'Suppress HH=Yes'
      }           :BOOL;            // Meldungsunterdr¸ckung oberer Alarm

  SP_OP_ON {S7_visible:='false';
      S7_dynamic:='true'            // CFC in Test/IBS: Anzeige aktueller Wert in AS)
      }           :BOOL := 1;       // 1 = Bedienfreigabe f¸r Sollwert-Eingabe

  SPBUMPON {S7_visible:='false';
       S7_link:='false';
       S7_m_c:='true';
       S7_string_0:='SP bumpless=Off';
       S7_string_1:='SP bumpless=On'
      }
                :BOOL := 1;       // 1 = Stoflfreiheit f¸r Sollwert
  SP_EXTON {S7_visible:='false';
      S7_dynamic:='true'            // CFC in Test/IBS: Anzeige aktueller Wert in AS)
      }
                :BOOL := 1;          // 1: Externer Sollwert ein

  SP_EXT   {S7_dynamic:='true'}

                 :REAL := 0;        // Externer Sollwert
```

```
  SP_HLM   {S7_visible:='false';
    S7_link:='false';
    S7_m_c:='true';
    S7_shortcut:='SP high limit';     // Text(max 16 Zeichen) zur Anzeige auf OS
    S7_unit:=''}                      // Einheit (max 16 Zeichen)
               :REAL := 100;          // oberer Bediengrenzwert f¸r Sollwert

  SP_LLM    {S7_visible:='false';
    S7_link:='false';
    S7_m_c:='true';
    S7_shortcut:='SP low limit';
    S7_unit:=''}
               :REAL := 0;            // unterer Bediengrenzwert f¸r Sollwert

PV_IN {S7_dynamic:='true';
    S7_m_c:='true';
    S7_unit:='%'} :   REAL := 0;      // Prozesswert (zu Begleitwert_PR04)

GAIN   {S7_link:='false';
    S7_edit:='para';                  // Parametrierung im IEA
    S7_m_c:='true';
    S7_shortcut:='Gain';
    S7_unit:=''}        :REAL := 1;   // Proportionalbeiwert

  EV_ID   {S7_visible:='false';
    S7_link:='false';
    S7_param :='false';               // Parameter im CFC nicht parametrierbar
    S7_server:='alarm_archiv';        // Vergabe der Meldenummer durch Server
    S7_a_type:='alarm_8p'             // Baustein meldet mit ALARM_8P
       }           :DWORD := 0;       // Meldungsnummer

                                      // Parameter f¸r SIMATIC BATCH
  STEP_NO {S7_visible := 'false';
      S7_m_c   := 'true'}  :DWORD;    // Batch Schrittnummer
  BA_ID {S7_visible  := 'false';
     S7_m_c    := 'true'}  :DWORD;    // Batch laufende Chargennummer
  BA_EN {S7_visible := 'false';       // Parameter im CFC-Plan unsichtbar
     S7_m_c := 'true'                 // Parameter ist B&B-f‰hig
     }            :BOOL := 0;         // Batch Belegt-Freigabe
  BA_NA {S7_visible := 'false';
     S7_m_c   := 'true'} :STRING[32] := '';  // Batch Chargenbezeichnung

  OCCUPIED {S7_visible := 'false';
       S7_m_c   := 'true'} :BOOL := 0;       // Batch Belegt-Kennung

  RUNUPCYC   {S7_visible:='false';
      S7_link:='false'} :INT := 3;    // Anzahl Erstlaufzyklen
  SUPPTIME       :REAL := 0;          // Verz^gerungszeit
  SUPP_IN        :REAL := 0;          // Eingangswert f¸r Verz^gerungszeit
END_VAR
```

```
VAR_OUTPUT
  LMN {S7_shortcut:='pressure';     // Bezeichnung des Parameters auf OS
    S7_unit := '%';                 // Einheit des Parameters
    S7_m_c := 'true'                // beobachtbar
    } :REAL;                        // Stellwert

  QH_ALM  :BOOL := false;           // 1 = Oberer Grenzwert Alarm hat angesprochen

  QL_ALM  :BOOL := false;           // 1 = Unterer Grenzwert Alarm hat angesprochen

  QSP_HLM   {S7_visible:='false';
    S7_dynamic:='true'}    :BOOL := 0;   // 1 = Sollwertausgang nach oben begrenzt

  QSP_LLM   {S7_visible:='false';
    S7_dynamic:='true'}    :BOOL := 0;   // 1 = Sollwertausgang nach unten begrenzt

  Q_SP_OP   {S7_visible:='false';
    S7_dynamic:='true';
    S7_m_c:='true'}        :BOOL := 0;   // Status: 1 = Bedienfreigabe f¸r Sollwert

  QOP_ERR   {S7_visible:='false';
    S7_dynamic:='true'}   :BOOL := 0;    // 1 = Bedienfehler

  QMSG_ERR   {S7_visible:='false';
    S7_dynamic:='true'}   :BOOL := 0;    // ALARM_8P: Meldungsfehler

  MSG_STAT   {S7_visible:='false';
    S7_dynamic:='true'}   :WORD := 0;    // Meldungsfehler-Informationen

  MSG_ACK   {S7_visible:='false';
    S7_dynamic:='true'}   :WORD := 0;    // Meldungen quittieren

  SUPP_OUT                :REAL := 0;    // Ausgangswert f¸r Verz^gerungszeit
  SP   {S7_dynamic:='true';
    S7_m_c:='true'}       :REAL := 0;    // Aktiver Sollwert

END_VAR

VAR_IN_OUT
  SP_OP {S7_visible:='false';
    S7_link:='false';
    S7_m_c:='true';
    S7_shortcut:='Setpoint';
    S7_unit:='%'} :   REAL := 0;         // Bedieneingabe Sollwert

 // freibelegbare Meldebegleitwerte des ALARM_8P

  AUX_PR05 {S7_visible := 'false'} : ANY;  // Begleitwert 5
  AUX_PR06 {S7_visible := 'false'} : ANY;  // Begleitwert 6
  AUX_PR07 {S7_visible := 'false'} : ANY;  // Begleitwert 7
  AUX_PR08 {S7_visible := 'false'} : ANY;  // Begleitwert 8
  AUX_PR09 {S7_visible := 'false'} : ANY;  // Begleitwert 9
  AUX_PR10 {S7_visible := 'false'} : ANY;  // Begleitwert 10

END_VAR
```

```
//*****************************************************************************
// Deklarationsteil: statische Variablen
//*****************************************************************************
VAR
  sbRESTART   :BOOL := TRUE;       // Erstlauf Merker
  siRUNUPCNT  :INT := 0;           // Z%hler f¸r RUNUPCYC-Bearbeitung
  sSUPP_IN    :REAL := 0;          // Altwert des Delay-Beispieleingangs
  ACT_TIME    :REAL := 0;          // Zeitz%hler

  srPV_IN     :REAL := 0;          // Begleitwert f¸r PV_IN
  swSTEP_NO   :DWORD;              // Batch Schrittnummer
  sdBA_ID     :DWORD;              // Batch Chargennummer

  sbyBA_NA    :ARRAY[1..32] OF BYTE := 32(0);

//*****************************************************************************
// Deklarationsteil Multiinstanzen
//*****************************************************************************
  OP_A_LIM_1:  OP_A_LIM;   // Bedienbaustein 1
  ALARM_8P_1:  ALARM_8P;   // Erzeugung max. 8 Meldungen mit max. 10 Begleitwerten
END_VAR

//*****************************************************************************
// Deklarationsteil: tempor%re Variablen
//*****************************************************************************
VAR_TEMP
  // Startinfo: Struktur mit Info f¸r den OB, der den Baustein gerade aufgerufen hat
  TOP_SI:    STRUCT
   EV_CLASS :BYTE;
   EV_NUM   :BYTE;
   PRIORITY :BYTE;
   NUM      :BYTE;
   TYP2_3   :BYTE;
   TYP1     :BYTE;
   ZI1      :WORD;
   ZI2_3    :DWORD;
  END_STRUCT;

  // Startinfo: Struktur mit Info f¸r den letzten aufgerufenen Anlauf-OB
  START_UP_SI:  STRUCT
   EV_CLASS :BYTE;
   EV_NUM   :BYTE;
   PRIORITY :BYTE;
   NUM      :BYTE;
   TYP2_3   :BYTE;
   TYP1     :BYTE;
   ZI1      :WORD;
   ZI2_3    :DWORD;
  END_STRUCT;

  S7DT   :DATE_AND_TIME;           // Lokale Zeitvariable
  DUMMY  :INT;                     // Hilfsvariable
END_VAR
```

```
//*****************************************************************************
// Codeteil
//*****************************************************************************
//*****************************************************************************
// Abh%ngigkeiten vom aufrufenden OB
//*****************************************************************************
   // Auslesen der Startinfo mittels SFC6 (RD_SINFO)
   DUMMY := RD_SINFO (TOP_SI := TOP_SI, START_UP_SI := START_UP_SI);

 IF sbRESTART THEN
   // Erstlauf
   TOP_SI.NUM := 100;          // Erstlauf als Neustart ausf¸hren
   sbRESTART := FALSE;         // R¸cksetzen Erstlauf
 END_IF;

   // Aus welchem OB wurde der Baustein aufgerufen ?
 CASE WORD_TO_INT(BYTE_TO_WORD(TOP_SI.NUM)) OF

//*****************************************************************************
// Behandlung von Fehler-OBs
//*****************************************************************************
// OB80: Zeitfehler
    80:
    QH_ALM  := 0;       // Fehlerausg%nge zur¸cksetzen
    QL_ALM  := 0;
//*****************************************************************************
// Anlauf
//*****************************************************************************
// OB100: Neustart
    100:
    QH_ALM   := 0;           // Fehlerausg%nge zur¸cksetzen
    QL_ALM   := 0;
    siRUNUPCNT := RUNUPCYC;  // RUNUPCYC-Wert abspeichern
 ELSE
//*****************************************************************************
// Bedieneingabe Sollwert SP_OP (Real-Wert) oder verschalteter Sollwert SP_EXT
//*****************************************************************************
   //Multiinstanz Aufruf OP_A_LIM (Bedeutung der Parameter siehe Online Hilfe OP_A_LIM)
   OP_A_LIM_1(U := SP_OP, U_HL:= SP_HLM, U_LL:= SP_LLM, OP_EN:= SP_OP_ON, BTRACK:=
SPBUMPON, LINK_ON:= SP_EXTON, LINK_U:= SP_EXT);

   OK := OK AND ENO;  //Enable Out des OP_A_LIM in OK-Flag des Bausteins ¸bernehmen
   Q_SP_OP := OP_A_LIM_1.QOP_EN;  // 1: Freigabe Bedieneingabe SP
   QOP_ERR := OP_A_LIM_1.QOP_ERR; // 1: Bedienfehler
   QSP_HLM := OP_A_LIM_1.QVHL;    // 1: Begrenzung Obergrenze
   QSP_LLM := OP_A_LIM_1.QVLL;    // 1: Begrenzung Untergrenze
   SP   := OP_A_LIM_1.V;          // wirksamer Sollwert

//*****************************************************************************
// Technologischer Teil
//*****************************************************************************
IF (SUPP_IN <> sSUPP_IN) THEN
   ACT_TIME := SUPPTIME;            // Zeitz%hler initialisieren
   sSUPP_IN := SUPP_IN;
END_IF;

  IF (ACT_TIME > 0) THEN            // Wenn Wartezeit noch nicht abgelaufen ist
   ACT_TIME := ACT_TIME-SAMPLE_T;   // Wartezeit herunterz%hlen
  ELSE
   SUPP_OUT := SUPP_IN;             // Eingang durchschalten
  END_IF;
```

```
  LMN := GAIN * (SP - PV_IN);       // Stellgrˆfle berechnen
  IF siRUNUPCNT = 0                 // RUNUPCYC-Zyklus bereits abgelaufen ?
  THEN
   IF (PV_IN > H_ALM) THEN      // Wenn der Prozesswert die obere Alarmgrenze verletzt
     QH_ALM  := 1;              // Fehlerausgang setzen
     QL_ALM  := 0;              // Fehlerausgang r¸cksetzen

   ELSIF (PV_IN < L_ALM) THEN  // Wenn der Prozesswert die untere Alarmgrenze verletzt
     QL_ALM  := 1;              // Fehlerausgang setzen,
     QH_ALM  := 0;              // Fehlerausgang r¸cksetzen
   ELSE
     QH_ALM  := 0;              // Fehlerausg%nge r¸cksetzen
     QL_ALM  := 0;

   END_IF;
  ELSE
   siRUNUPCNT := siRUNUPCNT - 1;
  END_IF;
END_CASE;

//*****************************************************************************************
// Melden mit ALARM_8P
//*****************************************************************************************

  // STRING-Variablen d¸rfen nicht als Begleitwert auf ALARM8_P verschaltet werden,
  // deshalb in array of byte ¸bertragen

    FOR DUMMY := 1 TO 32
     DO
      sbyBA_NA[DUMMY] := 0;    // array lˆschen als Vorbesetzung
     END_FOR;

     DUMMY := BLKMOV (SRCBLK:= BA_NA,DSTBLK:=sbyBA_NA);
     swSTEP_NO  := STEP_NO;    // Batch Schrittnummer (wegen I/O Begleitwert ALARM_8P)
     sdBA_ID   := BA_ID;       // Batch Chargennummer (wegen I/O Begleitwert ALARM_8P)
     srPV_IN   := PV_IN;       // Begleitwert darf kein INPUT sein

 ALARM_8P_1(EN_R := TRUE,        // Aktualisierung des Ausgangs ACK_STATE
      ID := 16#EEEE,             // Datenkanal f¸r Meldungen (immer 16#EEEE)
      EV_ID:= EV_ID,             // Meldungsnummer > 0
      SIG_1:= NOT M_SUP_AH AND QH_ALM,  // zu ¸berw. Signal 1 -> Meldung Alarm oben
      SIG_2:= NOT M_SUP_AL AND QL_ALM,  // zu ¸berw. Signal 2 -> Meldung Alarm unten
      SIG_3:= 0,                 // zu ¸berwachendes Signal 3 -> keine Meldung
      SIG_4:= 0,                 // zu ¸berwachendes Signal 4
      SIG_5:= 0,                 // zu ¸berwachendes Signal 5
      SIG_6:= 0,                 // zu ¸berwachendes Signal 6
      SIG_7:= 0,                 // zu ¸berwachendes Signal 7
      SIG_8:= 0,                 // zu ¸berwachendes Signal 8
      SD_1 := sbyBA_NA,          // Begleitwert 1
      SD_2 := swSTEP_NO,         // Begleitwert 2
      SD_3 := sdBA_ID,           // Begleitwert 3
      SD_4 := srPV_IN,           // Begleitwert 4
      SD_5 := AUX_PR05,          // Begleitwert 5
      SD_6 := AUX_PR06,          // Begleitwert 6
      SD_7 := AUX_PR07,          // Begleitwert 7
      SD_8 := AUX_PR08,          // Begleitwert 8
      SD_9 := AUX_PR09,          // Begleitwert 9
      SD_10:= AUX_PR10);         // Begleitwert 10

    QMSG_ERR := ALARM_8P_1.ERROR;      // Zustandsparameter ERROR
    MSG_STAT := ALARM_8P_1.STATUS;     // Zustandsparameter STATUS
    MSG_ACK := ALARM_8P_1.ACK_STATE;   // aktueller OS Quittierzustand
END_FUNCTION_BLOCK
```

2 Bildbausteine projektieren

2.1 Allgemeines zur Projektierung

In den nachfolgenden Kapiteln erhalten Sie alle Informationen, um einen Bildbaustein für PCS 7 zu erstellen. Das umfasst die Verwendung der für die Erstellung benötigten WinCC-Tools und im Besonderen das Arbeiten mit dem Faceplate Designer.

Hinweis

Die in dieser Beschreibung abgebildeten Bausteinsymbole und Bildbausteine sind exemplarisch und können in Details von der Darstellung der aktuellen Objekte abweichen.

Voraussetzungen und Vorkenntnisse

Die hier beschriebenen Bildbausteine sind für die Verwendung in WinCC gedacht. Für die Erstellung der Bausteine benötigen Sie das WinCC-Basispaket mit den leittechnischen Optionen "Basic Process Control" und "Advanced Process Control".

Folgende Kenntnisse werden vorausgesetzt:

- SIMATIC WinCC Systemkurs
  (angeboten vom A&D Trainingscenter unter ST-BWINCCS)
- SIMATIC WinCC Offenheit N
  (angeboten vom A&D Trainingscenter unter ST-BWINCCN)

2.1.1 Phasen der Erstellung

Erstellungsweg

Bei der Erstellung eines Bildbausteins hat sich die folgende Vorgehensweise als vorteilhaft erwiesen:

- Entwurf des Bildbausteins
- Projektierung des Bildbausteins
- Test des Bildbausteins

2.1.1.1 Entwurf des Bildbausteins

Darstellung

Ein Bildbaustein ist die B&B-Schnittstelle zu einem AS-Baustein. Für die Anzeige eines Bildbausteins gibt es zwei Darstellungsarten:

- **Gruppendarstellung:** Darstellung der AS-Werte in unterschiedlichen Sichten mit Anwahlelement für die Kreisbilddarstellung.
- **Kreisbilddarstellung:** Darstellung der Elemente aller Sichten der Gruppendarstellung.

Systemattribute

Welche Ein- und Aus- und Durchgangsparameter eines AS-Bausteins bedient und beobachtet werden können, wird bei der Erstellung des AS-Bausteins durch die Systemattribute festgelegt. Einzelheiten zu diesen Systemattributen können Sie im Kapitel "Aufbau eines AS-Bausteins" nachlesen (Tabelle 1-1 und Tabelle 1-2).

Parameter

Die Auswahl der Parameter erfolgt unter den folgenden Gesichtspunkten:

- Welche Daten benötigt das Bedienpersonal um schnell und zweifelsfrei den aktuellen Zustand zu erfassen?
- Wie sollen diese Werte dargestellt werden?
- Welche Werte soll das Bedienpersonal verändern können?
- Welche Berechtigungsstufe ist für die Bedienung erforderlich?
- Sind prozessabhängige Bedienverriegelungen erforderlich?
- In welcher Sicht des Bildbausteins sollen die einzelnen Werte dargestellt werden?

  Tipp: Gruppieren Sie dazu die einzelnen Parameter nach Funktionen. Platzieren Sie in der Sicht "Standard" die wichtigsten Elemente, vor allem jene, die sich fortlaufend ändern.

Gestaltung

Nach der Festlegung der Parameter und ihrer Darstellung erfolgt das Gestalten des Bildbausteins, d.h. die Auswahl der Bildelemente, deren Benennung, die Parametrierung der Bildelemente und deren Position. Verwenden Sie immer Namen, die einen Bezug zum dargestellten Objekt haben und auch ausgesprochen werden können.

Beispiel: In einer Zustandsanzeige soll die Variable "OCCUPIED" dargestellt werden → nennen Sie diese Zustandsanzeige "OCCUPIED".
Diese Vorgehensweise erleichtert die Projektdokumentation und die Wartung.

2.1.1.2 Projektierung des Bildbausteins

Für die Projektierung des Bildbausteins steht Ihnen das Werkzeug "WinCC Graphics Designer" zur Verfügung. Ausgehend von den Vorlagen des Faceplate Designers setzen Sie den Bildentwurf in WinCC-Bilder um. Eine ausführliche Beschreibung hierzu finden Sie im Kapitel 2.1.2, "Bildbausteinerstellung mit dem Faceplate Designer".

2.1.1.3 Test des Bildbausteins

Beim Test des Bildbausteins gehen Sie in 2 Schritten vor:

1. Prüfen Sie im WinCC Explorer die Eigenschaften der erstellten Bilder:
   - Sind die Parameternamen richtig geschrieben?
   - Haben mehrfach dargestellte Parameter den identischen WinCC-Zyklus? Unterschiedliche Zyklen verwirren das Bedienpersonal (z.B. wenn die Balkenanzeige inkonsistent zur numerischer Anzeige ist) und erhöhen die Kommunikationslast des Systems.
   - Sind die Direktverbindungen korrekt?
   - Sind alle ereignisgesteuerte Skripte vorhanden?
2. Prüfen Sie in WinCC Runtime:
   - Wird bei Mausklick auf das Bausteinsymbol der Bildbaustein in der Gruppendarstellung geöffnet?
   - Funktioniert in der Gruppendarstellung die Umschaltung der einzelnen Sichten?
   - Wird bei Mausklick auf die Taste "Kreisdarstellung" der Bildbaustein in der Kreisdarstellung geöffnet?
   - Werden die Werte des AS-Bausteins korrekt angezeigt?
   - Ist die Anzeige von Melde- und Trendsicht korrekt?
   - Funktioniert die Bedienfreigabe der bedienbaren Parameter?
   - Werden die Werte bei Bedienung in den AS-Baustein geschrieben?
   - Ist das Bedienprotokoll korrekt?

2.1.2 Bildbausteinerstellung mit dem Faceplate Designer

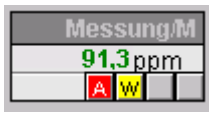
Bestandteil des WinCC-Optionspakets "Advanced Process Control" ist das Faceplate Designer-Tool. Dieses Tool generiert die Vorlagen zur PCS 7-konformen Erstellung von Bildbausteinen.

2.1.2.1 Vorlagen des Faceplate Designer

Für die Erstellung von Bildbausteinen stehen Ihnen unter WinCC folgende Vorlagen zur Verfügung:

- Bausteinsymbole (fertige Symbole für Prozessbilder)
- Vorlagenbilder
- Objekt-Baukasten mit den Objekten für die Bildbaustein-Erstellung
- Globale Skripte

2.1.2.2 Bausteinsymbol-Vorlagen



Die Bausteinsymbole sind im WinCC-Bild "@@PCS7Typicals.pdl" enthalten, z.B.: Ventil, Antrieb, Messwert, Regler usw.

- Die Beispiel-Vorlagen sind beliebig änderbar in Form, Farbe, Gestaltung usw. und schnell anpassbar, auch an projektspezifisch erstellte Bildbausteine.
- Die fertigen Aufruf-Skripte für die Bildbausteine sind schon enthalten und müssen nicht projektiert werden.
- Die Verschaltung erfolgt schnell und einfach mit dem Dynamic-Wizard "Bildbaustein mit Messstelle verbinden".

  Detailierte Beschreibung, siehe Kapitel 2.8, "Bausteinsymbole".

Hinweis

Für Änderungen ist das Bild unter dem Namen "@PCS7Typicals*.pdl“ zu speichern.
Nach diesen Namen sucht die Suchmaschine aus der TH zuerst. Nur wenn diese nicht gefunden werden, werden die Vorlagen aus "@@PCS7Typicals.pdl“ verwendet.

2.1.2.3 Vorlagenbilder

Die Bilder und Bitmaps befinden sich im Verzeichnis "WinCC\options\pdl\FaceplateDesigner_V6" und werden mit dem OS-Projekteditor in das Projekt kopiert. Siehe hierzu die Online-Hilfe "OS-Projekteditor Grunddaten“.

Objekt-Baukasten

Eine Reihe fertig vorbereiteter Objekte (Anwenderobjekte) zur Erstellung eines Bildbausteins, sind im WinCC-Bild "@PCS7Elements.pdl" enthalten, z.B. EA-Felder, Texte, usw.
Das Bild ist im Pfad "Siemens\WinCC\options\pdl\FaceplateDesigner_V6" abgelegt und wird beim Durchlauf des OS-Projekteditors in den Ordner "GraCS" des Projektverzeichnisses kopiert.

Detailierte Beschreibung, siehe Kapitel 2.3, "Basis-Elemente".

Globale Skripte

Die Bildbaustein-Aufrufe bzw. Runtime Funktionen befinden sich als globale Skripte im Verzeichnis "WinCC\aplib\FaceplateDesigner_V6".

Detailierte Beschreibung, siehe Kapitel 2.4, "Skripte".

2.1.2.4 Projektierungsschritte

Hinweise zur Projektierung

- Die mit dem Faceplate Designer erstellten Bildbausteine liegen zunächst im Verzeichnis "GraCS“des aktuell geöffneten Projekts.
  Für das projektübergeordnete Ablegen eigener Bildbausteine ist das Verzeichnis "\Siemens\WINCC\options\pdl\FaceplateDesigner“ vorgesehen.

  Mit dem OS-Projekteditor können Sie Datei-granular auswählen, welche Standard-Bildbausteine aus dem Ordner "\Siemens\WINCC\options\pdl\FaceplateDesigner_V6“ und welche selbst erstellten Bildbausteine aus dem Verzeichnis "\Siemens\WINCC\options\pdl\FaceplateDesigner“ beim Erzeugen der Grunddaten in das Projekt kopiert werden sollen.
- Die Daten aus dem Verzeichis "\Siemens\WINCC\options\pdl\FaceplateDesigner“ müssen Sie auch auf das entsprechende Verzeichnis von WinCC-Clients kopieren.

Die an eigenen Bildbausteinen projektierten Funktionen können Sie bei Bedarf im Editor "Global Script" gegen Einsicht und Änderung schützen.
Informationen hierzu finden Sie in der Dokumentation zum Editor "Global Script".

- Die Dynamik der mit dem Faceplate Designer erstellten Bildbausteine ist vollständig über die Projektierung steuerbar. Die Performance eines Bildbausteins wird somit im besonderen Maße durch die Wahl einer geeigneter Dynamikprojektierung beeinflusst. Besonders wichtig ist in diesem Zusammenhang, auf eine optimale, schlanke Schnittstelle zwischen den AS- und OS-Funktionen zu achten. Dies trifft vor allem auf die Bausteinsymbole zu.

  Hinweis
  Hierbei empfiehlt es sich, das in V6 an den AS-Bausteinen neu eingeführte Statuswort "VSTATUS“ (insbesondere bei den Bausteinsymbolen) zu verwenden, um Variablenanbindungen zu sparen.
  Da die Standard-Bausteinsymbole auch AS-Bausteine aus der V5 visualisieren können müssen, konnte hier das neue Statuswort nicht verwendet werden.

- In der Dynamik der Bildbausteine darf kein C-Skript mit fest kodiertem Instanznamen verwendet werden, da diese typspezifisch sind.

2.1.3 Bedienberechtigungen

Die Bedienberechtigungen können Sie bei den Bildbausteinen instanzspezifisch vergeben.

Die Projektierung bzw. die instanzspezifische Information wird an den Bausteinsymbolen abgelegt.

Am Bausteinsymbol gibt es zwei Eigenschaften "Processcontrolling_backup" und "HigherProcesscontrolling_backup" an denen Sie die Berechtigungsstufe für die Prozessbedienung und für die höherwertige Prozessbedienung einstellen können.

Die Informationen werden per Skript an den Bildbaustein übergeben.

Die Eigenschaften sind defaultmäßig auf die bisher gewohnten Berechtigungsstufen 5 und 6 eingestellt, können aber beliebig abgeändert werden. Hier ist jedoch zu beachten, dass die Stufen 1 bis 10 festen Bedeutungen bei WinCC zugeordnet sind.

Welche Parameter welcher der beiden Berechtigungsstufen unterliegen, wird von den bisherigen Bildbausteinen abgeleitet. Prozessbedienungen wie "Ein", "Aus", "Hand", "Automatik", "Sollwert stellen" sind der Berechtigungsstufe 5 und höhere Parametrierungen, wie "Grenzen", "Regelparameter" usw., der Berechtigungsstufe 6 zugeordnet.

Die bereichsspezifische Bedienberechtigung bleibt nach wie vor erhalten.

Die bereichspezifische Bedienberechtigung wird aufgrund der Zuordnung des Variablennamens (Tagname) geprüft. Hierzu ist es aber notwendig, dass in der Technologischen Hierarchie (TH) am Parameter "OS Bereich" der entsprechende Bereichsname parametriert ist und mit dem tatsächlichen OS-Bereich im Picture Tree übereinstimmt.

2.1.3.1 Bedienberechtigungen für Basis-Elemente projektieren

Für die Projektierung von Bedienberechtigungen sind in den vom Faceplate Designer angelegten Sichten die Objekte "@Level5" und "@Level6" vorhanden.

Werden neue Basis-Elemente in einer Sicht angelegt, so können diese per Direktverbindung mit den Objekten "@Level5" und "@Level6" verbunden werden.

Diese Objekte dürfen nicht gelöscht werden, da diese von Skripten beschrieben werden.

Verbunden werden die Eigenschaften "Hintergrundfarbe" (um bei nicht vorhandener Bedienberechtigung das Feld "grau" zu schalten) und "Bedienfreigabe".

Sollte die Bedienfreigabe von Objekten für Parameter aus dem AS benötigt werden, kann für die Bedienberechtigung auch die Passwort-Stufe per Direktverbindung weiter verschaltet werden. Die Objekte werden dann allerdings bei nicht vorhandener Berechtigung nicht grau geschaltet, sondern es kommt bei der Bedienung eine Meldung "Keine Berechtigung“.

Als geeignetere Variante für das Zusammenführen von WinCC-Bedienberechtigungen und prozessabhängigen Freigaben auf ein Bedienelement gibt es bei den Basis-Elementen das Objekt "Permission“. Die Projektierung dieses Objektes ist dort beschrieben.

Die Werte dieser Eigenschaften werden per Skript "PCS7_UpdatePermission_V6" geschaltet.

Defaultmäßig ist bei "@Level5" die Bedienberechtigungsstufe 5 und bei "@Level6" die Bedienberechtigungsstufe 6 hinterlegt. Die Werte werden vom Bausteinsymbol über die Eigenschaften "Processcontrolling_backup" und "HigherProcesscontrolling_backup" versorgt und können somit instanzspezifisch geändert werden.

Vorgehensweise:

1. Für die gewünschte Berechtigungsstufe das Objekt selektieren (z.B. @Level6) und "Objekteigenschaften" wählen.
2. Im Register "Ereignis → Propertythemen → Farben → Hintergrundfarbe" als Aktion eine Direktverbindung wählen.
3. Unter "Ziel" (rechtes Fenster) und "Objekt im Bild" das neu eingebaute Objekt auswählen, und dort als Eigenschaft die Hintergrundfarbe auswählen.
4. Im Register "Ereignis → Propertythemen → Sonstige → Bedienfreigabe" als Aktion eine Direktverbindung wählen. Bei diesem Ereignis ist bereits ein Skript hinterlegt, welches gleichzeitig die Hintergrundfarbe steuert, wenn die Bedienfreigabe über eine AS-Variable gesteuert wird; dieses Skript muss dann gelöscht werden.
5. Unter "Ziel" (rechtes Fenster) und "Objekt im Bild" das neu eingebaute Objekt auswählen, und dort als Eigenschaft die Bedienfreigabe auswählen.

Werden weitere Objekte in eine View eingebaut, so wird das letzte Objekt, das per Direktverbindung verschaltet wurde, ausgewählt und der oben beschriebene Vorgang wiederholt.

Wichtig!

Die Eigenschaften "Hintergrundfarbe" und "Bedienfreigabe" müssen von der Projektierung her so eingestellt sein, wie sie bei "@Level5" bzw. "@Level6" projektiert wurden (z.B. bei der Hintergrundfarbe "grau“ und bei der Bedienfreigabe "false“) damit die Skripte richtig funktionieren.

Hierzu besteht folgender Hintergrund:

Wird auf eine Eigenschaft per Skript der gleiche Wert geschrieben, den die Eigenschaft von der Projektierung her schon hatte, so wird dieser Wert über eine Direktverbindung nicht an andere Eigenschaften weiter gereicht. Deswegen müssen Sie zwingend darauf achten, dass die Ziel-Eigenschaften, die per Direktverbindung beschrieben werden, die gleichen Projektierungswerte haben wie die Quell-Eigenschaften.

2.1.4 Übersicht ändern

Erstellen Sie einen Bildbaustein ohne Meldungen bzw. ohne Sammelanzeige, so müssen Sie in dem Bild "@PG_XXXXX_Overview.Pdl" einige Elemente und Skripte wieder entfernen, um eine bessere Performance zu erreichen. Folgende Dinge sind zu entfernen:

- Button16, Meldungsquittierung
- Button17, Meldungen Sperren/ Freigeben
- @Level5, Bedienberechtigungen
- MSG_LOCK, Symbol für Meldungsunterdrückung
- Bei dem Bildobjekt selbst, am Ereignis "Bildanwahl" das Skript.
- Wenn kein Batch-Parameter "Occupied" vorhanden, ist auch das Batch-Symbol "OCCUPIED“ zu entfernen.

Beim Faceplate Designer können Sie dies durch die Optionskästchen "keine Sammelanzeige" bzw. "keine Batch-Parameter" berücksichtigen.

2.1.5 Multiinstanz projektieren

Multiinstanz bedeutet, dass in einem CFC Plan mehrere Bausteine projektiert sind, die von einem Bildbaustein in der OS aus bedien- und beobachtbar sind.

Bei der Projektierung wird grundsätzlich das Symbol auf einen kompletten Variablennamen verschaltet.

Für den ausgewählten Variablennamen wird die Bedienberechtigungsprüfung für den gesamten Bildbaustein durchgeführt.

Im Bild "@PG_xxxxx.pdl" und "@PL_xxxxx.pdl" müssen Sie am Objekt "@Faceplate" die Eigenschaft MULTI_INSTANCE auf 'true' setzen, damit der Typ "Multiinstanz" erkannt wird, und vom Variablenname nach der Berechtigungs-prüfung der Bausteinname für die einzelnen Sichten abgeschnitten wird.

Alle Variablenanbindungen im Bildbaustein müssen durch "/bausteinname“ ergänzt werden.

z.B. **/Regler.PV_IN.**

Wichtig: Das muss auch in den Skripten erfolgen!

Im Bild OVERVIEW müssen auch alle Verschaltungen entsprechend ergänzt werden.

Im Bild "@PG_xxxxx.pdl" und "@PL_xxxxx.pdl" müssen Sie am Objekt "@Faceplate" die Eigenschaften der Batch-Parameter für das Bedienprotokoll auf einen Baustein festlegen. Hier ist die Multiinstanztechnik nicht berücksichtigt. Sollen hier Spezialfälle abgedeckt werden, wie z.B. unterschiedliche BatchIDs, StepNr. etc. für die verschiedenen Bausteine, so muss dies mit Skripten bei der Bildanwahl der einzelnen Sichten gesteuert werden.

Bei der Trendsicht ist bei "Modus = 2“ die Multiinstanztechnik nicht berücksichtigt.

Online-Trends können nur zu dem Variablennamen angezeigt werden, der am Bausteinsymbol hinterlegt ist. Sollen Trends zu mehreren Bausteinen angezeigt werden, so ist dies über die anderen Modi zu erreichen.

Wenn Sie mehrere meldefähige Bausteine einbringen, müssen im Bild OVERVIEW folgende weiteren Punkte beachtet bzw. realisieren:

- Es müssen entsprechend der Anzahl meldefähiger Bausteine auch entsprechend viele Sammelanzeigen eingebaut werden.
- Das Skript für die Quittiertaste für Meldungen muss ergänzt werden.
  Skript "Button16/Mouseclick“.
  Bei der Quittierung muss für jede Instanz die Funktion "CSigAPIAcknowledgeTagAndCreateLTM“ ausgeführt werden.

  Beispiel:

```
TCHAR sztag1[_MAX_PATH + 1] = "";
TCHAR sztag2[_MAX_PATH + 1] = "";

GetComputerNameA(szStation, &dwSize);
pszParentPicture = GetParentPicture(lpszPictureName);
lpszCurrentUser  = GetPropChar(pszParentPicture ,"@Faceplate","CurrentUser");
pszTagname = GetPropChar(pszParentPicture ,"OverviewWindow","TagPrefix");

strcpy(sztag1,pszTagname);
strcpy(sztag2,pszTagname);
strcat(sztag1,"/Instanz1");
strcat(sztag2,"/Instanz2");
printf ("sztag2 : %s\r\n", sztag2);

if (!CSigAPIAcknowledgeTagAndCreateLTM(sztag1, lpszCurrentUser , szStation, &Err))
      printf ("Fehler bei CSigAPIAcknowledgePicture : %s\r\n", Err.szErrorText);
if (!CSigAPIAcknowledgeTagAndCreateLTM(sztag2, lpszCurrentUser , szStation, &Err))
      printf ("Fehler bei CSigAPIAcknowledgePicture : %s\r\n", Err.szErrorText);
```

- Das Skript für die Taste “Meldungen Sperren/ Freigeben“ muss ergänzt werden.
  Skript "Button17/Mouseclick“.
  Beim “Meldungen Sperren/ Freigeben” muss für jede Instanz die Funktion “CSigAPILockMessage“ ausgeführt werden.

  Beispiel:

```
DWORD dEventState1 =  GetPropWord(lpszPictureName,"EventState","CollectValue");
DWORD dEventState2 =  GetPropWord(lpszPictureName,"EventState2","CollectValue");
TCHAR sztag1[_MAX_PATH + 1] = "";
TCHAR sztag2[_MAX_PATH + 1] = "";

pszParentPicture = GetParentPicture(lpszPictureName);
pszTagname = GetPropChar(pszParentPicture ,"OverviewWindow","TagPrefix");
lpszCurrentUser  = GetPropChar(pszParentPicture ,"@Faceplate","CurrentUser");
GetComputerNameA(szStation, &dwSize);

strcpy(sztag1,pszTagname);
strcpy(sztag2,pszTagname);
strcat(sztag1,"/R");
strcat(sztag2,"/M");

if ((dEventState & 65536) == 65536)
{
    bRet =  CSigAPILockMessage(FALSE, sztag1, lpszCurrentUser  ,  szStation, &Err);
    SetPictureUp(lpszPictureName,lpszObjectName,"@Lock.bmp");
}
else
{
   bRet =  CSigAPILockMessage(TRUE, sztag1, lpszCurrentUser  ,  szStation, &Err);
   SetPictureUp(lpszPictureName,lpszObjectName,"@Unlock.bmp");
}

if ((dEventState2 & 65536) == 65536)
{
    bRet =  CSigAPILockMessage(FALSE, sztag2, lpszCurrentUser  ,  szStation, &Err);
    SetPictureUp(lpszPictureName,lpszObjectName,"@Lock.bmp");
}
else
{
   bRet =  CSigAPILockMessage(TRUE, sztag2, lpszCurrentUser  ,  szStation, &Err);
   SetPictureUp(lpszPictureName,lpszObjectName,"@Unlock.bmp");
}
```

- Für alle meldefähigen Bausteine muss ein Symbol zur Meldungsunterdrückung (MSG_LOCK) eingebaut werden.
- Für alle Batch-relevanten Bausteine muss ein OCCUPIED-Symbol eingebracht werden.

2.1.6 Zahlenformate projektieren

Die relevanten Bausteinsymbole bekommen drei neue Eigenschaften für Zahlenformatierungen:

Format_InputValue	für Eingangswerte am Bausteinsymbol wie "Setpoint" und "ProcessValue"
Format_OutputValue	für Stellgröße am Bausteinsymbol
Format_xx	für evtl. weitere Werte

Hinweis: Die Begriffe "Format_InputValue“ und "Format_OutputValue“ sind hier prozessorientiert gewählt worden und haben keinen direkten Bezug zu den Ein-/Ausgängen der AS-Bausteinparameter.

Beispiel: Beim FMT_PID ist der Parameter "PV“ zwar am AS-Baustein ein Ausgangsparameter, aber aus Prozesssicht eine Eingangsgröße.

Mit **Format_InputValue** und **Format_OutputValue** werden die Analogwerte des Bausteinsymbols versorgt.

Die Default-Einstellung für **Format_InputValue** zeigt gleitende Vorkommastellen und zwei gleitende Nachkommastellen; mindestens eine Vorkommastelle wird immer gezeigt (0.##)

Die Default-Einstellung für **Format_OutputValue** zeigt gleitende Vorkommastellen und zwei gleitende Nachkommastellen; mindestens eine Vorkommastelle wird immer gezeigt (0.##)

Der Projekteur kann diese Default-Einstellung bei Bedarf instanzspezifisch ändern.

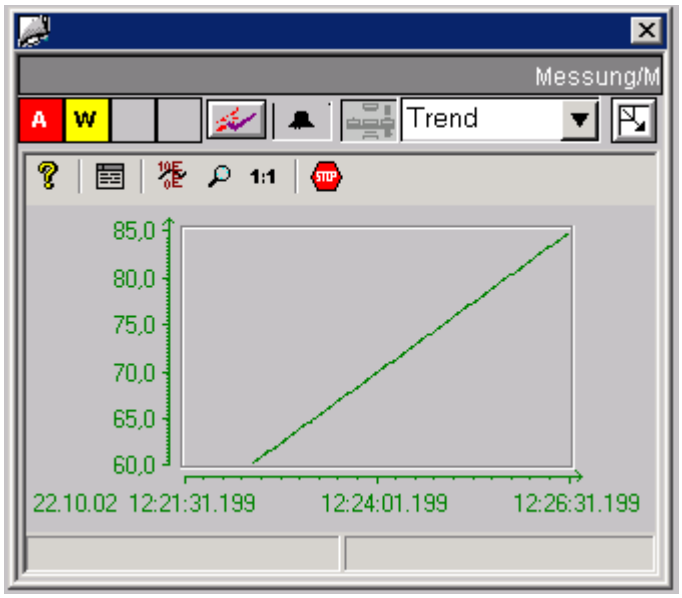
Alle drei Format-Eigenschaften des Bausteinsymbols werden per Skript in den Bildbaustein übertragen und können dort per Direktverbindung den Analogwerten zugewiesen werden.

Details:
Die Format-Eigenschaften des Bausteinsymbols werden per Skript "PCS7_OpenGroupDisplay_V6" in das übergeordnete Prototypbild "@PG_xxxxx.pdl" in das Objekt "Format“ übertragen und abgelegt.

Existiert in einer Sicht ein Objekt "Format“ mit den Eigenschaften "Format_InputValue, Format_OutputValue, Format_xx", so wird bei Bildanwahl per Skript "PCS7_Format_V6" die Formatierungen in dieses Objekt übertragen.

Von diesem Objekt aus können Sie nun per Direktverbindung die drei Formate beliebig den Analogwerten in dieser Sicht zuweisen.

2.1.7 Trendsicht projektieren



Für die Realisierung von Trendbildern gibt es bei den Bausteinsymbolen für Regler, Messung, usw. neue Eigenschaften:

- **ReturnPath:** Hier werden die Kurven-Daten für die entsprechende Messstelle übertragen.
- **StandardTrend:** Damit wird festgelegt, welche Trendfunktionalität in der Trendsicht dargestellt wird.

Standard-Trend:	**Modus**
0	Trendfunktionalität wie in V5; für jede Instanz muss ein eigenes Trendbild projektiert werden. Die für alle PCS7-Bildbausteine vorhandene Trendsicht "@PCS7_TREND.pdl" enthält das Bildfenster "TrendPicture". Damit an dieser Stelle für eine Messstelle ein Trendbild eingeblendet wird, müssen Sie folgendermaßen vorgehen: 1. Öffnen Sie den WinCC-Editor "Tag Logging". 2. Legen Sie mit Hilfe des Archiv Wizards ein Variablen-Archiv für die Trendsicht an. 3. Legen Sie die Variablen für dieses Archiv an. Eine ausführliche Anleitung zu den o.a. Schritten finden Sie in der WinCC-Online-Hilfe unter dem Index "Tag Logging". 4. Öffnen Sie im Graphics Designer das Bild "@CONL1_Standard.pdl" 5. Konfigurieren Sie das TlgOnlineTrend-Objekt "Control1" entsprechend Ihren Anforderungen; für weitere Informationen nutzen Sie die Taste "Hilfe". 6. Speichern Sie das Bild unter dem Namen "@CONL1_<tagname>.pdl". Der Variablenname <tagname> muss identisch sein mit dem Messstellennamen des AS-Bausteins. Das Zeichen "/" im Namen müssen Sie durch das Zeichen "_" ersetzen, da Windows "/" in Dateinamen nicht zulässt. Beispiel für den Fensternamen einer Messstelle mit dem Variablennamen "Messung/M“ @CONL1_Messung_M.pdl
2	Es wird ein Bild "@CONL1_Standard.pdl“ aufgerufen, in dem ein oder mehrere Online-Variablen eingetragen werden. Es wird kein Archiv benötigt. x Achse = 5 min Dies ist die Default-Einstellung, da hier von der Projektierung keine Aufwendungen notwendig sind und keine Namenskonventionen eingehalten werden müssen, um die Trends anzuzeigen. Über die Eigenschaft **ReturnPath** im Bausteinsymbol wird die Anzahl der Kurven, der Strukturelement-Name sowie die Farbe übergeben.

Standard-Trend:	**Modus**
>2	Es wird ein Bild "@CONL1_Standard.pdl“ aufgerufen, in dem ein oder mehrere Archivvariablennamen eingetragen werden. Damit diese Konfiguration ohne weiteren Projektierungsaufwand funktioniert, müssen folgende Konventionen erfüllt sein: • Das Archiv muss "Prozesswertarchiv“ heißen (Default-Name beim Erstellen des ersten Archives). • Das Archiv muss auf dem gleichen Server liegen wie die Variablen selbst. • Der Archivvariablenname muss so heißen, wie er standardmäßig beim Erstellen der Archivvariable angelegt wird. • Ist in der Eigenschaft "tagname" des Bausteinsymbols im eingetragenen Variablenname das Serverprefix enthalten, muss im Archivariablennamen das Serverprefix ohne "::" angegeben werden. • Ist in der Eigenschaft "tagname" der eingetragene Variablenname ohne Serverprefix, muss der Archivvariablenname genau so heißen wie der Variablenname. **Hinweis:** Der Archivvariablenname ist in älteren Versionen der "tagname" ohne die Trennzeichen "/“. In Versionen ab V6 ist dies der "tagname", und so muss die Archivvariable auch zwingend heißen, damit der Trend im Bildbaustein funktioniert. Über die Eigenschaft **ReturnPath** im Bausteinsymbol, wird die Anzahl der Kurven, der Variablennname sowie Farbe übergeben. Die Dauer der X-Achse wird für diesen Modus aus dem Parameter "StandardTrend“ gelesen (in min.) **Syntax für ReturnPath:** .PV_IN Strukturelement-Name mit Punkt für erste Kurve : Trennzeichen zwischen Strukturelement-Name und Farbe CO_GREEN Farbe für die erste Kurve , Komma für weitere Kurve Beispiel: .PV_IN:CO_DKGREEN,.SP:CO_BLUE,.LMNR_IN:CO_DKRED (Default für CTRL_PID)

Standard-Trend:	Modus
	Das Beispiel zeigt drei Kurven beim Regler (Istwert, Sollwert, Stellgröße Rückmeldung). Können die oben erwähnten 4 Konventionen nicht eingehalten werden, können Sie die Eigenschaft "ReturnPath“ mit weiteren Parametern ergänzen: ***tagname:** individueller Archivvariablenname ***archivname:** individueller Archivname (Eingabe ohne „\“) ***archivserver:** Kurve liegt auf einem Archivserver, (Eingabe des Serverprefix vom Archivserver ohne „::“) **Beispiel:** .PV_IN:CO_DKGREEN,.SP:CO_BLUE,.LMNR_IN:CO_DKRED*tagname: MySpecialTag*archivname:MySpecialArchivname*archivserver: MySpecialArchivServer **Hinweis:** Es kann hier nur ein “tagname“, "archivname“, "archivserver“ am Ende des Strings angegeben werden. Für den Fall von Multiinstanzen müssen die Archivvariablennamen entsprechend angepasst werden. Die Archivvariablen dürfen sich nur in der Membervariablen unterscheiden. **Beispiel für Multiinstanz:** Es gibt in einem CFC-Plan "MULTI“ zwei MEAS_MON-Bausteine "M1“ und "M2“. Als Archivvariablennamen für den Messwert "U“ werden vom Default her die Namen "MULTI/M1.U“ und "MULTI/M2.U“ erzeugt. Die Archivvariablen sind umzubenennen in "MULTI.M1_U“ und "MULTI.M2_U“ Textstring in der Eigenschaft **ReturnPath** im Bausteinsymbol**:** .M1_U:CO_DKGREEN,.M2_U:CO_BLUE*tagname:MULTI*archivname:SystemArchive ***asia:** add Serverprefix in Archivvariable Wird der Parameter *asia: nicht verwendet, so wird bei vorhandenem Serverprefix in "tagname" dieser auch mit in die Ableitung des Archivvariablennamens eingebracht. Ist kein Serverprefix in "tagname" vorhanden, so wird auch bei der Ableitung des Archivvariablennamens kein Serverprefix berücksichtigt. ***asia:MyServerPrefix** Der Archivvariablenname enthält das serverprefix ohne "::“, die Eigenschaft "tagname“ des Bausteinsymbols aber nicht. In diesem Fall kann hier das Serverprefix ergänzt werden (Eingabe ohne "::“). ***asia:** Der Archivvariablenname enthält kein Serverprefix. In diesem Fall kann hier das Serverprefix ausgeblendet werden. Diese Variante ist auch zu verwenden, wenn in "tagname" auf dem Server kein Serverprefix eingetragen ist, da bei einem Aufruf des Bildes vom Client das Serverprefix immer vorhanden ist. Beispiel für *asia: Serverprefix einblenden: .U:CO_DKGREEN*asia:OS(1)_PC2 Beispiel für *asia: Serverprefix ausblenden .U:CO_DKGREEN*asia:

In der folgenden Tabelle werden die notwendigen zusätzlichen Parametrierungen dargestellt, in Abhängigkeit

- vom Aufbau des Archivvariablennamens und Variablennamens im Bausteinsymbol
- davon, von wo der Bildbaustein aufgerufen wird
- von der Archivvariable auf dem eigenen Archivserver

	Server	**Client mit Server-Bild**	**Client mit Client-Bild**	**Client mit Server-Bild und Archivvariable auf separatem Archivserver**	**Client mit Client-Bild und Archivvariable auf separatem Archivserver**
Variablenname **ohne** Serverprefix Archivvariablenname **ohne** Serverprefix	*asia:	*asia:	Nicht möglich, Im "tagname" ist immer das Serverprefix	*asia: *archivname: MyArchivserver Prefix	Nicht möglich, Im "tagname" ist immer das Serverprefix
Variablenname **mit** Serverprefix Archivvariablenname **ohne** Serverprefix	*asia:	*asia:	*asia:	*asia: *archivname: MyArchivserver Prefix	*asia: *archivname: MyArchivserver Prefix
Variablenname **ohne** Serverprefix Archivvariablenname **mit** Serverprefix	*asia: MyServer Prefix	*asia: MyServer Prefix	Nicht möglich, Im "tagname" ist immer das Serverprefix	*archivname: MyArchivserver Prefix	Nicht möglich, Im "tagname" ist immer das Serverprefix
Variablenname **mit** Serverprefix Archivvariablenname **mit** Serverprefix	Keine Einstellung nötig bei empfohlener Variablen Projektierung	Keine Einstellung nötig bei empfohlener Variablen Projektierung	Keine Einstellung nötig bei empfohlener Variablen Projektierung	*archivname: MyArchivserver Prefix	*archivname: MyArchivserver Prefix

Bei Aufruf des Prototypbildes "@PG_xxx“ wird vom Bausteinsymbol der Wert der Eigenschaft "StandardTrend“ und "ReturnPath“ an den Bildbaustein übertragen.

Im Prototypbild ist ein Objekt "Trendfunktionen“ mit der Eigenschaft "StandardTrend“ und "ReturnPath“ hinterlegt, in dem die Informationen gespeichert werden und bei Bildanwahl des "@PCS7_Trend.pdl" per Skript "PCS7_Trend.fct" ausgewertet werden.

Die gleiche Funktionalität wird in die Loop-Darstellung übertragen.

2.1.8 Für einen AS Bausteintyp verschiedene Bausteinsymbole und Bildbausteintypen projektieren

Um eine Variation eines Bausteinsymbols zu einem AS-Baustein anzulegen, müssen Sie lediglich die Eigenschaft "type“ am Bausteinsymbol ändern.

Möchten Sie auch eine Variation eines Bildbausteines zu einem AS-Baustein anlegen, so müssen Sie einen vom AS-Bausteintyp abweichenden Namen verwenden. Die Variation für "MEAS_MON“ ist dann z.B. der Bildbausteinname "MEAS_NEU“.

Folgende Schritte sind am Bausteinsymbol notwendig:

1. Die Eigenschaft "Servername“ benennen Sie um in "PCS7 MEAS_NEU Control“.
2. Legen Sie die neue Eigenschaft "StructureType“ an dem Bausteinsymbol an, welches den AS-Bausteintyp beinhaltet ("MEAS_MON“).
   Dies ist notwendig, damit bei dem Wizard "Bildbaustein mit Messstelle verbinden“ die Variablenselektion noch funktioniert.

2.1.9 WebClient (Unterschiede zu WinCC)

Nachfolgend finden Sie die zu berücksichtigenden Unterschiede zwischen WinCC und WebClient,

2.1.9.1 Bildnamen

In WinCC-Skripten werden Bildnamen absolut adressiert, wie z.B.

@screen.@win12:@1001.@top09:@pg_elap_cnt.OverviewWindow:@PG_ELAP_CNT_OverView.pdl

oder relativ, wie z.B.

@PG_ELAP_CNT_OverView.pdl

Beim WebClient funktioniert nur die relative Adressierung von dem Kontext aus, wo das Skript läuft.

Weiterhin muss bei der WebClient-Skripverarbeitung der Bildfenstername und nicht der Bildname als Adresse angegeben werden.

Beispiel:
SetPropChar(lpszPictureName,lpszObjectName,lpszPropertyName,szValue);

WinCC: der lpszPictureName ist ein Zeiger auf den Bildnamen

WebClient: der lpszPictureName ist ein Zeiger auf den Bildfensternamen

Ausnahme ist das eigene Bild und das Vorgängerbild (ParentPicture), hier kann auch auf dem WebClient der Bildname als Zeiger verwendet werden. Es wird jedoch empfohlen, immer den Bildfensternamen als Zeiger zu verwenden, da dieser eindeutig ist, während der Bildname mehrdeutig sein kann.

Beispiel für die relative Bildadressierung:

In einem Bild @PG_xxx.pdl gibt es ein Bildfenster "View“ in dem z.B die Standardsicht eines Bildbausteins angezeigt wird.
Soll nun aus dem Bild (Standardsicht), das in dem Bildfenster "View“ geladen ist, ein anderes Bildfenster im Bild @PG_xxx.pdl adressiert werden, z.B. "OperationWindow“ zum Öffnen einer Bedienbox, so ist die relative Bildfensteradresse folgendermaßen zu ermitteln:

sprintf(szPictureName,"../OperationWindow");

2.1.9.2 Groß- / Kleinschreibung bei Dateinamen

Dateinamen (z.B. bei Zeigern auf Bildnamen) werden bei WinCC generell in Großbuchstaben geliefert z.B. @PG_MEAS_MON.PDL.

Auf dem WebClient werden die Dateinamen genau so geliefert, wie sie tatsächlich erstellt wurden, z.B. @Pg_Meas_Mon.pdl.

Bei String-Vergleichen müssen hier möglicherweise Anpassungen vorgenommen werden.

Beispiel:
(strcmp(GetPictureUp(lpszPictureName,lpszObjectName), "@KEEPVISIBLEON.EMF") == 0)

Das funktioniert in WinCC immer, aber auf dem Web nur, wenn der Dateiname ausschließlich mit Großbuchstaben angelegt wurde.

Erforderliche Anpassung:
(strcmp(_**strlwr**(GetPictureUp(lpszPictureName,lpszObjectName)), "@**keepvisibleon.emf**") == 0)

2.1.9.3 Laden von Bildern in Bildfenstern

In WinCC können Sie nach dem Laden eines Bildes in einem Bildfenster (mit z.B. SetVisible), die darin enthaltenen Objekte direkt per Skriptverarbeitung adressieren und die Eigenschaften verändern.

Auf dem WebClient erfolgt das Laden von Bildern asynchron. Deswegen muss in Skripten, nach dem Laden eines Bildes, die weitere Skriptverarbeitung verzögert werden, wenn im gleichen Skript Objekte des Bildes adressiert werden.
Hierfür wird für den WebClient die Funktion **WaitForDocumentReady("Bildfenstername");** bereitgestellt.
Die Funktion verzögert den Skriptablauf, bis der als Parameter angegebene Bildfenstername ein geladenes Bild zurückmeldet.

2.1.9.4 Bildabwahl innerhalb von Skripten

In WinCC kann das "Unsichtbar setzen“ eines Bildes aus einem Skript, das im Kontext dieses Bildes läuft, an einer beliebigen Stelle des Skriptes ausgeführt werden, da die Skriptverarbeitungs-Task unabhängig vom geschlossenen Bild das Skript weiter verarbeitet.

Auf dem WebClient muss das "Unsichtbar setzen“ zwingend als letzte Aktion im Skript ausgeführt werden.
Skriptbefehle, die nach dem "Unsichtbar setzen“ stehen, werden nicht mehr ausgeführt, da das Skript an dieser Stelle beendet wird.

2.1.9.5 Unterscheidung der Runtime-Umgebung WinCC <-> Web

In den Skripten von Grafikbildern oder Projektfunktionen ist es notwendig, die Runtime-Umgebung unterscheiden zu können, d.h. ob das Skript in WinCC oder auf dem WebClient läuft.
Hierzu gibt es die Compiler-Anweisung #ifdef RUN_ON_WEBNAVIGATOR bzw. die Negation #ifndef RUN_ON_WEBNAVIGATOR

Damit können die Unterschiede von WinCC zum WebClient berücksichtigt werden, wie z.B. Skriptverzögerung mit WaitForDocumentReady, unterschiedliche Bildadressierung, unterschiedliche Funktionsnamen bei Leittechnikfunktionen sowie Funktionen die im Web nicht unterstützt werden.

Beispiel aus dem Skript PCS7_ChangeView:

```
#ifdef RUN_ON_WEBNAVIGATOR
        SetPropChar("../", "View", "PictureName", szViewName);
        WaitForDocumentReady("../View");
#else
        SetPropChar(lpszParent, "View", "PictureName", szViewName);
#endif
```

Hinweis:

Die Syntax des Codeteils für Web wird nicht beim Compilieren des WinCC-Skriptes überprüft, sondern nur beim Veröffentlichen der Bilder.

2.1.9.6 Funktionsnamen in WinCC / Web

Funktionen, die eine Aktion auslösen, wurden mit anderen Funktionsnamen für das Web nachgebildet, damit die Aktion im Webnavigator-Client ausgelöst wird und nicht in WinCC-Runtime.

Beispiel:

SSMRTChangeWorkdfield ändert das Arbeitsfenster in WinCC-Runtime und

SSMChangeWorkdfield ändert das Arbeitsfenster beim WebClient.

Eine Liste der unterstüzten Funktionen und deren Namen finden Sie bei den entsprechenden Produktbeschreibungen. Zum Produkt "Faceplate Designer" werden folgende Funktionen unterstützt:

void PCS7_ChangeView(char* lpszPictureName, char* lpszObjectName);

BOOL PCS7_CheckPermission(char* lpszTagname, DWORD dwLevel);

void PCS7_UpdateGroupPermission(char* lpszPictureName);

void PCS7_UpdateGroupTagname(char* lpszPictureName, char* lpszObjectName, char* value);

void PCS7_UpdateLoopTagname(char* lpszPictureName, char* lpszObjectName, char* value);

void PCS7_UpdatePermission_V6(char* lpszParentPictureName, int CallFrom, char* lpszPictureName);

void PCS7_OpenGroupDisplay_V6(char *lpszPictureName, char *lpszObjectName);

void PCS7_OpenLoopDisplay_V6(char* lpszPictureName);

void PCS7_OpenInputBoxBin_V6(char *lpszPictureName,char *lpszObjectName, int CallFrom);

void PCS7_1vnStati_Variable_Changed_V6(char* lpszPictureName, char* lpszObjectName, char* lpszPropertyName, double value);

void PCS7_OperationLog_V6(char* lpszPictureName, double dOldValue, double dNewValue, char* lpszOperationText, char* lpszUnit);

void PCS7_Trend_V6(char* lpszPictureName, char* lpszObjectName, char* lpszPropertyName, char * ArchivVar, char* OnlineVar, DWORD TrendColor, int CallFrom);

void PCS7_UpdateGroupTagname_V6(char* lpszPictureName, char* lpszObjectName, char* value);

void PCS7_UpdateLoopTagname_V6(char* lpszPictureName, char* lpszObjectName, char* lpszTagName);

void PCS7_UpdateBarLimits_V6(char* lpszPictureName, char* lpszObjectName, int CallFrom);

void PCS7_UpdateBar_V6(char* lpszPictureName, char* lpszObjectName, int CallFrom);

void PCS7_OpenInputBoxAnalog_V6(char *lpszPictureName,char *lpszObjectName,int CallFrom);

void PCS7_OpenGroupDisplay_I_V6(char *lpszPictureName, char *lpszObjectName, char*lpszInterlokName);

void PCS7_OpenComboBox_V6(char *lpszPictureName,char *lpszObjectName,int CallFrom);

void PCS7_OpenCheckBox_V6(char *lpszPictureName,char *lpszObjectName, int CallFrom);

void PCS7_Open3ComboBox_V6(char *lpszPictureName,char *lpszObjectName,int CallFrom);

void PCS7_Format_V6(char* lpszPictureName, char* lpszObjectName, int CallFrom,char* lpszParent);

void PCS7_Combo_OK_V6(char* lpszPictureName, char* lpszObjectName, char* lpszPropertyName);

void PCS7_Check_OK_V6(char* lpszPictureName, char* lpszObjectName, char* lpszPropertyName);

void PCS7_Binary_OK_V6(char* lpszPictureName, char* lpszObjectName, char* lpszPropertyName);

void PCS7_AnalogPercent_V6(char* lpszPictureName, char* lpszObjectName, char* lpszPropertyName,double Prozent);

void PCS7_Analog_OK_V6(char* lpszPictureName, char* lpszObjectName, char* lpszPropertyName);

void PCS7_3Combo_OK_V6(char* lpszPictureName, char* lpszObjectName, char* lpszPropertyName);

void PCS7_2Stati_Variable_Changed_V6(char* lpszPictureName, char* lpszObjectName, char* lpszPropertyName, double value);

2.1.9.7 Global Script

Für den WebClient wird "Global Script" nicht unterstützt, da es hier keine eigene Applikation "Script.exe“ gibt.

Globale Funktionen sind als Projektfunktionen abzulegen, diese werden beim Veröffentlichen mit übersetzt.

2.1.9.8 VBS-Skript

Beim Erstellen von neuen Funktionen empfiehlt es sich, diese in VBS-Skripten zu erstellen, da bei VBS-Skripten keine Unterschiede zwischen WinCC und WebClient zu berücksichtigen sind.

2.1.9.9 Hinweise

Im Unterschied zu WinCC gibt es auf dem WebClient keine globalen bildübergreifenden Variablen in C-Skripten.

Auf zyklische, synchrone Funktionen sollten Sie verzichten, da die Laufzeit am Web-Client größer ist als bei WinCC.

2.1.10 Sprachumschaltung

Die Vorlagen des Faceplate Designers sind dreisprachig angelegt (Deutsch, Englisch, Französisch), d.h. bei einer Sprachumschaltung in WinCC werden die Texte in der eingestellten Sprache angezeigt. Wenn Sie weitere Textelemente in ein Bild einbauen und eine Sprachumschaltung wünschen, müssen Sie diese Texte in allen gewünschten Sprachen im Graphics Designer eingeben. Die Sprachumschaltung erfolgt im Graphics Designer über den Menübefehl **Ansicht > Sprache... > Sprache wählen**.

2.1.11 Texte für Analogwert- und Binärwert-Bedienung aus ES

Für die Beschriftung von Binäranzeigen, Analogwertanzeigen, Kombinationsfeldern, Optionskästchen und Bedienprotokollen werden die Texte aus den Bausteininstanzen von den Parameter-Attributen gelesen.

Beachten Sie dabei, dass die Default-Texte im ES (s7_shortcut, s7_unit, S7_string_0, S7_string_1) alle in Englisch projektiert sind.

In den Bildbausteinen wurden bisher die ES-Texte nur im Bedienprotokoll verwendet. In den Anzeigen des Bildbausteins wurden bislang die Texte in WinCC gebildet und waren dreispachig.

Für die Migration müssen diese Texte im ES (ggf. unter Verwendung des IEA) in die entsprechend gewünschte Sprache umgesetzt werden.

Für den Fall, dass eine Mehrsprachigkeit der Texte benötigt wird, können Sie die Texte mit dem Textlibrary-Editor in WinCC in andere Sprachen übersetzen. Sie müssen dann aber im SIMATIC Manager als Standardsprache für Anzeigegeräte genau die Sprache einstellen, in der die Bedien- und Anzeigetexte in STEP 7 und im ES projektiert sind (im Normalfall "Englisch"). Nur dann werden die übersetzten Texte beim nächsten Übersetzen der OS nicht wieder überschrieben.

Wichtiger Hinweis:

Aus oben genanntem Grund (Default-Texte existieren im ES in Englisch), ist es wichtig, dass Sie bei einem Projekt, das nicht in Englisch projektiert wird, eine Projektbibliothek für die AS-Bausteine anlegen und dort die englischen Default-Texte auf die gewünschte Sprache umstellen.

Beachten Sie hierbei, dass die Texte nicht länger als die Default-Texte werden. Sind längere Texte nicht zu vermeiden, muss am Bildbaustein geprüft werden, ob der Text noch korrekt angezeigt wird.

Hinweise zu Erstellung von Projektbibliotheken finden Sie im ES-Projektierungs-handbuch.

Das Attribut "s7_unit“ muss für die Übersetzung nicht in Betracht gezogen werden, da hier als Vorbesetzung entweder Leerzeichen oder international gebräuchliche Kurzbezeichnungen verwendet wurden.

Liste der Parameter-Attribute, die angepasst werden müssen:

Baustein	Parameter	S7_shortcut	S7_string_0	S7_string_1
CTRL_PID	AUT_ON_OP		Mode= Manual	Mode= Auto
	DEADB_W	Deadband		
	ER_HYS	ER hysteresis		
	ERH_ALM	ER: HH alarm		
	ERL_ALM	ER: LL alarm		
	GAIN	GAIN		
	HYS	Hysteresis		
	LMNR_IN	OUT		
	M_SUP_AH		Suppress HH=No	Suppress HH=Yes
	M_SUP_AL		Suppress LL=No	Suppress LL=Yes
	M_SUP_ER		Enable ER Alarm	Suppr. ER Alarm
	M_SUP_WH		Suppress H=No	Suppress H=Yes
	M_SUP_WL		Suppress L=No	Suppress L=Yes
	MAN_HLM	MAN high limit		
	MAN_LLM	MAN low limit		
	MAN_OP	MAN		
	MO_PVHR	Bar HL		
	MO_PVLR	Bar LL		
	OOS		In Service	Out of Service
	OPTI_EN		Disable Optimiz.	Enable Optimiz.
	PV_IN	PV		
	PVH_ALM	PV: HH alarm		
	PVH_WRN	PV: H alarm		
	PVL_ALM	PV: LL alarm		
	PVL_WRN	PV: L alarm		
	SP_HLM	SP high limit		
	SP_LLM	SP low limit		
	SP_OP	SP		
	SP_TRK_ON		No Tracking	Track SP:= PV
	SPBUMPON		SP may bump	SP bumpless
	SPDRLM	Neg. ramp		
	SPEXTSEL_OP		SP= Internal	SP= External
	SPRAMPOF		SP Ramp On	SP Ramp Off
	SPURLM	Pos. ramp		
	TM_LAG	Lag time		
	TN	TI		
	TV	TD		

Baustein	Parameter	S7_shortcut	S7_string_0	S7_string_1
CTRL_S	AUT_ON_OP		Mode= Manual	Mode= Auto
	BREAK_TM	Break time		
	DEADB_W	Deadband		
	ER_HYS	ER hysteresis		
	ERH_ALM	ER: HH alarm		
	ERL_ALM	ER: LL alarm		
	GAIN	GAIN		
	HYS	Hysteresis		
	LMNDN_OP		Stop	Close
	LMNR_IN	OUT		
	LMNUP_OP		Stop	Open
	M_SUP_AH		Suppress HH=No	Suppress HH=Yes
	M_SUP_AL		Suppress LL=No	Suppress LL=Yes
	M_SUP_ER		Enable ER Alarm	Suppr. ER Alarm
	M_SUP_WH		Suppress H=No	Suppress H=Yes
	M_SUP_WL		Suppress L=No	Suppress L=Yes
	MAN_HLM	MAN high limit		
	MAN_LLM	MAN low limit		
	MAN_OP	MAN		
	MO_PVHR	Bar HL		
	MO_PVLR	Bar LL		
	MTR_TM	Motor time		
	OOS		In Service	Out of Service
	OPTI_EN		Disable Optimiz.	Enable Optimiz.
	PULSE_TM	Pulse time		
	PV_IN	PV		
	PVH_ALM	PV: HH alarm		
	PVH_WRN	PV: H alarm		
	PVL_ALM	PV: LL alarm		
	PVL_WRN	PV: L alarm		
	RESET		Do Nothing	Reset QMSS_ST
	SP_HLM	SP high limit		
	SP_LLM	SP low limit		
	SP_OP	SP		
	SP_TRK_ON		No Tracking	Track SP:= PV
	SPBUMPON		SP may bump	SP bumpless
	SPDRLM	Neg. ramp		
	SPEXTSEL_OP		SP= Internal	SP= External
	SPRAMPOF		Ascent limit=On	Ascent limit=Off
	SPURLM	Pos. ramp		
	TM_LAG	Lag time		
	TN	TI		
	TV	TD		
DIG_MON	I		Off	On
	SUPPTIME	Suppress time		
	OOS		In Service	Out of Service

Baustein	Parameter	S7_shortcut	S7_string_0	S7_string_1
DOSE	ACK_TOL_OP		0	Acknowl.
	CANCEL_OP		0	Cancel
	COMP_CHG		Comp. change=Off	Comp. change=On
	DRIB_COR		Dribb. corr.=Off	Dribb. corr.=On
	DRIBB	Dribbling init.		
	DRIBBMAX	Max. dribbling		
	M_SUP_1		Suppr normal =No	Suppr normal=Yes
	M_SUP_2		Suppr over =No	Suppr over =Yes
	M_SUP_3		Suppr under =No	Suppr under =Yes
	MO_PVHR	Bar UL		
	MO_PVLR	Bar LL		
	OOS		In Service	Out of Service
	PAUSE_OP		Process=Continue	Process=Pause
	PDOS_TME	Postdose time		
	POSTDOSE		0	Postdose
	PV_IN	PV		
	RELAXTME	Relax time		
	REVERSE		Reverse=No	Reverse=Yes
	SP_HLM	SP high limit		
	SP_LLM	SP low limit		
	SP_OP	Setpoint		
	SPBUMPON		SP may bump	SP bumpless
	SPEXTSEL_OP		SP= Internal	SP= External
	START_OP		0	Dose=Start
	TOL_N	Lower tol. band		
	TOL_P	Upper tol. band		
ELAP_CNT	HOURS	Hours		
	HOURS_AH	HH alarm		
	HOURS_OP	Preset value		
	HOURS_WH	H alarm		
	M_SUP_AH		Suppress HH=No	Suppress HH=Yes
	M_SUP_WH		Suppress H=No	Suppress H=Yes
	MO_HOUHR	Bar UL		
	MO_HOULR	Bar LL		
	OOS		In Service	Out of Service
	TRACK_OP		0	Preset

Baustein	Parameter	S7_shortcut	S7_string_0	S7_string_1
FMCS_PID	AUT_ON_OP		Mode= Manual	Mode= Auto
	BREAK_TM	Break time		
	DEADB_W	Deadband		
	GAIN	Gain		
	H_ALM	HH alarm		
	H_WRN	H alarm		
	HYS	Hysteresis		
	L_ALM	LL alarm		
	L_WRN	L alarm		
	LMN	OUT		
	LMN_HLM	LMN high limit		
	LMN_LLM	LMN low limit		
	LMN_OP	MAN		
	LMN_SAFE	LMN safety		
	LMNDN_OP		Stop	Close
	LMNUP_OP		Stop	Open
	M_SUP_AH		Suppress HH=No	Suppress HH=Yes
	M_SUP_AL		Suppress LL=No	Suppress LL=Yes
	M_SUP_WH		Suppress H=No	Suppress H=Yes
	M_SUP_WL		Suppress L=No	Suppress L=Yes
	MO_PVHR	Bar UL		
	MO_PVLR	Bar LL		
	MTR_TM	MTR time		
	OOS		In Service	Out of Service
	OP_SEL		OP operation=Off	OP operation=On
	OPTI_EN		Optim.=disable	Optim.=enable
	PULSE_TM	Pulse time		
	PV	PV		
	SDB_SEL		SDBParameter=On	SDBParameter=Off
	SP_HLM	SP high limit		
	SP_LLM	SP low limit		
	SP_OP	SP		
	SP_TRK_ON		SP track=Off	SP track=On
	SPBUMPON		SP may bump	SP bumpless
	SPEXTSEL_OP		SP= Internal	SP= External
	TD	TD		
	TI	TI		
	TM_LAG	Time lag		

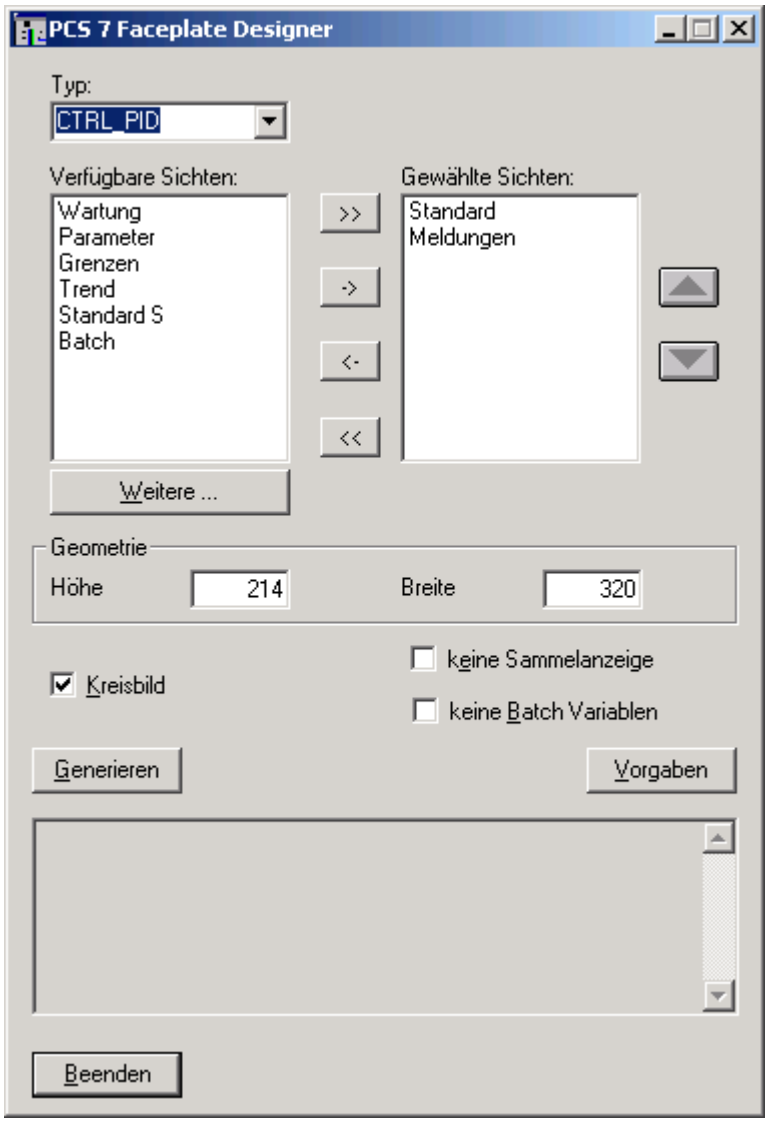
Baustein	Parameter	S7_shortcut	S7_string_0	S7_string_1
FMT_PID	AUT_ON_OP		Mode= Manual	Mode= Auto
	BREAK_TM	Break time		
	D_F	deriv. fac.		
	DEADB_W	Deadband		
	GAIN	Gain		
	H_ALM	HH alarm		
	H_WRN	H alarm		
	HYS	Hysteresis		
	L_ALM	LL alarm		
	L_WRN	L alarm		
	LMN	OUT		
	LMN_HLM	LMN high limit		
	LMN_LLM	LMN low limit		
	LMN_OP	MAN		
	LMN_SAFE	LMN safety		
	LMNDN_OP		Stop	Close
	LMNUP_OP		Stop	Open
	M_SUP_AH		Suppress HH=No	Suppress HH=Yes
	M_SUP_AL		Suppress LL=No	Suppress LL=Yes
	M_SUP_WH		Suppress H=No	Suppress H=Yes
	M_SUP_WL		Suppress L=No	Suppress L=Yes
	MO_PVHR	Bar UL		
	MO_PVLR	Bar LL		
	MTR_TM	MTR time		
	OOS		In Service	Out of Service
	PFAC_SP	Gain		
	PULSE_TM	Pulse time		
	PV	PV		
	SDB_SEL		SDBParameter=On	SDBParameter=Off
	SP_HLM	SP high limit		
	SP_LLM	SP low limit		
	SP_OP	SP		
	SP_TRK_ON		SP track=Off	SP track=On
	SPBUMPON		SP may bump	SP bumpless
	SPEXTSEL_OP		SP= Internal	SP= External
	TD	TD		
	TI	TI		
INTERLOK	I1_1		0	in_1
	I1_2		0	in_2
	I1_3		0	in_3
	I1_4		0	in_4
	I1_5		0	in_5
	I2_1		0	in_6
	I2_2		0	in_7
	I2_3		0	in_8
	I2_4		0	in_9
	I2_5		0	in_10
	OVERWRITE		Overwrite=Off	Overwrite=On

Baustein	Parameter	S7_shortcut	S7_string_0	S7_string_1
MEAS_MON	HYS	Hysteresis		
	M_SUP_AH		Suppress HH=No	Suppress HH=Yes
	M_SUP_AL		Suppress LL=No	Suppress LL=Yes
	M_SUP_WH		Suppress H=No	Suppress H=Yes
	M_SUP_WL		Suppress L=No	Suppress L=Yes
	MO_PVHR	Bar UL		
	MO_PVLR	Bar LL		
	OOS		In Service	Out of Service
	U	PV		
	U_AH	HH alarm		
	U_AL	LL alarm		
	U_WH	H alarm		
	U_WL	L alarm		
MOTOR	AUT_ON_OP		Mode=Manual	Mode=Auto
	MAN_ON		Motor=Stop	Motor=Start
	MONITOR		Monitoring=Off	Monitoring=On
	OOS		In Service	Out of Service
	RESET		0	Error=Reset
	TIME_MON	Mon. time		
MOT_REV	AUT_ON_OP		Mode=Manual	Mode=Auto
	FORW_ON		0	Motor=Forward
	MONITOR		Monitoring=Off	Monitoring=On
	MOT_OFF		0	Motor=Off
	OOS		In Service	Out of Service
	RESET		0	Error=Reset
	REV_ON		0	Motor=Reverse
	TIME_OFF	Mon. time off		
	TIME_ON	Mon. time on		
MOT_SPED	AUT_ON_OP		Mode=Manual	Mode=Auto
	MONITOR		Monitoring=Off	Monitoring=On
	MOT_OFF		0	Motor=Off
	OOS		In Service	Out of Service
	RESET		0	Error=Reset
	SP1_ON		0	Motor=Speed 1
	SP2_ON		0	Motor=Speed 2
	TIME_MON	Mon. time		
OP_A	BTRACK		Bumpless=Off	Bumpless=On
	U	U		
OP_A_LIM	BTRACK		Bumpless=Off	Bumpless=On
	U	U		
OP_A_RJC	BTRACK		Bumpless=Off	Bumpless=On
	U	U		

Baustein	Parameter	S7_shortcut	S7_string_0	S7_string_1
OP_D	BTRACK		Bumpless=Off	Bumpless=On
	I0		Off	On
OP_D3	BTRACK		Bumpless=Off	Bumpless=On
	I1		0	Switch 1
	I2		0	Switch 2
	I3		0	Switch 3
OP_TRIG	I0		0	Reset
RATIO_P	IN_EX		Internal	External
	MO_U1HR	Bar UL		
	MO_U1LR	Bar LL		
	U1	U1		
	U2	U2		
	U2_HL	High limit U2		
	U2_LL	Low limit U2		
	V_HL	High limit V		
	V_LL	Low limit V		
SWIT_CNT	M_SUP_AH		Suppress HH=No	Suppress HH=Yes
	M_SUP_WH		Suppress H=No	Suppress H=Yes
	MO_VHR	Bar UL		
	MO_VLR	Bar LL		
	OOS		In Service	Out of Service
	TRACK_OP		0	Preset
	V	V		
	VAH	HH alarm		
	VTRACK_OP	Preset value		
	VWH	H alarm		
VALVE	AUT_ON_OP		Mode=Manual	Mode=Auto
	MAN_OC		Valve=Close	Valve=Open
	MONITOR		Monitoring=Off	Monitoring=On
	OOS		In Service	Out of Service
	RESET		0	Error=Reset
	TIME_MON	Mon. time		
VAL_MOT	AUT_ON_OP		Mode=Manual	Mode=Auto
	CLOS_VAL		0	Valve=Close
	MONITOR		Monitoring=Off	Monitoring=On
	OOS		In Service	Out of Service
	OPEN_VAL		0	Valve=Open
	RESET		0	Error=Reset
	SS_POS			
	STOP_VAL		0	Valve=Stop
	TIME_OFF	Mon. time off		
	TIME_ON	Mon. time on		

2.2 Arbeiten mit dem Faceplate Designer

Die EXE und die DLLs zum Faceplate Designer V6.0 sind im Verzeichnis "..\WinCC\bin\FaceplateDesigner" abgelegt.



Prinzipielle Vorgehensweise

1. Starten Sie im WinCC Explorer den Faceplate Designer.
2. Im Kombinationsfeld “Typ” tragen Sie den Namen für den neuen Bildbaustein ein.

   Sie können den Namen aus der Klappliste der Strukturtypen vom WinCC-Variablenhaushalt auswählen oder einen neuen Namen vergeben.
3. Im rechten Fenster “Gewählte Sichten” tragen Sie die einzelnen Sichten des Bildbausteins ein, die vom Faceplate Designer angelegt werden sollen. Diese können Sie aus dem linken Fenster “Verfügbare Sichten” auswählen oder über die Schaltfläche “Weitere” neue Namen von Sichten anlegen. Neue Sichten-Namen müssen Sie dann allerdings ggf. auf Mehrsprachigkeit für die Anzeige der Sichten-Liste nachprojektieren. Beim Generieren dieser neuen Sicht erhalten Sie hierzu einen Hinweis.

   Die Vorgehensweise wird im Beispiel beschrieben (Kapitel 2.2.1).
4. Die Bildhöhe und die Bildbreite der Sichten projektieren Sie über die "Geometrie". Voreingestellt ist die Defaultgröße wie z.B. auch die Batch-Sicht oder die Meldesicht.

Beachten Sie beim Projektieren:

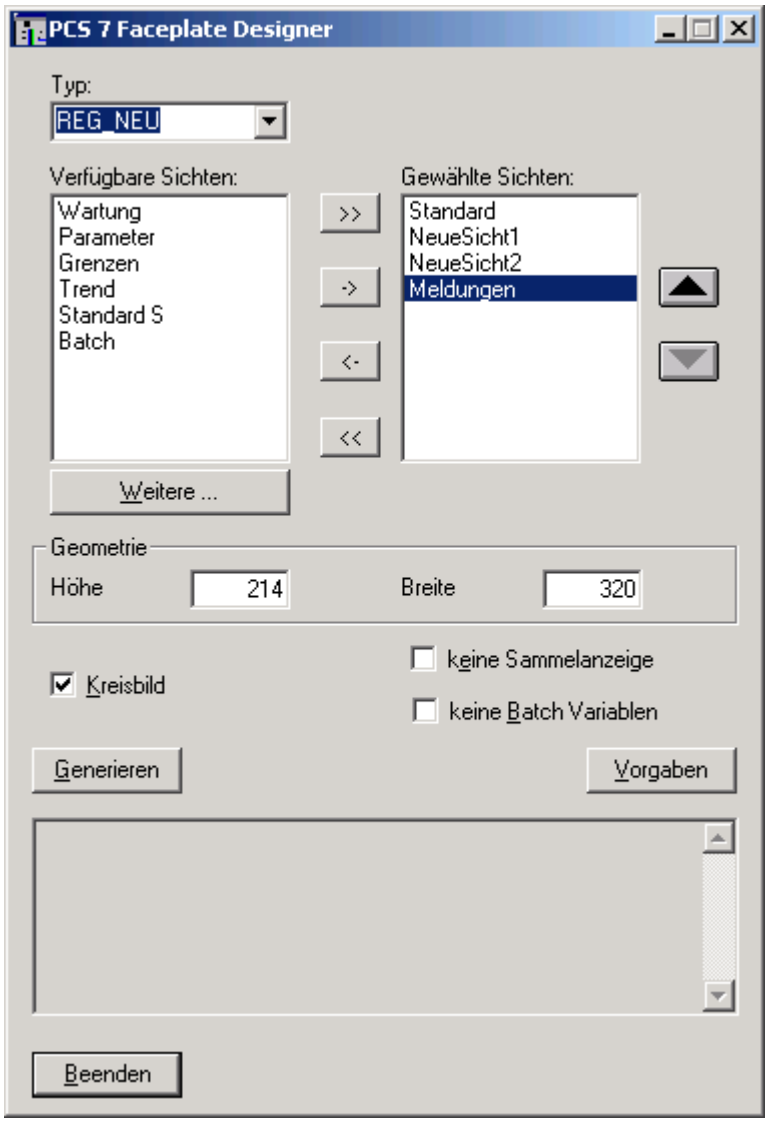
- Die Sichten der Bilder werden im Gruppenbild und im Kreisbild ausgegeben.
- Für Alarm, Batch und Trend werden universal gültige Bilder verwendet.
- Eine Unterdrückung der Kreisbildgenerierung ist möglich (gleichzeitig wird auch die Taste für die Anwahl des Kreisbildes unsichtbar geschaltet).
- Die Sprache des Faceplate Designers ist die, wie sie beim WinCC-Explorer eingestellt ist.
- Der Faceplate Designer ermittelt beim Aufschlag die aktuelle WinCC-Oberflächensprache und präsentiert seinen Dialog in derselben Sprache.
- Nachfolgende Umschaltungen der WinCC-Oberflächensprache bei aufgeschlagenem Dialog werden ignoriert.
- Die Schaltfläche "Vorgaben“ stellt die Grundeinstellung des Faceplate Designers wieder her.
- Mit der "Schaltfläche "Generieren" wird die Generierung der Bilder gestartet. Generiert werden (z.B. für einen Bildbausteinnamen “TEST” mit einer ausgewählten Sicht "Selected Views = Standard") folgende Bilder:

neues Bild	**aus dem Vorlagenbild**
@PG_TEST.pdl	@PG_%TYPE%.pdl
@PL_TEST.pdl	@PL_%TYPE%.pdl
@PG_TEST_OVERVIEW.pdl	@PG_%Type%_OVERVIEW.pdl
@PG_TEST_VIEWLIST.pdl	@PG_%Type%_VIEWLIST.pdl
@PG_TEST_STANDARD.pdl	@PG_%Type%_%View%.pdl

2.2.1 Beispiel: Erstellung eines neuen Bildbausteins für einen Regler

2.2.1.1 Vorlagen erstellen

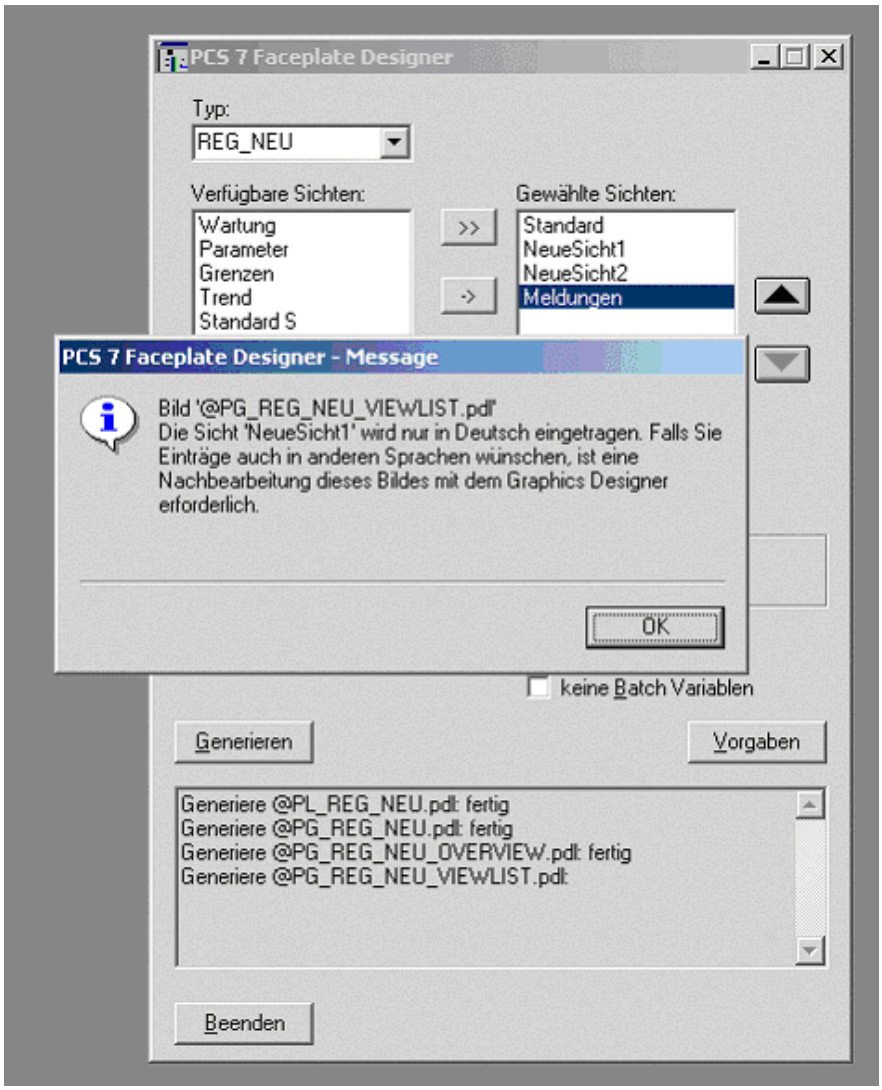
1. Starten Sie im WinCC Explorer den Faceplate Designer.
2. Im Feld "Typ" tragen Sie "REG_NEU“ ein.
3. In "Verfügbare Sichten“ legen Sie mittels der Schaltfläche "Weitere...." zwei neue Sichten an: "NeueSicht1“ und "NeueSicht2“.
4. Mit der Schaltfläche "->" (Pfeil-Rechts) übertragen Sie diese nach "Gewählte Sichten".
5. Schieben Sie ggf. die Sicht "Meldungen" mit der Schaltfläche "Pfeil nach unten" ans Ende der Liste.



Mit der Schaltfläche "Generieren" werden nun folgende Vorlagenbilder erzeugt:

neues Bild	**aus dem Vorlagenbild**
@PG_REG_NEU.pdl	@PG_%TYPE%.pdl
@PL_REG_NEU.pdl	@PL_%TYPE%.pdl
@PG_REG_NEU_OVERVIEW.pdl	@PG_%Type%_OVERVIEW.pdl
@PG_REG_NEU_VIEWLIST.pdl	@PG_%Type%_VIEWLIST.pdl
@PG_REG_NEU_STANDARD.pdl	@PG_%Type%_%View%.pdl
@PG_REG_NEU_NEUESICHT1.pdl	@PG_%Type%_%View%.pdl
@PG_REG_NEU_NEUESICHT2.pdl	@PG_%Type%_%View%.pdl

Beim Generieren der neuen Sichten kommt ein Hinweis zur Mehrsprachigkeit.



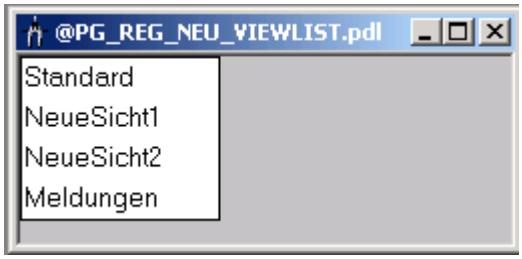
Im unteren Ausgabefenster werden die generierten Dateien aufgelistet.

2.2.1.2 Vorlagen Bearbeiten

Nach der Generierung der Vorlagenbilder können Sie mit der Bearbeitung der einzelnen Sichten beginnen:

@PG_REG_NEU_VIEWLIST.pdl
@PG_REG_NEU_STANDARD.pdl
@PG_REG_NEU_NEUESICHT1.pdl
@PG_REG_NEU_NEUESICHT2.pdl

Das Bild "@PG_REG_NEU_VIEWLIST.pdl" muss nur dann bearbeitet werden, wenn der Bildbaustein auch in anderen Sprachen als in der Erstellersprache genutzt werden soll.
Das Beispiel wurde in deutscher Sprache generiert und zeigt im Graphics Designer bei der Spracheinstellung "Deutsch" folgendes Bild.



Schalten Sie nun im Graphics Designer auf die englische Sprache um, so wird folgendes Bild angezeigt:

Hier müssen Sie nun für die statischen Texte, deren Objektname "NeueSicht1“ und "NeueSicht2“ ist, die Eigenschaften "Text" anpassen, z.B. "NewView1“ und "NewView2“.

Das Gleiche ist auch bei der französischen Sprache vorzunehmen.

2.2.1.3 @PG_REG_NEU_STANDARD.pdl bearbeiten

Wenn Sie eine Modifikation am Standard-Bildbaustein des Reglers vornehmen, so empfiehlt sich folgende Vorgehensweise:

1. Öffnen Sie "@PG_REG_NEU_STANDARD.pdl"
2. Löschen Sie die Objekte "@Level6" und "@Level5" (siehe hierzu auch die Beschreibung "Bedienberechtigungen projektieren").
3. Öffnen Sie den Standard-Bildbaustein "@PG_CTRL_PID" für den Regler, markieren und kopieren Sie alles und fügen Sie es in das Bild "@PG_REG_NEU_STANDARD.pdl" ein.

   Die Elemente "@Level6" und "@Level5", die für die Bedienberechtigung vorgesehen sind, wurden mitkopiert und haben schon die entsprechenden Direktverbindungen auf die bedienbaren Elemente.

   Hinweis! Beim Kopieren von Objekten werden Sonderzeichen beim Objektnamen der Kopie entfernt. Deswegen heißen die Objekte "@Level6“ und "@Level5“ im Bild "@PG_REG_NEU_STANDARD.pdl" zunächst "Level6“ und "Level5“ und müssen zwingend auf die ursprünglichen Namen umbenannt werden, da diese per Skript mit Werten versorgt werden.
4. Anschließend können Sie die Objekte im Bild verändern, neue Objekte hinzufügen oder auch Objekte löschen.

Beachten Sie beim Löschen von Objekten, dass diese über eine Direktverbindung eine Information bekommen oder weiterleiten. Siehe hierzu "Dokumentation der Standardbildbausteine". Dort sind die Direktverbindungsketten aufgelistet, einschließlich der Objekte, die per Direktverbindung Informationen weiterleiten. In der Regel sind das

- "@Level6" und "@Level5" zum Übertragen der Bedienberechtigungen,
- das Obkjekt "Format" zum Übertragen von instanzspezifischen Zahlenformaten (siehe hierzu die Beschreibung der Basis-Elemente ab Kapitel 2.3).

2.2.1.4 @PG_REG_NEU_NEUESICHT1.pdl bearbeiten

Aus dem Vorlagenbild "@PCS7Elements.pdl" können Sie Objekte einfügen und dynamisieren.

Vorgehensweise:

1. Öffnen Sie im WinCC Graphics Designer das Bild "@PG_REG_NEU_NEUESICHT1.pdl" und "@PCS7Elements.pdl".
2. Ordnen Sie die Bilder nebeneinander an (Menübefehl: Fenster > Nebeneinander).
3. Kopieren Sie die benötigten Bildelemente vom Bild "@PCS7Elements.pdl" in das Bild "@PG_REG_NEU_NEUESICHT1.pdl".
4. Vergeben Sie sinnvolle (objektbezogene) Objektnamen.
5. Stellen Sie die Position für die einzelnen Objekte ein.
6. Verschalten Sie die dynamischen Attribute der Bildelemente mit den Parametern des AS.
7. Bauen Sie die "Bedienberechtigungsketten" mit den Objekten "@Level5" und "@Level6" auf.
8. Speichern Sie das Bild.

2.2.1.5 Dynamisierung von Bildbausteinen

Es gibt unterschiedliche Wege um eine Dynamisierung zu erreichen:

- Die Extension der benötigten Variablen sind bekannt.

  Für das zu dynamisierende Objekt (z.B. Balken) rufen Sie die Objekteigenschaften auf. In der Spalte "Dynamik" des Eigenschaften-Fensters doppelklicken Sie das Symbol der Glühlampe für das gewünschte Attribut. Im Eingabefeld können Sie nun die Extension mit Punkt eintragen, z.B. ".PV_IN“

- Variable aus der Variablenliste

  Der bei Prozessbildern übliche Weg führt über die Variablenliste. Hier sind allerdings alle Variablen aufgeführt. Mit der rechten Maustaste klicken Sie im Eigenschaften-Fenster auf das Glühlampen-Symbol und wählen "Variable" aus.

  Suchen Sie die gewünschte Variable und führen Sie darauf einen Doppelklick aus.
  Jetzt steht in der Dynamisierung allerdings der vollständige Variablenname. Löschen Sie den Text bis zum Punkt (".“) vor der Extension.

2.2.1.6 Loop-Darstellung erstellen

Es empfiehlt sich, immer eine Loop-Darstellung zu erzeugen, auch wenn nur eine Sicht am Bildbaustein benötigt wird, da es bei der Funktion "Bildanwahl über Messstelle“ immer die Möglichkeit gibt, in die Loop Darstellung zu springen.

Vorgehensweise:

1. Generieren Sie zuerst mit dem Faceplate Designer die Bilder ohne die Loop-Darstellung, dann wird die Gruppenbild-Darstellung ohne die Loop-Bild-Anwahltaste generiert.
2. Danach generieren Sie nochmals mit Loop-Bild-Darstellung. Bei allen Bildern, außer "Loop-Bild überschreiben", quittieren Sie mit "Nein“.

2.2.1.7 Nachträglich eine neue Sicht generieren

Wenn Sie nachträglich eine neue Sicht generieren wollen und die bereits vorhandenen Sichten schon modifiziert haben, so empfiehlt sich folgende Vorgehensweise:

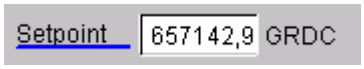
1. Im Faceplate Designer tragen Sie den Typnamen und die gewählten Sichten genau so wieder ein, wie in der ersten Generierung.
2. Tragen Sie zusätzlich die neue Sicht ein und starten Sie die Generierung.

   Während der Generierung kommt die Abfrage für jede bereits vorhandene Datei, ob diese überschrieben werden soll.
3. Diese Meldungen quittieren Sie mit "Nein".

2.3 Basis-Elemente

Die Basis-Elemente sind alle in dem Muster Bild "@PCS7Elements.pdl" abgelegt. Das Bild wird im Verzeichnis "..\Wincc\options\pdl\FaceplateDesigner_V6" abgelegt und vom OS-Projekteditor in das Projekt kopiert.

2.3.1 Analogwertdarstellung und Analogwertbedienung



Objekttyp : PCS7_AnalogValue

Objektname: PCS7_AnalogValue1, PCS7_AnalogValue2, PCS7_AnalogValue3

Bildname: @PCS7Elements.pdl

Die beiden Objekte “PCS7_AnalogValue1“ und "PCS7_AnalogValue2“ sind identisch und hier als Beispiel parametriert.

- “PCS7_AnalogValue1“ für Analogwertbedienung und
- “PCS7_AnalogValue2“ für Analogwertanzeige.

Bei diesen beiden Objekten wird die Gleitkommadarstellung des EA-Feldes verwendet. Die Objekte können Sie bei Zahlendarstellungen bzw. Zahleneingaben verwenden, die nicht vom Prozess verändert werden, da hier das gleitende Komma nicht als störend wirkt.

Empfehlenswert ist aber das Objekt "PCS7_AnalogValue3“, welches mit dem neuen AdvancedAnalogDisplay erstellt wurde. Hier gleiten nur die Vorkommastellen, während die Nachkommastellen instanzspezifisch projektierbar sind.

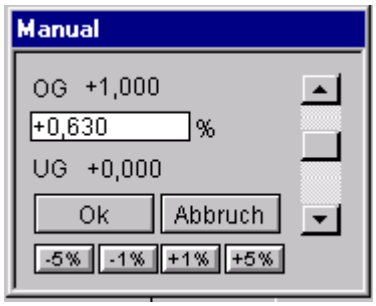
Bei Verwendung als reine Analogwertanzeige parametrieren Sie die Eigenschaft "Bedienfreigabe“ auf "FALSE“. Die Eigenschaft "Hintergrundfarbe“ stellen Sie auf "grau".

Bei Verwendung als Analogwertbedienung wird bei einem Mausklick das Skript "PCS7_OpenInputBoxAnalog_V6“ aufgerufen.

Beim Skript-Übergabeparameter CallFrom = 0

"PCS7_OpenInputBoxAnalog_V6(lpszPictureName,lpszObjectName,0);“

wird das Bedienbild "@PCS7_BedAnalog.pdl“ aufgerufen. Dieses Bedienbild enthält einen Schiebebalken mit prozentualer Verstellung und ist für Analogbedienungen mit Bediengrenzen vorgesehen.



Beim Skript-Übergabeparameter "CallFrom = 1"

"PCS7_OpenInputBoxAnalog_V6(lpszPictureName,lpszObjectName,1);“

wird das Bedienbild "@PCS7_BedAnalog_NL.pdl“ aufgerufen. Dieses Bedienbild enthält keinen Schiebebalken und ist für Analogbedienungen ohne Bediengrenzen vorgesehen.

Um die Änderung einer prozessgesteuerten Bedienfreigabe direkt bei der Änderung in das Bedienbild zu übertragen, können Sie auch in der Eigenschaft "Bedienfreigabe" des Objektes "PCS7_AnalogValue“

- für das Bedienbild mit Grenzen das Skript "PCS7_OpenInputBoxAnalog_V6(lpszPictureName,lpszObjectName,10);“

  und

- für das Bedienbild ohne Grenzen das Skript "PCS7_OpenInputBoxAnalog_V6(lpszPictureName,lpszObjectName,11);“

aufrufen.

Dies deckt den Fall ab, wenn bei gerade angewähltem Bedienbild in diesem Moment die Bedienfreigabe für genau diesen Wert entzogen wird, dass auch der Wert im Bedienbild unbedienbar wird.

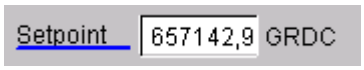
Eigenschaften	**Funktion**
ProcessValue/OutputValue	Strukturelement des Analogwertes, der bedient wird.
ProcessValue/VisibleValue	Strukturelement des Analogwertes, der dynamisiert wird.
Limits/UpperLimit	Strukturelement für Bediengrenze "Oben" des Analogwertes
Limits/LowerLimit	Strukturelement für Bediengrenze "Unten" des Analogwertes
Sonstige/Bedienfreigabe	Bedienfreigabe von der AS-Seite, z.B. "Q_SP_OP" oder fest auf "Nein" parametriert, wenn nur eine Anzeige erfolgen soll.
Sonstige/Anzeige_Analogwert	Muss immer sichtbar geschaltet sein.
Sonstige/Passwort	Sollte nicht benutzt werden, wird über die Bedienberechtigung abgewickelt.
Sonstige/Anzeige_Linie	Anzeige der blauen Linie.
Sonstige/Linienstärke	Linienstärke kann eingestellt werden.
Sonstige/ Open_BedBox_from_Bar	Eigenschaften zu indirekten Aufrufen des Bedienbildes "@PCS7_BedAnalog.pdl“, z.B. über Balken.
Schrift/Text	Beschriftung des EA-Feldes (Sollwert).
Schrift/ Unit	Einheit des Analogwertes dynamisch über .MEMBER#unit, z. B. .SP_OP#unit
Schrift/X-Ausrichtung_Text	Linksbündig, wenn mehrere EA-Felder untereinander stehen (wegen der Ausrichtung). Rechtsbündig empfiehlt sich bei einzeln stehenden Analogwerten.
Farben/Schriftfarbe	Auswahl der Schriftfarbe.
Farben/Linienfarbe	Auswahl der Linienfarbe.
Farben/Hintergrundfarbe_Value	Auswahl der Hintergrundfarbe "Value" wird mit Skript dynamisiert beim Ereignis "Änderung der Bedienfreigabe“.
Farben/Schriftfarbe_Value	Auswahl Schriftfarbe "Value".
Geometrie/Breite_Analogwert	Breite des gesamten Analogfeldes.
Geometrie/Breite_Linie_Text	Länge der Linie und des Textes kann eingestellt werden.
Geometrie/PositionX1_Linie_Text	Muss angepasst werden bei Änderung der Text- und Linienlänge.
Geometrie/PositionY1_Linie	Linienhöhe gegenüber der Schrift.

Die Eigenschaft "Schrift/Text“ ist defaultmäßig mit dem Beispiel-Parameter ".SP_OP#shortcut“ verschaltet, das heißt, der Anzeigetext des Bausteinsymbols sowie die Überschrift des Analog-Bedienbildes werden aus dem AS gelesen (s7_shortcut). Bei Variablen, die keinen Anzeigetext besitzen (z.B. PV_IN), müssen Sie die Verschaltung löschen und den Text direkt in WinCC (ggf. dreisprachig) parametrieren.

Bei der Änderung der Eigenschaft "Open_BedBox_from_Bar“ wird auch das Skript "PCS7_OpenInputBoxAnalog_V6" aufgerufen. Auch hier wird zuvor überprüft, ob die Bedienfreigabe vorhanden ist.

Das Skript kann auch für die indirekte Bedienung verwendet werden. Z. B. wird beim Ereignis "Anklicken des Sollwert-Balkens“ das entsprechende Bedienbild aufgerufen. Siehe hierzu auch die Beschreibung bei der zweifachen Balkendarstellung für Sollwert und Istwert. Dort wird die Änderung einer Eigenschaft per Direktverbindung umgeleitet auf die Eigenschaft "Open_BedBox_from_Bar“.

2.3.2 Analogwertdarstellung mit "AdvancedAnalogDisplay"



Objekttyp: PCS7_AnalogValue

Objektname: PCS7_AnalogValue3

Bildname: @PCS7Elements.pdl

Für die Zahlendarstellungen, bei denen keine Gleitkommas (vor allem für die Nachkommastellen) erwünscht sind, gibt es die Zahlendarstellung mit dem AdvancedAnalogDisplay. Hier kann das Zahlenformat instanzspezifisch über das Bausteinsymbol vorgegeben werden.
Siehe Kapitel 2.1.6 "Zahlenformate projektieren".

Die Beschreibung der Objekte des Anwenderobjekts und die Funktion der Eigenschaften ist identisch mit der Beschreibung im Kapitel 2.3.1.
Zusätzliche ist die Eigenschaft "Format" vorhanden. Das Zahlenformat wird per Direktverbindung angesteuert, wenn instanzspezifisch Zahlenformate erwünscht sind.

2.3.3 Statischer Text

Der statische Text wird für die Beschriftungen von Optionskästchen, Ein-/Aus-Tasten usw. verwendet.

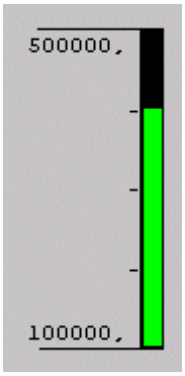
Objekttyp : PCS7_StaticText

Objektname: StaticText

Bildname: @PCS7Elements.pdl

Zeichensatz Arial, Schriftgröße 12

2.3.4 Einfache Balkendarstellung für Analogwerte



Objekttyp: PCS7_BarStandard_1

Objektname: BarStandard_1

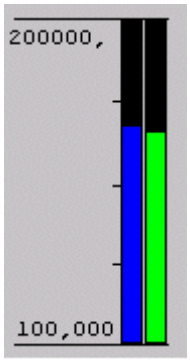
Bildname: @PCS7Elements.pdl

Eigenschaften	Funktion
Links/PV	Strukturelement des Analogwertes, der als Balken dynamisiert wird.
Links/ Range_LL	Strukturelement für "Balkengrenze unten" (Messanfangswert).
Links/ Range_UL	Strukturelement für "Balkengrenze oben" (Messendwert).
Sonstige / PVunderLimit	Anzeige Messbereichsunterschreitung. Der Istwert darf vom Anwender nicht verändert werden.
Sonstige / PVoverLimit	Anzeige Messbereichsüberschreitung. Der Istwert darf vom Anwender nicht verändert werden.
Farben / Balkenfarbe_PV	Farbe des Balkens.

An den verschalteten Eigenschaften PV, RANGE_LL, RANGE_UL sind jeweils Skripte hinterlegt (Aufruf des Skriptes "PCS7_UpdateBar_V6.fct"), die bei Änderung der Werte eine evtl. Messbereichsüberschreitung mit Pfeilen an den Balkengrenzen anzeigen.

2.3.5 Zweifache Balkendarstellung für Analogwerte

Die zweifache Balkendarstellung wird z.B. für die gleichzeitige Darstellung von Sollwert und Istwert verwendet.



Objekttyp: PCS7_BarStandard_2

Objektname: BarStandard_2

Bildname: @PCS7Elements.pdl

Eigenschaften	Funktion
Links/PV	Strukturelement des Istwertes, der als Balken dynamisiert wird.
Links/SP	Strukturelement des Sollwertes, der als Balken dynamisiert wird.
Links/ Range_LL	Strukturelement für "Balkengrenze unten" (Messanfangswert).
Links/ Range_UL	Strukturelement für "Balkengrenze oben" (Messendwert).
Sonstige / SPunderLimit	Anzeige Messbereichsunterschreitung. Der Sollwert darf vom Anwender nicht verändert werden.
Sonstige / PVunderLimit	Anzeige Messbereichsunterschreitung. Der Istwert darf vom Anwender nicht verändert werden.
Sonstige / SpoverLimit	Anzeige Messbereichsüberschreitung. Der Sollwert darf vom Anwender nicht verändert werden.
Sonstige / PVoverLimit	Anzeige Messbereichsüberschreitung. Der Istwert darf vom Anwender nicht verändert werden.
Farben / Balkenfarbe_SP	Farbe des Balkens für den Sollwert.
Farben / Balkenfarbe_PV	Farbe des Balkens für den Istwert.

Wird die Bedienfreigabe der Balkendarstellung auf "TRUE“ gesetzt, so wird beim Anklicken der Balkendarstellung die Bedienfreigabe per Skript kurz auf "FALSE“ und anschließend sofort wieder auf "TRUE“ gesetzt.

Die Eigenschaft "Bedienfreigabe" kann anschließend per Direktverbindung auf das zugehörige Basis-Element "PCS7_AnalogValue" (Eigenschaft "Open_BedBox_from_Bar“) weitergeleitet werden und bewirkt dort den Aufruf des analogen Bedienbildes zum Sollwert (siehe auch Beschreibung der Analogwertbedienung, Kapitel 2.3.1).

Bei der Balkendarstellung wird die Pfeil-Darstellung für Werte unter- und oberhalb der Balkengrenzen ergänzt.

An den verschalteten Eigenschaften PV, SP, RANGE_LL, RANGE_UL sind jeweils Skripte hinterlegt (Aufruf des Skriptes "PCS7_UpdateBar.fct"), die bei Änderung der Werte eine evtl. Messbereichsüberschreitung mit Pfeilen an den Balkengrenzen anzeigen.

2.3.6 Balkendarstellung horizontal

Die horizontale Darstellung des Balkens kann z.B. für Stellgrößen verwendet werden.

Objekttyp: PCS7_BarStandard_3

Objektname: BarStandard_3

Bildname: @PCS7Elements.pdl

Die Eigenschaft "Bedienfreigabe" kann per Direktverbindung auf das zugehörige Basis-Element "PCS7_AnalogValue" (Eigenschaft "Open_BedBox_from_Bar“) weitergeleitet werden und bewirkt dort den Aufruf des analogen Bedienbildes zum Sollwert (siehe auch Beschreibung der Analogwertbedienung, Kapitel 2.3.1).

Bei der Balkendarstellung wird die Pfeil-Darstellung für Werte unter- und oberhalb der Balkengrenzen ergänzt.

Eigenschaften	Funktion
Sonstige/ Balkenfarbe	Farbe des Balkens
Sonstige / PVunderLimit	Anzeige der Messbereichsunterschreitung. Der Istwert darf vom Anwender nicht verändert werden.
Sonstige / PVoverLimit	Anzeige der Messbereichsüberschreitung. Der Istwert darf vom Anwender nicht verändert werden.
Sonstige/Unit	Einheit der Stellgröße.
Links/ RANGE_LL	Mess-Anfang der Balken-Anzeige.
Links/ 50PWERT	50%-Anzeige wird bei Änderung von RANGE_LL oder RANGE_UL mit Skript errechnet.
Links/ RANGE_UL	Mess-Ende der Balken-Anzeige.
Links/PV	Strukturelement der Stellgröße, die als Balken dynamisiert wird.

2.3.7 Balkendarstellung "Grenzwertanzeige"

Objekttyp: PCS7_BarLimits
Objektname: BarLimits
Bildname: @PCS7Elements.pdl

Eigenschaften	Balken-Eigenschaft	Funktion
Grenzen / RANGE_UL	Maximalwert und Prozessanschluss	Messbereichsende des Balkens (.MO_PVHR
Grenzen / RANGE_LL	Minimalwert, Nullpunktwert und Obergrenze RH5	Messbereichsanfang des Balkens (.MO_PVLR)
Grenzen / VALUE_AH	Obergrenze AH	Grenzenanzeige "Alarm high" (.PVH_ALM)
Grenzen / VALUE_WH	Obergrenze WH	Grenzenanzeige "Warning high" (.PVH_WRN)
Grenzen / VALUE_WL	Untergrenze WL	Grenzenanzeige "Warning low" (.PVL_WRN)
Grenzen / VALUE_AL	Untergrenze AL	Grenzenanzeige "Alarm low" (.PVL_ALM)
Grenzen / @ObergrenzeTH	Obergrenze TH	Steuerung der unteren Grenze des zweituntersten Segmentes. Die Eigenschaft ist nur im Konfigurationsdialog sichtbar.
Grenzen / @ObergrenzeRH4	Obergrenze RH4	Steuerung der unteren Grenze des untersten Segmentes. Die Eigenschaft ist nur im Konfigurationsdialog sichtbar.
Grenzen / @BalkenfarbeRH4	Balkenfarbe RH4	Farbbestimmung des zweituntersten Segmentes. Die Eigenschaft ist nur im Konfigurationsdialog sichtbar.
Grenzen / @BalkenfarbeRH5	Balkenfarbe RH5	Farbbestimmung des untersten Segmentes. Die Eigenschaft ist nur im Konfigurationsdialog sichtbar.
Farben/FarbeAlarm	Balkenfarbe AH	
Farben/FarbeWarnung	Balkenfarbe WH	
Farben/ FarbeHintergrund	Balkenfarbe TH	Farbe des mittleren Segmentes.

Bei einer Änderung der unteren Grenzwerte "VALUE_WL", "VALUE_AL" und "RANGE_LL" wird das Skript "PCS7_UpdateBarLimits_V6" aufgerufen, da hier die Balkendarstellung für "Alarm" und "Warnung unten" nicht mit Mitteln des Standard-Balkens erzeugt werden kann. Für die oberen Grenzwerte ist kein Skript notwendig.

2.3.8 Anzeige "Meldungsunterdrückung"

Objekttyp: PCS7_MSG_LOCK

Objektname: MSG_LOCK

Bildname: @PCS7Elements.pdl

Eigenschaften	Funktion
Benutzerdefiniert2 / aktueller Zustand	Steuerung "Zustandsanzeige 2". Einmal durchgestrichene Meldungen sperren Anforderung vom AS (.MSG_LOCK) 
Benutzerdefiniert2 / aktueller Zustand1	Steuerung "Zustandsanzeige 3". Zweimal durchgestrichene Meldungen sind gesperrt. Rückmeldung von WinCC (.QMSG_SUP) 
Benutzerdefiniert2 / Anzeige1	"Zustandsanzeige 3" wird eingeblendet und überblendet "Zustandsanzeige 2" (.QMSG_LOCK)

2.3.9 Anzeige "Batch Occupied"

Objekttyp: PCS7_OCCUPIED

Objektname: OCCUPIED

Bildname: @PCS7Elements.pdl

Eigenschaften	Funktion
Benutzerdefiniert2 / aktueller Zustand	Steuerung der Zustandsanzeige 1 / Anzeige, ob Baustein von Batch belegt ist. Strukturelement (.OCCUPIED).

2.3.10 Quittierung der Meldungen vom angewählten Bildbaustein

Objekttyp: Button

Objektname: ButtonQS

Bildname: @PCS7Elements.pdl

Die Sammelquittierung von Meldungen ist die gleiche Taste wie im Standard-Tastensatz "@Buttons11.pdl", jedoch mit einem anderem Skript für die instanzspezifische Meldequittierung. Das funktioniert nur im Zusammenhang mit einer Sammelanzeige.

2.3.11 Anzeige-Baustein "Verriegelt" (Ventil, Motor)


Objekttyp: Zustandsanzeige

Objektname: Lock

Bildname: @PCS7Elements.pdl

Eigenschaften	Funktion
Zustand / aktueller Zustand	Anzeige, ob Baustein verriegelt ist. Strukturelement (.LOCK)

2.3.12 Sammelanzeige

Objekttyp: Sammelanzeige

Objektname: EventState

Bildname: @PCS7Elements.pdl

Eigenschaften	Funktion
Sonstige / Sammelwert	Anzeige der Alarm- und Warnungszustände. Strukturelement (.EventState)

2.3.13 Binärwertbedienung mit Check Box_R

@PCS7_BedCheck.pdl

Objekttyp: CHECKBOX_R

Objektname: Checkbox_R1

Bildname: @PCS7Elements.pdl

Das Bild "@PCS7_BedCheck.pdl" wird über das Skript "PCS7_OpenCheckbox_V6" aufgerufen.

Eigenschaften	Funktion
Sonstige/ DisplayActivWith	DisplayActivWith = 1 → Bei gesetztem Häkchen im Optionskästchen wird 'TRUE' auf die binäre Variable geschrieben. DisplayActivWith = 0 → Bei gesetztem Häkchen im Optionskästchen wird 'FALSE' auf die binäre Variable geschrieben.
Sonstige/ Bedienfreigabe	Bedienfreigabe des Optionskästchens.
Verschaltung/Input	Überschrift für Bedienbild zum Optionskästchen "@PCS7_BedCheck.pdl".
Verschaltung/ Variable	Verschaltung auf das binäre Strukturelement, z.B. (.MSG_LOCK).
Verschaltung/NegatedVariable	Optionskästchen, das angeklickt wird; darf vom Anwender nicht verändert werden.
Verschaltung/Read_Text_From_AS	Text wird von "String_0" oder "String_1" gelesen und der Eigenschaft "Input" zugewiesen.
Verschaltung/ CaptionCheckBoxOn	JA = Text von "Input" wird am Optionskästchen rechts angezeigt. NEIN = Bei wenig Platz wird kein Text zum Optionskästchen angezeigt.

Bei einem Mausklick auf das Optionskästchen wird die interne Option "Inverted Variable“ angewählt. Es wird der Zustand der zu bedienenden binären Variable ermittelt und auf "Inverted Variable" zurück geschrieben (Eigenschaft "Verschaltung/NegatedVariable"). Weiterhin wird bei Änderung der Eigenschaft "Verschaltung/Variable“ und bei Bildanwahl ein Skript durchlaufen, das die Anzeige von "NegatedVariable" aktualisiert (siehe auch Flussdiagramm Check_Box). Die Eigenschaft "Verschaltung/Variable“ muss von der Projektierung auf "ungleich 1“ und "ungleich 0“ (0x8) eingestellt sein, damit in allen Fällen das Skript auch bei Bildanwahl durchlaufen wird.

Die Texte für die Beschriftung der Optionsfelder werden aus den Parameterattributen "String_0" und "String_1" der Bausteininstanz gelesen.

Für die Beschriftung des Optionskästchens gilt:

- Wird die Eigenschaft "Read_Text_From_AS" auf 1 gesetzt, so wird bei "DisplayActivWith = 1" der komplette Text von "String_1" rechts vom Optionskästchen angezeigt.
- Bei "DisplayActivWith = 0" wird der komplette Text von "String_0" rechts vom Optionskästchen angezeigt.
- Wird die Eigenschaft "Read_Text_From_AS" auf 0 gesetzt, so wird der an “Input“ projektierte Text angezeigt. Dies ist insbesondere bei der Anzeige von Ausgangsparametern einzustellen, da hier kein "String_0" oder "String_1" existiert und somit Skriptfehler erzeugt würden. Ebenfalls darf das Optionskästchen für diesen Fall nicht bedienbar geschaltet werden.

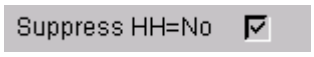
Für die Beschriftung der Optionsfelder gilt:

- Enthält der Text der Strings ("String_0" und "String_1") ein "=“, so wird der rechte Teil der Strings (nach dem "=“) für die Beschriftung der Optionsfelder verwendet.
- Enthält der Text der Strings kein "=“, so wird der komplette Text für die Beschriftung der Optionsfelder verwendet.

Für die Überschrift des Bedienbildes gilt:

- Enthält der Text der Strings ("String_0" und "String_1") ein "=“, so wird der linke Teil vor dem "=“ für die Überschrift des Bedienbildes verwendet, bei "DisplayActivWith = 1" von "String_1" und bei "DisplayActivWith = 0" von "String_0".
- Enthält der Text der Strings kein "=“, so wird der Text von der Beschriftung des Optionskästchens genommen.

2.3.14 Binärwertbedienung mit "Check Box_L"

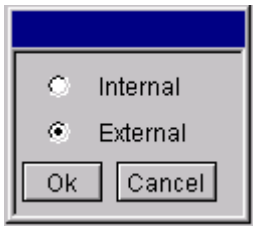


Die Funktionalität ist die gleiche wie bei "CheckBox_R", nur wird der Anzeigetext auf der linken Seite dargestellt.

Auch wenn der Text nicht angezeigt wird, werden beide Varianten (Text rechts oder Text links) benötigt, abhängig davon, ob rechts oder links vom Optionskästchen ein weiteres bedienbares Element liegt (z.B. bei den Alarmgrenzen von MEAS_MON).

Eigenschaften und Funktion siehe 2.3.13.

2.3.15 Binärwertbedienung mit Kombinationsfeld (Combobox)



@PCS7_BedCombo.pdl

Objekttyp: PCS7_COMBOBOX
Objektname: COMBOBOX1
Bildname: @PCS7Elements.pdl

Das Bild "@PCS7_BedCombo.pdl" wird über das Skript "PCS7_OpenComboBox_V6.fct" aufgerufen.

Die Texte werden von "String_0" und "String_1" gelesen

Eigenschaften	Funktion
Farben/BackColor_Text1	Hintergrundfarbe für Text1
Farben/BackColor_Text2	Hintergrundfarbe für Text2
Schrift/Text1	Anzeigetext für Text1 (wird aus AS gelesen).
Schrift/Text2	Anzeigetext für Text2 (wird aus AS gelesen).
Displays/Display_Text1	Anzeige-Text1 (Steuerung der Anzeige, welcher Wert selektiert ist).
Displays/Display_Text2	Anzeige-Text2 (Steuerung der Anzeige, welcher Wert selektiert ist).
Sonstige/ OP_enabled_Text1	Bedienfreigabe für Text1.
Sonstige/ OP_enabled_Text2	Bedienfreigabe für Text2.
Sonstige/ Bedienfreigabe	Bedienfreigabe für komplettes Kombinationsfeld für Steuerung über "@Level5/6".
Sonstige/ Password_Text1	Bedienberechtigung für Text1 (wird nicht verwendet).
Sonstige/ Password_Text2	Bedienberechtigung für Text2 (wird nicht verwendet).
Links/Write_Variable1	Strukturelement, das mit Anwahl-Text1 beschrieben wird.
Links/Write_Variable2	Strukturelement, das mit Anwahl-Text2 beschrieben wird.
Links/Display_Variable1	Strukturelement, welches den ersten Binärzustand (Text1) anzeigt.
Links/Display_Variable2	Strukturelement, welches den zweiten Binärzustand (Text2) anzeigt.
Parameters/ Write_value_Text1	Gibt an, welcher Wert auf das Strukturelement "Write_Variable1“ mit Anwahl-Text1 geschrieben wird ('TRUE' oder 'FALSE').
Parameters/Display_Text1_with	Gibt an, mit welchem Wert ('TRUE' oder 'FALSE) das Strukturelement "Display_Variable1" den Text1 anzeigt.
Parameters/ Write_value_Text2	Gibt an, welcher Wert auf das Strukturelement, "Write_Variable2“ mit Anwahl-Text2 geschrieben wird ('TRUE' oder 'FALSE).
Parameters/Display_Text2_with	Gibt an, mit welchem Wert ('TRUE' oder 'FALSE) die "Display_Variable2" den Text2 anzeigt.

Die Texte für die Beschriftung der Optionsfelder werden aus den Parameterattributen "String_0" und "String_1" der Bausteininstanz gelesen.

Enthält der Text der Strings ein "=“, so wird der rechte Teil der Strings (nach dem "=“) für die Beschriftung der Optionsfelder und der linke Teil (vor dem "=“) für die Überschrift des Bedienbildes verwendet.

Enthält der Text der Strings kein "=“, so wird der komplette Text für die Beschriftung der Optionsfelder verwendet. Die Überschrift des Bedienbildes wird in diesem Fall mit Leerzeichen gefüllt.

Hinweise für die Bedienfreigabe:

Für das Anzeigen bzw. Hervorheben einer fehlenden Bedienfreigabe kann der Hintergrund des Textes im Kombinationsfeld grau geschaltet werden. Dies geschieht über eine Direktverbindung von "@Level5"- oder "@Level6"-Hintergrundfarbe auf "BackColor_Text1" in diesem Objekt und einer weiteren Direktverbindung von "BackColor_Text1" auf "BackColor_Text2"-Hintergrundfarbe. Die Hintergrundfarben müssen dann beide vom Default her grau sein.

Diese Möglichkeit besteht jedoch nur, wenn die verschiedenen Schaltzustände des Kombinationsfeldes immer mit weißem Hintergrund angezeigt werden, damit die Bedienberechtigung zwischen grau und weiß schaltet.

Beispiel für Regler-Umschaltung "Hand/Automatik":

Beim Kombinationsfeld zur Betriebsartenumschaltung "Hand/Automatik" wird die Hintergrundfarbe auch zur Kennzeichnung der Betriebsart verwendet (Weiß = Hand, Grün = Automatik)

Aus diesem Grund kann hier bei fehlender Bedienberechtigung nicht auf den grauen Hintergrund geschaltet werden. Bei fehlender Bedienfreigabe wird deshalb bei den Kombinationsfeldern zusätzlich noch die Pfeil-Schaltfläche unsichtbar geschaltet.

Bedienfreigabe "Ja"

Bedienfreigabe "Nein"

2.3.16 Binärwertbedienung mit Kombinationsfeld (3ComboBox)

@PCS7_3BedCombo.pdl

Objekttyp: PCS7_3COMBOBOX

Objektname: 3COMBOBOX1

Bildname: @PCS7Elements.pdl

Das Bild "@PCS7_3BedCombo.pdl" wird über das Skript "PCS7_Open3ComboBox_V6.fct" aufgerufen.

Die Texte werden von "String_0" und "String_1" gelesen.

Eigenschaften	Funktion
Farben/BackColor_Text1	Hintergrundfarbe für Text1.
Farben/BackColor_Text2	Hintergrundfarbe für Text2.
Farben/BackColor_Text3	Hintergrundfarbe für Text3.
Schrift/Text1	Anzeigetext für Text1 (wird aus AS gelesen).
Schrift/Text2	Anzeigetext für Text2 (wird aus AS gelesen).
Schrift/Text3	Anzeigetext für Text3 (wird aus AS gelesen).
Displays/Display_Text1	Anzeige Text1 (Steuerung der Anzeige, welcher Wert selektiert ist).
Displays/Display_Text2	Anzeige Text2 (Steuerung der Anzeige, welcher Wert selektiert ist).
Displays/Display_Text3	Anzeige Text3 (Steuerung der Anzeige, welcher Wert selektiert ist).
Sonstige/ OP_enabled_Text1	Bedienfreigabe für Text1.
Sonstige/ OP_enabled_Text2	Bedienfreigabe für Text2.
Sonstige/ OP_enabled_Text3	Bedienfreigabe für Text3.
Sonstige/ Bedienfreigabe	Bedienfreigabe für komplettes Kombinationsfeld für Steuerung über "@Level5/6".
Sonstige/ Password_Text1	Bedienberechtigung für Text1 (wird nicht verwendet).
Sonstige/ Password_Text2	Bedienberechtigung für Text2 (wird nicht verwendet).
Sonstige/ Password_Text3	Bedienberechtigung für Text3 (wird nicht verwendet).
Links/Write_Variable1	Strukturelement, das mit Anwahl-Text1 beschrieben wird.
Links/Write_Variable2	Strukturelement, das mit Anwahl-Text2 beschrieben wird.
Links/Write_Variable3	Strukturelement, das mit Anwahl-Text3 beschrieben wird.

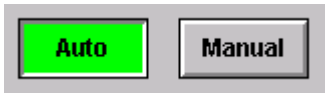
Eigenschaften	Funktion
Links/Display_Variable1	Strukturelement, das den ersten Binärzustand (Text1) anzeigt.
Links/Display_Variable2	Strukturelement, das den zweiten Binärzustand (Text2) anzeigt.
Links/Display_Variable3	Strukturelement, das den zweiten Binärzustand (Text3) anzeigt.
Parameters/ Write_value_Text1	Gibt an, welcher Wert auf das Strukturelement "Write_Variable1“ mit Anwahl-Text1 geschrieben wird ('TRUE' oder 'FALSE').
Parameters/Display_Text1_with	Gibt an, mit welchem Wert die "Display_Variable" den Text1 anzeigt.
Parameters/ Write_value_Text2	Gibt an, welcher Wert auf das Strukturelement "Write_Variable2“ mit Anwahl-Text2 geschrieben wird ('TRUE' oder 'FALSE').
Parameters/Display_Text2_with	Gibt an,mit welchem Wert ('TRUE' oder 'FALSE') die "Display_Variable" den Text2 anzeigt.
Parameters/ Write_value_Text3	Gibt an, welcher Wert auf das Strukturelement "Write_Variable3“ mit Anwahl-Text3 geschrieben wird ('TRUE' oder 'FALSE').
Parameters/Display_Text3_with	Gibt an, mit welchem Wert ('TRUE' oder 'FALSE') die "Display_Variable" den Text3 anzeigt.

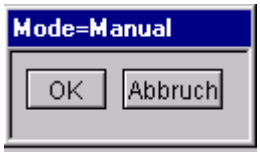
Die Texte für die Beschriftung der Optionsfelder werden aus den Parameterattributen "String_0" und "String_1" der Bausteininstanz gelesen.

Enthält der Text der Strings ein "=“, so wird der rechte Teil der Strings (nach dem "=“) für die Beschriftung der Optionsfelder und der linke Teil (vor dem "=“) für die Überschrift des Bedienbildes verwendet.

Enthält der Text der Strings kein "=“, so wird der komplette Text für die Beschriftung der Optionsfelder genommen. Die Überschrift des Bedienbildes wird in diesem Fall mit Leerzeichen gefüllt.

2.3.17 Binärwertbedienung mit Taste (Button) und Farbwechsel





@PCS7_BedBinaer.pdl

Objekttyp: PCS7_BinOp

Objektname: BinOp0 / BinOp1

Bildname: @PCS7Elements.pdl

Mit "CMD2Steps" kann festgelegt werden, ob die Binärwertbedienung eine Ein- oder Zwei-Tastenbedienung ist.

CMD2Steps = Ja
Bei Anwahl von "Auto" oder "Manual" wird über das Skript "PCS7_OpenInputBoxBin_V6.fct" das Bild "@PCS7_BedBinaer.pdl" aufgerufen. Mit "OK" in diesem Bild wird dann der über "Write_Value" festgelegte Wert an die an "Write_Variable" verbundene Variable geschrieben.

CMD2Steps = Nein
Der über "Write_Value" festgelegte Wert wird direkt beim Anklicken von STOP oder LAUF an die mit "Write_Variable" verbundene Variable geschrieben.

Die Texte für die Beschriftung der Tasten werden aus den Parameterattributen "String_0" und "String_1" der Bausteininstanz gelesen.

Enthält der Text der Strings ein "=“, so wird der rechte Teil der Strings (nach dem "=“) für die Beschriftung der Tasten verwendet und der komplette String für die Überschrift des Bedienbildes verwendet.

Enthält der Text der Strings kein "=“, so wird der komplette Text für die Beschriftung der Tasten verwendet.

Eigenschaften	Funktion
Farben/Button_Colour	Hintergrundfarbe für Text, wenn aktiv und nicht bedienbar.
Sonstige/ Bedienfreigabe	Bedienfreigabe
Sonstige/ DisplayActive	Eigenschaften Anzeige von "Button On". Darf nicht verändert werden; wird per Skript gesteuert.
Links/Write_Variable	Strukturelement, das mit Anwahl-Text beschrieben wird.
Links/Display_Variable	Strukturelement, das den ersten Binärzustand (Text) anzeigt.
Parameters/ Write_value	Gibt an, welcher Wert auf das Strukturelement "Write_Variable“ mit Anwahl-Text geschrieben wird ('TRUE' oder 'FALSE').

Eigenschaften	Funktion
Parameters/ Display_is_active_with	Gibt an, mit welchem Wert ('TRUE' oder 'FALSE') die "Display_Variable" den Text mit der ausgewählten Farbe anzeigt (Darstellung aktiv und unbedienbar).
Parameters/ButtonText	Anzeigetext für Text darf nicht verändert werden; wird per Skript gesteuert.
Parameters/ CMD2Steps	1- oder 2-Tastenbedienung

2.3.18 Binärwertbedienung mit Taste (Button)

@PCS7_BedBinär.pdl

Objekttyp: PCS7_ButtonBit

Objektname: ButtonBit

Bildname: @PCS7Elements.pdl

Gleiche Funktion wie sie in Kapitel 2.3.17 beschrieben ist, außer:

- kein Farbumschlag
- Taste bleibt von der Darstellung her immer bedienbar (sie hat keine Darstellung "Button On“).

Mit CMD2Steps kann festgelegt werden, ob die Binärwertbedienung eine Ein- oder Zwei-Tastenbedienung ist.

- CMD2Steps = Ja
  Bei Anwahl von "Reset" wird über das Skript "PCS7_openinputboxbin_V6.fct" das Bild "@PCS7_BedBinär.pdl" aufgerufen. Mit "OK" in diesem Bild wird dann der über "Write_Value" festgelegte Wert an die mit "Write_Variable" verbundene Variable geschrieben.
- CMD2Steps = Nein
  Der über "Write_Value" festgelegte Wert wird direkt beim Anklicken von "Reset" an die mit "Write_Variable" verbundene Variable geschrieben.

2.3.19 Statusanzeige mit 2 Alternativen

Objekttyp: PCS7_Status_2_Alternative

Objektname: Status_2_Alternative

Bildname: @PCS7Elements.pdl

Eigenschaften	Funktion
Links/ Link	Verschaltung auf das binäre Strukturelement, z.B. (.MSG_LOCK)
Sonstige/ Read_Text_From_AS	Beschriftung der Texte für die Statusanzeige aus dem AS ("String_0" und "String_1") lesen.
Sonstige/Anzeige	Darf nicht vom Anwender verändert werden, muss 'TRUE' bleiben.
Sonstige/ Bedienfreigabe	Bedienfreigabe
Schrift/ Text_On	Beschreibung für Zustand 'True'.
Schrift/ Text_Off	Beschreibung für Zustand 'FALSE'.
Farben/BackColor_OFF	Hintergrundfarbe für Zustand 'Off'.
Farben/BackColor_ON	Hintergrundfarbe für Zustand 'On'.
Farben/ TextColor_OFF	Textfarbe für Zustand 'Off'.
Farben/ TextColor_ON	Textfarbe für Zustand 'On'.
Displays/Off	Anzeige vom statischen Text "T_off". Darf nicht vom Anwender verändert werden, wird von Skript gesteuert.
Displays/On	Anzeige von dem statischen Text "T_on". Darf nicht vom Anwender verändert werden, wird von Skript gesteuert.

Bei Änderung der Variable die auf die Eigenschaft "Links/Link" verschaltet ist, wird das Skript "PCS7_2Stati_Variable_Changed_V6.fct" aufgerufen.

In diesem Skript wird aufgrund des Zustandes von "Links/Link" die Anzeige von "T_off" und "T_on" gesteuert.

Legen Sie den Link auf einen Eingangsparameter (z.B. AUT_ON_OP), können die Statustexte aus dem AS gelesen werden (Read_Text_From_AS = TRUE). Bei Ausgangsparametern ist meistens kein "String_0" bzw. "String_1" vorhanden und somit müssen Sie den Text direkt am Objekt (Text_On, Text_Off) hinterlegen (Read_Text_From_AS = FALSE).

2.3.20 Statusanzeige mit n Alternativen

Objekttyp: PCS7_Status_1_v_n

Objektname: Status_1_v_n

Bildname: @PCS7Elements.pdl

Eigenschaften	Funktion
Links/ Link	Verschaltung auf das binäre Strukturelement (.MSG_LOCK)
Sonstige/ Read_Text_From_AS	Beschriftung "Text_On" für die Statusanzeige aus dem AS (String_1) lesen.
Sonstige/Anzeige	Darf nicht vom Anwender verändert werden; wird vom Skript gesteuert.
Sonstige/ Bedienfreigabe	Bedienfreigabe
Schrift/ Text_On	Beschreibung für Zustand 'True'.
Farben/BackColor_ON	Hintergrundfarbe für Zustand 'On'.
Farben/ TextColor_ON	Textfarbe für Zustand 'On'
Masks/ Mask	Wert, mit dem das Textfeld sichtbar geschaltet wird.

Bei der Statusanzeige mit n Alternativen werden jeweils so viele Objekte vom Typ **Status_1_v_n** übereinander gelegt, wie Alternativen benötigt werden.

Bei Änderung der Variable, die auf die Eigenschaft "Links/Link" verschaltet ist, wird das Skript "PCS7_1vnStati_Variable_Changed_V6.fct" aufgerufen.

Über die Eigenschaft "Masks/Mask" können Sie jeweils für das Objekt einstellen, mit welchem Wert es angezeigt werden soll. Diese Möglichkeit eignet sich z.B. für eine Alternativensteuerung über einen INTEGER oder REAL.

Bei einer Alternativensteuerung über mehrere Binärwerte, stellen Sie bei allen Objekten "Mask" auf den Wert 1. Die Priorität bei gleichzeitigem Anstehen von mehreren Binärwerten, können Sie über die Eigenschaft "Ebene“ einstellen. Bei WinCC gehen die Ebenen von 0 bis 15 und haben ebenfalls aufsteigende Priorität. **Beispiel:** Wenn von zwei Objekten eines in Ebene 4 und eines in Ebene 5 liegt, und beide liegen übereinander und beide sind sichtbar geschaltet, so wird das Objekt in Ebene 5 über dem Objekt in Ebene 4 angezeigt.

Legen Sie den Link auf einen Eingangsparameter (z.B. REV_ON), kann der Statustext aus dem AS gelesen werden (Read_Text_From_AS = TRUE). Bei Ausgangsparametern ist meistens kein "String_1" vorhanden und somit müssen Sie denText direkt am Objekt (Text_On) hinterlegen (Read_Text_From_AS = FALSE).

2.3.21 Zustandsanzeige "Ventil"

Die folgende Ventil-Zustandsanzeige wurde aus PCS 7 V5 übernommen.

Für die Realisierung von Zustandsanzeigen empfiehlt sich ab PCS 7 V6 die Verwendung der "Erweiterten Zustandsanzeige“. Siehe Beschreibung WinCC Leittechnikoptionen.

Objekttyp: PCS7_Valve_Stat

Objektname: Valve_Stat1,Valve_Stat0,Valve_Stat2

Bildname: @PCS7Elements.pdl

Eigenschaften	Funktion
Sonstige/Bedienfreigabe	Bedienfreigabe
Sonstige/Anzeige	Anzeige des Anwenderobjektes.
Link/VariableLink	Verschaltung auf das Strukturelement, wenn Steuerung der Zustände über REAL oder INTEGER erfolgen soll.
State/Index	Manuelle Einstellung der Zustandsanzeige für binäre Steuerung über "State/Display".
State/Display	Steuerung der Zustände über binäre Variablen.

Das Anwenderobjekt "Valve_Stat" kann auf zwei Arten genutzt werden.

- Die erste Möglichkeit besteht darin, das Objekt als normale Zustandsanzeige zu nutzen. Dann werden die Zustände über die Eigenschaft "Link/VariableLink“ gesteuert. "State/Display" muss auf "Ja“ stehen. Es gibt die drei oben aufgeführten Zustände. Diese sind aber nur möglich, wenn die Ventilzustände in einer Variable zur Verfügung stehen, was normalerweise nicht der Fall ist.
- Die zweite Möglichkeit besteht darin, für jeden der drei Zustände ein Objekt "Valve_Stat" anzulegen und diese übereinander zu legen. Mit "State/Index" wird der anzuzeigende Zustand eingestellt. Mit "State/Display" wird das entsprechende Strukturelement verbunden. Diese Variante wird beim Bildbaustein VALVE verwendet.

2.3.22 Zustandsanzeige "Motor"

Objekttyp: Zustandsanzeige

Objektname: Motor

Bildname: @PCS7Elements.pdl

Normale Zustandsanzeige mit zwei Alternativen und der Steuerung über die Eigenschaft "Zustand/aktuellerZustand".

2.3.23 Permission

Objekttyp: Permission

Objektname: Permission_Setpoint

Bildname: @PCS7Elements.pdl

Das Objekt "Permission" ist zur Erzeugung einer Gesamt-Bedienberechtigung für ein bedienbares Element "Setpoint" vorgesehen.

Es gibt bedienbare Elemente, die mehreren verschiedenen Berechtigungsprüfungen unterliegen.

Beispiel:

- Einmal unterliegt "Setpoint" der Berechtigungsprüfung der User-Administration von WinCC, die beim Benutzerwechsel ablaufen muss (Steuerung über "@Level5" oder "@Level6").
- Desweiteren unterliegt "Setpoint" einer Variablen des AS (Q_SP_OP, Bedienung nur erlaubt, wenn Sollwert intern eingestellt wird), Überprüfung der Berechtigung bei Änderung der Variablen.

Beim FMCS_PID wird noch ein weiterer Parameter des AS (QPARF) zur Berechtigungsprüfung hinzugezogen.

Projektierung:

1. Erstellen der Direktverbindung von "@Level5“ oder “@Level6“, Eigenschaft "Bedienfreigabe“ auf Objekt "Permission“, Eigenschaft "Level_Source“.
2. C-Skript bei Änderung der Eigenschaft "Level_Source“:

```
BOOL bTag1 =!GetTagBitWait(".QPARF");
BOOL bTag2 =GetTagBitWait(".Q_SP_OP");
if (bTag1 &&  bTag2 && value)
{
  SetPropBOOL(lpszPictureName,lpszObjectName,"Target_Bedienfreigabe",TRUE);
  SetPropWord(lpszPictureName,lpszObjectName,"Target_Hintergrundfarbe",CO_WHITE);
}
else
{
   SetPropBOOL(lpszPictureName,lpszObjectName,"Target_Bedienfreigabe",FALSE);

SetPropWord(lpszPictureName,lpszObjectName,"Target_Hintergrundfarbe",CO_LTGRAY)
;
}
SetPropBOOL(lpszPictureName,lpszObjectName,"Level_Target",value);
```

Das Skript wird aufgerufen, wenn ein neuer Benutzer sich anmeldet oder bei der Bildanwahl.

Gleichzeitig müssen hier aber auch immer die Variablen des AS mit überprüft werden, die mit in die Berechtigungsprüfung eingehen. In diesem Fall "QPARF" und "Q_SP_OP".

Sind alle drei Kriterien erfüllt, wird auf die Eigenschaft "TARGET_BEDIENFREIGABE“ 'TRUE' geschrieben und auf die Eigenschaft "TARGET_Hintergrundfarbe“ die Farbe "weiß" rangiert.

Weiterhin wird auf die Eigenschaft "Level_Target“ der Wert von der Eigenschaft "Level_Source“ übertragen.

"Level_Target“ ist für das Verschalten der Direktverbindung auf weitere Objekte bestimmt.

3. C-Skript unter der Eigenschaft "Tags".

   Dieses Skript wird von den Variablen aufgerufen, die mit in die Berechtigungsprüfung eingehen (QPARF und Q_SP_OP).

```
BOOL bTag1 =!GetTagBitWait(".QPARF");
BOOL bTag2 =GetTagBitWait(".Q_SP_OP");
**  BOOL bLevel = GetPropBOOL(lpszPictureName,"@Level5","Operation");
if (bTag1 &&  bTag2 && bLevel)
{
  SetPropBOOL(lpszPictureName,lpszObjectName,"Target_Bedienfreigabe",TRUE);
  SetPropWord(lpszPictureName,lpszObjectName,"Target_Hintergrundfarbe",CO_WHITE);
}
else
{
   SetPropBOOL(lpszPictureName,lpszObjectName,"Target_Bedienfreigabe",FALSE);

SetPropWord(lpszPictureName,lpszObjectName,"Target_Hintergrundfarbe",CO_LTGRAY)
;
}
   return TRUE;
```

   Bei Änderung einer dieser Variablen werden diese aus dem AS gelesen. Weiterhin muss auch der Zustand von "@Level5" und "@Level6" überprüft werden.

   Wenn alle Kriterien erfüllt sind, wird auf die Eigenschaft "TARGET_BEDIENFREIGABE“ 'TRUE' geschrieben und auf die Eigenschaft "TARGET_Hintergrundfarbe“ die Farbe "weiß" rangiert.

   Hinweis: Abhängig davon, von welchem "@Level" aus die Direktverbindung gesteuert wird, muss diejenige auch im Skript abgefragt werden.

4. Eine Direktverbindung von der Eigenschaft "Target_Bedienfreigabe“ auf das eigentliche Objekt erstellen, das der Berechtigungsprüfung unterzogen wird, z.B. "Setpoint_AnalogValue", Eigenschaft "Bedienfreigabe“.

   An diesem Objekt darf die Bedienfreigabe nicht mehr verschaltet sein und muss auf 'FALSE' stehen.

   Die Hintergrundfarbe muss ebenfalls auf "grau" stehen und wird entweder von der Bedienfreigabe aus per Skript gesteuert oder von einer Direktverbindung mit der Eigenschaft "TARGET_Hintergrundfarbe" vom Objekt "Permission".

   Die Eigenschaft "Passwort" wird für das "Permisson"-Objekt nicht mehr verwendet.

5. Bei Bedarf eine weitere Direktverbindung von Eigenschaft "Level_Target“ auf ein weiteres "Permission"-Objekt erstellen.

2.3.24 Taste "OpenNextFaceplate"

Das Objekt "OpenNextFaceplate" ist dafür vorgesehen, aus einem geöffneten Bildbaustein einen weiteren Bildbaustein eines AS-Bausteins aus dem gleichen Plan aufzurufen.

Ein typischer Anwendungsfall ist der Aufruf des zugehörigen INTERLOK-Bausteins aus einem Motor- oder Ventil-Bildbaustein heraus.

- In der Eigenschaft "blockname“ wird der Bausteinname des aufzurufenden Bausteins eingetragen.
- In der Eigenschaft "Servername“ wird der Bausteintyp nach der üblichen Schreibweise eingetragen, z.B "PCS7 INTERLOK Control“ für den INTERLOK-Baustein.
- Mit Klick auf die Taste wird dann der Variablenname des aktuell göffneten Bildbausteins genommen, der Bausteinname abgeschnitten und der unter "blockname“ eingetragene Bausteinname angehängt.
- Weiterhin wird das Strukturelement "#Comment“, das in jedem Baustein vorkommt, angehängt und geprüft, ob sie im Datenmanager vorhanden ist.

Ist dies der Fall, so wird der neu gebildete Variablenname in die Eigenschaft "tagname“ geschrieben und der Bildbaustein mit dem Skript "PCS7_OpenGroupDisplay_V6“ aufgerufen.

Über ein Skript, welches bei Änderung der Eigenschaft "check_tag“ aufgerufen wird, wird ebenfalls geprüft, ob die Variable vorhanden ist. Die Taste wird nur bedienbar geschaltet, wenn die Variable im Datenmanager vorhanden ist.

Damit das Skript in der Eigenschaft "check_tag“ auch bei Bildanwahl durchlaufen wird, muss ein gültiges Strukturelement darauf verschaltet sein, deren Text sich vom Default-Text der Eigenschaft unterscheidet. Somit wird eine Änderung der Eigenschaft erkannt. Hierfür wird wieder das in allen Bausteinen vorhandene "#Comment“-Strukturelement verwendet.

Ein weiterer häufiger Anwendungsfall ist, dass für einen Motor oder ein Ventil mehrere INTERLOK-Bausteine verwendet werden.

Für diesen Fall kann das gleiche Objekt in den INTERLOK-Baustein eingebaut werden. In der Eigenschaft "blockname" wird für den kaskadierbaren Aufruf eine "1“ eingetragen.

Hat die momentan aufgerufene Variable (tagname) des Bildbausteins als AS-Bausteinname einen Namen ohne angehängte Zahl, wie z.B. "L“, so wird als nächstes ein Baustein mit dem AS-Bausteinname "L1“ aus dem gleichen Plan aufgerufen, wenn der Variablenname hierzu im Datenmanager vorhanden ist.

Hat der momentan aufgerufene Variablenname des Bildbausteins als AS-Bausteinname "L1“, wird als nächstes "L2“ erwartet usw.

Somit können im Prinzip beliebig viele INTERLOK-Bausteine aufgerufen werden.

Die Projektierungsvorschrift für dieses Szenario lautet:

- In einem VALVE-Bildbaustein wird eine "OpenNextFaceplate"-Taste mit dem Blocknamen "LOCK" eingebaut.
- Im INTERLOK-Baustein wird eine "OpenNextFaceplate-Tasten mit dem Blocknamen "1" eingebaut.
- INTERLOK-Bausteine, die vom Bildbausteintyp "VALVE" mit dem oben genannten Mechanismus aufgerufen werden, müssen "LOCK", "LOCK1", "LOCK2" usw. heißen.

Bedienberechtigungen:

Das Objekt "OpenNextFaceplate" hat ebenfalls die Eigenschaften "Processcontrolling_backup“ und "HigherProcesscontrolling_backup“ zum Einstellen der Bedienberechtigungen des aufzurufenden Bildbausteines. Die Default-Einstellung ist hier, wie bisher gewohnt, die Berechtigungsstufe 5 und 6. Möchten Sie die aus dem Quellsymbol eingetragenen Berechtigungen weiterleiten, so können Sie dies über die Direktverbindungen von "@Level5" und "@Level6" nach "OpenNextFaceplate" erreichen.

Direktverbindungen:

@Level5/Berechtigung/Änderung → OpenNextFaceplate/Processcontrolling

@Level6/Berechtigung/Änderung → OpenNextFaceplate/HigherProcesscontrolling

2.4 Skripte

Liste der Skripte

Die hier aufgelisteten Skripte werden im Verzeichnis "\Siemens\WinCC\aplib\FacePlateDesigner_V6" bzw. "\Siemens\WinCC\aplib\FacePlateDesigner" installiert und müssen **nicht** in den GraCS-Ordner des Projektverzeichnisses kopiert werden. Die Skripte sind nicht projektspezifisch sondern rechnerspezifisch.

Skript-Name	Funktion
PCS7_OpenGroupDisplay_V6.Fct	Öffnen eines Bildbausteins "Gruppendarstellung"
PCS7_OpenGroupDisplay_I_V6.Fct	Öffnen eines zugehörigen INTERLOK-Bausteins bei Antrieben mit Mausklick "Rechts"
PCS7_UpdateGroupTagname_V6.fct	Das Skript wird aufgerufen bei Änderung der Eigenschaft "tagname" des Objekts "@Faceplate". Die Eigenschaft "tagname“ wird bei Aufruf des Bildbausteins beschrieben.
PCS7_OpenLoopDisplay_V6.Fct	Öffnen der Loop-Darstellung, Aufruf von Taste "Loop" in "@PG_%Type%.pdl"
PCS7_UpdateLoopTagname_V6.Fct	Das Skript wird aufgerufen bei Änderung der Eigenschaft "tagname" des Objekts "@Faceplate" im Loop-Bild. Die Eigenschaft "tagname“ wird bei Aufruf des Bildbausteins beschrieben.
PCS7_CheckPermission.fct	Bedienberechtigung prüfen
PCS7_UpdatePermission_V6.Fct	Aufruf bei Änderung von "@CurrentUser" und Bildanwahl im Bild "@PG_%Type%.pdl" und im Bild "@PG_%Type%.pdl"
PCS7_ChangeView.fct	Aufruf einer anderen Sicht im Bildbaustein, Aufruf aus "@PG_%Type%_Viewlist.pdl"
PCS7_OperationLog_V6.fct	Bedienmeldung fuer WinCC
PCS7_OpenCheckbox_V6.Fct	Aufruf aus Basis-Element "CHECKBOX_L1/R1". Öffnen des Bedienbildes "@PCS7_BedCheck.pdl"
PCS7_Check_OK_V6.fct	Skript wird aufgerufen von "OK"-Taste aus "@PCS7_BedCheck.pdl"
PCS7_OpenComboBox_V6.fct	Öffnen von Bedienbild "@PCS7_BedCombo.pdl" aus dem Basis-Element "PCS7_COMBOBOX1"
PCS7_Combo_OK_V6.fct	Skript wird aufgerufen von "OK"-Taste aus "@PCS7_BedCombo.pdl"
PCS7_Open3ComboBox_V6.fct	Öffnen von Bedienbild "@PCS7_3BedCombo.pdl" aus dem Basis-Element "PCS7_3COMBOBOX1"
PCS7_3Combo_OK_V6.fct	Skript wird aufgerufen von "OK"-Taste aus "@PCS7_3BedCombo.pdl"
PCS7_OpenInputBoxBin_V6.fct	Öffnen von Bedienbild "@PCS7_BedBinaer.pdl" aus dem Basis-Element "PCS7_BinOp" und "PCS7_ButtonBit"
PCS7_Binary_OK_V6.fct	Skript wird aufgerufen von "OK"-Taste aus "@PCS7_BedBinaer.pdl"
PCS7_2Stati_Variable_Changed_V6.fct	Aufruf in Basis-Element "PCS7_Status_2_Alternative“
PCS7_1vnStati_Variable_Changed_V6.fct	Aufruf in Basis-Element "PCS7_Status_1_v_n“
PCS7_OpenInputBoxAnalog_V6.fct	Aufruf von Bedienbild "@PCS7_BedAnalog.pdl" oder

Skript-Name	Funktion
	"@PCS7_BedAnalog_NL.pdl" aus dem Basis-Element "PCS7_AnalogValue"
PCS7_Analog_OK_V6.fct	Skript wird aufgerufen von "OK"-Taste aus "@PCS7_BedAnalog.pdl" oder "@PCS7_BedAnalog_NL.pdl"
PCS7_AnalogPercent_V6 .fct	Skript wird aufgerufen bei inkrementeller Bedienung in "@PCS7_BedAnalog.pdl"
PCS7_UpdateBarLimits_V6.Fct	Skript wird von Basis-Element "PCS7_BarLimits" aufgerufen
PCS7_UpdateBar_V6.Fct	Skript wird von Basis-Element "PCS7_BarStandard1" und "PCS7_BarStandard2" aufgerufen
PCS7_Format_V6	Skript zur Zahlenformat-Übertragung
PCS7_Trend_V6.Fct	Skript für Trend-Darstellung im Bildbaustein

Die beiden Skripte "PCS7_ChangeView.fct“ und "PCS7_CheckPermission.fct“ sind unverändert aus der V5 übernommen und haben deswegen keine Endung _V6 im Namen.

2.5 Bitmaps

Die Bitmaps werden ins Verzeichnis "...\Siemens\WinCC\options\pdl\FaceplateDesigner_V6" installiert.

Bei Ablauf des OS-Projekteditors werden die Bitmaps in den "GraCS-Ordner" des Projektverzeichnisses kopiert.

Die Bitmaps werden in den entsprechenden Bildern bei Bildanwahl dynamisch nachgeladen.

Dateiname des Bitmap	Symbol-darstellung	Verwendung in	Verzeichnis
@FP_PopUpIcon.bmp		Bildbaustein global	pdl\FaceplateDesigner
@PCS7_AlarmCrossed.bmp		Bildbaustein global	pdl\FaceplateDesigner
@PCS7_AlarmDisabled.bmp		Bildbaustein global	pdl\FaceplateDesigner
@PCS7_AlarmEnabled.bmp		Bildbaustein global	pdl\FaceplateDesigner
@PCS7_NotOccupied.bmp		Bildbaustein global	pdl\FaceplateDesigner
@PCS7_Occupied.bmp		Bildbaustein global	pdl\FaceplateDesigner
@PCS7_OpenLoop.bmp		Bildbaustein global	pdl\FaceplateDesigner
@PCS7_Lock.bmp		Verwendung Bildbaustein Motor / Ventil	pdl\FaceplateDesigner
@PCS7_UnLock.bmp		Verwendung Bildbaustein Motor / Ventil	pdl\FaceplateDesigner
@CollectValue_S.emf		Bausteinsymbole global ASD	
@CollectValue_F.emf		Bausteinsymbole global ASD	
@CollectValue_transparent.		Bausteinsymbole global ASD	
@CollectValue_empty.emf		Bausteinsymbole global ASD	
@Ctrl_Manual.emf		Regler / Bausteinsymbol	
@Ctrl_intern.emf		Regler / Bausteinsymbol	
@Ctrl_extern.emf		Regler / Bausteinsymbol	
@Ctrl_Auto.emf		Regler / Bausteinsymbol	
@Ctrl_Track.emf		Regler / Bausteinsymbol	
@off.emf		Bausteinsymbol OPD	pdl\Base_Data_Poo
@on.emf		Bausteinsymbole OPD	pdl\Base_Data_Poo
@Auto.emf		Bausteinsymbole allgemein	pdl\Base_Data_Poo
@Manual.emf		Bausteinsymbole allgemein	pdl\Base_Data_Poo
MOTOR_IS_OFF.emf		Bildbaustein und Bausteinsymbol Motor	pdl\FaceplateDesigner
MOTOR_Error.emf		Bildbaustein und Bausteinsymbol Motor	pdl\FaceplateDesigner

Dateiname des Bitmap	Symbol-darstellung	Verwendung in	Verzeichnis
MOTOR_IS_ON		Bildbaustein und Bausteinsymbol Motor	pdl\FaceplateDesigner
MOTOR_OFF.emf		Bildbaustein und Bausteinsymbol Motor	pdl\FaceplateDesigner
MOTOR_ON.emf		Bildbaustein und Bausteinsymbol Motor	pdl\FaceplateDesigner
VAZ_H.emf		Bildbaustein und Bausteinsymbol Ventil	pdl\FaceplateDesigner
VAZ_H_CLOSE.emf		Bildbaustein und Bausteinsymbol Ventil	pdl\FaceplateDesigner
VAZ_H_OPEN.emf		Bildbaustein und Bausteinsymbol Ventil	pdl\FaceplateDesigner
Valve_NL.emf		Bausteinsymbol Ventil (Verriegelung)	pdl\FaceplateDesigner
Valve_L.emf		Bausteinsymbol Ventil (Verriegelung)	pdl\FaceplateDesigner
VHO_closed.emf		Bildbaustein und Bausteinsymbol Ventil	pdl\FaceplateDesigner
VHO_opened.emf		Bildbaustein und Bausteinsymbol Ventil	pdl\FaceplateDesigner
VHO_Error.emf		Bildbaustein und Bausteinsymbol Ventil	
VHO_undef.emf		Bildbaustein und Bausteinsymbol Ventil	pdl\FaceplateDesigner
VHZ_closed.emf		Bildbaustein und Bausteinsymbol Ventil	pdl\FaceplateDesigner
VHZ_opened.emf		Bildbaustein und Bausteinsymbol Ventil	pdl\FaceplateDesigner
VHZ_undef.emf		Bildbaustein und Bausteinsymbol Ventil	pdl\FaceplateDesigner
VVE_closed.emf		Bausteinsymbol Ventil	pdl\FaceplateDesigner
VVE_opened.emf		Bausteinsymbol Ventil	pdl\FaceplateDesigner
VVE_Error.emf		Bausteinsymbol Ventil	pdl\FaceplateDesigner
VVE_undef.emf		Bausteinsymbol Ventil	pdl\FaceplateDesigner
VVT_closed.emf		Bausteinsymbol Ventil	pdl\FaceplateDesigner
VVT_opened.emf		Bausteinsymbol Ventil	pdl\FaceplateDesigner
VVT_undef.emf		Bausteinsymbol Ventil	pdl\FaceplateDesigner

2.6 Bilder

Die hier aufgelisteten Bilder werden ins Verzeichnis "...\Siemens\WinCC\options\pdl\FaceplateDesigner_V6" installiert.

Bei Ablauf des OS-Projekteditors werden diese in den "GraCS-Ordner" des Projektverzeichnisses kopiert.

Bilder	
@PCS7Elements.Pdl	Vorlagenbild für Basis-Elemente
@@PCS7Typicals.pdl	Vorlagenbild von Bausteinsymbole für Technologische Hierarchie, Bausteinsymbole ergänzen.
@Template.pdl	Vorlagenbild von Bausteinsymbole für Grafics Object Update Unterschied zu "@@PCS7Typicals.pdl" nur in der Eigenschaft "type“.
@PCS7_BedAnalog.pdl	Bedienbild für Analogwerte
@PCS7_BedAnalog_NL.pdl	Bedienbild für Analogwerte ohne Grenzen, ohne Schiebebalken und ohne inkrementelle Bedienung.
@PCS7_BedBinaer.pdl	Bedienbild für Binär, 1 Punkt-Bedienung, nur Bestätigung (z. B. Auf / Zu/ Hand /Auto bei Ventil / Motor).
@PCS7_BedCheck.pdl	Bedienbild binär mit 2-Punkt-Bedienung Button (z.B. Alarm / Warnung aktiv setzen)
@PCS7_BedKombo.pdl	Bedienbild binär für 2-Punkt-Bedienung Liste (z.B. Hand/Auto bei Regler)
@PCS7_3BedKombo.pdl	Bedienbild binär für 3-Punkt-Bedienung Liste (z.B. Hand/Auto bei Regler)
@PCS7_AnalogInputwithLimits.pdl	Bedienbild für Analogwerte (wird nicht verwendet, aber wegen Kompatibilität zu V5.1 weiterhin geliefert)
@PCS7_BinaryInput1of2.pdl	Bedienbild für Binärwerte (wird nicht verwendet, aber wegen Kompatibilität zu V5.1 weiterhin geliefert)
@PCS7_ALARM.pdl	Darstellung der Melde-Sicht in den Bildbaustein mit Alarm
@PCS7_BATCH.pdl	Darstellung der Batch-Sicht in den Bildbausteinen
@PCS7_TREND.pdl	Darstellung der Trend-Sicht in den analogen Bildbausteinen
@PG_%Type%.pdl	Vorlagenbild für Prototypbild-Bildbaustein
@PG_%Type%_%View%.pdl	Vorlagenbild Bildfenster für die unterlagerten Sichten
@PG_%Type%_VIEWLIST.pdl	Vorlagenbild für Sichten-Auswahlmenü
@PG_%Type%_OverView.pdl	Vorlagenbild für Übersicht
@PL_%Type%.pdl	Vorlagenbild für Loop-Darstellung

2.7 Bildbausteine

Allgemeines

Mit welchen Parametern der AS-Bausteininstanz die einzelnen Anzeigeobjekte im Bildbaustein visualisiert werden, kann im Graphics Designer Offline-Dialog nachgeschaut werden.

Bausteinkommentar

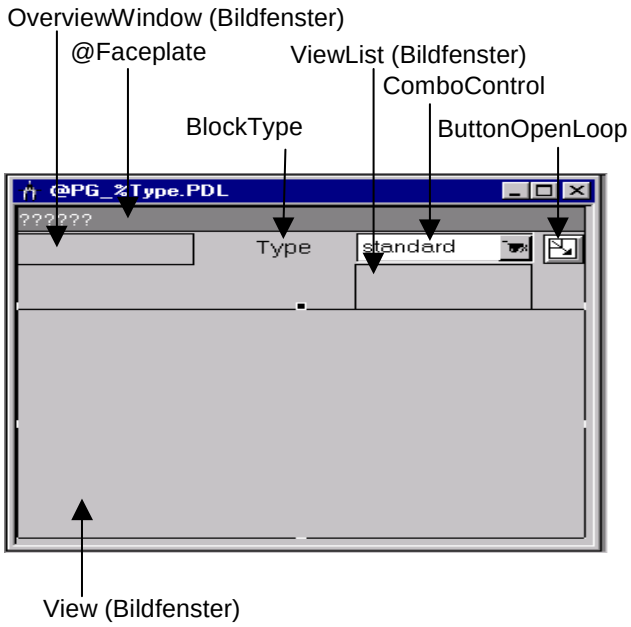
Der CFC-Bausteinkommentar wird im Bildbaustein als Kurzinformationstext (Tooltip) vom Variablennamen angezeigt.

Damit wird erreicht, dass im Bedarfsfall die Beschreibung der technologischen Funktion auch im Bildbaustein verfügbar ist.

2.7.1 Grunddaten der Vorlagenbilder

2.7.1.1 @PG_%Type%.pdl

Für die Projektierung steht die Vorlage "@PG_%Type%.pdl" zur Verfügung.



Die einzelnen Elemente haben folgende Funktionalität:

OverviewWindow

Bildfenster zur Darstellung des Alarmzustandes, Batch-Zustandes, instanzspezifischer Meldungsquittierung und Meldungsfreigabe für den Bildbaustein. Normalerweise wird hier eine Sammelanzeige eingeblendet. Wird der Bildbaustein für einen Multiinstanz-Baustein verwendet, können hier auch mehrere Sammelanzeigen angezeigt werden. Die Anzeige wird über die Projektierung des Bildes "@PG_<Type>_Overview.pdl“ festgelegt. Das gleiche Bild wird auch in der Loop-Sicht des Bildbausteins angezeigt.

@Faceplate

Dient zur Anzeige der Bausteininstanz bzw. der Variablen aus dem aufrufenden Objekt. Hier werden auch weitere Zusatzinfos für den Bildbaustein hinterlegt, wie FirstView, Name der Batch-Variablen, CurrentUser usw.

TrendFunktionen

Anwenderobjekt mit EA-Feld, für Speicherung von Daten der Trendfunktionen "Trendseite". Das Objekt ist online nicht sichtbar.

BlockType

Name des Bildbaustein-Typs. Dieser Typname geht in die Namen aller zum Bildbaustein gehörigen Sichten ein. Das Objekt ist online nicht sichtbar.

ViewList

Bildfenster zur Darstellung und Anwahl der vorhandenen Sichten.

ComboControl

Anwahl- und Anzeigeelement für die verschiedenen Sichten. Dieses Element zeigt immer den Namen der aktuellen Sicht an.

ButtonOpenLoop

Objekt zur Anwahl des Loop-Bildes. Dieses Objekt wird vom Faceplate Designer automatisch unsichtbar geschaltet, wenn dort die Generierung bzw. das Update der Loop-Sicht nicht angewählt ist.

View

Bildfenster zur Darstellung der einzelnen Sichten des Bildbausteins.

OperationWindow

Bildfenster zum Aufblenden der Analogwertbedienungen.

ComboWindow

Bildfenster zum Aufblenden der Binärwertbedienungen.

2.7.1.2 @PG_%TYPE%

Bildobjekt @PG_%TYPE%

Geometrie/Pos	X=0, y = 0
Geometrie/Größe	Breite = 320, Höhe = 260

Bildfenster "View"

Geometrie/Pos	X=1, y = 47
Geometrie/Größe	Breite = 320, Höhe = 214

Anwenderobjekt @Faceplate

Das Anwenderobjekt "@Faceplate" besteht aus folgenden Elementen:

Eigenschaft im Anwenderobjekt	Element	Typ	Vorbesetzung
Geometrie/Pos	X=0, y = 0		
Geometrie/Größe	Breite = 320, Höhe = 20		
Tagname	Tagname	Stat. Text	Keine
Tag	Tag	Stat. Text	Text = MKZ
FirstView	FirstView	EA-Feld	Ausgabewert wird vom Faceplate Designer gesetzt
CurrentUser	CurrentUser	EA-Feld	Ausgabewert = Verschaltung auf interne Variable @Current User
BName	VarBatchname	EA-Feld	Ausgabewert = .BA_NA
BATCH_ID	VarBatchID	EA-Feld	Ausgabewert = .BA_ID
STEP_NO	VarBatch Schrittnummer	EA-Feld	Ausgabewert = .STEP_NO
STEP_N1	VarBatch Schrittnummer_N1	EA-Feld	Ausgabewert = .STEP_N1
Areaname	Areaname	EA-Feld	Ausgabewert = .#areaname
Processcontrolling_backup	POP	EA-Feld	Berechtigung = Processcontrolling
HigherProcess controlling_backup	HIPOP	EA-Feld	Berechtigung = Higher Processcontrolling
MULTI_INSTANCE	HIPOP.Verdeckte Eingabe	EA-Feld	True = MultiInstanceFaceplate

2.7.1.3 @PG_%Type%_%View%.pdl

Bildobjekt @PG_%Type%_%View%

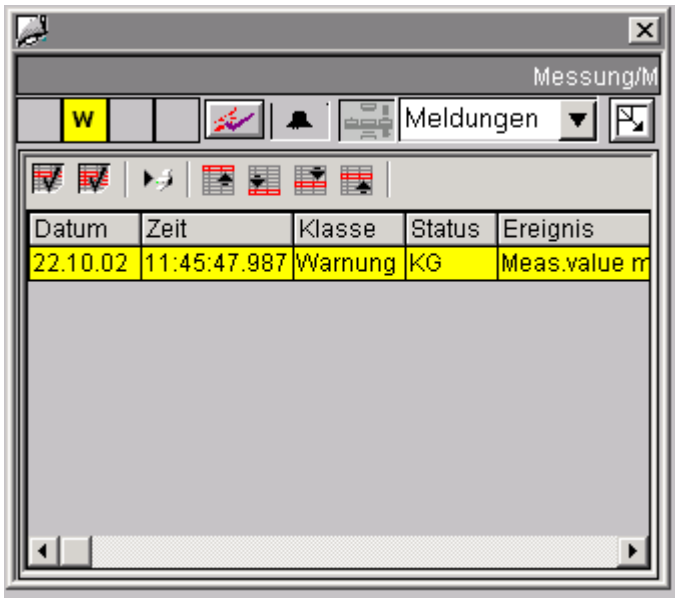
Geometrie/Pos	X=0, y = 0
Geometrie/Größe	Breite = 320, Höhe = 214

Rechteck @Frame

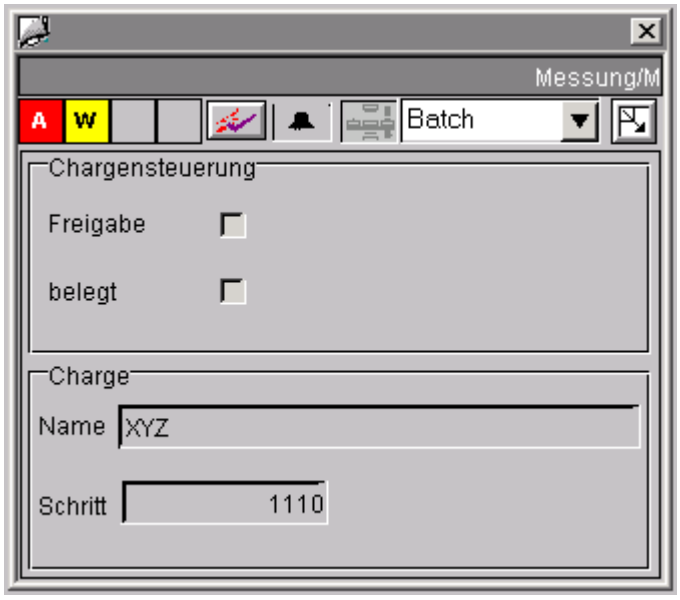
Geometrie/Pos	X=1, y = 50
Geometrie/Größe	Breite = 320, Höhe = 214

2.7.2 Globale Sichten

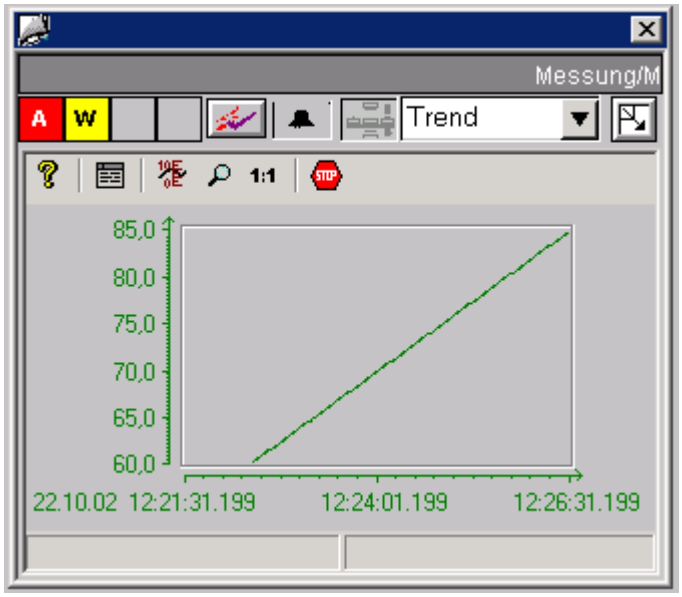
2.7.2.1 Meldesicht



2.7.2.2 Batch-Sicht



2.7.2.3 Trendsicht



Siehe dazu auch Kapitel 2.1.7, "Trendsicht projektieren".

2.7.3 CTRL_PID

Nachfolgend wird stellvertretend für alle PCS 7-Bildbausteine der Bildbaustein CTRL_PID mit seinen Sichten "Standardsicht", "Wartungssicht", "Parametersicht" und "Grenzensicht" beschrieben.

Informationen über alle Bildbausteine finden Sie in der Online-Hilfe der PCS 7 Faceplates.

2.7.3.1 CTRL_PID: Standardsicht

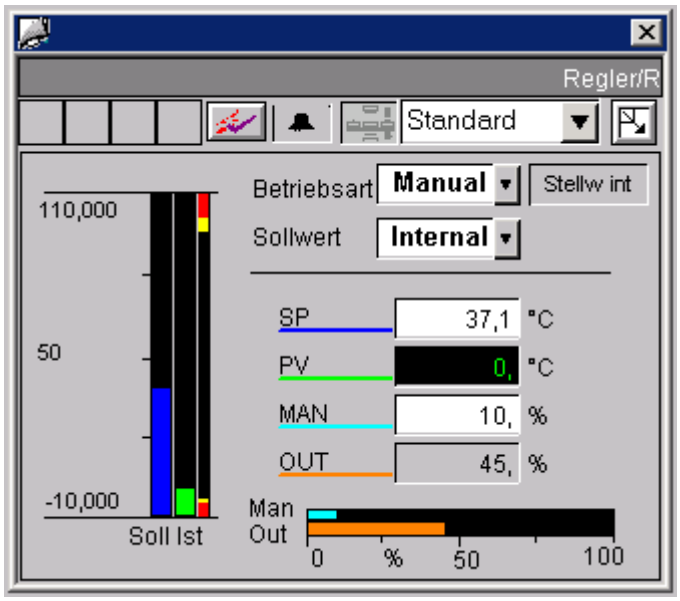
Bildbaustein-Standardsicht ab V6.0

Alle Analoganzeigen sind mit dem "AdvancedAnalogDisplay" realisiert und das Zahlenformat wird über das Bausteinsymbol (Eigenschaft "Format_InputValue“ und "Format_OutputValue“) versorgt. Siehe Kapitel 2.1.6, "Zahlenformate projektieren".

Weiterhin hat die Sicht 2 "Permission“ Objekte für Sollwert und Stellgrößeneingabe, da für diese Größen die Bedienberechtigung von verschiedenen Faktoren abhängig ist. Siehe auch Kapitel 2.3.23, Basis-Elemente, Permission-Objekt.

Das Permission-Objekt "Permission_Setpoint“ wertet außer den WinCC-Berechtigungsstufen auch den Parameter "Q_SP_OP = TRUE“ aus.

Das Permission-Objekt "Permission_Manual“ wertet außer den WinCC Berechtigungsstufen auch den Parameter "QLMNOP = TRUE“ aus.



Die Bedienung für den PID-Tuner erfolgt in der Parametersicht (Optimierung ein/aus).

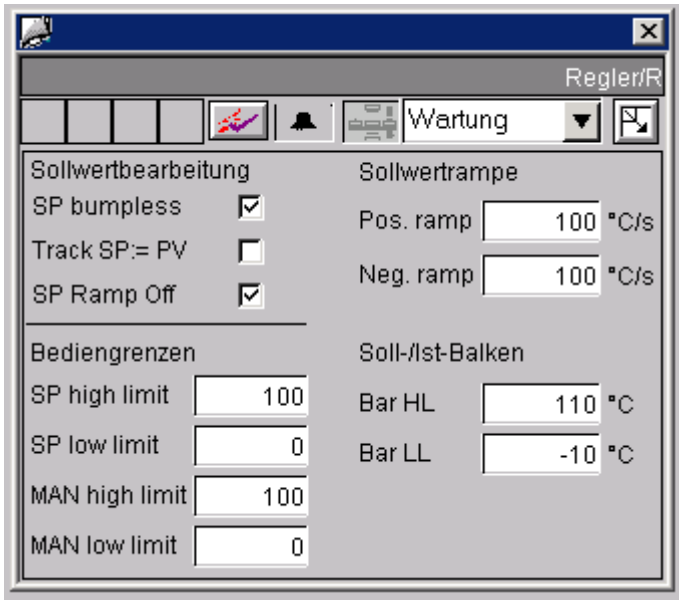
Wird die Optimierung eingeschaltet, so wird in der Standardsicht ein Kombinationsfeld über das Betriebsarten-Kombinationsfeld "Hand/Automatik" eingeblendet, mit der die Optimierung auch aus der Standardsicht wieder ausgeschaltet werden kann. Bei "Optimierung Ein“ sind alle anderen Bedienungen des Reglers gesperrt.

Reihenfolge und Rangierung von Direktverbindungen auf die bedienbaren Objekte

@Level5	->	**Bedienfreigabe**
Manual_COMBOBOX	->	Bedienfreigabe
External_COMBOBOX	->	Bedienfreigabe
Permission_Setpoint	->	Level_Source -> Level_Target
Permission_Manual	->	Level_Source
Permission_Setpoint	->	**Target_Bedienfreigabe**
Setpoint_AnalogValue	->	Bedienfreigabe
Permission_Manual	->	**Target_Bedienfreigabe**
Manual_AnalogValue	->	Bedienfreigabe
Format	->	**Format_InputValue**
Setpoint_AnalogValue	->	Format
ProcessValue_AnalogValue	->	Format
Format	->	**Format_OutputValue**
Manual_AnalogValue	->	Format
Output_AnalogValue	->	Format

2.7.3.2 CTRL_PID: Wartungssicht

Das Permission-Objekt "Permission_SP_Bumpless“ wertet außer den WinCC-Berechtigungsstufen auch den Parameter "OPTI_EN = FALSE“ aus.



Reihenfolge und Rangierung von Direktverbindungen auf die bedienbaren Objekte

@Level6	->	**Bedienfreigabe**
Permission_SP_Bumpless	->	Level_Source
Permission_SP_Bumpless	->	**Target_Bedienfreigabe**
Bumbless_CHECKBOX_L	->	Bedienfreigabe
SP_TRK_ON_CHECKBOX_L	->	Bedienfreigabe
SPRAMP_OFF_CHECKBOX_L	->	Bedienfreigabe
SPHighLimit_AnalogValue	->	Bedienfreigabe
SPLowLimit_AnalogValue	->	Bedienfreigabe
ManHighLimit_AnalogValue	->	Bedienfreigabe
ManLowLimit_AnalogValue	->	Bedienfreigabe
SPURLM_AnalogValue	->	Bedienfreigabe
SPDRLM_AnalogValue	->	Bedienfreigabe
MO_PVHR_AnalogValue	->	Bedienfreigabe
MO_PVLR_AnalogValue	->	Bedienfreigabe

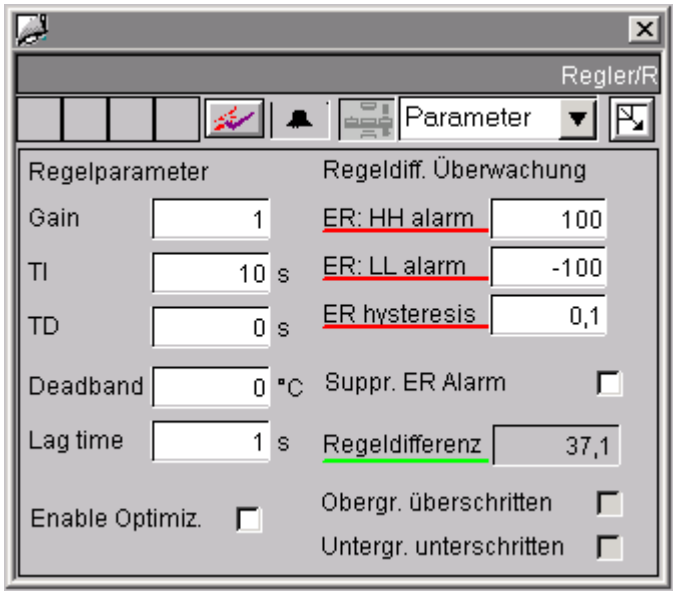
Permission_SP_Bumpless	->	**Target_Hintergrundfarbe**
SPHighLimit_AnalogValue	->	Hintergundfarbe_Value
SPLowLimit_AnalogValue	->	Hintergundfarbe_Value
ManHighLimit_AnalogValue	->	Hintergundfarbe_Value
ManLowLimit_AnalogValue	->	Hintergundfarbe_Value
SPURLM_AnalogValue	->	Hintergundfarbe_Value
SPDRLM_AnalogValue	->	Hintergundfarbe_Value
MO_PVHR_AnalogValue	->	Hintergundfarbe_Value
MO_PVLR_AnalogValue	->	Hintergundfarbe_Value

2.7.3.3 CTRL_PID: Parametersicht

Der Prozesswert "Regeldifferenz_AnalogValue“ ist mit dem "AdvancedAnalogDisplay" realisiert und das Zahlenformat wird über das Bausteinsymbol (Eigenschaft "Format_InputValue“) versorgt.

Alle anderen Analoganzeigen sind mit dem herkömmlichen EA-Feld "Gleitpunktformat“ realisiert.

Das Permission-Objekt "Permission_Gain“ wertet außer den WinCC-Berechtigungsstufen auch den Parameter "OPTI_EN = FALSE“ aus.



Reihenfolge und Rangierung von Direktverbindungen auf die bedienbaren Objekte

@Level6	->	**Bedienfreigabe**
Permission_Gain	->	Level_Source
OPTI_EN_CHECKBOX_L	->	Bedienfreigabe
Permission_Gain	->	**Target_Bedienfreigabe**
Gain_AnalogValue	->	Bedienfreigabe
TN_AnalogValue	->	Bedienfreigabe
TV_AnalogValue	->	Bedienfreigabe
DEADB_W_AnalogValue	->	Bedienfreigabe
TM_LAG_AnalogValue	->	Bedienfreigabe
ERH_ALM_AnalogValue	->	Bedienfreigabe
ERL_ALM_AnalogValue	->	Bedienfreigabe
ER_HYS_AnalogValue3	->	Bedienfreigabe
M_SUP_ER_CHECKBOX_L	->	Bedienfreigabe

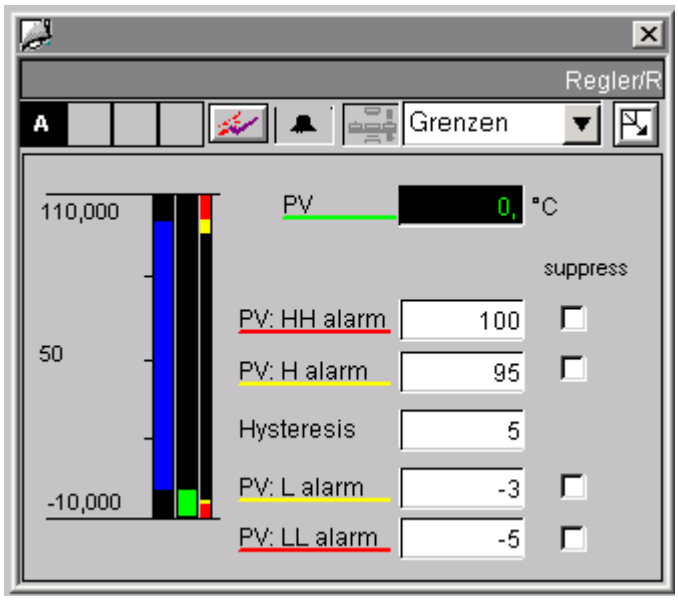
Permission_Gain	->	**Target_Hintergrundfarbe**
Gain_AnalogValue	->	Hintergundfarbe_Value
TN_AnalogValue	->	Hintergundfarbe_Value
TV_AnalogValue	->	Hintergundfarbe_Value
DEADB_W_AnalogValue	->	Hintergundfarbe_Value
TM_LAG_AnalogValue	->	Hintergundfarbe_Value
ERH_ALM_AnalogValue	->	Hintergundfarbe_Value
ERL_ALM_AnalogValue	->	Hintergundfarbe_Value
ER_HYS_AnalogValue3	->	Hintergundfarbe_Value
Format	->	**Format_InputValue**
Regeldifferenz_AnalogValuem	->	Format

2.7.3.4 CTRL_PID: Grenzensicht

Der Prozesswert "ProcessValue_AnalogValue“ ist mit dem "AdvancedAnalogDisplay" realisiert und das Zahlenformat wird über das Bausteinsymbol (Eigenschaft "Format_InputValue“) versorgt.

Alle anderen Analoganzeigen sind mit dem herkömmlichen EA-Feld "Gleitpunktformat“ realisiert.

Das Permission-Objekt "Permission_AlarmHigh_AnalogValue“ wertet außer den WinCC-Berechtigungsstufen auch den Parameter "OPTI_EN = FALSE“ aus.



Der Sollwert-Balken zeigt hier die Bediengrenzen für den Sollwert an, bezogen auf die Balkengrenzen.

Die Einstellung der Bediengrenzen des Sollwertes erfolgt in der Wartungssicht.

Reihenfolge und Rangierung von Direktverbindungen auf die bedienbaren Objekte

@Level6	->	**Bedienfreigabe**
Permission_AlarmHigh_AnalogValue	->	Level_Source
Permission_AlarmHigh_AnalogValue	**->**	**Target_Bedienfreigabe**
AlarmHigh_AnalogValue	->	Bedienfreigabe
WarningHigh_AnalogValue	->	Bedienfreigabe
Hysterese_AnalogValue	->	Bedienfreigabe
WarningLow_AnalogValue	->	Bedienfreigabe
AlarmLow_AnalogValue	->	Bedienfreigabe
AlarmHigh_CHECKBOX_R	->	Bedienfreigabe
WarningHigh_CHECKBOX_R	->	Bedienfreigabe
WarningLow_CHECKBOX_R	->	Bedienfreigabe
AlarmLow_CHECKBOX_R	->	Bedienfreigabe
Permission_AlarmHigh_AnalogValue	->	**Target_Hintergrundfarbe**
AlarmHigh_AnalogValue	->	Hintergrundfarbe_Value
WarningHigh_AnalogValue	->	Hintergrundfarbe_Value
Hysterese_AnalogValue	->	Hintergrundfarbe_Value
WarningLow_AnalogValue	->	Hintergrundfarbe_Value
AlarmLow_AnalogValue	->	Hintergrundfarbe_Value
Format	->	**Format_InputValue**
ProcessValue_AnalogValue	->	Format

2.8 Bausteinsymbole

Hinweis: Bei den Bausteinsymbolen sind keine Prozessbedienungen vorgesehen. Alle Prozessbedienung erfolgen aus den Bildbausteinen.

2.8.1 Vorlagenbilder @@PCS7Typicals.pdl und @Template.pdl

Die bisher ausgelieferten Bausteinsymbole (V5.x) in dem Bild @@**PCS7Typicals.pdl und @Template.pdl** können alle Varianten von Bildbausteinen (OCX oder Bildbaustein V5.1 / V5.2 / V6.0) im entsprechenden Prototypbild öffnen.

Die neuen Funktionen stehen allerdings nur zur Verfügung, wenn die neuen Bausteinsymbole verwendet werden.

Die neuen Bausteinsymbole sind im Bild @@**PCS7Typicals.pdl** und @**Template.pdl** abgelegt.

Das Bild @@**PCS7Typicals.pdl** wird für das automatische Anlegen von Bausteinsymbolen aus der TH benutzt.

Ist in der Technologischen Hierarchie (TH) ein Bild enthalten, bei dem die Option "Bausteinsymbole aus der TH ableiten“ gesetzt ist, so werden für alle OS-relevanten CFC-Bausteine in den Plänen dieses Hierarchieordners und je nach Einstellung die unterlagerten Ordnern die Bausteinsymbole in diesem Bild angelegt, wenn Sie

- in der TH den Menübefehl "Bausteinsymbole erstellen/ändern" verwenden

  oder

- beim "OS übersetzen" die entsprechende Option im Assistenten eingeschaltet haben.

Hierbei gilt:

Für eine CFC-Bausteininstanz mit dem symbolischen Typ-Namen CTRL_PID wird in diesem Bild eine Kopie von einem Bausteinsymbol aus dem Bild @@**PCS7Typicals.pdl** angelegt, an dessen Eigenschaft "type“ der String "@CTRL_PID/1“ eingetragen ist.

Wenn Sie das Bild "@@PCS7Typicals.pdl" ändern möchten, müssen Sie es unter dem Namen "@PCS7Typicals.pdl" abspeichern und dieses verändern. Es wird beim Ableiten aus der TH automatisch das Bild "@PCS7Typicals.pdl" verwendet, falls dieses im Projekt vorhanden ist.

Hinweis

Beim automatischen Erzeugen werden alle Bausteinsymbole in dem Bild gelöscht, die auch in "@@PCS7Typicals.pdl" vorkommen, aber nicht über die TH generiert wurden. Deshalb ist als Vorlage für Bausteinsymbole bei händischer Projektierung und Nachbearbeitung in solchen Bildern, zwingend das Bild "@Template.pdl" zu verwenden, da hier die Eigenschaft "type" anders vorbelegt ist.

Neu ab V6 ist, dass an einer CFC-Bausteininstanz diese Referenz nun projektierbar ist und keine zwingende Namenskonvention an der Eigenschaft "type“ besteht. Desweiteren können im ES für einen Bausteintyp mehrere verschiedene Bausteinsymbole erzeugt werden.

Beispiel: Wird an einer CTRL_PID-Instanz als Symbolnamen "XXX“ eingetragen, so wird in dem Bild @@**PCS7Typicals.pdl** nach einem Bausteinsymbol referenziert, welches in der Eigenschaft "type“ den String "@CTRL_PID/XXX“ eingetragen hat.

Das Bild @**Template.pdl** gilt vor allem als Vorlage für händische Projektierung von Bausteinsymbolen in WinCC-Bildern. Der Unterschied der Bausteinsymbole in diesen beiden Bildern liegt lediglich in der Eigenschaft "**type**“, die in dem Bild "@@PCS7Typicals.pdl" nicht verändert werden darf (Namenskonvention z.B. @MEAS_MON/1), da dies die Referenz zur Bestimmung der Objekte ist, welche bei der Generierung über die TH erzeugt und gelöscht werden.

In @**Template.pdl** darf diese Eigenschaft verändert werden.

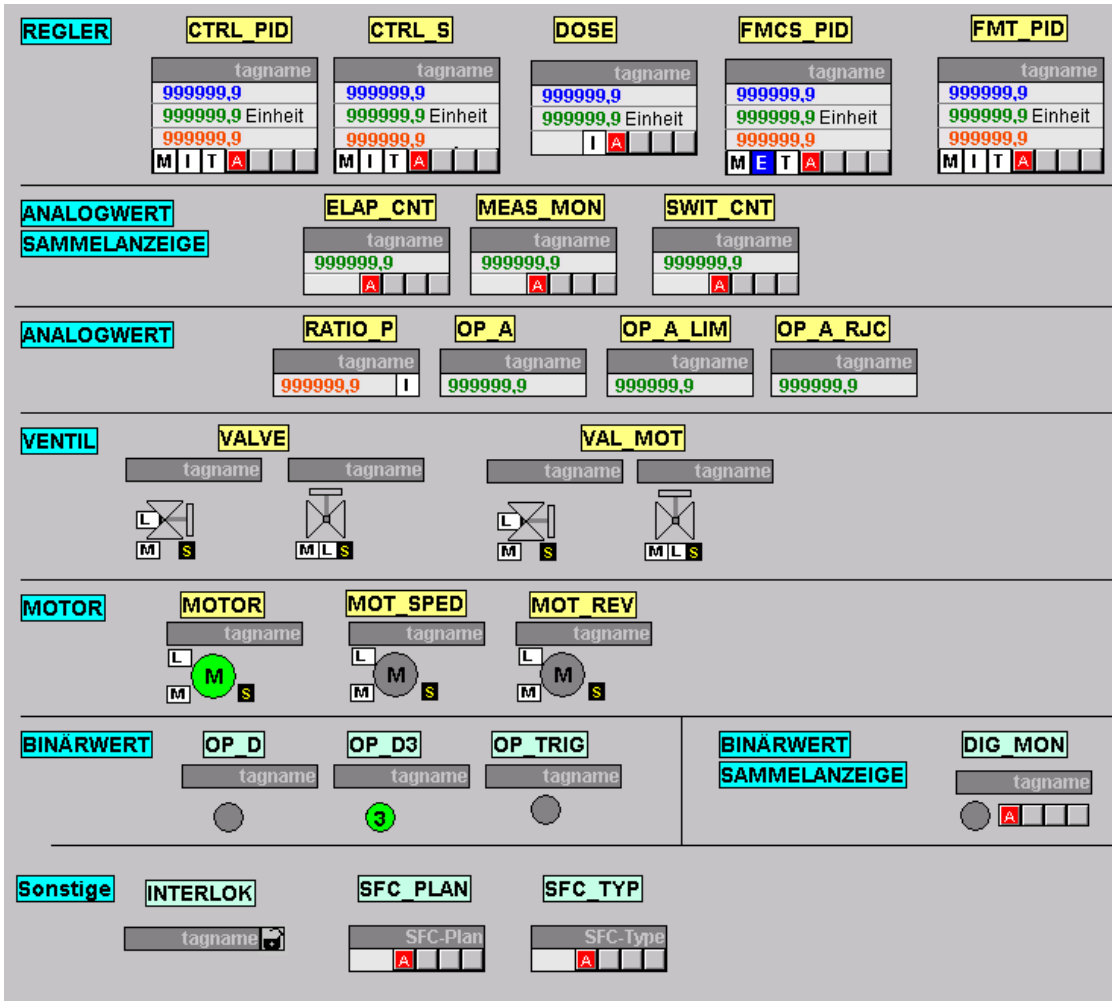
Die Eigenschaft sollten Sie aber niemals so benennen, wie sie bei den Bausteinsymbolen in @@PCS7Typicals bereits bestehen, da sonst die Gefahr besteht, dass Bausteinsymbole, die aus dieser Vorlage kopiert wurden, in den Bildern gelöscht werden, die über die TH generiert werden.

Besteht der Bedarf, die vorhandenen Symbole zu verändern, so ist es sinnvoll, nicht das Bild "@Template.pdl" zu ändern, sondern dieses unter einen anderen Namen abzuspeichen und dann zu ändern. Das Bild wird sonst beim OS-Projekteditor (früher Split Screen Wizard) wieder zurückgesetzt.

Für den Wizard "**Aktualisieren der Bildobjekte**“ kann sowohl das Bild @@**PCS7Typicals.pdl** als auch das Bild @**Template.pdl** verwendet werden.

Auch hier gilt die Eigenschaft "type“ als Referenz, welche Objekte ausgetauscht werden sollen.

2.8.2 Bausteinsymbole im Bild @@PCS7_Typicals



2.8.3 Eigenschaften der Bausteinsymbole

2.8.3.1 Allgemeine Eigenschaften

Folgende Eigenschaften sollten grundsätzlich bei allen Bausteinsymbolen aus dem Bild "@@PCS7Typicals" nicht verändert werden:

- Geometrie/Breite
- Geometrie/Höhe
- Sonstige/Bedienfreigabe
- Sonstige/Passwort
- Sonstige/Anzeige
- General/Servername
- Styles/Sammelrelevant (nur bei Bausteinen mit Alarm_8P-Meldungen)

Folgende Eigenschaften sind bei allen Bausteinsymbolen vorhanden:

Eigenschaften	**Element und Eigenschaft im Anwenderobjekt**	**Objekt**	**Beschreibung**
Sonstige / Processcontrolling_backup	POP.Berechtigung	EA-Feld	Instanzspez. Bedienberechtigung, Default = 5
Sonstige / HigherProcesscontrolling_backup	HIPOP.Berechtigung	EA-Feld	Instanzspez. Bedienberechtigung, Default = 6
General / tag	NameOfTag.Ausgabewert	EA-Feld	Angezeigter Text im Symbol
General / type	Type.Ausgabewert	EA-Feld	Refererenz für Symbolgenerierung aus der TH und für Wizards
General / tagname	Tagname.Ausgabewert	EA-Feld	Tatsächlicher Variablenname, welcher an die Variablen-Prefixe der Bildfenster weitergeleitet wird
General / Servername	Servername.Ausgabewert	EA-Feld	Bausteintyp bzw. Bildbausteintyp
General / Version	Version.Ausgabewert	EA-Feld	Versionsnummer
Styles / View_Tag	NameOfTag.Anzeige Rechteck17.Anzeige (wenn vorhanden)	Rechteck EA-Feld	Hier kann die Anzeige des Variablennamens ausgeblendet werden
MouseClick links	PCS7_OpenGroupDisplay_V6 (lpszPictureName, lpszObjectName)		Aufruf des Bildbausteins

2.8.3.2 CTRL_PID

Siehe auch Kapitel 2.8.3.1 "Allgemeine Eigenschaften"

Eigenschaften	**Element und Eigenschaft im Anwenderobjekt**	**Objekt**	**Beschreibung**
Geometrie	Breite = 110 / Höhe = 77		
General / UnitPV	UnitPV.Text	Stat.Text	Anzeige: Einheit PV
General / Unit_MAN_OP	Unit_MAN_OP.Text	Stat.Text	Anzeige: Einheit MAN_OP
Links / CollectValue	Sammelanzeige.Sammelwert	Sammelanzeige.	.EventState
Links / SetpointValue	SetpointValue_Analoganzeige Erweitert.Wert	AdvancedAnalogDis.	Anzeige: Sollwert
Links / ProcessValue	ProcessValue_Analoganzeige Erweitert.Wert	AdvancedAnalogDis.	Anzeige: Istwert
Links / OutputValue	OutputValue_Analoganzeige Erweitert.Wert	AdvancedAnalogDis.	Anzeige: Stellgröße
Links / LMN_SEL	Tracking_Zustandsanzeige Erweitert.Status	AdvancedStatusDis.	Anzeige: Stellwert nachführen
Links / Mode_MAN_AUT	Manual_Zustandsanzeige Erweitert.Status	AdvancedStatusDis.	Anzeige: Hand/ Automatik
Links / Mode_INT_EXT	External_Zustandsanzeige Erweitert.Status	AdvancedStatusDis.	Anzeige: Extern/ Intern
Styles / ReturnPath	TrendFunktionen2 .Ausgabewert	EA-Feld	Siehe Kapitel 2.1.7, "Trendsicht projektieren"
Styles / StandardTrend	TrendFunktionen2 .Zeichensatzgröße	EA-Feld	Siehe Kapitel 2.1.7, "Trendsicht projektieren"
Styles / Format_InputValue	ProcessValue_Analoganzeige Erweitert.Format SetpointValue_Analoganzeige Erweitert.Format	AdvancedAnalogDis. AdvancedAnalogDis.	Zahlenformatierung für Istwert und Sollwert
Styles/ Format_OutputValue	OutputValue_Analoganzeige Erweitert.Format	AdvancedAnalogDis.	Zahlenformatierung für Stellwert
Styles / Format_xx	Format_xx.Ausgabewert	EA-Feld	Weiteres Format, siehe Kap. 2.1.6, "Zahlenformate projektieren"

2.8.3.3 CTRL_S

Siehe auch Kapitel 2.8.3.1 "Allgemeine Eigenschaften"

Eigenschaften	Element und Eigenschaft im Anwenderobjekt	Objekt	Beschreibung
Geometrie	Breite = 110 / Höhe = 77		
General / UnitPV	Unit.Text / .PV_IN#unit	Stat. Text	Anzeige: Einheit PV
General / Unit_MAN_OP	Unit.Text / .MAN_OP#unit	Stat. Text	Anzeige: Einheit Stellgröße
Links / CollectValue	Sammelanzeige.Sammelwert /	Sammelanzeige	.EventState
Links / SetpointValue	SetpointValue_Analoganzeige Erweitert.Wert	AdvancedAnalogDis.	Anzeige: Sollwert
Links / ProcessValue	ProcessValue_Analoganzeige Erweitert.Wert	AdvancedAnalogDis.	Anzeige: Istwert
Links / OutputValue	OutputValue_Analoganzeige Erweitert.Wert	AdvancedAnalogDis.	Anzeige: Stellgröße
Links / Mode_MAN_AUT	Manual_Zustandsanzeige Erweitert.Status	AdvancedStatusDis.	Anzeige: Hand/ Automatik
Links / Mode_INT_EXT	External_Zustandsanzeige Erweitert.Status	AdvancedStatusDis.	Anzeige: Extern/ Intern
Links /LMN_SEL	Tracking_Zustandsanzeige Erweitert.Status	AdvancedStatusDis.	Anzeige: Stellwert nachführen
Links /QLMNR_ON	OutputValue_Analoganzeige Erweitert.Anzeige Unit_MAN_OP.Anzeige	AdvancedAnalogDis. Stat. Text	Beschr. Siehe unten
Links /QLMNUP	LMNUP_StatusAnzeige	Stat. Text	Anzeige: QLMNUP
Links /QLMNDN	LMNDN_StatusAnzeige	Stat. Text	Anzeige: QLMNDN
Styles / ReturnPath	TrendFunktionen2 .Ausgabewert	EA-Feld	Siehe Kapitel 2.1.7, "Trendsicht projektieren"
Styles / StandardTrend	TrendFunktionen2 .Zeichensatzgröße	EA-Feld	Siehe Kapitel 2.1.7, "Trendsicht projektieren"
Styles / Format_InputValue	ProcessValue_Analoganzeige Erweitert.Format SetpointValue_Analoganzeige Erweitert.Format	AdvancedAnalogDis. AdvancedAnalogDis.	Zahlenformatierung für Istwert und Sollwert
Styles/ Format_OutputValue	OutputValue_Analoganzeige Erweitert.Format	AdvancedAnalogDis.	Zahlenformatierung für Stellwert
Styles / Format_xx	Format_xx.Ausgabewert	EA-Feld	Weiteres Format, siehe Kap. 2.1.6, "Zahlenformate projektieren"

Das Bausteinsymbol vom CTRL_S unterscheidet sich gegenüber dem CTRL_PID dahingehend, dass bei nicht vorhandener Stellungsrückmeldung (LMNR_ON = 0) anstatt der Stellgröße die binären Ansteuersignale QLMNUP und QLMNDN angezeigt werden.

Die Sichtbarkeit dieser Texte wird auch über Skripte gesteuert, die bei Änderung der Eigenschaften QLMNUP und QLMNDN aufgerufen werden.

Hinweis: Die Objekte "OutputValue_AnaloganzeigeErweitert“ und "Unit_MAN_OP“ müssen im Anwenderobjekt zwingend im Vordergrund liegen, damit die Sichtbarkeitssteuerung korrekt funktioniert.

2.8.3.4 DOSE

Siehe auch Kapitel 2.8.3.1 "Allgemeine Eigenschaften"

Eigenschaften	Element und Eigenschaft im Anwenderobjekt	Objekt	Beschreibung
Geometrie	Breite = 110 / Höhe = 63		
General / UnitPV	UnitPV.Text	Stat. Text	Anzeige: Einheit PV
Links / CollectValue	Sammelanzeige.Sammelwert	Sammelanzeige	.EventState
Links / ProcessValue	ProcessValue_Analoganzeige Erweitert.Wert	AdvancedAnalogDis.	Anzeige: Istwert
Links / SetpointValue	SetpointValue_Analoganzeige Erweitert.Wert	AdvancedAnalogDis.	Anzeige: Sollwert
Links / SetpointExtern	External_Zustandsanzeige Erweitert.Status SetpointExternValue_ AnaloganzeigeErweitert .Anzeige	AdvancedStatusDis AdvancedAnalogDis	.QSPEXTON Beschreibung siehe unten
Links / WertSetpointExtern	SetpointExternValue_ AnaloganzeigeErweitert.Wert	AdvancedAnalogDis	Wird bei .QSPEXTON über den Sollwert eingeblendet
Styles / ReturnPath	TrendFunktionen2 .Ausgabewert	EA-Feld	Siehe Kapitel 2.1.7, "Trendsicht projektieren"
Styles / StandardTrend	TrendFunktionen2 .Zeichensatzgröße	EA-Feld	Siehe Kapitel 2.1.7, "Trendsicht projektieren"
Styles / Format_InputValue	ProcessValue_Analoganzeige Erweitert.Format SetpointValue_Analoganzeige Erweitert.Format	AdvancedAnalogDis. AdvancedAnalogDis.	Zahlenformatierung für Istwert und Sollwert
Styles/ Format_OutputValue	OutputValue_Analoganzeige Erweitert.Format	AdvancedAnalogDis.	Zahlenformatierung für Stellwert
Styles / Format_xx	Format_xx.Ausgabewert	EA-Feld	Weiteres Format, siehe Kap. 2.1.6, "Zahlenformate projektieren"

Da es beim DOSE-Baustein keinen Parameter gibt, der den wirksamen Sollwert repräsentiert, wird in Abhängigkeit von QSPEXTON die Sollwertanzeige eingeblendet.

QSPEXTON = 0 → "SetpointValue_AnaloganzeigeErweitert" wird eingeblendet

QSPEXTON = 1 → "SetpointExternValue_AnaloganzeigeErweitert" wird eingeblendet

2.8.3.5 FMCS_PID / FMT_PID

Siehe auch Kapitel 2.8.3.1 "Allgemeine Eigenschaften"

Eigenschaften	Element und Eigenschaft im Anwenderobjekt	Objekt	Beschreibung
Geometrie	Breite = 110 / Höhe = 77		
General / UnitPV	Unit.Text / .PV#unit	Stat. Text	Anzeige: Einheit PV
General / Unit_MAN_OP	Unit.Text / .LMN#unit	Stat. Text	Anzeige: Einheit Stellgröße
Links / CollectValue	Sammelanzeige.Sammelwert /	Sammelanzeige	.EventState
Links / SetpointValue	SetpointValue_Analoganzeige Erweitert.Wert / .SP	AdvancedAnalogDis.	Anzeige: Sollwert
Links / ProcessValue	ProcessValue_Analoganzeige Erweitert.Wert / .PV	AdvancedAnalogDis.	Anzeige: Istwert
Links / OutputValue	OutputValue_Analoganzeige Erweitert.Wert / .LMN	AdvancedAnalogDis.	Anzeige: Stellgröße
Links / Tracking	Tracking_Zustandsanzeige Erweitert.Status	AdvancedStatusDis.	Anzeige: Tracking LMN
Links / Mode_MAN_AUT	Manual_Zustandsanzeige Erweitert.Status	AdvancedStatusDis.	Anzeige: Hand/ Automatik
Links / Mode_INT_EXT	External_Zustandsanzeige Erweitert.Status	AdvancedStatusDis.	Anzeige: Extern/ Intern
Styles / ReturnPath	TrendFunktionen2 .Ausgabewert	EA-Feld	Siehe Kapitel 2.1.7, "Trendsicht projektieren"
Styles / StandardTrend	TrendFunktionen2 .Zeichensatzgröße	EA-Feld	Siehe Kapitel 2.1.7, "Trendsicht projektieren"
Styles / Format_InputValue	ProcessValue_Analoganzeige Erweitert.Format SetpointValue_Analoganzeige Erweitert.Format	AdvancedAnalogDis. AdvancedAnalogDis.	Zahlenformatierung für Istwert und Sollwert
Styles/ Format_OutputValue	OutputValue_Analoganzeige Erweitert.Format	AdvancedAnalogDis.	Zahlenformatierung für Stellwert
Styles / Format_xx	Format_xx.Ausgabewert	EA-Feld	Weiteres Format, siehe Kap. 2.1.6, "Zahlenformate projektieren"

2.8.3.6 ELAP_CNT

Siehe auch Kapitel 2.8.3.1 "Allgemeine Eigenschaften"

Eigenschaften	Element und Eigenschaft im Anwenderobjekt	Objekt	Beschreibung
Geometrie	Breite = 97 / Höhe = 45		
General / Unit	Unit.Text	Stat. Text	Anzeige: Einheit
Links / CollectValue	Sammelanzeige.Sammelwert	Sammelanzeige	.EventState
Links / Output_Value	ProcessValue_Analoganzeige Erweitert.Wert	AdvancedAnalogDis.	.HOURS Anzeige max. 7 Stellen
Styles / Format_InputValue	ProcessValue_Analogan-zeigeErweitert.Format	AdvancedAnalogDis.	Zahlenformatierung für Istwert und Sollwert
Styles/ Format_OutputValue	Format_OutputValue. Ausgabewert	EA-Feld	Zahlenformatierung für Stellwert
Styles / Format_xx	Format_xx.Ausgabewert	EA-Feld	Weiteres Format, siehe Kap. 2.1.6, "Zahlenformate projektieren"

2.8.3.7 MEAS_MON

Siehe auch Kapitel 2.8.3.1 "Allgemeine Eigenschaften"

Eigenschaften	Element und Eigenschaft im Anwenderobjekt	Objekt	Beschreibung
Geometrie	Breite = 97 / Höhe = 45		
General / Unit	Unit.Text	Stat. Text	Anzeige: Einheit
Links / CollectValue	Sammelanzeige.Sammelwert	Sammelanzeige	.EventState
Links / OutputValue	ProcessValue_Analoganzeige Erweitert.Wert	AdvancedAnalogDis.	Anzeige: Istwert
Styles / ReturnPath	TrendFunktionen2 .Ausgabewert	EA-Feld	Siehe Kapitel 2.1.7, "Trendsicht projektieren"
Styles / StandardTrend	TrendFunktionen2 .Zeichensatzgröße	EA-Feld	Siehe Kapitel 2.1.7, "Trendsicht projektieren"
Styles / Format_InputValue	ProcessValue_Analoganzeige Erweitert.Format	AdvancedAnalogDis.	Zahlenformatierung für Istwert und Sollwert
Styles/ Format_OutputValue	Format_OutputValue. Ausgabewert	EA-Feld.	Zahlenformatierung für Stellwert
Styles / Format_xx	Format_xx.Ausgabewert	EA-Feld	Weiteres Format, siehe Kap. 2.1.6, "Zahlenformate projektieren"

2.8.3.8 SWIT_CNT

Siehe auch Kapitel 2.8.3.1 "Allgemeine Eigenschaften"

Eigenschaften	Element und Eigenschaft im Anwenderobjekt	Objekt	Beschreibung
Geometrie	Breite = 97 / Höhe = 45		
General / Unit	Unit.Text	Stat. Text	.V#UNIT
Links / CollectValue	Sammelanzeige.Sammelwert	Sammelanzeige	.EventState
Links / OutputValue	ProcessValue_Analoganzeige Erweitert.Wert	AdvancedAnalogDis.	Anzeige: Istwert
Styles / Format_InputValue	ProcessValue_Analoganzeige Erweitert.Format	AdvancedAnalogDis.	Zahlenformatierung für Istwert und Sollwert
Styles/ Format_OutputValue	Format_OutputValue .Ausgabewert	EA-Feld.	Zahlenformatierung für Stellwert
Styles / Format_xx	Format_xx.Ausgabewert	EA-Feld	Weiteres Format, siehe Kap. 2.1.6, "Zahlenformate projektieren"

2.8.3.9 RATIO_P

Siehe auch Kapitel 2.8.3.1 "Allgemeine Eigenschaften"

Eigenschaften	Element und Eigenschaft im Anwenderobjekt	Objekt	Beschreibung
Geometrie	Breite = 97 / Höhe = 32		
General / Unit	Unit.Text	Stat. Text	Anzeige: Einheit
Links / OutputValue	ProcessValue_Analoganzeige Erweitert.Wert	AdvancedAnalogDis.	Anzeige: Istwert
Links / Mode_INT_EXT	External_Zustandsanzeige Erweitert.Status	AdvancedStatusDis.	Anzeige: Extern/ Intern
Styles / Format_InputValue	ProcessValue_Analoganzeige Erweitert.Format	AdvancedAnalogDis.	Zahlenformatierung für Istwert und Sollwert
Styles/ Format_OutputValue	Format_OutputValue .Ausgabewert	EA-Feld.	Zahlenformatierung für Stellwert
Styles / Format_xx	Format_xx.Ausgabewert	EA-Feld	Weiteres Format, siehe Kap. 2.1.6, "Zahlenformate projektieren"

2.8.3.10 OP_A

Siehe auch Kapitel 2.8.3.1 "Allgemeine Eigenschaften"

Eigenschaften	Element und Eigenschaft im Anwenderobjekt	Objekt	Beschreibung
Geometrie	Breite = 97 / Höhe = 32		
General / Unit	Unit.Text	Stat. Text	Anzeige: Einheit
Links / OutputValue	ProcessValue_Analoganzeige Erweitert.Wert	AdvancedAnalogDis.	Anzeige: Istwert
Styles / Format_InputValue	ProcessValue_Analoganzeige Erweitert.Format	AdvancedAnalogDis.	Zahlenformatierung für Istwert und Sollwert
Styles / Format_OutputValue	Format_OutputValue. Ausgabewert	EA-Feld.	Zahlenformatierung für Stellwert
Styles / Format_xx	Format_xx.Ausgabewert	EA-Feld	Weiteres Format, siehe Kap. 2.1.6, "Zahlenformate projektieren"

2.8.3.11 OP_A_LIM

Eigenschaften und Darstellung wie OP_A. Siehe Kapitel 2.8.3.10, OP_A

2.8.3.12 OP__A_RJC

Eigenschaften und Darstellung wie OP_A. Siehe Kapitel 2.8.3.10, OP_A

2.8.3.13 VALVE

Siehe auch Kapitel 2.8.3.1 "Allgemeine Eigenschaften"

Eigenschaften	Element und Eigenschaft im Anwenderobjekt	Objekt	Beschreibung
Geometrie	Breite = 90 / Höhe = 67		
Links / CollectValue	Sammelanzeige_mitASD .Sammelwert	AdvancedStatusDis.	.EventState
Links/QMAN_AUT	Mode.Status1	AdvancedStatusDis.	Anzeige: Auto/Manual
Links / V_LOCK	Interlock.Status1	AdvancedStatusDis.	Anzeige: Lock
Links / QOPENED	Valve_Status.Status1	AdvancedStatusDis.	Anzeige: Ventil
Links / QCLOSED	Valve_Status.Status2	AdvancedStatusDis.	Anzeige: Ventil
Links / QOPENING	Valve_Status.Status3	AdvancedStatusDis.	Anzeige: Ventil
Links / QCLOSING	Valve_Status.Status4	AdvancedStatusDis.	Anzeige: Ventil

Beim Mausklick links wird der VALVE-Bildbaustein und beim Mausklick rechts wird der zugehörige INTERLOK-Bildbaustein aufgerufen.

Der Bausteinname des INTERLOK-Bausteins ist als Skript-Übergabeparameter hinterlegt, siehe Kapitel 2.4 "Skripte"

Der Default des Bausteinnamens ist "L“. Der INTERLOK-Baustein muss im gleichen CFC-Plan wie der VALVE platziert sein.

2.8.3.14 VAL_MOT

Eigenschaften und Darstellung wie VALVE. Siehe Kapitel 2.8.3.13, VALVE

2.8.3.15 MOTOR

Siehe auch Kapitel 2.8.3.1 "Allgemeine Eigenschaften"

Eigenschaften	Element und Eigenschaft im Anwenderobjekt	Objekt	Beschreibung
Geometrie	Breite = 90 / Höhe = 54		
Links / CollectValue	Sammelanzeige_mitASD .Sammelwert	AdvancedStatusDis.	.EventState
Links/QMAN_AUT	Mode.Status1	AdvancedStatusDis.	Anzeige: Auto/Manual
Links / LOCK	Interlock.Status1	AdvancedStatusDis.	Anzeige: Lock
Links / QRUN	Motor_Status.Status1	AdvancedStatusDis.	Anzeige: Motor
Links / QSTOP	Motor_Status.Status2	AdvancedStatusDis.	Anzeige: Motor

Beim Mausklick links wird der MOTOR-Bildbaustein und beim Mausklick rechts wird der zugehörige INTERLOK-Bildbaustein aufgerufen.

Der Bausteinname des INTERLOK-Bausteins ist als Skript-Übergabeparameter hinterlegt, siehe Kapitel 2.4 "Skripte"

Der Default des Bausteinnamens ist "L“. Der INTERLOK-Baustein muss im gleichen CFC-Plan wie der MOTOR platziert sein.

2.8.3.16 MOT_SPED

Siehe auch Kapitel 2.8.3.1 "Allgemeine Eigenschaften"

Eigenschaften	Element und Eigenschaft im Anwenderobjekt	Objekt	Beschreibung
Geometrie	Breite = 90 / Höhe = 53		
Links / CollectValue	Sammelanzeige_mitASD1 .Sammelwert	AdvancedStatusDis.	.EventState
Links/QMAN_AUT	Mode.Status1	AdvancedStatusDis.	Anzeige: Auto/Manual
Links / LOCK	Interlock.Status1	AdvancedStatusDis.	Anzeige: Lock
Links / QRUN	Motor_Status.Status1	AdvancedStatusDis.	Anzeige: Motor
Links / QSTOP	Motor_Status.Status2	AdvancedStatusDis.	Anzeige: Motor
Links / QSPEED	Motor_Status.Status3	AdvancedStatusDis.	Anzeige: Motor
Links / QSTOPING	Motor_Status.Status4	AdvancedStatusDis.	Anzeige: Motor

Beim Mausklick links wird der MOT_SPED-Bildbaustein und beim Mausklick rechts wird der zugehörige INTERLOK-Bildbaustein aufgerufen.

Der Bausteinname des INTERLOK-Bausteins ist als Skript-Übergabeparameter hinterlegt, siehe Kapitel 2.4 "Skripte"

Der Default des Bausteinnamens ist "L“. Der INTERLOK-Baustein muss im gleichen CFC-Plan wie der MOT_SPED platziert sein.

2.8.3.17 MOT_REV

Siehe auch Kapitel 2.8.3.1 "Allgemeine Eigenschaften"

Eigenschaften	Element und Eigenschaft im Anwenderobjekt	Objekt	Beschreibung
Geometrie	Breite = 90 / Höhe = 53		
Links / CollectValue	Sammelanzeige_mitASD1 .Sammelwert	AdvancedStatusDis.	.EventState
Links/QMAN_AUT	Mode.Status1	AdvancedStatusDis.	Anzeige: Auto/Manual
Links / LOCK	Interlock.Status1	AdvancedStatusDis.	Anzeige: Lock
Links / QRUN	Motor_Status1.Status1	AdvancedStatusDis.	Anzeige: Motor
Links / QSTOP	Motor_Status1.Status2	AdvancedStatusDis.	Anzeige: Motor
Links / QDIR	Motor_Status1.Status3	AdvancedStatusDis.	Anzeige: Motor

Beim Mausklick links wird der MOT_REV-Bildbaustein und beim Mausklick rechts wird der zugehörige INTERLOK-Bildbaustein aufgerufen.

Der Bausteinname des INTERLOK-Bausteins ist als Skript-Übergabeparameter hinterlegt, siehe Kapitel 2.4 "Skripte"

Der Default des Bausteinnamens ist "L“. Der INTERLOK-Baustein muss im gleichen CFC-Plan wie der MOT_REV platziert sein.

2.8.3.18 INTERLOK

Siehe auch Kapitel 2.8.3.1 "Allgemeine Eigenschaften"

Eigenschaften	Element und Eigenschaft im Anwenderobjekt	Objekt	Beschreibung
Geometrie	Breite = 108 / Höhe = 20		
Links / Link	Lock.AktuellerZustand	ZusAnz.	Lock Symbol

2.8.3.19 OP_D3

Eigenschaften	Element und Eigenschaft im Anwenderobjekt	Objekt	Beschreibung
Geometrie	Breite = 90 / Höhe = 45		
Links / Output1	Zustandsanzeige1.Anzeige	Zustandsanzeige	.Q1
Links / Output2	Zustandsanzeige2.Anzeige	Zustandsanzeige.	.Q2
Links / Output3	Zustandsanzeige3.Anzeige	Zustandsanzeige.	.Q3

2.8.3.20 OP_D

Siehe auch Kapitel 2.8.3.1 "Allgemeine Eigenschaften"

Eigenschaften	Element und Eigenschaft im Anwenderobjekt	Objekt	Beschreibung
Geometrie	Breite = 90 / Höhe = 45		
Links / Status	Zustandsanzeige.Aktueller Zustand	Zustandsanzeige	.Q0

2.8.3.21 OP_TRIG

Siehe auch Kapitel 2.8.3.1 "Allgemeine Eigenschaften"

Eigenschaften	Element und Eigenschaft im Anwenderobjekt	Objekt	Beschreibung
Geometrie	Breite = 90 / Höhe = 40		
Links / Status	Zustandsanzeige.Aktueller Zustand	Zustandsanzeige	.SIGNAL

2.8.3.22 DIG_MON

Siehe auch Kapitel 2.8.3.1 "Allgemeine Eigenschaften"

Eigenschaften	Element und Eigenschaft im Anwenderobjekt	Objekt	Beschreibung
Geometrie	Breite = 90 / Höhe = 40		
Links / Status	Zustandsanzeige.Aktueller Zustand	Zustandsanzeige	.Q
Links / CollectValue	Sammelanzeige3.Sammelwert	Sammelanzeige	.EventState

3 Online-Hilfe erstellen

Voraussetzungen

Sie benötigen:

- den in WINDOWS integrierten ASCII-Editor "Notepad" o.ä. zur Erstellung einer Registrierungsdatei.
- ein Erstellungswerkzeug für die Hilfethemen (z.B. "RoboHelp").

3.1 Aufbau der Hilfedatei

Falls Sie für Ihre Bausteine eine Online-Hilfe erstellen wollen, schreiben Sie mit dem Hilfe-Erstellungs-System eine Hilfedatei. Der Name dieser Datei ist frei wählbar, aus Übersichtlichkeitsgründen sollten Sie jedoch den Namen ihrer Bibliothek (bzw. einen gemeinsamen Namen Ihrer Bausteine) verwenden, z.B. "**MYLIB__a.HLP**".

Erstellen Sie für jeden Ihrer Bausteine ein eigenes Hilfethema (Topic). Anschließend müssen Sie für jedes Topic die Einsprungadresse zur Online-Hilfe definieren ("Topic-ID" + "Map #") und auch in die Registrierungsdatei eintragen (vgl. Kapitel 3.2). Diese müssen innerhalb der jeweiligen Online-Hilfe eindeutig sein, sind aber sonst frei vergebbar.

Handelt es sich um eine relativ umfangreiche Bibliothek, können Sie auch eine hm-Datei erstellen, die alle verwendeten IDs enthält. Bei der Vergabe der MAP-IDs kann dann das Erstellungswerkzeug RoboHelp diese Datei verwenden.

Einträge in der hm-Datei:

```
// Headerfile für Onlinehilfe Mylib-Funktionsbausteine
//
#define CONTROL   0x10                  // dez. 16
#define CONTROL2  0x11                  // dez. 17
#define CONTROL3  0x12                  // dez. 18
....
```

Jedes Hilfethema enthält neben dem Hilfetext die folgenden Angaben:

Topic-Titel	Überschrift des Hilfethemas für diesen Baustein (üblicherweise gleich dem Bausteinnamen, evtl. mit Kurzbezeichnung der Funktion).
Topic-ID	Name und Einsprungadresse des Topics (Name = Header des Bausteins, siehe Bild 1-3)
Index	Stichwörter, mit denen vom Indexverzeichnis zum Hilfethema gesprungen werden kann

Die Hilfe kann aus zwei Dateien bestehen, einer HLP-Datei (Hilfethemen) und einer CNT-Datei (Inhaltsverzeichnis).

Die CNT-Datei ist dann sinnvoll, wenn die Baustein-Hilfe nicht ausschließlich als Kontext-Hilfe (F1 auf den selektierten Baustein) verwendet werden soll. Bei einer Bibliothek mit mehreren Bausteinen können die einzelnen Hilfethemen in einem Inhaltsverzeichnis (Contents) aufgeführt werden. Damit besteht die Möglichkeit, auch zu den Hilfethemen der anderen Bausteine zu wechseln, ohne dass der betreffende Baustein vorhanden sein muss.

Diese CNT-Datei kann auch in der CNT-Datei eines anderen Hilfeprojektes mit einer INCLUDE-Anweisung aufgenommen werden. (z.B. ":include Mylib__a.cnt"). Die eingebundene CNT-Datei wird dann im Inhaltsverzeichnis des anderen Hilfeprojekts mit aufgeführt, wenn beide im gleichen Ordner der Installation vorhanden sind.

Falls Sie Ihre Online-Hilfe in mehreren Sprachen anbieten wollen, müssen Sie für jede gewünschte Sprache eine eigene Hilfedatei erstellen. Bei PCS 7 besteht der Name aus 8 Zeichen, von denen das letzte Zeichen für die Sprachkennung verwendet wird:

a	deutsch	
b	englisch	
c	französisch	
d	spanisch	wird z. Zt von PCS7 nicht unterstützt
e	italienisch	wird z. Zt von PCS7 nicht unterstützt
y	sprachunabhängig	z.B. für Reg-Datei

Mittels der Registrierung (vgl. Kapitel 3.2) wird von PCS 7 dann die zur jeweiligen über den Dialog "**Options > Customize > Language**" eingestellten Landessprache passende Hilfedatei aufgerufen.

Zuletzt müssen Sie die Hilfedatei (falls vorhanden auch die zugehörige CNT-Datei) in das Unterverzeichnis des STEP 7-Verzeichnisses kopieren, in dem Ihre Bibliothek bzw. das Projekt mit Ihren Bausteinen installiert wird.

3.2 Aufbau der Registrierungsdatei

Schreiben Sie mit dem ASCII-Editor eine Registrierungsdatei, die die Informationen für Ihre Bausteine in die WINDOWS- Registrierung einträgt. Der Name der Registrierungsdatei ist frei wählbar, aus Übersichtlichkeitsgründen sollten Sie jedoch den Namen ihrer Bibliothek (bzw. einen gemeinsamen Namen Ihrer Bausteine) verwenden, z.B. "**Mylib_y.reg**".

Beispiel einer Reg-Datei für 3 Bausteine und 5 Sprachversionen (Spanisch und Italienisch mit englischem Hilfetext).

```
REGEDIT4

[HKEY_LOCAL_MACHINE\SOFTWARE\Siemens\STEP7\2.0\Applications\s7libs\mylib\ABC]
"Version"="0.2"
"VersionDate"="23.11.2003"
"HelpFileGerman"="S7libs\\mylib\\MYLIB__a.hlp"
"HelpFileEnglish"="S7libs\\mylib\\MYLIB__b.hlp"
"HelpFileFrench"="S7libs\\mylib\\MYLIB__c.hlp"
"HelpFileSpanish"="S7libs\\mylib\\MYLIB__b.hlp"
"HelpFileItalian"="S7libs\\mylib\\MYLIB__b.hlp"

[HKEY_LOCAL_MACHINE\SOFTWARE\Siemens\STEP7\2.0\Applications\s7libs\mylib\ABC\XYZ]
"CONTROL"=dword:00000010
"CONTROL2"=dword:00000011
"CONTROL3"=dword:00000012
```

Hinweis

Beachten Sie bitte, dass es durch fehlerhafte Einträge in der Registry zu Störungen im Programmablauf kommen kann oder die gewünschte Funktion nicht ausgeführt wird.

Verwenden Sie daher die Schlüssel so wie in dem hier aufgeführten Beispiel.

Es müssen die folgenden Werte in den Schlüssel der Registrierung eingetragen werden:

[HKEY_LOCAL_MACHINE\SOFTWARE\Siemens\STEP7\2.0\Applications\s7libs\NameOfLibrary\Author]

Dabei steht der Bibliotheksname für einen von Ihnen frei gewählten Namen für Ihre Bibliothek (hier: mylib). Er entspricht dem Namen des STEP 7-Unterverzeichnisses, in dem Ihre Hilfedatei abgelegt ist. Unter diesem Namen wird Ihre Bibliothek im CFC-Editor angezeigt. **Author** steht für den von Ihnen beim Attribut AUTHOR im Bausteinkopf angegebenen Namen (hier: ABC).

Version
Enthält die Versionsnummer der gesamten Bibliothek. Dieser Eintrag ist optional.

VersionDate
Enthält das Erstellungsdatum der gesamten Bibliothek. Dieser Eintrag ist optional.

Pfad zur Hilfedatei
Enthält für alle gewünschten Sprachen den zum STEP 7-Verzeichnis relativen Pfad zur jeweiligen Hilfedatei, z.B.:

"HelpFileGerman"="S7libs\\mylib\\MYLIB__a.hlp".

Beachten Sie, dass die Trennzeichen doppelt angegeben werden müssen (\\ ). Über diesen Eintrag wird die zur im SIMATIC Manager eingestellten Landessprache passende Hilfedatei aufgerufen.

Hinweis

Die Sprachen Italienisch und Spanisch werden bisher nicht von PCS 7 unterstützt, deshalb werden bei diesen Sprachen die Hilfen in Englisch aufgerufen..

Danach ist der folgende Schlüssel einzutragen:

[HKEY_LOCAL_MACHINE\SOFTWARE\Siemens\STEP7\2.0\Applications\s7libs\Name OfLibrary \Author\Family]

Dabei steht **Family** für den von Ihnen beim Attribut FAMILY im Bausteinkopf angegebenen Namen (hier: XYZ).

Darunter muss für jeden Baustein der Bausteinname, wie im Bausteinkopf (Header) aufgeführt (siehe Bild 1-3), mit der Einsprungadresse in die Hilfedatei angegeben werden, z.B.: "CONTROL"=dword:00000010. Die Einsprungadresse ist die Nummer der Topic-ID.

Falls Sie Ihre Bausteine in mehrere Familien gruppiert haben, müssen Sie für jede Familie einen eigenen Schlüssel in die Registrierungsdatei einfügen.

Wenn diese Registrierungsdatei ausgeführt worden ist (z.B. mit Doppelklick), wird nach dem Markieren eines Bausteins im CFC oder im SIMATIC Manager mit der F1-Taste über die eingestellte Landessprache und die Baustein-Attribute AUTHOR, FAMILY und FUNCTION_BLOCK in der WINDOWS-Registrierung die zugehörige Hilfedatei bestimmt und die betreffende Hilfe angezeigt.

3.3 Besonderheiten für die Hilfe-Erstellung von SFC-Typen

Abweichend bzw. ergänzend zu den Informationen in den Kapiteln 3.1 und 3.2 ist für die Erstellung von Online-Hilfen für SFC-Typen. Folgendes zu berücksichtigen:

Ablageort

Die Hilfen für die SFC-Typen müssen Sie in Ihr Installationsverzeichnis kopieren. Wir empfehlen für die Ablage den bereits existierenden Ordner "S7Hlp". Das ist der Ordner, der auch die SFC-Hilfen enthält. Nur damit ist gewährleistet, dass Sie aus der selbst erstellten Hilfe auch zur SFC-Hilfe wechseln können, siehe unten. Das folgende Beispiel für die Registrierung bezieht sich auf diesen Ablageort.

Einträge für den Schlüssel

Author	Der Eintrag für "Autor" wird vom SFC vergeben, da dieser die SFC-Typen in ablauffähige Bausteine übersetzt. Immer: **ES_SFC**.
Family	Den Familiennamen entnehmen Sie dem Eigenschaften-Dialog des SFC-Typs aus dem Feld "Familie" (kein Leerstring). Im Beispiel: **SFC**
Name	Den SFC-Typ-Namen entnehmen Sie dem Eigenschaften-Dialog des SFC-Typs aus dem Feld "Name" (kein Leerstring). Im Beispiel: "**Dosi1"** und **"Heat1"**

Beispiel für eine Registrierungsdatei

für 2 Bausteintypen und 2 Sprachversionen.

```
Windows Registry Editor Version 5.00

[HKEY_LOCAL_MACHINE\SOFTWARE\Siemens\STEP7\2.0\Applications\s7libs\SFCTypes]

[HKEY_LOCAL_MACHINE\SOFTWARE\Siemens\STEP7\2.0\Applications\s7libs\SFCTypes\ES_SFC]
"HelpFileGerman"="S7Hlp\\SFC_Typa.hlp"
"HelpFileENGLISH"="S7Hlp\\SFC_Typb.hlp"
"Version"="1.1"
"VersionDate"="14.11.2003"

[HKEY_LOCAL_MACHINE\SOFTWARE\Siemens\STEP7\2.0\Applications\s7libs\SFCTypes\ES_SFC\SFC]
"Dosi1"=dword:00000888
"Heat1"=dword:00000889
```

Zur SFC-Hilfe wechseln

Damit Sie aus der selbst erstellten Hilfe zur SFC-Hilfe gelangen, können Sie in den einzelnen Topics einen Sprung auf das (interne) Inhaltsverzeichnis der SFC-Hilfe einfügen. Dieser Sprung sieht wie folgt aus:

SFC-Hilfe!JumpID(`s7jsfcaa.hlp',`IDH_CONTENTS')

Hinweis: Die Sprungadresse nach dem grünen doppelt unterstrichenen Text → !JumpID(`s7jsfcaa.hlp',`IDH_CONTENTS') wird als verborgener Text formatiert.

4 Bibliothek und Setup erstellen

Voraussetzungen

Zur Erstellung einer lieferfähigen Bibliothek inkl. dem dazu gehörigen Setup benötigen Sie ein Programm zur Erstellung von Installationsprogrammen, z.B. "InstallShield".

4.1 Bibliothek erstellen

Falls Sie Ihre Bausteine und /oder SFC-Typen in einer Bibliothek zusammenfassen wollen, gehen Sie folgendermaßen vor:

1. Legen Sie eine neue S7-Bibliothek an und erzeugen darin ein S7-Programm.
2. Tragen Sie die Namen und Nummern Ihrer Bausteine sowie die dazugehörigen Kommentare in die Symboltabelle der Bibliothek ein (dies gilt nicht für SFC-Typen).
3. Quellen:
   Falls Sie die Quellen der Bausteine mitliefern wollen, kopieren Sie die Quellen aus dem Quellordner Ihres Projektes in den Quellordner der Bibliothek.
4. Bausteine:
   Kopieren Sie Ihre Bausteine aus dem Bausteinordner Ihres Projektes in den Bausteinordner der Bibliothek.

   Hinweis:

   - Falls Sie in Ihren Multiinstanzbausteinen nicht allgemein verfügbare Bausteine aufrufen (SFBs, SFCs), kopieren Sie diese ebenfalls in den Bausteinordner der Bibliothek.
   - Die beim Übersetzen von SFC-Typen erzeugten Bausteine dürfen Sie nicht kopieren.
5. SFC-Typen:
   Falls noch nicht vorhanden, legen Sie einen Planordner an.
   Kopieren Sie die SFC-Typen aus dem Planordner Ihres Projekts in den Planordner der Bibliothek.

   Hinweis: Die zu den SFC-Typen zugehörigen Bausteine werden dabei ebenfalls mitkopiert und im Baustein-Ordner abgelegt.

4.2 Setup erstellen

Falls Sie Ihre Bibliothek per Setup auf dem Zielrechner installieren wollen, entwerfen Sie mit dem Setup-Erstellungs-Werkzeug ein Installationsskript, das die folgenden Aktionen durchführt:

- Kopieren der Bausteinbibliothek in das Unterverzeichnis **S7LIBS** des STEP 7 Verzeichnisses.
- Aufruf des Programms **S7BIN\S7ALIBXX.EXE** im STEP 7-Verzeichnis, um die neue Bibliothek beim SIMATIC Manager bekannt zu machen.
- Kopieren der Hilfedatei (HLP- und CNT-Datei) in das Unterverzeichnis des STEP 7-Verzeichnisses, in den die Bausteinbibliothek kopiert worden ist (z. B. das Unterverzeichnis **S7LIBS\MYLIB**).
- Aufruf der zur Hilfedatei gehörenden Registrierungsdatei
- Kopieren der Prototypbilder in das Unterverzeichnis **options\pdl** \FaceplateDesigner_V6 des WinCC-Verzeichnisses.
- Kopieren der Skripts in ein beliebiges Unterverzeichnis im Unterverzeichnis **aplib** des WinCC-Verzeichnisses. Sinnvollerweise sollte dieses Verzeichnis den selben Namen haben, wie das, in das die Bausteinbibliothek kopiert worden ist (z.B. das Unterverzeichnis **options\pdl\mylib**).
- Einrichten einer Deinstallationsmöglichkeit.

Beachten Sie, dass die Bausteinbibliothek und die Online-Hilfe nur installiert werden können, wenn auf dem Zielrechner STEP 7 vorhanden ist. Die Prototypbilder können nur in ein Unterverzeichnis von WinCC installiert werden. Sehen Sie deshalb im Installationsdialog Abfragen auf das Vorhandensein von STEP 7 und WinCC vor. Dazu können Sie in der Registrierung im Schlüssel

HKEY_LOCAL_MACHINE\SOFTWARE\Siemens\STEP7
den Namen STEP7_VERSION auf den Wert der gewünschten Version (z.B. "5.2)" und im Schlüssel HKEY_LOCAL_MACHINE\SOFTWARE\Siemens\WinCC\Setup den Namen der Version (z.B. V6.0 SP1 abfragen.

A Beispiele: Quellcode der Bausteine MEAS_MON, MOTOR und VALVE

A.1 MEAS_MON

```
// @(#) $Header::   L:/PCS7/PCS7LibV51.SW/TechBlocks/VCS/Algorithm/meas_mon.csv$
//Copyright (C) Siemens AG 1995. All Rights Reserved. Confidential

// PCS 7 Library Vx.x
// Function: Meas.value monitoring block
// Label Version 3.0
// Macro Version :V0.92
FUNCTION_BLOCK "MEAS_MON"
TITLE = 'Meas.value monitoring block'
//
{
        S7_tasklist:=        'OB100';
        S7_alarm_ui:=       '1';
        S7_m_c:=            'true'
}
AUTHOR:              TECHN61
NAME:                MEAS_MON
VERSION:             '3.0'
FAMILY:              CONTROL
KNOW_HOW_PROTECT

VAR_INPUT
OOS  {S7_visible:='false';
        S7_link:='false';
        S7_m_c:='true';
        S7_string_0:='In Service';
        S7_string_1:='Out of Service'} :    BOOL := 0; // 1= Out of Service
M_SUP_AH     {S7_visible:='false';
        S7_link:='false';
        S7_m_c:='true';
        S7_string_0:='Suppress HH=No';
        S7_string_1:='Suppress HH=Yes'} :   BOOL := 0; // 1=Suppress HH Alarm
M_SUP_AL     {S7_visible:='false';
        S7_link:='false';
        S7_m_c:='true';
        S7_string_0:='Suppress LL=No';
        S7_string_1:='Suppress LL=Yes'} :   BOOL := 0; // 1=Suppress LL Alarm
M_SUP_WH     {S7_visible:='false';
        S7_link:='false';
        S7_m_c:='true';
        S7_string_0:='Suppress H=No';
        S7_string_1:='Suppress H=Yes'} :    BOOL := 0; // 1=Suppress H Alarm (Warning)
M_SUP_WL     {S7_visible:='false';
        S7_link:='false';
        S7_m_c:='true';
        S7_string_0:='Suppress L=No';
        S7_string_1:='Suppress L=Yes'} :    BOOL := 0; // 1=Suppress L Alarm (Warning)
CSF  {S7_dynamic:='true'} :     BOOL := 0; // Control System Fault 1=External Error
MSG_LOCK     {S7_visible:='false';
        S7_dynamic:='true';
        S7_m_c:='true'} :   BOOL := 0;     // Enable 1=Messages locked
MO_PVHR  {S7_visible:='false';
        S7_m_c:='true';
        S7_shortcut:='Bar UL';
        S7_unit:=''} :  REAL := 110;       // High Limit Bar Range
MO_PVLR  {S7_visible:='false';
        S7_m_c:='true';
        S7_shortcut:='Bar LL';
        S7_unit:=''} :  REAL := -10;       // Low Limit Bar Range
```

```
USTATUS  {S7_visible:='false'} :    WORD := 0; // User Status Bits
U    {S7_qc:='true';
       S7_dynamic:='true';
       S7_m_c:='true';
       S7_shortcut:='PV';
       S7_unit:=''} :  REAL := 0;   // Analog Input (Measured Value)
QC_U : BYTE := 16#80;               // Quality Code for Input U
U_AH     {S7_link:='false';
       S7_edit:='para';
       S7_m_c:='true';
       S7_shortcut:='HH alarm';
       S7_unit:=''} :  REAL := 100; // HH Alarm Limit
U_WH     {S7_link:='false';
       S7_edit:='para';
       S7_m_c:='true';
       S7_shortcut:='H alarm';
       S7_unit:=''} :  REAL := 95;  // H Alarm Limit (Warning)
U_WL     {S7_link:='false';
       S7_edit:='para';
       S7_m_c:='true';
       S7_shortcut:='L alarm';
       S7_unit:=''} :  REAL := -3;  // L Alarm Limit (Warning)
U_AL     {S7_link:='false';
       S7_edit:='para';
       S7_m_c:='true';
       S7_shortcut:='LL alarm';
       S7_unit:=''} :  REAL := -5;  // LL Alarm Limit
HYS  {S7_link:='false';
       S7_m_c:='true';
       S7_shortcut:='Hysteresis';
       S7_unit:=''} :  REAL := 5;   // Hysteresis of Analog Input
MSG_EVID     {S7_visible:='false';
       S7_link:='false';
       S7_param :='false';
       S7_server:='alarm_archiv';S7_a_type:='alarm_8p'} :  DWORD := 0; // Message ID
BA_EN    {S7_visible:='false';
       S7_m_c:='true'} :   BOOL := 0;        // Batch Enable
OCCUPIED     {S7_visible:='false';
       S7_m_c:='true'} :   BOOL := 0;        // Occupied by Batch
BA_ID    {S7_visible:='false';
       S7_m_c:='true'} :   DWORD := 0;       // Batch ID
BA_NA    {S7_visible:='false';
       S7_m_c:='true'} :   STRING[32] := ''; // Batch Name
STEP_NO  {S7_visible:='false';
       S7_m_c:='true'} :   DWORD := 0;       // Batch Step Number
RUNUPCYC     {S7_visible:='false';
       S7_link:='false'} :     INT := 3;     // Number of Run Up Cycles
END_VAR

VAR_IN_OUT
AUX_PR05     {S7_visible:='false'} :    ANY; // Auxiliary Value 5
AUX_PR06     {S7_visible:='false'} :    ANY; // Auxiliary Value 6
AUX_PR07     {S7_visible:='false'} :    ANY; // Auxiliary Value 7
AUX_PR08     {S7_visible:='false'} :    ANY; // Auxiliary Value 8
AUX_PR09     {S7_visible:='false'} :    ANY; // Auxiliary Value 9
AUX_PR10     {S7_visible:='false'} :    ANY; // Auxiliary Value 10
END_VAR

VAR_OUTPUT
QERR     {S7_visible:='false';
       S7_m_c:='true'} :   BOOL := 1; // 1=Error
QH_ALM   {S7_dynamic:='true'} :     BOOL := 0; // 1=HH-Alarm active
QL_ALM   {S7_dynamic:='true'} :     BOOL := 0; // 1=LL Alarm active
QH_WRN   {S7_dynamic:='true'} :     BOOL := 0; // 1=H Alarm active (Warning)
QL_WRN   {S7_dynamic:='true'} :     BOOL := 0; // 1=L Alarm active (Warning)
QMSG_ERR     {S7_visible:='false';
       S7_dynamic:='true'} :   BOOL := 0;     // 1=Message ERROR
QMSG_SUP     {S7_visible:='false';
       S7_dynamic:='true';
       S7_m_c:='true'} :   BOOL := 0;         // 1=Message Suppression Active
MSG_STAT     {S7_visible:='false';
       S7_dynamic:='true'} :   WORD := 0;     // Message: STATUS output
MSG_ACK  {S7_visible:='false';
       S7_dynamic:='true'} :   WORD := 0;     // Message: ACK_STATE output
VSTATUS  {S7_visible:='false';
       S7_m_c:='true'} :   DWORD := 0;        // Status word
END_VAR
```

```
CONST
C_OOS := 0;        // 1= Out of Service
C_M_SUP_AH := 0;   // 1=Suppress HH Alarm
C_M_SUP_AL := 0;   // 1=Suppress LL Alarm
C_M_SUP_WH := 0;   // 1=Suppress H Alarm (Warning)
C_M_SUP_WL := 0;   // 1=Suppress L Alarm (Warning)
C_CSF := 0;        // Control System Fault 1=External Error
C_MSG_LOCK := 0;   // Enable 1=Messages locked
C_MO_PVHR := 110;  // High Limit Bar Range
C_MO_PVLR := -10;  // Low Limit Bar Range
C_USTATUS := 0;    // User Status Bits
C_U := 0;          // Analog Input (Measured Value)
C_QC_U := 16#80;   // Quality Code for Input U
C_U_AH := 100;     // HH Alarm Limit
C_U_WH := 95;      // H Alarm Limit (Warning)
C_U_WL := -3;      // L Alarm Limit (Warning)
C_U_AL := -5;      // LL Alarm Limit
C_HYS := 5;        // Hysteresis of Analog Input
C_AUX_PR05 := 0;   // Auxiliary Value 5
C_AUX_PR06 := 0;   // Auxiliary Value 6
C_AUX_PR07 := 0;   // Auxiliary Value 7
C_AUX_PR08 := 0;   // Auxiliary Value 8
C_AUX_PR09 := 0;   // Auxiliary Value 9
C_AUX_PR10 := 0;   // Auxiliary Value 10
C_MSG_EVID := 0;   // Message ID
C_BA_EN := 0;      // Batch Enable
C_OCCUPIED := 0;   // Occupied by Batch
C_BA_ID := 0;      // Batch ID
C_BA_NA := '';     // Batch Name
C_STEP_NO := 0;    // Batch Step Number
C_RUNUPCYC := 3;   // Lag: Number of Run Up Cycles
C_QERR := 1;       // 1=Error
C_QH_ALM := 0;     // 1=HH-alarm Active
C_QL_ALM := 0;     // 1=LL Alarm Active
C_QH_WRN := 0;     // 1=H Alarm active (Warning)
C_QL_WRN := 0;     // 1=L Alarm Active (Warning)
C_QMSG_ERR := 0;   // 1=Message ERROR
C_QMSG_SUP := 0;   // 1=Message Suppression Active
C_MSG_STAT := 0;   // Message: STATUS Output
C_MSG_ACK := 0;    // Message: ACK_STATE output
C_VSTATUS := 0;    // Status word
END_CONST

// Static Variables
VAR
siRUNUPCNT:    int := 0;   // Counter for RUNUPCYC editing
 sb_SIG_1: bool    := FALSE;   //Merker ALARM_8P Signal 1
 sb_SIG_2: bool    := FALSE;   //Merker ALARM_8P Signal 2
 sb_SIG_3: bool    := FALSE;   //Merker ALARM_8P Signal 3
 sb_SIG_4: bool    := FALSE;   //Merker ALARM_8P Signal 4
 sb_SIG_5: bool    := FALSE;   //Merker ALARM_8P Signal 5
ALARM_8P_1:    ALARM_8P;   // Multiple instance ALARM_8P
siBA_ID:   dword := 0;     // Old value BA_ID
sbyBA_NA:  array[1..32] of byte := 32(0);
VSTATUS_LOC : DWORD :=16#0;// Local static variable, in which the output VSTATUS
                           // is copied.
STEP_NO_LOC : DWORD;       // Local variable, in which the input STEP_NO is
                           // saved.
PV_IN_LOC : REAL;          // Local variable, in which the input PV_IN is saved.
dwDUMMY:    DWORD := 0;    // Stand-by
BA_ID_LOC : DWORD;         // Local static variable, in which the input BA_ID
                           // is copied.
U_LOC : REAL;  // Local static variable, in which the input U is copied, because of
               // message in ALARM_8P

// Variables for the status word "VSTATUS" and "USTATUS"
VSTATUS_STR AT VSTATUS_LOC :  STRUCT
                                USTATUS : WORD;
                                VSTATUS_LOW_BIT_8 : BOOL;
                                VSTATUS_LOW_BIT_9 : BOOL;
                                VSTATUS_LOW_BIT_10 : BOOL;
                                VSTATUS_LOW_BIT_11 : BOOL;
                                VSTATUS_LOW_BIT_12 : BOOL;
                                VSTATUS_LOW_BIT_13 : BOOL;
                                VSTATUS_LOW_BIT_14 : BOOL;
                                VSTATUS_LOW_BIT_15 : BOOL;
                                VSTATUS_LOW_BIT_0 : BOOL;
                                VSTATUS_LOW_BIT_1 : BOOL;
                                VSTATUS_LOW_BIT_2 : BOOL;
```

```
                                VSTATUS_LOW_BIT_3 : BOOL;
                                VSTATUS_LOW_BIT_4 : BOOL;
                                VSTATUS_LOW_BIT_5 : BOOL;
                                VSTATUS_LOW_BIT_6 : BOOL;
                                VSTATUS_LOW_BIT_7 : BOOL;
                              END_STRUCT;
END_VAR

// Temporary Variables
VAR_TEMP
 pbALARM:  BOOL;       // Call up ALARM_8P
 pbM_SUP:  BOOL;       // Message suppression
 pb_SIG_1,pb_SIG_2,pb_SIG_3,pb_SIG_4:BOOL;
TOP_SI:    STRUCT
   EV_CLASS:   BYTE;
   EV_NUM:     BYTE;
   PRIORITY:   BYTE;
   NUM:        BYTE;
   TYP2_3:     BYTE;
   TYP1:       BYTE;
   ZI1:        WORD;
   ZI2_3:      DWORD;
   END_STRUCT;
START_UP_SI:   STRUCT
   EV_CLASS:   BYTE;
   EV_NUM:     BYTE;
   PRIORITY:   BYTE;
   NUM:        BYTE;
   TYP2_3:     BYTE;
   TYP1:       BYTE;
   ZI1:        WORD;
   ZI2_3:      DWORD;
   END_STRUCT;
 ERR :     INT;    // Error at startup
END_VAR

BEGIN
    // Write VSTATUS, STEP_NO resave as STEP_NO_LOC and
      // BA_ID resave as BA_ID_LOC.
    // Supply of VSTATUS output variables with the inputs.
    VSTATUS_STR.VSTATUS_LOW_BIT_0 := OCCUPIED;
    VSTATUS_STR.VSTATUS_LOW_BIT_1 := BA_EN;
    VSTATUS_STR.VSTATUS_LOW_BIT_2 := MSG_LOCK;
    // Supply of VSTATUS output variables (HIGH BYTE) with the values,
      // which come from the user via the input variable USTATUS (from external).
    VSTATUS_STR.USTATUS := USTATUS;
    STEP_NO_LOC := STEP_NO; // Resave, due to output of STEP_NO_LOC in ALARM_8P
    BA_ID_LOC := BA_ID;     // Resave, due to output of BA_ID_LOC in ALARM_8P
    U_LOC := U;             // Resave, due to output of U_LOC in ALARM_8P
    ERR := RD_SINFO (TOP_SI := TOP_SI, START_UP_SI := START_UP_SI);
    // Read out start info
    pbM_SUP := MSG_LOCK;
    IF TOP_SI.NUM = 100   THEN     // When startup
        siRUNUPCNT := RUNUPCYC;    // Saving the value of the RUNUPCYC input
        // Initialization outputs
        QMSG_ERR := C_QMSG_ERR;
        QMSG_SUP := C_QMSG_SUP;
        MSG_STAT := C_MSG_STAT;
        MSG_ACK  := C_MSG_ACK;
        QH_WRN := C_QH_WRN;
        QL_WRN := C_QL_WRN;
        QH_ALM := C_QH_ALM;
        QL_ALM := C_QL_ALM;
        // pbALARM := NOT OOS; // Initialization first call ALARM_8P
        pb_SIG_1:= QH_ALM;     // Alarm high
        pb_SIG_2:= QH_WRN;     // Warning high
        pb_SIG_3:= QL_WRN;     // Warning low
        pb_SIG_4:= QL_ALM;     // Alarm low
        pbALARM :=TRUE;        // Initialization ALARM
    ELSE;
        // LIMITS_P.1 f¸r Alarmpr¸fung (optimiert)
        IF (U <= U_AL) THEN            // Low limit responded
            QL_ALM := TRUE;
        ELSE;
            IF (U >= (U_AL+HYS)) THEN  // Reset low limit responded
                QL_ALM := FALSE;
            ELSE;                      // QL_ALM remains unchanged
```

```
        END_IF;
    END_IF;
    IF (U >= U_AH) THEN           // High limit responded
                    QH_ALM := TRUE;
    ELSE;
        IF (U <= (U_AH-HYS)) THEN // Reset high limit responded
            QH_ALM := FALSE;
        ELSE;                     // QH_ALM remains unchanged
        END_IF;
    END_IF;
    // LIMITS_P.2 for Warning check (optimized)
    IF (U <= U_WL) THEN           // Low limit responded
        QL_WRN := TRUE;
    ELSE;
        IF (U >= (U_WL+HYS)) THEN // Reset low limit responded
            QL_WRN := FALSE;
        ELSE;                     // QL_WRN remains unchanged
        END_IF;
    END_IF;
    IF (U >= U_WH) THEN           // High limit responded
        QH_WRN := TRUE;
    ELSE;
        IF (U <= (U_WH-HYS)) THEN // Reset high limit responded
            QH_WRN := FALSE;
        ELSE;                     // QH_WRN remains unchanged
        END_IF;
    END_IF;
    IF siRUNUPCNT = 0  THEN       // Initialize alarms
        IF M_SUP_AH OR MSG_LOCK THEN
            pb_SIG_1:= 0;         // Report possible outgoing
        ELSE;
            pb_SIG_1:= QH_ALM;    // Alarm high
        END_IF;
        IF M_SUP_WH OR MSG_LOCK THEN
            pb_SIG_2:= 0;         // Report possible outgoing
        ELSE;
            pb_SIG_2:= QH_WRN;    // Warning high
        END_IF;
        IF M_SUP_WL OR MSG_LOCK THEN
            pb_SIG_3:= 0;         // Report possible outgoing
        ELSE;
            pb_SIG_3:= QL_WRN;    // Warning low
        END_IF;
        IF M_SUP_AL OR MSG_LOCK THEN
            pb_SIG_4:= 0;         // Report possible outgoing
        ELSE;
            pb_SIG_4:= QL_ALM;    // Alarm low
        END_IF;
        // pbALARM := (sb_SIG_1 <> pb_SIG_1) OR (sb_SIG_2 <>  pb_SIG_2)
          // OR (sb_SIG_3 <>  pb_SIG_3) OR (sb_SIG_4 <>  pb_SIG_4)
            // OR (sb_SIG_5 <>  CSF);
        pbALARM :=TRUE;           // Initialization ALARM
    ELSE;
        pbALARM :=FALSE;          // Initialization no ALARM
        pbM_SUP := TRUE;
        siRUNUPCNT := siRUNUPCNT - 1;
    END_IF;
END_IF;
IF pbALARM  THEN                  // Call up ALARM_8P
    IF siBA_ID <> BA_ID_LOC THEN
        // STRING variables may not be interconnected to ALARM8_P as auxiliary
          //process values, therefore transferred in ARRAY OF BYTE.
        FOR ERR := 1 TO 32
         DO
            sbyBA_NA[ERR] := 0;   // Delete array as default
        END_FOR;
        ERR := BLKMOV (SRCBLK:= BA_NA,DSTBLK:=sbyBA_NA);
        siBA_ID := BA_ID_LOC;     // Save modified BA_ID
    END_IF;
    // Call ALARM_8P with lock logic (MSG_LOCK).
    ALARM_8P_1(    EN_R := TRUE,  // Update the ACKL_STATE output
                   ID := 16#EEEE, // PMC communication channel
       EV_ID:= MSG_EVID,
       SIG_1:= pb_SIG_1,
       SIG_2:= pb_SIG_2,
       SIG_3:= pb_SIG_3,
       SIG_4:= pb_SIG_4,
       SIG_5:= CSF,
       SIG_6:= 0,
```

```
          SIG_7:= 0,
          SIG_8:= 0,
          SD_1 := sbyBA_NA,
          SD_2 := STEP_NO_LOC,
          SD_3 := BA_ID_LOC,
          SD_4 := U_LOC,
          SD_5 := AUX_PR05,
          SD_6 := AUX_PR06,
          SD_7 := AUX_PR07,
          SD_8 := AUX_PR08,
          SD_9 := AUX_PR09,
          SD_10 := AUX_PR10);
       QMSG_ERR := ALARM_8P_1.ERROR;
       MSG_STAT := ALARM_8P_1.STATUS;
       MSG_ACK := ALARM_8P_1.ACK_STATE;
       IF (NOT QMSG_ERR) THEN        // Note historical signals.
           sb_SIG_1:= pb_SIG_1;
           sb_SIG_2:= pb_SIG_2;
           sb_SIG_3:= pb_SIG_3;
           sb_SIG_4:= pb_SIG_4;
           sb_SIG_5:= CSF;
       END_IF;
   END_IF;
   IF (MSG_STAT = 21) THEN           // Block locked
       pbM_SUP := TRUE;
   END_IF;
   QMSG_SUP := pbM_SUP;
   QERR := NOT OK;                   // Note negated OK-Flag result in the block.
   // Power supply of the VSTATUS output variable with the outputs.
   VSTATUS_STR.VSTATUS_LOW_BIT_14 := QMSG_SUP OR M_SUP_AH OR M_SUP_WH OR M_SUP_WL OR
M_SUP_AL ;
   VSTATUS_STR.VSTATUS_LOW_BIT_15 := OOS;
   VSTATUS := VSTATUS_LOC;
END_FUNCTION_BLOCK
```

A.2 MOTOR

```
// @(#) $Header::   L:/PCS7/PCS7LibV51.SW/TechBlocks/VCS/Algorithm/motor.csv   $
//Copyright (C) Siemens AG 1995-1997. All Rights Reserved. Confidential

PCS 7 Library Vx.x
// Function: motor
// Label Version 3.0
// Macro Version :V0.92

FUNCTION_BLOCK "MOTOR"
TITLE = 'motor '
//
{
        S7_tasklist:=        'OB100';
        S7_alarm_ui:=       '1';
        S7_m_c:=            'true'
}
AUTHOR:              TECHN61
NAME:                MOTOR
VERSION:             '4.0'
FAMILY:              CONTROL
KNOW_HOW_PROTECT

VAR_INPUT
OOS  {S7_visible:='false';
        S7_link:='false';
        S7_m_c:='true';
        S7_string_0:='In Service';
        S7_string_1:='Out of Service'} :    BOOL := 0; // 1= Out of Service
LOCK     {S7_dynamic:='true';
        S7_m_c:='true'} :   BOOL := 0;     // 1=Lock to OFF
LOCK_ON  {S7_dynamic:='true';
        S7_m_c:='true'} :   BOOL := 0;     // 1=Lock to ON
AUTO_ON  {S7_dynamic:='true';
        S7_contact:='true'} :   BOOL := 0; // AUTO Mode:1=ON, 0=OFF
L_RESET  {S7_dynamic:='true';
        S7_contact:='true'} :   BOOL := 0; // Linkable Input RESET
MSS  {S7_dynamic:='true'} :     BOOL := 1; // Motor Protecting Switch: 0=Active
CSF  {S7_dynamic:='true'} :     BOOL := 0; // Control System Fault 1=External Error
FB_ON    {S7_qc:='true';
        S7_dynamic:='true';
        S7_m_c:='true'} :   BOOL := 0; // Feedback: 1=ON
QC_FB_ON : BYTE := 16#80;              // Quality Code for FB_ON
QC_QSTART_I : BYTE := 16#80;           // Quality Code for Input QSTART
ON_OP_EN   {S7_visible:='false'} :   BOOL := 1; // Enable 1=Operator may input "ON"
OFFOP_EN   {S7_visible:='false'} :   BOOL := 1; // Enable 1=Operator for "OFF"
MANOP_EN   {S7_visible:='false'} :   BOOL := 1; // Enable: 1=Operator may input "MANUAL"
AUTOP_EN   {S7_visible:='false'} :   BOOL := 1; // Enable: 1=Operator may input "AUTO"
LIOP_SEL   {S7_dynamic:='true';
        S7_contact:='true'} :   BOOL := 0; // Select: 1=Linking, 0=Operator Active
AUT_L    {S7_dynamic:='true';
        S7_contact:='true'} :   BOOL := 0; // Linkable Input for  MANUAL/AUTO Mode
MONITOR  {S7_link:='false';
        S7_m_c:='true';
        S7_string_0:='Monitoring=Off';
        S7_string_1:='Monitoring=On'} :     BOOL := 1;  // Select: 1=Monitoring ON,
                                                        // 0=Monitoring OFF
TIME_MON     {S7_link:='false';
        S7_edit:='para';
        S7_m_c:='true';
        S7_shortcut:='Mon. Time';
        S7_unit:='s'} :    REAL := 3;          // Monitoring Time for ON [s]
SAMPLE_T     {S7_visible:='false';
        S7_sampletime:='true'} :    REAL := 1; // Sample Time [s]
MSG_EVID     {S7_visible:='false';
        S7_link:='false';
        S7_param :='false';
        S7_server:='alarm_archiv';S7_a_type:='alarm_8p'} :  DWORD := 0; // Message ID
BA_EN    {S7_visible:='false';
        S7_m_c:='true'} :   BOOL := 0;        // Batch Enable
OCCUPIED     {S7_visible:='false';
        S7_m_c:='true'} :   BOOL := 0;        // Occupied by Batch
```

```
BA_ID    {S7_visible:='false';
         S7_m_c:='true'} :     DWORD := 0;      // Batch ID
BA_NA    {S7_visible:='false';
         S7_m_c:='true'} :   STRING[32] := ''; // Batch Name
STEP_NO  {S7_visible:='false';
         S7_m_c:='true'} :   DWORD := 0;       // Batch Step Number
RUNUPCYC     {S7_visible:='false';
         S7_link:='false'} :     INT := 3; // Lag: Number of Run Up Cycles
START_OFF    {S7_visible:='false'} :    BOOL := 1; // 1=Start up with Motor OFF
FAULT_OFF    {S7_visible:='false'} :    BOOL := 1; // 1=In case of Fault: Motor OFF
MSS_OFF  {S7_visible:='false'} :    BOOL := 1; // 1=In case of MSS-Fault: Motor OFF
USTATUS  {S7_visible:='false'} :    WORD := 0; // User STATUS Bits
END_VAR

VAR_IN_OUT
RESET    {S7_visible:='false';
         S7_link:='false';
         S7_m_c:='true';
         S7_string_0:='0';
         S7_string_1:='Error=Reset'} :   BOOL := 0; // Operator Input Error Reset
MAN_ON   {S7_visible:='false';
         S7_link:='false';
         S7_m_c:='true';
         S7_string_0:='Motor=Stop';
         S7_string_1:='Motor=Start'} :   BOOL := 0; // Operator Input: 1=ON, 0=OFF
AUT_ON_OP    {S7_visible:='false';
         S7_link:='false';
         S7_m_c:='true';
         S7_string_0:='Mode=Manual';
         S7_string_1:='Mode=Auto'} :     BOOL := 0; // Operator Input Mode 1=AUTO,
                                                    // 0= MANUAL
AUX_PR04     {S7_visible:='false'} :    ANY; // Auxiliary Value 4
AUX_PR05     {S7_visible:='false'} :    ANY; // Auxiliary Value 5
AUX_PR06     {S7_visible:='false'} :    ANY; // Auxiliary Value 6
AUX_PR07     {S7_visible:='false'} :    ANY; // Auxiliary Value 7
AUX_PR08     {S7_visible:='false'} :    ANY; // Auxiliary Value 8
AUX_PR09     {S7_visible:='false'} :    ANY; // Auxiliary Value 9
AUX_PR10     {S7_visible:='false'} :    ANY; // Auxiliary Value 10
END_VAR

VAR_OUTPUT
QERR     {S7_visible:='false';
         S7_dynamic:='true';
         S7_m_c:='true'} :   BOOL := 1; // 1=Error
QMSS_ST  {S7_dynamic:='true';
         S7_m_c:='true'} :   BOOL := 0; // Unacknowledged Motor Protective Switch
QMON_ERR     {S7_dynamic:='true';
         S7_m_c:='true'} :   BOOL := 0; // 1=Monitoring Error
QGR_ERR  {S7_dynamic:='true';
         S7_contact:='true'} :   BOOL := 0; // 1=Group Error
QOP_ERR  {S7_visible:='false';
         S7_dynamic:='true'} :   BOOL := 0; // 1=Operator Error
QRUN     {S7_dynamic:='true';
         S7_contact:='true';
         S7_m_c:='true'} :   BOOL := 0; // Status: 1=Motor running
QSTOP    {S7_dynamic:='true';
         S7_contact:='true';
         S7_m_c:='true'} :   BOOL := 0; // Status: 1=Motor STOP
QSTART   {S7_qc:='true';
         S7_dynamic:='true';
         S7_m_c:='true'} :   BOOL := 0; // Control Output 1=START Active
QC_QSTART : BYTE := 16#80; // Quality Code for Output QSTART
QON_OP   {S7_visible:='false';
         S7_dynamic:='true';
         S7_m_c:='true'} :   BOOL := 0; // Status: 1=Operator enabled for "ON"
QOFF_OP  {S7_visible:='false';
         S7_dynamic:='true';
         S7_m_c:='true'} :   BOOL := 0; // Status: 1=Operator enabled for "OFF"
QMAN_AUT     {S7_dynamic:='true';
         S7_contact:='true';
         S7_m_c:='true'} :   BOOL := 0; // 1=AUTO, 0=MANUAL Mode
QMANOP   {S7_visible:='false';
         S7_dynamic:='true';
         S7_m_c:='true'} :   BOOL := 0; // Status: 1=Oper. ena. for "MANUAL" Mode
```

```
QAUTOP   {S7_visible:='false';
         S7_dynamic:='true';
         S7_m_c:='true'} :   BOOL := 0; // Status: 1=Operator enabled for "AUTO"
QMSG_ERR     {S7_visible:='false';
         S7_dynamic:='true'} :   BOOL := 0; // 1=Message ERROR
QMSG_SUP     {S7_visible:='false';
         S7_dynamic:='true';
         S7_m_c:='true'} :   BOOL := 0;     // 1=Message Suppression Active
MSG_STAT     {S7_visible:='false';
         S7_dynamic:='true'} :   WORD := 0; // Message: STATUS Output
MSG_ACK  {S7_visible:='false';
         S7_dynamic:='true'} :   WORD := 0; // Message: ACK_STATE output
VSTATUS  {S7_visible:='false';
         S7_m_c:='true'} :   DWORD := 0;    // Status word
END_VAR

CONST
C_OOS := 0;              // 1= Out of Service
C_LOCK := 0;             // 1=Lock to OFF
C_LOCK_ON := 0;          // 1=Lock to ON
C_AUTO_ON := 0;          // AUTO Mode:1=ON, 0=Off
C_RESET := 0;            // Operator Input Error Reset
C_L_RESET := 0;          // Linkable Input RESET
C_MSS := 1;              // Motor Protecting Switch: 0=Active
C_CSF := 0;              // Control System Fault 1=External Error
C_FB_ON := 0;            // Feedback: 1=ON
C_QC_FB_ON := 16#80;     // Quality Code for FB_ON
C_QC_QSTART_I := 16#80; // Quality Code for Input QSTART
C_MAN_ON := 0;           // Operator Input: 1=ON, 0=OFF
C_ON_OP_EN := 1;         // Enable 1=Operator may input ON
C_OFFOP_EN := 1;         // Enable 1=Operator for "OFF"
C_AUT_ON_OP := 0;        // Operator Input Mode 1=AUTO, 0= MANUAL
C_MANOP_EN := 1;         // Enable: 1=Operator may input MANUAL
C_AUTOP_EN := 1;         // Enable: 1=Operator may input AUTO
C_LIOP_SEL := 0;         // Select: 1=Linking, 0=Operator Active
C_AUT_L := 0;            // Linkable Input for  MANUAL/AUTO Mode
C_MONITOR := 1;          // Select: 1=Monitoring ON, 0=Monitoring OFF
C_TIME_MON := 3;         // Monitoring Time for ON [s]
C_SAMPLE_T := 1;         // Sample Time [s]
C_AUX_PR04 := 0;         // Auxiliary Value 4
C_AUX_PR05 := 0;         // Auxiliary Value 5
C_AUX_PR06 := 0;         // Auxiliary Value 6
C_AUX_PR07 := 0;         // Auxiliary Value 7
C_AUX_PR08 := 0;         // Auxiliary Value 8
C_AUX_PR09 := 0;         // Auxiliary Value 9
C_AUX_PR10 := 0;         // Auxiliary Value 10
C_MSG_EVID := 0;         // Message ID
C_BA_EN := 0;            // Batch Enable
C_OCCUPIED := 0;         // Occupied by Batch
C_BA_ID := 0;            // Batch ID
C_BA_NA := '';           // Batch Name
C_STEP_NO := 0;          // Batch Step Number
C_RUNUPCYC := 3;         // Lag: Number of Run Up Cycles
C_START_OFF := 1;        // 1=Start up with Motor OFF
C_FAULT_OFF := 1;        // 1=In case of fault: Motor OFF
C_MSS_OFF := 1;          // 1=In case of MSS fault: Motor OFF
C_USTATUS := 0;          // User STATUS Bits
C_QERR := 1;             // 1=Error
C_QMSS_ST := 0;          // Unacknowledged Motor Protective Switch
C_QMON_ERR := 0;         // 1=Monitoring Error
C_QGR_ERR := 0;          // 1=Group Error
C_QOP_ERR := 0;          // 1=Operator Error
C_QRUN := 0;             // Status: 1=Motor running
C_QSTOP := 0;            // Status: 1=Motor STOP
C_QSTART := 0;           // Control Output 1=START Active
C_QC_QSTART := 16#80;    // Quality Code for Output QSTART
C_QON_OP := 0;           // Status: 1=Operator enabled for "ON"
C_QOFF_OP := 0;          // Status: 1=Operator enabled for "OFF"
C_QMAN_AUT := 0;         // 1=AUTO, 0=MANUAL Mode
C_QMANOP := 0;           // Status: 1=Oper. enabled for "MANUAL" Mode
C_QAUTOP := 0;           // Status: 1=Operator enabled for "AUTO" Mode
C_QMSG_ERR := 0;         // 1=Message ERROR
C_QMSG_SUP := 0;         // 1=Message Suppression Active
C_MSG_STAT := 0;         // Message: STATUS Output
C_MSG_ACK := 0;          // Message: ACK_STATE output
C_VSTATUS := 0;          // Status word
END_CONST
```

```
// Static Variables
VAR
sbI_OD1:        BOOL := 0;       // Flag of old operating value for OP_D.1
sbI_OD2:        BOOL := 0;       // Flag of old operating value for OP_D.2
sbQ_OD1:        BOOL := 0;       // Binary output for OP_D.1
sbALT_LINK_I_OT1:  BOOL := 0;  // Old value of interconnectable input for OP_TRIG.1
sbALT_QSTART:       BOOL := 0;  // Historical process data for QSTART
 sb_SIG_1: BOOL     := FALSE;   // ALARM_8P signal 1 flag
 sb_SIG_2: BOOL     := FALSE;   // ALARM_8P signal 2 flag
 sb_SIG_3: BOOL     := FALSE;   // ALARM_8P signal 3 flag
srAktZeit:      REAL := 0;       // Time passed
siRUNUPCNT:         INT  := 0;  // Counter for RUNUPCYC editing

//------------------------ ALARM_8P.1 -----------------------------
ALARM_8P_1:         ALARM_8P;   // Multiple instanced ALARM_8P
dwDUMMY:    DWORD := 0;         // Stand-by
siBA_ID:   DWORD := 0;          // Old value BA_ID
sbyBA_NA:  ARRAY[1..32] OF BYTE := 32(0);

VSTATUS_LOC : DWORD :=16#0;  // Local variable, into which the VSTATUS output is copied
STEP_NO_LOC : DWORD;         // Local variable, into which the STEP_NO input is copied
BA_ID_LOC : DWORD;           // Local variable, into which the BA_ID input is copied

// Variables for STATUSWORT "VSTATUS" and "USTATUS"
VSTATUS_STR AT VSTATUS_LOC :  STRUCT
                                USTATUS : WORD;
                                VSTATUS_LOW_BIT_8 : BOOL;
                                VSTATUS_LOW_BIT_9 : BOOL;
                                VSTATUS_LOW_BIT_10 : BOOL;
                                VSTATUS_LOW_BIT_11 : BOOL;
                                VSTATUS_LOW_BIT_12 : BOOL;
                                VSTATUS_LOW_BIT_13 : BOOL;
                                VSTATUS_LOW_BIT_14 : BOOL;
                                VSTATUS_LOW_BIT_15 : BOOL;
                                VSTATUS_LOW_BIT_0 : BOOL;
                                VSTATUS_LOW_BIT_1 : BOOL;
                                VSTATUS_LOW_BIT_2 : BOOL;
                                VSTATUS_LOW_BIT_3 : BOOL;
                                VSTATUS_LOW_BIT_4 : BOOL;
                                VSTATUS_LOW_BIT_5 : BOOL;
                                VSTATUS_LOW_BIT_6 : BOOL;
                                VSTATUS_LOW_BIT_7 : BOOL;
                               END_STRUCT;
END_VAR

// Temporary variables
VAR_TEMP
TOP_SI:    STRUCT
   EV_CLASS:    BYTE;
   EV_NUM:      BYTE;
   PRIORITY:    BYTE;
   NUM:         BYTE;
   TYP2_3:      BYTE;
   TYP1:        BYTE;
   ZI1:         WORD;
   ZI2_3:       DWORD;
   END_STRUCT;
START_UP_SI:    STRUCT
   EV_CLASS:    BYTE;
   EV_NUM:      BYTE;
   PRIORITY:    BYTE;
   NUM:         BYTE;
   TYP2_3:      BYTE;
   TYP1:        BYTE;
   ZI1:         WORD;
   ZI2_3:       DWORD;
   END_STRUCT;
ERR :              INT;             // Startup error
pbLINK_ON_OD1:     BOOL;            // Operating/Interconnection flag for OP_D.1.
pbLINK_I_OD1:      BOOL;            // Interconnectable input for OP_D.1.
pbQOP_ERR:         BOOL;            // Operating error pointer for OP_D.1.
pbLINK_I_OT1:      BOOL;            // Interconnectable input for OP_TRIG.1.
pbQ_OT1:           BOOL;            // Binary output for OP_TRIG.1.
pbALARM:           BOOL;            // Call up ALARM_8P
pbM_SUP:           BOOL;            // Message suppression
END_VAR
```

```
BEGIN
    // Supply of the VSTATUS, resave STEP_NO as STEP_NO_LOC,
      // Resave BA_ID as BA_ID_LOC
    // Supply of the VSTATUS output variable with the inputs.
    VSTATUS_STR.VSTATUS_LOW_BIT_0 := OCCUPIED;
    VSTATUS_STR.VSTATUS_LOW_BIT_1 := BA_EN;
    // Supply of the VSTATUS output variable (HIGH BYTE) with the values
      // that come from the user via the input variable USTATUS (from external).
    VSTATUS_STR.USTATUS := USTATUS;
    // Resave STEP_NO as STEP_NO_LOC and resave BA_ID as BA_ID_LOC
    STEP_NO_LOC := STEP_NO; // Resave, due to output of STEP_NO_LOC in ALARM_8P
    BA_ID_LOC := BA_ID;     // Resave, due to output of BA_ID_LOC in ALARM_8P
    ERR := RD_SINFO (TOP_SI := TOP_SI, START_UP_SI := START_UP_SI); // Read out
                                                                    // start info
    // pbM_SUP := FALSE;
    IF (TOP_SI.NUM = 100) AND START_OFF THEN
        siRUNUPCNT := RUNUPCYC;     // Save the value of the RUNUPCYC input
        // Initial states
        // Outputs to be set
        RESET     := C_RESET;
        MAN_ON    := C_MAN_ON;
        AUT_ON_OP := C_AUT_ON_OP;
        QERR      := C_QERR;
        QMSS_ST   := C_QMSS_ST;
        QMON_ERR  := C_QMON_ERR;
        QGR_ERR   := C_QGR_ERR;
        QOP_ERR   := C_QOP_ERR;
        QRUN      := C_QRUN;
        QSTOP     := C_QSTOP;
        QSTART    := C_QSTART;
        QC_QSTART := QC_QSTART_I; // Quality output described with value of input
        QON_OP    := C_QON_OP;
        QOFF_OP   := C_QOFF_OP;
        QMAN_AUT  := C_QMAN_AUT;
        QMANOP    := C_QMANOP;
        QAUTOP    := C_QAUTOP;
        QMSG_ERR  := C_QMSG_ERR;
                                  // QMSG_SUP := C_QMSG_SUP;
        MSG_STAT  := C_MSG_STAT;
        MSG_ACK   := C_MSG_ACK;
        // Static variables to be set
        sbI_OD1 := C_MAN_ON;         // Reset flag of old operating value for OP_D.1.
                                       // to zero
        sbI_OD2 := C_AUT_ON_OP;      // Reset flag of old operating value for OP_D.2.
                                       // to zero
        sbQ_OD1 := 0;                // Has to be set for incorrect operator
        sbALT_LINK_I_OT1 := L_RESET; // Previous value set by LINK_I
        sbALT_QSTART := C_QSTART;    // Set previous STOP-Modi
        srAktZeit := 0;              // Set acttime to NULL (zero)
        pbALARM :=TRUE;              // Initialization ALARM_8P
    ELSE;
        // MOTOR Algorithm
        // OP_D.2        Select manual or automatic mode
        pbQOP_ERR := FALSE; // Is only set to 1 in fault scenarios
        IF LIOP_SEL THEN
            // Interconnection
            IF AUT_ON_OP = sbI_OD2 THEN
                ;// No  operation occurs
            ELSE;
                pbQOP_ERR := TRUE;  // Prohibited operator
            END_IF;
            AUT_ON_OP := AUT_L;
            // Write AUT_L back to operator input
            QMAN_AUT := AUT_L;  // Interconnection
            QMANOP := FALSE;    // No operator allowed
            QAUTOP := FALSE;
        ELSE;
            // Operator
            IF (sbI_OD2 <> AUT_ON_OP) AND
               NOT ((AUT_ON_OP AND AUTOP_EN) OR (NOT(AUT_ON_OP) AND MANOP_EN)) THEN
                // Prohibited operator
                pbQOP_ERR := TRUE;
                AUT_ON_OP := sbI_OD2;
            ELSE;                // No or allowed operator
                QMAN_AUT := AUT_ON_OP;
            END_IF;
            QMANOP := MANOP_EN;
```

```
    // Copy the inputs to the outputs
    QAUTOP := AUTOP_EN;
END_IF;
sbI_OD2 := AUT_ON_OP;               // Note old operating values
// OP_TRIG.1.
pbLINK_I_OT1 := L_RESET AND MSS;    // Link_I for OP_TRIG.1.( RESET )
// Activating RESET
pbQ_OT1 := FALSE;           // Reset Q
IF ( pbLINK_I_OT1 AND ( NOT sbALT_LINK_I_OT1 ) )   THEN
    pbQ_OT1 := TRUE;        // Set pbQ_OT1
ELSE;
    IF RESET THEN
        pbQ_OT1 := TRUE;    // Set pbQ_OT1
    END_IF;
END_IF;
sbALT_LINK_I_OT1 := pbLINK_I_OT1;   // Save previous values
RESET := FALSE;
// Motor protector circuit-breaker
// Establish motor protector circuit-breaker
QMSS_ST := QMSS_ST AND (NOT pbQ_OT1) OR (NOT MSS);
// Watchdog   - 1 -
IF pbQ_OT1  THEN        // If RESET came in this cycle,
                          // The monitor error has to be corrected
    QMON_ERR := 0;
END_IF;
// Inputs for OP_D.1.(manual mode)
pbLINK_ON_OD1 := LOCK OR LOCK_ON OR QMAN_AUT OR (QMSS_ST AND MSS_OFF)
                  OR (QMON_ERR AND FAULT_OFF);
pbLINK_I_OD1 :=  (LOCK_ON OR (AUTO_ON AND (NOT(QMON_ERR AND FAULT_OFF))))
                  AND (NOT LOCK) AND (NOT(QMSS_ST AND MSS_OFF));
// OP_D.1.
// Outputs of the manual mode for control
IF pbLINK_ON_OD1 THEN
    // Interconnection
    IF MAN_ON = sbI_OD1 THEN
        ; // No  operation occurred
    ELSE;
        pbQOP_ERR := TRUE;
        // Prohibited operator
    END_IF;
    MAN_ON := pbLINK_I_OD1;  // Write pbLINK_I_OD1 back to operator input
    sbQ_OD1 := pbLINK_I_OD1; // Interconnection
    QON_OP := FALSE;          // No operator allowed
    QOFF_OP := FALSE;
ELSE;   // Operator
    IF (sbI_OD1 <> MAN_ON) AND
       NOT ((MAN_ON AND ON_OP_EN) OR (NOT(MAN_ON) AND OFFOP_EN)) THEN
        // Prohibited operator
        pbQOP_ERR := TRUE;
        MAN_ON := sbI_OD1;
    ELSE;   // No or allowed operator
            sbQ_OD1 := MAN_ON;
    END_IF;
    QON_OP := ON_OP_EN;     // Copy the inputs to the outputs
    QOFF_OP := OFFOP_EN;
END_IF;
sbI_OD1 := MAN_ON;          //Note old operating values
QOP_ERR := pbQOP_ERR;       // Operator error
// Watchdog   - 2 -
// Switch the motor on or off within a configured timespan.
IF (( sbALT_QSTART <> QSTART ) AND ( NOT QMON_ERR ))
   // Change through control
THEN
    IF TIME_MON < SAMPLE_T   THEN
        srAktZeit := SAMPLE_T;
    ELSE;
        srAktZeit := TIME_MON;
        // Set acttime
    END_IF;
END_IF;
IF srAktZeit >= SAMPLE_T THEN
    srAktZeit := srAktZeit - SAMPLE_T; // Bring down time
END_IF;
```

```
        IF MONITOR THEN
           // If the monitor is active, the change from Q_START has to be written
             // immediately to the output.
            IF FB_ON = QSTART THEN    // Before the monitor time expires,
                                        // same Feedback and QSTART
                srAktZeit := 0;
                IF QSTART THEN       // Report the corresponding feedback status
                    QRUN := 1;
                ELSE;
                    QSTOP := 1;
                END_IF;
            END_IF;
        END_IF;
        IF srAktZeit < SAMPLE_T  THEN // If the monitor time expires, does the  <=
            IF MONITOR  THEN          // watchdog then < instead function?
                IF ( FB_ON <> QSTART ) AND ( NOT pbQ_OT1 ) THEN
                   // After the monitor time, feedback and QSTART are different
                    QMON_ERR := 1;
                ELSE; END_IF;
            ELSE;
                IF QSTART THEN  // Set QRUN and QSTOP
                    QRUN := 1;
                    QSTOP := 0;
                ELSE;
                    QRUN := 0;
                    QSTOP := 1;
                END_IF;
            END_IF;
        END_IF;
        // Control signal
        sbALT_QSTART := QSTART;
        QGR_ERR := QMON_ERR OR QMSS_ST OR CSF;
        QSTART := sbQ_OD1 AND ((NOT(QMON_ERR AND FAULT_OFF)) OR LOCK_ON);
        QC_QSTART := QC_QSTART_I; // Describe quality output with value of input.
        // Watchdog   - 3 -
        // Set QRUN and QSTOP
        IF QSTART THEN  // Adapt QRUN and QSTOP to the previous value of QSTART.
            QSTOP := 0;
        ELSE;
            QRUN := 0;
        END_IF;
        IF siRUNUPCNT = 0 THEN
            // Initialize alarm
            // pbALARM := (sb_SIG_1 <> QMSS_ST)
            // OR (sb_SIG_2 <>  QMON_ERR) OR (sb_SIG_3 <>  CSF);
            pbALARM :=TRUE;   // Initialization ALARM_8P
        ELSE;
            siRUNUPCNT := siRUNUPCNT - 1;
            QMSG_SUP := TRUE;
            pbALARM := FALSE; // Initialization no ALARM
        END_IF;
    END_IF;
    IF pbALARM THEN           // Call up ALARM_8P
        IF siBA_ID <> BA_ID_LOC THEN
            // STRING variables may not be interconnected to ALARM8_P as auxiliary
              // process values, therefore transferred in ARRAY OF BYTE.
            FOR ERR := 1 TO 32
             DO
                sbyBA_NA[ERR] := 0;    // Delete array as default
            END_FOR;
            ERR := BLKMOV (SRCBLK:= BA_NA,DSTBLK:=sbyBA_NA);
            siBA_ID := BA_ID_LOC;      // Save modified BA_ID
        END_IF;
```

```
        // Call up ALARM_8P with lock logic (MSG_LOCK)
        ALARM_8P_1(    EN_R := TRUE,   // Update the ACKL_STATE output
                       ID := 16#EEEE,  // PMC communication channel
           EV_ID:= MSG_EVID,
           SIG_1:= QMSS_ST,
           SIG_2:= QMON_ERR,
           SIG_3:= CSF,
           SIG_4:= 0,
           SIG_5:= 0,
           SIG_6:= 0,
           SIG_7:= 0,
           SIG_8:= 0,
           SD_1 := sbyBA_NA,
           SD_2 := STEP_NO_LOC,
           SD_3 := BA_ID_LOC,
           SD_4 := AUX_PR04,
           SD_5 := AUX_PR05,
           SD_6 := AUX_PR06,
           SD_7 := AUX_PR07,
           SD_8 := AUX_PR08,
           SD_9 := AUX_PR09,
           SD_10 := AUX_PR10);
        QMSG_ERR := ALARM_8P_1.ERROR;
        MSG_STAT := ALARM_8P_1.STATUS;
        MSG_ACK := ALARM_8P_1.ACK_STATE;
        sb_SIG_1:= QMSS_ST;           // Note historical signals.
        sb_SIG_2:= QMON_ERR;
        sb_SIG_3:= CSF;
        QMSG_SUP := (MSG_STAT = 21);
    END_IF;
    QERR := NOT OK;
    // Power supply of the VSTATUS output variable with the outputs.
    VSTATUS_STR.VSTATUS_LOW_BIT_3 := QMAN_AUT;
    VSTATUS_STR.VSTATUS_LOW_BIT_5 := QMSS_ST;
    VSTATUS_STR.VSTATUS_LOW_BIT_7 := QMON_ERR;
    VSTATUS_STR.VSTATUS_LOW_BIT_8 := LOCK;
    VSTATUS_STR.VSTATUS_LOW_BIT_9 := QRUN;
    VSTATUS_STR.VSTATUS_LOW_BIT_10 := QSTOP;
    VSTATUS_STR.VSTATUS_LOW_BIT_14 := QMSG_SUP;
    VSTATUS_STR.VSTATUS_LOW_BIT_15 := OOS;
    VSTATUS := VSTATUS_LOC;
END_FUNCTION_BLOCK
```

A.3 Valve

```
// @(#) $Header::   L:/PCS7/PCS7LibV51.SW/TechBlocks/VCS/Algorithm/valv$
//Copyright (C) Siemens AG 1995. All Rights Reserved. Confidential

(*------------------------------------------------------------------------------

Order of the part blocks:

1. OP_D.2
2. OP_TRIG.1
3. Watchdog RESET Logic
4. SEL_BOOL.1
5. OP_D.1
6. Watchdog (remaining)
7. XOR
8. MESSAGE

------------------------------------------------------------------------------*)

//PCS 7 Library Vx.x
//Funktion: Single-Drive/Dual-Feedback Valve
//Etikett-Version 3.0
//Makro-Version :V0.92

FUNCTION_BLOCK "VALVE"
TITLE = 'Single-Drive/Dual-Feedback Valve'
//
{
        S7_tasklist:=        'OB100';
        S7_alarm_ui:=       '1';
        S7_m_c:=            'true'
}
AUTHOR:              TECHN61
NAME:                VALVE
VERSION:             '3.0'
FAMILY:              CONTROL
KNOW_HOW_PROTECT

VAR_INPUT
OOS  {S7_visible:='false';
        S7_link:='false';
        S7_m_c:='true';
        S7_string_0:='In Service';
        S7_string_1:='Out of Service'} :    BOOL := 0; // 1= Out of Service
V_LOCK   {S7_dynamic:='true';
        S7_m_c:='true'} :   BOOL := 0;     // 1=Lock to SAVE position
VL_OPEN  {S7_dynamic:='true';
        S7_m_c:='true'} :   BOOL := 0;     // 1=Lock to OPEN
VL_CLOSE     {S7_dynamic:='true';
        S7_m_c:='true'} :   BOOL := 0;     // 1=Lock to CLOSE
AUTO_OC   {S7_dynamic:='true';
        S7_contact:='true'} :   BOOL := 0; // AUTO Mode:1=Open, 0=Close
SS_POS   {S7_dynamic:='true';
        S7_edit:='para';
        S7_m_c:='true'} :   BOOL := 0;     // Safe Position. 1=Open, 0=Close
START_SS     {S7_visible:='false'} :    BOOL := 1; // 1=Start with Safe State Position
                                                   // and Manual Mode.
FAULT_SS     {S7_visible:='false'} :    BOOL := 1; // 1=In Case of fault: Safe State
                                                   // Position.
L_RESET  {S7_dynamic:='true';
        S7_contact:='true'} :   BOOL := 0; // Linkable Input RESET
CSF  {S7_dynamic:='true'} :     BOOL := 0; // Control System Fault 1=External Error
FB_OPEN  {S7_qc:='true';
        S7_dynamic:='true';
        S7_m_c:='true'} :   BOOL := 0;     // Feedback: 1=OPEN
QC_FB_OPEN : BYTE := 16#80;                // Quality Code for FB_OPEN
FB_CLOSE     {S7_qc:='true';
        S7_dynamic:='true';
        S7_m_c:='true'} :   BOOL := 0;     // Feedback: 1=CLOSE
QC_FB_CLOSE : BYTE := 16#80;               // Quality Code for FB_CLOSE
QC_QCONTROL_I : BYTE := 16#80;             // Quality Code for Input QCONTROL
NO_FB_OP     {S7_visible:='false'} :    BOOL := 0; // 1=No Feedback OPEN present
NO_FB_CL     {S7_visible:='false'} :    BOOL := 0; // 1=No Feedback CLOSE present
```

```
MONITOR  {S7_link:='false';
         S7_m_c:='true';
         S7_string_0:='Monitoring=Off';
         S7_string_1:='Monitoring=On'} :     BOOL := 1; // Select: 1=Monitoring ON,
                                                        // 0=Monitoring OFF
NOMON_OP      {S7_visible:='false'} :    BOOL := 0; // 1=No  Monitoring OPEN
NOMON_CL      {S7_visible:='false'} :    BOOL := 0; // 1=No  Monitoring CLOSE
OP_OP_EN      {S7_visible:='false'} :    BOOL := 1; // Enable 1=Operator may input OPEN
CL_OP_EN      {S7_visible:='false'} :    BOOL := 1; // Enable: 1=Operator may input CLOSE
MANOP_EN      {S7_visible:='false'} :    BOOL := 1; // Enable: 1=Operator may input
                                                    // MANUAL
AUTOP_EN      {S7_visible:='false'} :    BOOL := 1; // Enable: 1=Operator may input AUTO
LIOP_SEL      {S7_contact:='true'} :     BOOL := 0; // Select: 1=Linking,
                                                    // 0=Operator Active
AUT_L    {S7_dynamic:='true';
         S7_contact:='true'} :   BOOL := 0;  // Linkable Input for MANUAL/AUTO Mode
TIME_MON      {S7_link:='false';
         S7_edit:='para';
         S7_m_c:='true';
         S7_shortcut:='Mon. Time';
         S7_unit:='s'} :     REAL := 3;           // Monitoring Time [s]
SAMPLE_T      {S7_visible:='false';
         S7_sampletime:='true'} :    REAL := 1;   // Sample Time [s]
MSG_EVID      {S7_visible:='false';
         S7_link:='false';
         S7_param :='false';
         S7_server:='alarm_archiv';S7_a_type:='alarm_8p'} :  DWORD := 0; // Message ID
BA_EN    {S7_visible:='false';
         S7_m_c:='true'} :   BOOL := 0;          // Batch Enable
OCCUPIED      {S7_visible:='false';
         S7_m_c:='true'} :   BOOL := 0;          // Occupied by Batch
BA_ID    {S7_visible:='false';
         S7_m_c:='true'} :   DWORD := 0;         // Batch ID
BA_NA    {S7_visible:='false';
         S7_m_c:='true'} :   STRING[32] := '';   // Batch Name
STEP_NO  {S7_visible:='false';
         S7_m_c:='true'} :   DWORD := 0;         // Batch Step Number
RUNUPCYC      {S7_visible:='false';
         S7_link:='false'} :     INT := 3;       // Lag: Number of Run Up Cycles
USTATUS  {S7_visible:='false'} :    WORD := 0;  // User STATUS Bits
END_VAR

VAR_IN_OUT
RESET    {S7_visible:='false';
         S7_link:='false';
         S7_m_c:='true';
         S7_string_0:='0';
         S7_string_1:='Error=Reset'} :   BOOL := 0; // Operator Input Error Reset
MAN_OC   {S7_visible:='false';
         S7_link:='false';
         S7_m_c:='true';
         S7_string_0:='Valve=Close';
         S7_string_1:='Valve=Open'} :    BOOL := 0; // Operator Input: 1=OPEN, 0=CLOSE
AUT_ON_OP    {S7_visible:='false';
         S7_link:='false';
         S7_m_c:='true';
         S7_string_0:='Mode=Manual';
         S7_string_1:='Mode=Auto'} :     BOOL := 0; // Operator Input Mode 1=AUTO,
                                                    // 0= MANUAL
AUX_PR04      {S7_visible:='false'} :    ANY;       // Auxiliary Value 4
AUX_PR05      {S7_visible:='false'} :    ANY;       // Auxiliary Value 5
AUX_PR06      {S7_visible:='false'} :    ANY;       // Auxiliary Value 6
AUX_PR07      {S7_visible:='false'} :    ANY;       // Auxiliary Value 7
AUX_PR08      {S7_visible:='false'} :    ANY;       // Auxiliary Value 8
AUX_PR09      {S7_visible:='false'} :    ANY;       // Auxiliary Value 9
AUX_PR10      {S7_visible:='false'} :    ANY;       // Auxiliary Value 10
END_VAR

VAR_OUTPUT
QERR     {S7_visible:='false';
         S7_m_c:='true'} :   BOOL := 1;      // 1=Error
QCONTROL    {S7_qc:='true';
         S7_dynamic:='true';
         S7_m_c:='true'} :   BOOL := 0;      // Control Output: 0=Standstill
QC_QCONTROL : BYTE := 16#80;                 // Quality Code Output for QCONTROL
QMON_ERR     {S7_dynamic:='true';
         S7_m_c:='true'} :   BOOL := 0;      // 1=Monitoring Error
QGR_ERR  {S7_dynamic:='true';
         S7_contact:='true'} :   BOOL := 0;  // 1=Group Error
```

```
QMAN_AUT     {S7_dynamic:='true';
        S7_contact:='true';
        S7_m_c:='true'} :   BOOL := 0;      // 1=AUTO, 0=MANUAL Mode
QOP_ERR  {S7_visible:='false';
        S7_dynamic:='true'} :    BOOL := 0; // 1=Operator Error
QOP_OP   {S7_visible:='false';
        S7_dynamic:='true';
        S7_m_c:='true'} :   BOOL := 0;      // Status: 1=Operator enabled for "OPEN"
QCL_OP   {S7_visible:='false';
        S7_dynamic:='true';
        S7_m_c:='true'} :   BOOL := 0;      // Status: 1=Operator enabled for "CLOSE"
QAUTOP   {S7_visible:='false';
        S7_dynamic:='true';
        S7_m_c:='true'} :   BOOL := 0;      // Status: 1=Operator enabled for "AUTO"
QMANOP   {S7_visible:='false';
        S7_dynamic:='true';
        S7_m_c:='true'} :   BOOL := 0;      // Status: 1=Oper. ena. for "MANUAL" Mode
QOPENING     {S7_dynamic:='true';
        S7_m_c:='true'} :   BOOL := 0;      // 1=Valve is Opening
QOPENED  {S7_dynamic:='true';
        S7_contact:='true';
        S7_m_c:='true'} :   BOOL := 0;      // 1=Valve is OPEN
QCLOSING     {S7_dynamic:='true';
        S7_m_c:='true'} :   BOOL := 0;      // 1=Valve is Closing
QCLOSED  {S7_dynamic:='true';
        S7_contact:='true';
        S7_m_c:='true'} :   BOOL := 0;      // 1=Valve is CLOSED
QMSG_ERR     {S7_visible:='false';
        S7_dynamic:='true'} :   BOOL := 0;  // 1=Message ERROR
QMSG_SUP     {S7_visible:='false';
        S7_dynamic:='true';
        S7_m_c:='true'} :   BOOL := 0;      // 1=Message Suppression Active
MSG_STAT     {S7_visible:='false';
        S7_dynamic:='true'} :   WORD := 0;  // Message: STATUS Output
MSG_ACK  {S7_visible:='false';
        S7_dynamic:='true'} :   WORD := 0;  // Message: ACK_STATE-output
VSTATUS  {S7_visible:='false';
        S7_m_c:='true'} :   DWORD := 0;     // Status-word
END_VAR

CONST
C_OOS := 0;                 // 1= Out of Service
C_V_LOCK := 0;              // 1=Lock to SAVE position
C_VL_OPEN := 0;             // 1=Lock to OPEN
C_VL_CLOSE := 0;            // 1=Lock to CLOSE
C_AUTO_OC := 0;             // AUTO Mode:1=Open, 0=Close
C_SS_POS := 0;              // Safe Position. 1=Open, 0=Close
C_START_SS := 1;            // 1=Start with Safe State Position and Manual Mode
C_FAULT_SS := 1;            // 1=In Case of Fault: Safe State Position
C_RESET := 0;               // Operator Input Error Reset
C_L_RESET := 0;             // Linkable Input RESET
C_CSF := 0;                 // Control System Fault 1=External Error
C_FB_OPEN := 0;             // Feedback: 1=OPEN
C_QC_FB_OPEN := 16#80;      // Quality Code for FB_OPEN
C_FB_CLOSE := 0;            // Feedback: 1=CLOSE
C_QC_FB_CLOSE := 16#80;     // Quality Code for FB_CLOSE
C_QC_QCONTROL_I := 16#80;   // Quality Code for Input QCONTROL
C_NO_FB_OP := 0;            // 1=No Feedback OPEN present
C_NO_FB_CL := 0;            // 1=No Feedback CLOSE present
C_MONITOR := 1;             // Select: 1=Monitoring ON, 0=Monitoring OFF
C_NOMON_OP := 0;            // 1=No  Monitoring OPEN
C_NOMON_CL := 0;            // 1=No  Monitoring CLOSE
C_MAN_OC := 0;              // Operator Input: 1=OPEN, 0=CLOSE
C_OP_OP_EN := 1;            // Enable 1=Operator may input OPEN
C_CL_OP_EN := 1;            // Enable: 1=Operator may input CLOSE
C_AUT_ON_OP := 0;           // Operator Input Mode 1=AUTO, 0= MANUAL
C_MANOP_EN := 1;            // Enable: 1=Operator may input MANUAL
C_AUTOP_EN := 1;            // Enable: 1=Operator may input AUTO
C_LIOP_SEL := 0;            // Select: 1=Linking, 0=Operator Active
C_AUT_L := 0;               // Linkable Input for  MANUAL/AUTO Mode
C_TIME_MON := 3;            // Monitoring Time [s]
C_SAMPLE_T := 1;            // Sample Time [s]
C_AUX_PR04 := 0;            // Auxiliary Value 4
C_AUX_PR05 := 0;            // Auxiliary Value 5
C_AUX_PR06 := 0;            // Auxiliary Value 6
C_AUX_PR07 := 0;            // Auxiliary Value 7
C_AUX_PR08 := 0;            // Auxiliary Value 8
C_AUX_PR09 := 0;            // Auxiliary Value 9
```

```
C_AUX_PR10 := 0;            // Auxiliary Value 10
C_MSG_EVID := 0;            // Message ID
C_BA_EN := 0;               // Batch Enable
C_OCCUPIED := 0;            // Occupied by Batch
C_BA_ID := 0;               // Batch ID
C_BA_NA := '';              // Batch Name
C_STEP_NO := 0;             // Batch Step Number
C_RUNUPCYC := 3;            // Lag: Number of Run Up Cycles
C_USTATUS := 0;             // User STATUS Bits
C_QERR := 1;                // 1=Error
C_QCONTROL := 0;            // Control Output: 0=Standstill
C_QC_QCONTROL := 16#80;     // Quality Code Output for QCONTROL
C_QMON_ERR := 0;            // 1=Monitoring Error
C_QGR_ERR := 0;             // 1=Group Error
C_QMAN_AUT := 0;            // 1=AUTO, 0=MANUAL Mode
C_QOP_ERR := 0;             // 1=Operator Error
C_QOP_OP := 0;              // Status: 1=Operator enabled for "OPEN"
C_QCL_OP := 0;              // Status: 1=Operator enabled for "CLOSE"
C_QAUTOP := 0;              // Status: 1=Operator enabled for "AUTO"
C_QMANOP := 0;              // Status: 1=Oper. ena. for "MANUAL" Mode
C_QOPENING := 0;            // 1=Valve is Opening
C_QOPENED := 0;             // 1=Valve is OPEN
C_QCLOSING := 0;            // 1=Valve is Closing
C_QCLOSED := 0;             // 1=Valve is CLOSED
C_QMSG_ERR := 0;            // 1=Message ERROR
C_QMSG_SUP := 0;            // 1=Message Suppression Active
C_MSG_STAT := 0;            // Message: STATUS Output
C_MSG_ACK := 0;             // Message: ACK_STATE-output
C_VSTATUS := 0;             // Status-word
END_CONST

// Static variables
VAR
sbRESTART: BOOL := 1;           // First run(Default is1)
 sb_SIG_1: BOOL    := FALSE;    // ALARM_8P signal 1 flag
 sb_SIG_2: BOOL    := FALSE;    // ALARM_8P signal 2 flag

//------------------------- OP_D.2 -----------------------------
sbAUT_ON_OP:   BOOL := C_AUT_ON_OP;  // Flag of old operating value for(I0) (0=MAN)

//------------------------- OP_D.1 -----------------------------
sbMAN_OC:  BOOL := C_MAN_OC;         // Flag of old operating value for(I0) (0=CLOSED)
sbD1_Q_OC: BOOL := FALSE;            // Q Ausgang

//------------------------- OP_TRIG.1  ---------------------------
sbL_RESET: BOOL := C_L_RESET;        // Flag for historical signal L_RESET (LINK_I)

//------------------------- Watchdog-----------------------------
sbOC:      BOOL := FALSE;            // Old values direction
srAKTZEIT: REAL := 0;                // Current time of the on-delay
siRUNUPCNT:    INT := 0;             // Counter for RUNUPCYC editing

//------------------------- ALARM_8P.1 ---------------------------
ALARM_8P_1:    ALARM_8P;           // Multi-instanced ALARM_8P
dwDUMMY: DWORD := 0;               // Stand-by
siBA_ID:   DWORD := 0;             // Old value BA_ID
sbyBA_NA:  ARRAY[1..32] OF BYTE := 32(0);

VSTATUS_LOC : DWORD := 16#0; // Local variable, into which the VSTATUS output is copied.
STEP_NO_LOC : DWORD;       // Local variable, into which the STEP_NO input is copied.
BA_ID_LOC : DWORD;         // Local variable, into which the BA_ID input is copied.

// Variables for STATUSWORT "VSTATUS" and "USTATUS"
VSTATUS_STR AT VSTATUS_LOC :  STRUCT
                                USTATUS : WORD;
                                VSTATUS_LOW_BIT_8 : BOOL;
                                VSTATUS_LOW_BIT_9 : BOOL;
                                VSTATUS_LOW_BIT_10 : BOOL;
                                VSTATUS_LOW_BIT_11 : BOOL;
                                VSTATUS_LOW_BIT_12 : BOOL;
                                VSTATUS_LOW_BIT_13 : BOOL;
                                VSTATUS_LOW_BIT_14 : BOOL;
                                VSTATUS_LOW_BIT_15 : BOOL;
                                VSTATUS_LOW_BIT_0 : BOOL;
                                VSTATUS_LOW_BIT_1 : BOOL;
                                VSTATUS_LOW_BIT_2 : BOOL;
                                VSTATUS_LOW_BIT_3 : BOOL;
                                VSTATUS_LOW_BIT_4 : BOOL;
                                VSTATUS_LOW_BIT_5 : BOOL;
```

```
                                VSTATUS_LOW_BIT_6 : BOOL;
                                VSTATUS_LOW_BIT_7 : BOOL;
                              END_STRUCT;
END_VAR

// Temporary variables
VAR_TEMP
TOP_SI:    STRUCT
   EV_CLASS:   BYTE;
   EV_NUM:     BYTE;
   PRIORITY:   BYTE;
   NUM:        BYTE;
   TYP2_3:     BYTE;
   TYP1:       BYTE;
   ZI1:        WORD;
   ZI2_3:      DWORD;
   END_STRUCT;
START_UP_SI:   STRUCT
   EV_CLASS:   BYTE;
   EV_NUM:     BYTE;
   PRIORITY:   BYTE;
   NUM:        BYTE;
   TYP2_3:     BYTE;
   TYP1:       BYTE;
   ZI1:        WORD;
   ZI2_3:      DWORD;
   END_STRUCT;
ERR :      INT;        // Startup error
pbALARM:   BOOL;       // Call up ALARM_8P
pbM_SUP:   BOOL;       // Message suppression
//--------------------------- SEL_BOOL.1 -------------------------
pbSEL_BOOL_Q:      BOOL;   // SEL_BOOL 1 output
//--------------------------- OP_D.1 and 2 -------------------------
pbQOP_ERR:     BOOL;       // Temporary output value
//--------------------------- OP_TRIG.1   -------------------------
pbRESET:       BOOL;       // RESET (Ausgang Q)
//--------------------------- Watchdog -------------------------
pbQMON_ERR:        BOOL;   // Watchdog error
pbVerfahren:       BOOL;   // Valve travel
//--------------------------- Feedback -------------------------
pbFB_OPEN:     BOOL;
pbFB_CLOSE:    BOOL;
END_VAR
BEGIN
    // Write VSTATUS, resave STEP_NO as STEP_NO_LOC, resave BA_ID as
      //BA_ID_LOC.
    // Supply of the VSTATUS output variable with the inputs.
    VSTATUS_STR.VSTATUS_LOW_BIT_0 := OCCUPIED;
    VSTATUS_STR.VSTATUS_LOW_BIT_1 := BA_EN;
    // Supply of the VSTATUS output variable (HIGH BYTE) with the values
      // that come from the user via the input variable USTATUS (from external).
    VSTATUS_STR.USTATUS := USTATUS;
    //Resave STEP_NO as STEP_NO_LOC und resave BA_ID as BA_ID_LOC
    STEP_NO_LOC := STEP_NO; // Resave, due to output of STEP_NO_LOC in ALARM_8P
    BA_ID_LOC := BA_ID;     // Resave, due to output of BA_ID_LOC in ALARM_8P
    ERR := RD_SINFO (TOP_SI := TOP_SI, START_UP_SI := START_UP_SI); // Read out
                                                                    // start info.
    pbM_SUP := FALSE;
    IF TOP_SI.NUM = 100  OR sbRESTART THEN  // Startup or first run
        sbRESTART := FALSE;                 // No first run anymore
        QMON_ERR := C_QMON_ERR;             // No error
        pbALARM :=TRUE;                     // Initialization ALARM_8P
        IF START_SS  THEN                   // At Start SafeStatePosition?
            AUT_ON_OP := false;             // Manual mode
            sbAUT_ON_OP := false;           // Manual mode
            sbMAN_OC := SS_POS;             // Idle position
            MAN_OC := SS_POS;               // Write back idle position
            sbD1_Q_OC := SS_POS;            // Idle position
            srAKTZEIT := 0;                 // Initialize watchdogs time
            QMAN_AUT := C_QMAN_AUT;         // Manual
            QOPENING := C_QOPENING;         // No movement
            QCLOSING := C_QCLOSING;
            sbL_RESET := C_L_RESET;         // No RESET
            RESET := C_RESET;               // No RESET
            QCONTROL := C_QCONTROL;         // Idle position
            QC_QCONTROL := QC_QCONTROL_I;   // Describe quality output with values of
                                              // input.
```

```
            QMSG_ERR := C_QMSG_ERR;
            QMSG_SUP := C_QMSG_SUP;
            MSG_STAT := C_MSG_STAT;
            MSG_ACK  := C_MSG_ACK;
            QGR_ERR  := C_QGR_ERR;
            QCL_OP := C_QCL_OP;
            QOP_OP := C_QOP_OP;
            QOP_ERR := C_QOP_ERR;
            QMANOP := C_QMANOP;
            QAUTOP := C_QAUTOP;
        ELSE;
            IF (NOT AUT_ON_OP AND                    // Manual mode
                MONITOR)    THEN                     // Watchdog active
                IF NOT NO_FB_OP AND                  // Feedback OPEN available
                   FB_OPEN  THEN                     // Feedback available
                    MAN_OC := TRUE;                  // Track operator input
                    QCONTROL := (MAN_OC XOR SS_POS);
                    QC_QCONTROL := QC_QCONTROL_I;  // Describe quality output with
                                                     // values of  input.
                    QOPENED := TRUE;
                    QOPENING := FALSE;
                    QCLOSED := FALSE;
                    QCLOSING := FALSE;
                END_IF;
                IF NOT NO_FB_CL AND                  // Feedback CLOSE available
                   FB_CLOSE   THEN                   // Feedback available
                    MAN_OC := FALSE;                 // Track operator input
                    QCONTROL := (MAN_OC XOR SS_POS);
                    QC_QCONTROL := QC_QCONTROL_I;  // Describe quality output with
                                                     // values of input.
                    QOPENED := FALSE;
                    QOPENING := FALSE;
                    QCLOSED := TRUE;
                    QCLOSING := FALSE;
                END_IF;
            END_IF;
            QCL_OP := CL_OP_EN;
            QOP_OP := OP_OP_EN;
        END_IF;
    END_IF;
    IF TOP_SI.NUM = 100 THEN     // Initialization at startup
        siRUNUPCNT := RUNUPCYC;  // Save the value of the RUNUPCYC input
    ELSE;
        // VALVE algorithm
        pbQOP_ERR := FALSE;      // Optimistic initializations
        pbQMON_ERR := FALSE;
        pbVerfahren := FALSE;    // No OPEN/CLOSE operation
        // OP_D.2 (optimized)                          - 1 -
        IF LIOP_SEL THEN
            // Interconnection
            IF AUT_ON_OP = sbAUT_ON_OP THEN
                ;// No operation occurred
            ELSE; // Prohibited operator
                pbQOP_ERR := TRUE;
            END_IF;
            AUT_ON_OP := AUT_L;       // Write AUT_L back to operator input
                                      // (BTRACK is always1).
            QMAN_AUT := AUT_L;        // Interconnection
            QMANOP := FALSE;          // No operation allowed
            QAUTOP := FALSE;
        ELSE;         // Bedienung
            IF (sbAUT_ON_OP <> AUT_ON_OP) AND
               NOT ((AUT_ON_OP AND AUTOP_EN) OR (NOT(AUT_ON_OP) AND MANOP_EN)) THEN
                pbQOP_ERR := TRUE;          // Prohibited operator
                AUT_ON_OP := sbAUT_ON_OP;   // Write back old operating values
            ELSE;     // No or allowed operation
                QMAN_AUT := AUT_ON_OP;
            END_IF;
            QMANOP := MANOP_EN;         // Copy the inputs to the outputs
            QAUTOP := AUTOP_EN;
        END_IF;
        sbAUT_ON_OP := AUT_ON_OP;       //Note old operating values
        //OP_TRIG.1 (optimized)                       - 2 -
        IF ((L_RESET AND (NOT sbL_RESET)) OR RESET) THEN   // Rising edge was for
                                                           // L_RESET or RESET.
            pbRESET := TRUE;            // RESET
        ELSE;
            pbRESET := FALSE;           // No RESET
        END_IF;
```

```
sbL_RESET := L_RESET;           // Save previous values
RESET := FALSE;                 // Reset operator-controllable input
// Watchdog RESET Logic                              - 3 -
IF QMON_ERR  THEN               // QMON_ERR Flip-Flop
    IF NOT pbRESET THEN
        pbQMON_ERR := TRUE;
    ELSE;
        // pbQMON_ERR bleibt 0
        // pbRESET bleibt wirksam
        IF TIME_MON < SAMPLE_T  THEN // Reposition watchdog time
            srAKTZEIT := SAMPLE_T;
        ELSE;
            srAKTZEIT := TIME_MON;
        END_IF;
    END_IF;
ELSE;
    // RESET without error not effective
    pbRESET := FALSE;
END_IF;
IF V_LOCK OR VL_OPEN OR VL_CLOSE OR (pbQMON_ERR AND FAULT_SS)// SEL_BOOL.1 - 4 -
THEN
    pbSEL_BOOL_Q := SS_POS;          // Idle position
    IF (NOT V_LOCK) THEN
        IF VL_CLOSE THEN
            pbSEL_BOOL_Q := 0;       // Close valve
        ELSE;
            IF VL_OPEN THEN
                pbSEL_BOOL_Q := 1;  // Open valve
            END_IF;
        END_IF;
    END_IF;
ELSE;
    pbSEL_BOOL_Q := AUTO_OC;         // Automatic value
END_IF;
// OP_D.1 (optimized and expanded with LINK_ON calculation)          - 5 -
IF QMAN_AUT OR V_LOCK OR VL_OPEN OR VL_CLOSE OR (pbQMON_ERR AND FAULT_SS)
                                    // LINK_ON
THEN // Interconnection
    IF MAN_OC = sbMAN_OC THEN
        ; // Keine  Bedienung erfolgt
    ELSE;
        pbQOP_ERR := TRUE;             // Prohibited operator
    END_IF;
    IF sbD1_Q_OC <> pbSEL_BOOL_Q  THEN // Valve travel?
        pbVerfahren := TRUE;
        sbD1_Q_OC := pbSEL_BOOL_Q;     // LINK_I
    END_IF;
    MAN_OC := sbD1_Q_OC;    // Write output back to operator input
                              // (BTRACK is always1).
ELSE;   // Bedienung
    IF (sbMAN_OC <> MAN_OC) AND
       NOT ((MAN_OC AND OP_OP_EN) OR (NOT(MAN_OC) AND CL_OP_EN)) THEN
        pbQOP_ERR := TRUE;             // Prohibited operator
        MAN_OC := sbMAN_OC;
        // Write back old operating values
    ELSE;
        IF sbD1_Q_OC <> MAN_OC  THEN   // Valve travel?
            // Allowed operator
            pbVerfahren := TRUE;
            sbD1_Q_OC := MAN_OC;
        END_IF;
    END_IF;
END_IF;
sbMAN_OC := MAN_OC;    //Note old operating values
IF pbVerfahren         // Valve is traveled -> Reposition watchdog time
THEN
    IF TIME_MON < SAMPLE_T THEN
        srAKTZEIT := SAMPLE_T;
    ELSE;
        srAKTZEIT := TIME_MON;
    END_IF;
END_IF;
```

```
// Watchdog and position display                          - 6 -
IF NOT MONITOR  THEN                  // No feedback watchdogs
    IF (srAKTZEIT >= SAMPLE_T) THEN         // Timer is running
        srAKTZEIT := srAKTZEIT - SAMPLE_T; // Bring down time
        pbFB_OPEN := FALSE;
        pbFB_CLOSE := FALSE;
    ELSE;
        pbFB_OPEN := sbD1_Q_OC;
        pbFB_CLOSE := NOT sbD1_Q_OC;
    END_IF;
ELSE;
    IF NO_FB_OP  THEN   // No limit switch OPEN
        pbFB_OPEN := sbD1_Q_OC;
    ELSE;
        pbFB_OPEN := FB_OPEN;
    END_IF;
    IF NO_FB_CL  THEN   // No limit switch CLOSE
        pbFB_CLOSE := NOT sbD1_Q_OC;
    ELSE;
        pbFB_CLOSE := FB_CLOSE;
    END_IF;
END_IF;
IF MONITOR AND (((sbD1_Q_OC  AND NOT pbFB_OPEN) AND NOT NOMON_OP) OR
   ((NOT sbD1_Q_OC AND NOT pbFB_CLOSE) AND NOT NOMON_CL) OR
   (pbFB_OPEN AND pbFB_CLOSE) AND NOT (NOMON_OP AND NOMON_CL))THEN
    IF (srAKTZEIT < SAMPLE_T) THEN
        pbQMON_ERR := TRUE;
    END_IF;
    IF (srAKTZEIT >= SAMPLE_T) THEN         // Timer is running
        srAKTZEIT := srAKTZEIT - SAMPLE_T; // Bring down time
    END_IF;
ELSE;
    IF (sbD1_Q_OC  AND pbFB_OPEN) OR
       (NOT sbD1_Q_OC AND pbFB_CLOSE) THEN
        srAKTZEIT := 0;    // End position reached-> Reset watchdog time
    END_IF;
END_IF;
QMON_ERR := pbQMON_ERR;
IF QMAN_AUT OR V_LOCK OR VL_OPEN OR VL_CLOSE OR (QMON_ERR AND FAULT_SS) THEN
    // No OPEN/CLOSE operation allowed
    QOP_OP := FALSE;
    QCL_OP := FALSE;
ELSE;
    // Copy the inputs to the outputs
    QOP_OP := OP_OP_EN;
    QCL_OP := CL_OP_EN;
END_IF;
// Position displays
QOPENED  := sbD1_Q_OC AND pbFB_OPEN AND NOT pbFB_CLOSE;
QCLOSED  := NOT sbD1_Q_OC  AND pbFB_CLOSE AND NOT pbFB_OPEN;
IF QOPENED THEN
    QOPENING := FALSE;
ELSE;
    QOPENING := sbD1_Q_OC AND NOT QMON_ERR;
END_IF;
IF QCLOSED THEN
    QCLOSING := FALSE;
ELSE;
    QCLOSING := NOT sbD1_Q_OC  AND NOT QMON_ERR;
END_IF;
QOP_ERR  := pbQOP_ERR;              // Replace outputs
QGR_ERR  := QMON_ERR OR CSF;        // Calculate group monitor error
IF FAULT_SS AND (NOT(V_LOCK OR VL_OPEN OR VL_CLOSE))
THEN // Safe State Position at error?
    QCONTROL := (sbD1_Q_OC XOR SS_POS) AND NOT QMON_ERR;
    QC_QCONTROL := QC_QCONTROL_I;   // Describe quality output with values of
                                      // input
ELSE;
    QCONTROL := (sbD1_Q_OC XOR SS_POS);
    QC_QCONTROL := QC_QCONTROL_I;   // Describe quality output with values of
                                      // input
END_IF;
IF siRUNUPCNT = 0  THEN             // Initialize alarms
    // pbALARM := (sb_SIG_1 <> QMON_ERR) OR (sb_SIG_2 <>  CSF);
    pbALARM :=TRUE;                 // Initialization ALARM_8P
ELSE;
```

```
            pbALARM :=FALSE;                // Initialization no ALARM
            siRUNUPCNT := siRUNUPCNT - 1;
            pbM_SUP := TRUE;
        END_IF;
    END_IF;
    IF pbALARM THEN    // Call up ALARM_8P
        IF siBA_ID <> BA_ID_LOC THEN
            // STRING variables may not be interconnected to ALARM8_P as auxiliary
              // process values, therefore transferred in ARRAY OF BYTE.
            FOR ERR := 1 TO 32
             DO
                sbyBA_NA[ERR] := 0;    // Delete array as default
            END_FOR;
            ERR := BLKMOV (SRCBLK:= BA_NA,DSTBLK:=sbyBA_NA);
            siBA_ID := BA_ID_LOC;      // Save modified BA_ID
        END_IF;

        // Call up ALARM_8P with lock logic (MSG_LOCK)
        ALARM_8P_1(    EN_R := TRUE,   // Update the ACKL_STATE output
                       ID := 16#EEEE,  // PMC communication channel
           EV_ID:= MSG_EVID,
           SIG_1:= QMON_ERR,
           SIG_2:= CSF,
           SIG_3:= 0,
           SIG_4:= 0,
           SIG_5:= 0,
           SIG_6:= 0,
           SIG_7:= 0,
           SIG_8:= 0,
           SD_1 := sbyBA_NA,
           SD_2 := STEP_NO_LOC,
           SD_3 := BA_ID_LOC,
           SD_4 := AUX_PR04,
           SD_5 := AUX_PR05,
           SD_6 := AUX_PR06,
           SD_7 := AUX_PR07,
           SD_8 := AUX_PR08,
           SD_9 := AUX_PR09,
           SD_10 := AUX_PR10);
        QMSG_ERR := ALARM_8P_1.ERROR;
        MSG_STAT := ALARM_8P_1.STATUS;
        MSG_ACK := ALARM_8P_1.ACK_STATE;
        sb_SIG_1:= QMON_ERR;     // Note historical signals
        sb_SIG_2:= CSF;
    END_IF;
    IF (MSG_STAT = 21) THEN      // Block locked
        pbM_SUP := TRUE;
    END_IF;
    QMSG_SUP := pbM_SUP;
    QERR := NOT OK;              // Note negated OK-Flag result in the block

    //Power supply of the VSTATUS output variable with the outputs.
    VSTATUS_STR.VSTATUS_LOW_BIT_3 := QMAN_AUT;
    VSTATUS_STR.VSTATUS_LOW_BIT_7 := QMON_ERR;
    VSTATUS_STR.VSTATUS_LOW_BIT_8 := V_LOCK;
    VSTATUS_STR.VSTATUS_LOW_BIT_9 := QOPENED;
    VSTATUS_STR.VSTATUS_LOW_BIT_10 := QCLOSED;
    VSTATUS_STR.VSTATUS_LOW_BIT_11 := QOPENING;
    VSTATUS_STR.VSTATUS_LOW_BIT_12 := QCLOSING;
    VSTATUS_STR.VSTATUS_LOW_BIT_14 := QMSG_SUP;
    VSTATUS_STR.VSTATUS_LOW_BIT_15 := OOS;
    VSTATUS := VSTATUS_LOC;
END_FUNCTION_BLOCK
```

Glossar

A

Anwenderdefinierte Datentypen (UDT)

Anwenderdefinierte Datentypen sind vom Anwender erzeugte spezielle Datenstrukturen, die nach ihrer Definition im gesamten CPU-Programm verwendet werden können. Sie können wie elementare oder zusammengesetzte Datentypen in der Variablendeklaration von Codebausteinen (FC, FB, OB) verwendet werden oder als Vorlage für die Erstellung von Datenbausteinen mit gleicher Datenstruktur dienen.

AKZ

Anlagenkennzeichen: Wird gebildet aus den Namen der THO, die die Eigenschaft "namensbildend" haben und aus dem Namen des CFC-Plans und des Bausteins im CFC.

AS-Baustein

Objekt einer Bibliothek oder einer Bausteinstruktur, das einen Teil des S7-Anwenderprogramms enthält. Als AS-Bausteine werden die Bausteine bezeichnet, die in der CPU eines AS ablaufen.

Asynchrone OBs

Asynchrone OBs (Organisationsbausteine) werden vom Betriebssystem der CPU beim Auftreten asynchroner Ereignisse (z.B. Fehler) aufgerufen.

AUTHOR

➔ *Baustein-Attribut*:

Enthält bei Verwendung einer Bausteinbibliothek den Bibliotheksnamen. Wird zur Identifikation der zur Bibliothek passenden Hilfedatei verwendet.

AWL

Anweisungsliste: Die Anweisungsliste ist eine maschinennahe, textuelle Programmiersprache gemäß IEC 1131-3.

B

BATCH *flexible*

ab V6.0 ➔ SIMATIC BATCH.

Baustein-Attribut

Mit Hilfe der Baustein-Attribute (➔ *FUNCTION_BLOCK*, ➔ *TITLE*, ➔ *Liste der Systemattribute*, ➔ *AUTHOR*, ➔ *NAME*, ➔ *VERSION*, ➔ *FAMILY*, ➔ *KNOW_HOW_PROTECT*) des Bausteins können Sie die Objekteigenschaften Ihres Bausteins beeinflussen.

Bausteinkopf

1. Abschnitt des ➔ *AS-Bausteins* mit Verwaltungsinformationen (➔ *Baustein-Attribute*).
2. Oberer Teil des Bausteins in der grafischen Darstellung des CFC, der u.a. die Namen (Bausteintyp, Bausteinname), Kommentar und die Task-Zuordnung (Ablaufeigenschaft) enthält.

Bausteinsymbol

Symbolische Darstellung der wichtigsten Informationen eines ➔ *AS-Bausteins*. Über das Bausteinsymbol kann der zugehörige ➔ *Bildbaustein* aufgerufen werden.

Bedienen

Vorgang, bei dem der Anlagenfahrer Wert- bzw. Zustandsveränderungen bei einem Baustein veranlasst. In der Regel wird dies durch Eingaben an der OS eingeleitet, überprüft und an den Baustein im AS weitergeleitet. Dort erfolgt eine letzte Überprüfung vor der Zuweisung an den Baustein, weil sich in der Zeit zwischen dem OS-Senden und dem AS-Empfang die Prozessbedingungen verändern könnten.

Bedienberechtigung

Berechtigung des aktuellen Anlagenfahres zum ➔ *Bedienen* des ➔ *AS-Bausteinparameters*

Beobachten

Teil der Aufgaben einer OS, die die Visualisierung der Prozessparameter und Zustände in verschiedenen Formen (numerisch, grafisch) ermöglicht.

Bibliothek

Softwarepaket mit nach gemeinsamen Merkmalen zusammengefassten ➔ *AS-* und/oder ➔ *Bildbausteinen*.

Bildbaustein

Grafische Darstellung aller Elemente eines AS-Bausteins, die zum Bedienen und Beobachten auf einer OS vorgesehen sind.

C

CFC

Continuous Function Chart. Synonym für:

1. Funktionsplan, auf dem Bausteine platziert, verschaltet und parametriert werden können. Ein CFC-Plan besteht aus 1 bis 26 Teilplänen mit jeweils 6 Blättern.
2. Editor zur technologieorientierten, grafischen Projektierung der Automatisierungsaufgabe. Mit dem CFC-Editor wird aus vorgefertigten Bausteinen eine Gesamt-Softwarestruktur (CFC-Plan) erstellt.

CNT-Datei

Optionaler Bestandteil einer Online-Hilfe. Die CNT-Datei enthält das Inhaltsverzeichnis der Online-Hilfe.

Codeteil

Bestandteil eines Bausteins, der den Algorithmus des Bausteins enthält.

D

Deklarationsteil

Bestandteil eines Bausteins, der die Schnittstelle des Bausteins sowie seine intern verwendeten Daten definiert.

E

Erstlauf

Aus Bausteinsicht der Vorgang, bei dem der Baustein nach seiner Instanziierung zum ersten Mal bearbeitet wird. Anschließend befindet sich der Baustein in einem definierten Zustand bezüglich der Parameter bzw. Betriebsarten.

F

FAMILY

➔ *Baustein-Attribut*:

Enthält bei Verwendung einer Bausteinbibliothek einen gemeinsamen Namen für eine Teilmenge der Bausteine. FAMILY und ➔ NAME sind Teilschlüssel zur Suche des Hilfetextes eines Bausteins in der Online-Hilfe.

Funktion (FC)

In IEC 1131–3 festgelegt als Software-Einheit, die beim Ausführen ein einziges Ergebnis liefert (es kann auch ein strukturierter Datentyp sein) und keine speichernde Datenablage (Gedächtnis) hat. Der wesentliche Unterschied zum FB ist die fehlende Datenablage.

FUNCTION_BLOCK

➔ *Baustein-Attribut*:

Enthält den symbolischen Namen des Bausteins. Wird für die Anzeige des Namens im SIMATIC Manager und im CFC-Plan verwendet.

Funktionsbaustein (FB)

Ein Funktionsbaustein ist gemäß IEC 1131-3 ein Codebaustein mit statischen Daten. Mit dem FB bietet können im Anwenderprogramm Parameter übergeben werden. Dadurch eignen sich Funktionsbausteine zur Programmierung von häufig wiederkehrenden komplexen Funktionen, z.B. Regelungen, Betriebsartenanwahl. Da ein FB über ein Gedächtnis (Instanz-Datenbaustein) verfügt, kann auf dessen Parameter (z.B. Ausgänge) zu jeder Zeit an jeder beliebigen Stelle im Anwenderprogramm zugegriffen werden.

G

Global Script

Global Script ist innerhalb von ➔ *WinCC* der Oberbegriff für vom Anwender erzeugte C-Funktionen, die projektweit oder auch projektübergreifend verwendet werden können.

Graphics Designer

Grafischer Editor in ➔ *WinCC* zur Erstellung von Bildbausteinen.

K

KNOW_HOW_PROTECT

➔ *Baustein-Attribut*:

Das gesetzte Attribut schützt den Algorithmus des Bausteins gegen Einsichtnahme und Änderung, wenn die Quelle nicht im gleichen Programm vorhanden ist.

M

Meldeliste

Aus dem Runtimesystem von ➔ *WinCC* heraus besteht die Möglichkeit, Listen mit Meldungen anzeigen zu lassen und zu bearbeiten. Die in den Listen angezeigten Meldungen beziehen sich ausschließlich auf das aktuelle Projekt.

Multiinstanzbaustein

Baustein, der sich aus mehreren Bausteinen zusammensetzt. Seine Instanz (Datenablage) beinhaltet die Instanzen (Datenablagen) der in ihm zusammengefassten (aufgerufenen) FBs.

N

NAME

➔ *Baustein-Attribut*:

Enthält den symbolischen Namen des Bausteins; identisch mit ➔ FUNCTION_BLOCK. NAME und ➔ FAMILY sind Teilschlüssel zur Suche des Hilfetextes eines Bausteins in der Online-Hilfe.

O

OK-Flag

Das OK-Flag ist eine systeminterne Variable. Tritt während der Ausführung einer Operation ein Fehler auf z.B. Überlauf bei arithmetischen Operationen, so wird das OK-Flag vom System beeinflusst und an den Bausteinausgang ENO durchgereicht.

P

Prototypbild

Prototypbilder werden von ➔ *WinCC* dazu verwendet, bereits projektierte Bildkomponenten wieder zu verwenden. Die Technik der Prototypbilder arbeitet mit sogenannten Templatebildern, die mehrfach in ein oder mehrere Vaterbilder eingebunden werden. Ein Templatebild ist nur eine Schablone, die erst in einem echten Objekt zum Leben erweckt wird. Ein Objekt auf Basis einer Schablone (=Prototypbild) entsteht durch eine sogenannte Instanzierung. Es können mehrere Instanzen (d.h. echte Objekte) zu einer Schablone erstellt werden.

R

Registrierungsdatei

Eine ASCII-Datei (.reg) die alle Informationen enthält wie Pfad, Einsprungadressen für Online-Hilfe etc., um z.B. einen Baustein in der WINDOWS-Registrierung einzutragen. Über diese registrierten Informationen kann im CFC oder SIMATIC Manager für einen selektierten Baustein die Online-Hilfe in der gewünschten Landessprache aufgerufen werden.

S

SCL

Höhere Programmiersprache zur Formulierung technologischer Problemlösungen in der SIMATIC S7 (PASCAL-ähnlich), entsprechend der in IEC 1131–3 festgelegten Sprache ST (structured text).

Sicht

Darstellungsart eines Bildbausteins, in der bestimmte Werte eines AS-Bausteins angezeigt werden (z.B. Trendsicht, Meldesicht, Standardsicht usw.).

SIMATIC BATCH

Programmpaket für die Erstellung komplexer Rezeptsteuerungen für den gesamten Bereich von kleinen bis hin zu großen Anwendungen.

Split Screen Wizard

Bestandteil von ➔ *WinCC*: Initialisiert die Bildschirm- und Bildeinstellungen der OS.

Standardsicht

➔ *Sicht* eines ➔ *Bildbausteins*, bei der die wichtigsten Werte des zugehörigen ➔ *AS-Bausteins* visualisiert werden.

Startinfo

Die Startinformation ist Bestandteil eines Organisationsbausteins (OB). Sie informiert den S7-Anwender detailliert über das Ereignis, das den Aufruf des OB ausgelöst hat.

Systemattribute für Bausteine

Spezielle Attribute, die den ➔ *AS-Baustein* für die Verbindung mit der OS vorbereiten oder den Einbau des Bausteins in einen CFC-Plan beeinflussen. Beispiel: "S7_tasklist" enthält eine Liste der OBs in die der Baustein vom CFC eingebaut werden soll.

Systemattribute für Parameter

Spezielle Attribute, die die Darstellung des Parameters durch den ➔ *Bildbaustein* oder seine Behandlung im CFC beeinflussen. Beispiel: "S7_m_c" definiert, ob der Parameter von einer OS aus bedient oder beobachtet werden kann.

T

TITLE

➔ *Baustein-Attribut*:

Diese Information wird unter PCS 7 nicht ausgewertet, sie wird jedoch im SIMATIC Manager in den Objekteigenschaften des Bausteins im Kommentarfeld angezeigt.

Trendsicht

➔ *Sicht* eines ➔ *Bildbausteins*, bei der der zeitliche Verlauf der wichtigsten Werte des zugehörigen ➔ *AS-Bausteins* visualisiert wird.

U

UDT

➔ *Anwenderdefinierte Datentypen.*

V

VERSION

➔ *Baustein-Attribut*:

Enthält die Versionsnummer des Bausteins.

W

WinCC

Windows Control Center: Ein Software-Paket zur technologieorientierten grafischen Entwicklung von ➔ *Bildbausteinen* sowie zur Bedienung und Beobachtung des AS.

Index

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

V

W

Z