Automatisierungstechnik nach internationaler Norm programmieren (3)

Autor: Dr. Ulrich Becker
Fachzentrum Automatisierungstechnik und vernetzte Systeme im BTZ Rohr-Kloster
Mail: Ulrich.Becker@BTZ-Rohr.de

Folge 3: Einführung in die Handhabung von POE's und Bibliotheken

Um die steuertechnische Aufgabe 1 in Folge 1 IEC-gerecht zu lösen, wurden in Folge 2 in das Programmiersystem CoDeSys eingeführt. Viele Fragen, insbesondere die nach Nutzung von Funktionsbausteinen und Bibliotheken, blieben offen. In dieser Folge werden weitere Fragen beantwortet und das Programm ergänzt.

Der Programmcode als Ergebnis der bisherigen Arbeit

Bild 8 zeigt das Programm „Band_Grundfunktion" als Lösung von Aufgabe 1 in der Sprache FUP. Es wurde in einen Funktionsbaustein geschrieben und verwendet grafische Boolsche Operatoren. Im gleichen Baustein wurden auch alle erforderlichen Variablen deklariert und auf Adressen gelegt.
Neben Kommentaren, die durch die Zeichen (*.....*) eingeschlossen werden, erscheint in diesem Programm ein neuartiger Baustein für die Flankenauswertung der Lichtschranke vom Typ R_TRIG. Hierbei handelt es sich um einen instanzierten Funktionsbaustein aus der Standard-Bibliothek. Diese Fachbegriffe werden nachfolgend näher erklärt.

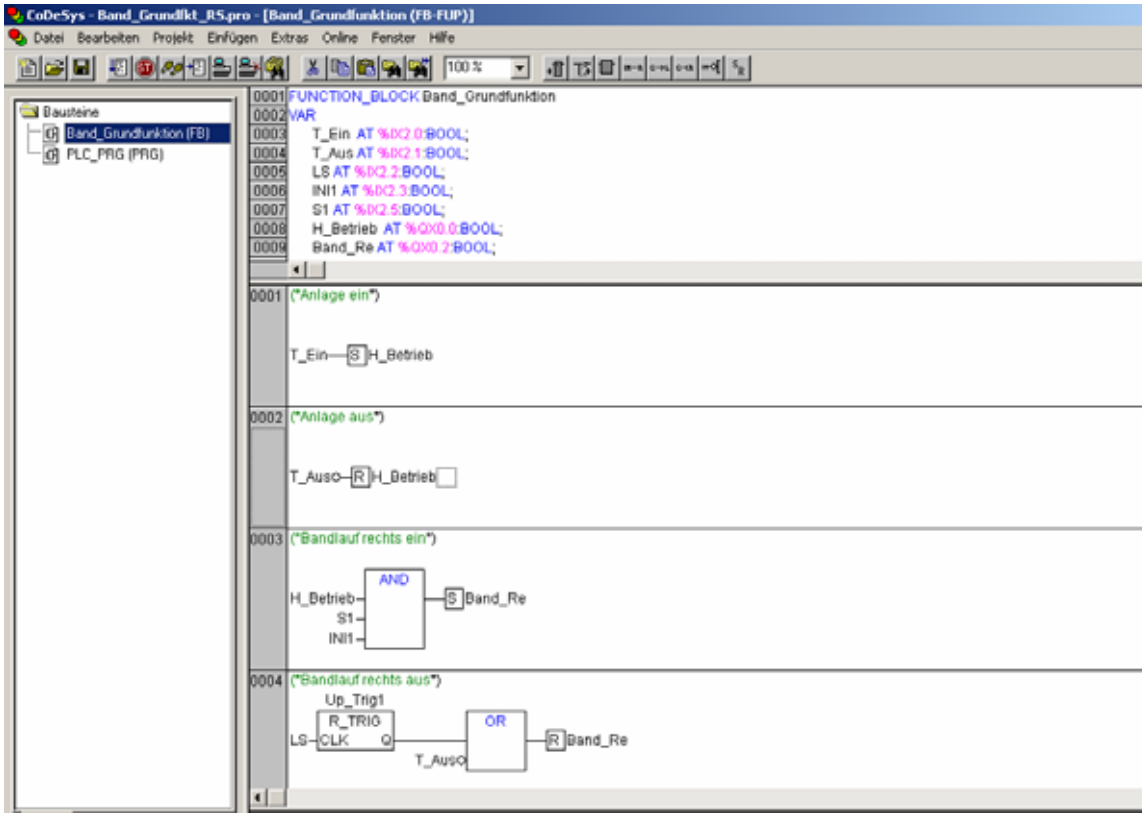

Bild 8: Programmcode „*Band_Grundfunktion*" in der Sprache FUP

Funktionen

Funktionen (FUN) sind POE, die bei gleichen Eingangsparametern stets dasselbe Ergebnis liefern. Sie haben genau einen Ausgangswert und keine internen Variablen, d.h. „kein Gedächtnis“. Der Ausgangswert wird auch als „Rückgabewert“ bezeichnet.
Alle Operatoren sind einfachste Funktionen. Funktionen können aber weit umfangreicher gestaltet werden. **Bild 9** zeigt eine Funktion mit 5 Eingangsvariablen und definitionsgemäß genau einem Rückgabewert.

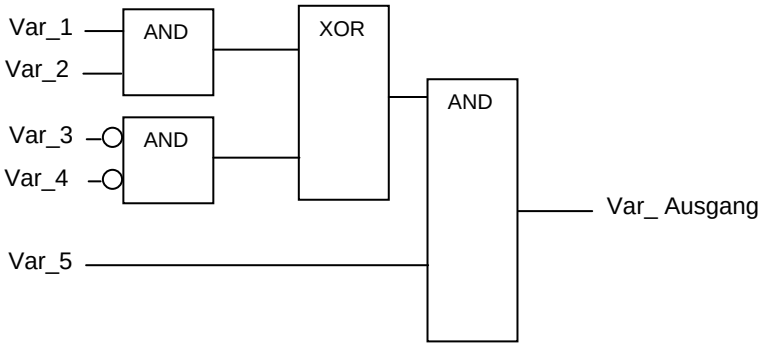

Bild 9: Beispiel einer Funktion

Funktionsbausteine

Funktionsbausteine (FB) sind POE mit internen Variablen, d.h. „mit Gedächtnis“. Die Ausgangswerte hängen auch von den internen Variablen ab. Diese bleiben zwischen den Aufrufen des Bausteins erhalten. Ein FB kann beliebig viele Ausgangsparameter haben.
Ein einfachster FB ist der bekannte Flip-Flop-Baustein. **Bild 10** zeigt ihn in der Ausführung „Dominierend aus“, erkennbar am Index „1“ des RESET-Eingangs. Es ist sofort einzusehen, dass der Wert der Ausgangsvariablen nicht nur von den Signalen der Eingänge, sondern auch vom aktuellen inneren Schaltzustand des Flip-Flop abhängt.

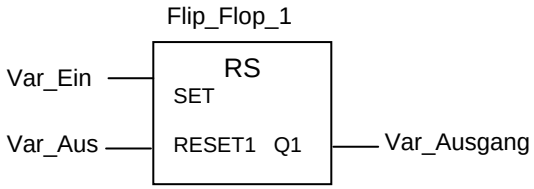

Bild 10: Beispiel eines Funktionsbausteins

Fügt man einen FB in ein Programm ein, so muß das System einen bestimmten Speicherplatz für die internen Variablen reservieren. Im Beispiel des Bildes 9 wird lediglich ein Bit Speicher für die Ablage des Schaltzustandes 0 oder 1 des Flip-Flop benötigt. Das Reservieren des Speicherplatzes erfolgt durch Instanzieren des FB.

Funktionen und Funktionsbausteine sind auch in Step7 bekannt. Nach IEC 61131-3 werden diese jedoch in strengerer Weise abgegrenzt. Step7 weicht hier von der Norm in einem ergänzenden Sinne ab. Beispielsweise kann ein FC in Step7 auch mehrere Werte am Ausgang zurückgeben.

Funktionsbausteine sind sowohl in Step7 als auch in Programmen nach IEC 61131-3 die „Hauptträger“ eines Anwenderprogramms.

Instanzierung von Funktionsbausteinen

Indem wir einen FB instanzieren, fertigen wir eine Kopie des Bausteins für die spezielle Verwendung in unserem Programm an. Diese Kopie erhält einen Namen und wird dem System per Variablendeklaration bekannt gemacht. Im Beispiel Bild 10 wurde als Instanzname Flip_Flop_1 gewählt. Im Deklarationsteil ist deshalb zu schreiben
VAR
Flip_Flop_1:RS;
END_VAR
Werden nun in einem Anwenderprogramm mehrere RS-Speicher benötigt, so deklariert man für jeden derartigen FB einen neuen Namen. Damit wird der entsprechende Speicherplatz bereitgestellt.

In Step7 erfolgt ein ähnlicher Vorgang durch Benennung des Instanzdatenbausteins beim Aufruf eines FB. Dieser nimmt u.a. die internen Variablen auf, gekennzeichnet durch den lokalen Datentyp „stat“.

Standard-Funktionsblöcke, Parametrierung und Bibliotheken

Wiederholt benötigte Programmdetails programmiert man nicht selbst, sondern entnimmt sie den im Programmiersystem vorbereiteten Lösungen. Diese werden in Bibliotheken bereitgestellt. Zumindest die Standard-Bibliothek (File standard.lib) ist Bestandteil jedes IEC-Programmiersystems. Sie enthält neben Standard-Funktionen die Funktionsblöcke RS-Speicher, Zeitrelais (Timer), Zähler (Counter) und Trigger.

Gewünschte Bibliotheks-Bausteine werden ähnlich wie Operatoren in das Anwender-programm eingefügt und parametriert. Darunter versteht man den Eintrag der konkreten Variablen, mit denen der Baustein beim aktuellen Aufruf arbeiten soll. Ein aus einer Bibliothek entnommener Baustein ist stets ein parametrierbarer Baustein. Mit diesem Typ werden wir uns in einer späteren Folge näher befassen, wenn wir selbst solche Bausteine schreiben. Dann müssen wir uns auch detaillierter mit der Deklaration unterschiedlicher Variablentypen befassen.

Ruft man die Ressourcen des Projektes auf, werden die aktuellverfügbaren Bibliotheken angezeigt. Mit Hilfe der Bibliotheksverwaltung können wir weitere Bibliotheken in ein Projekt einfügen und danach nutzen. Bibliotheken findet man auf den Homepage von Hard- und Softwareanbietern wie z.B.www.wago.com.

Umgang mit Bibliotheken im Programmiersystem CoDeSys

Bild 11 zeigt die Programmieroberfläche für die Arbeit mit Bibliotheken. Wählt man die Ressourcen (Folge 2 Bild 5), so werden die im Projekt verfügbaren Bibliotheken angezeigt. Im Beispiel ist dies die Standard. Lib. Nach Anwahl im Menu -> „*Fenster*“ -> *Bibliotheks-verwaltung* und -> *Einfügen* können weitere Bibliotheken in das Projekt eingefügt werden, soweit sie in Files verfügbar sind.

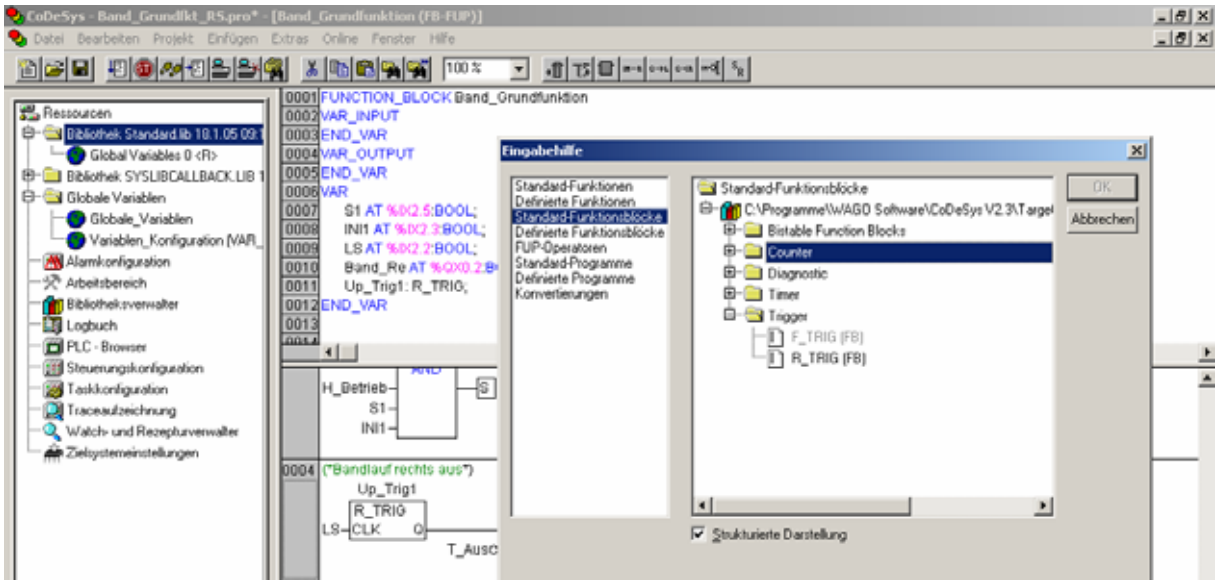

Bild 11: Einfügen von Standard-Funktionsblöcken aus der Standard.lib

Benötigen wir nun beim Programmieren in FUP Standard-FC oder –FB, können wir wie gewohnt einen AND-Operator einfügen und das Schlüsselwort überschreiben, z.B. mit RS.

Komfortabler ist die Nutzung der Einfügetaste F2. Nach dem Markieren der Programm-element-Vorlage führt Taste F2 zu Auswahlfenstern, aus denen man die gewünschten Bibliothekselemente auswählen kann.

Im Programm Bild 8 ist der Baustein R_TRIG ein Standard-Funktionsblock von Typ Trigger mit der Funktion positive Flankenauswertung. Er ist ein parametrierbarer Baustein aus der Standard-Bibliothek zur freien und wiederholten Nutzung der Flankenauswertung. Als Instanzname wurde Up_Trig1 gewählt. Deshalb erscheint im Variablenblock der Eintrag Up_Trig1: R_TRIG;. Diese Instanz von R_TRIG kann dann nur in dieser POE verwendet werden, in welcher sie deklariert wurde.
Die Parametrierung erfolgt durch Anschalten der Variablen „*LS*“ an den Eingang CLK und das Verbinden von Ausgang Q mit dem nachfolgenden OR-Operator.

Dem mit Step7 vertrauten Leser sind die Setz- und Rück-Befehle des Programmcodes Bild 8 bekannt. Er ist es aber auch gewohnt, beide Befehle in einem RS-Flip-Flop gemeinsam einzusetzen. Ein RS- Speicher steht in Step7 als „fertiger Befehl“ zur Verfügung. Anstelle der Instanzierung ist dort einfach ein Merker-, Ausgangs- oder Datenbit zu verwenden, und dessen Adresse muss man selbst benennen und verwalten. Bei CoDeSys müssen stattdessen ein Bistabiler Function Block (Bild 10) aus der Standard-Bibliothek in das Programm eingefügt werden.

Nunmehr liegt der Programmcode Band_Grundfunktion in einer vertrauten Form vor, wie ihn **Bild 12** als Programmausdruck zeigt. Als Instanznamen der beiden R-S-Blocks wurden M0 und M1 gewählt und im Variablenblock eingetragen.

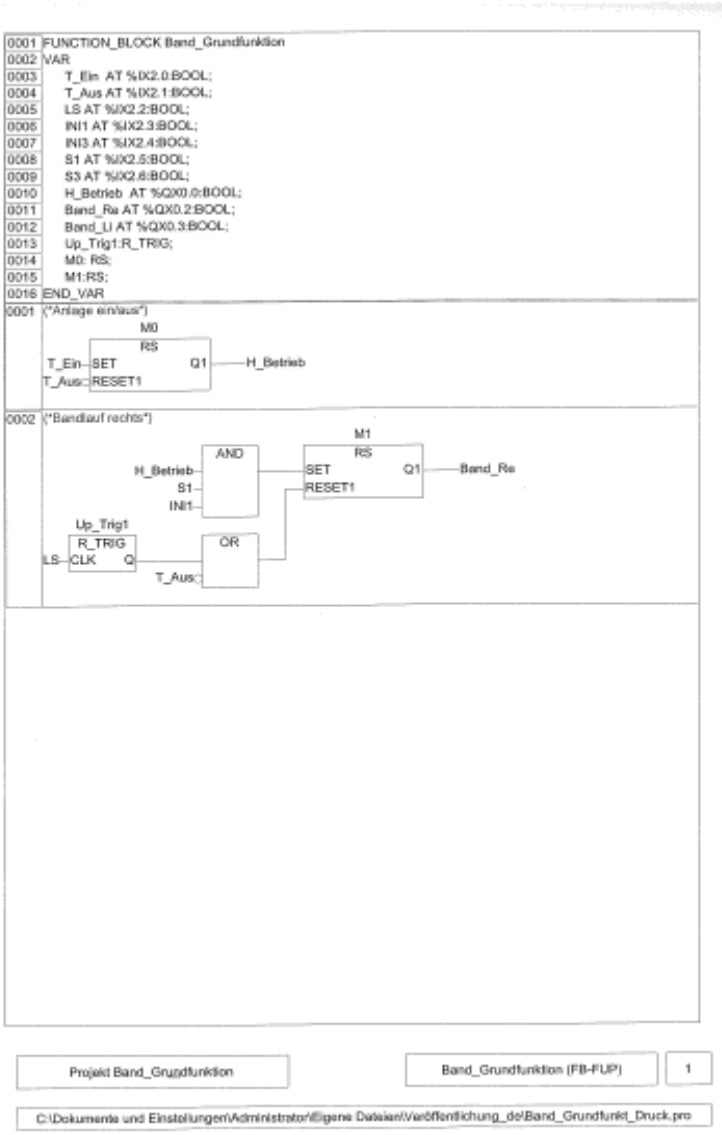

Bild 12: Ausdruck des Programmcodes des FB Band_Grundfunktion

An dieser Stelle wollen wir uns erinnern, dass wir unser gesamtes Programm gleichfalls in einem Funktionsbaustein geschrieben haben, weil wir interne Variable und mehrere Ausgänge nutzen wollen. In der nächsten Folge wollen wir das Programm in den programmierbaren Feldbus-Controller unseres Trainingsracks eintragen. Bei der Inbetriebnahme des Verteilerbandes müssen wir prüfen, ob unser Programm bearbeitet wird oder ob auch dieser Block aufgerufen, instanziert und ggf. parametriert werden muss.

Glossar:

Bibliothek	hier: Verzeichnis, in welchem parametrierbare POE für vorgegebene Aufgaben bereitstehen. Diese können Funktionen (Operatoren), Funktionsbausteine oder auch selbstgeschriebene POE sein.
Funktion (FUN)	POE, die bei gleichen Eingangsparametern stets dasselbe Ergebnis liefert. Sie hat „kein Gedächtnis“.
Funktionsbaustein (FB)	POE, deren Ausgangswerte nicht nur von den Eingangs-parametern, sondern auch von der Vorgeschichte abhängen. FB besitzen internen Variable, d.h. „sie haben ein Gedächtnis“. Interne Werte bleiben zwischen den Aufrufen des Bausteins erhalten.
Instanz	Datenkopie eines Funktionsbausteins für die Nutzung desselben mit aktuellen Parametern
Lokale / globale Daten	In Step 7: Alle Variablen, welche in der Deklarationstabelle eines FC oder FB festgelegt werden, sind lokale Daten (Kennzeichnung durch #). Diese sind nur im Baustein selbst bekannt. Ein Datentyp davon sind die statischen Daten, welche im Instanzdatenbaustein verwaltet werden. Im Gegensatz dazu sind globale Daten in allen Bausteinen des Programms zu verwenden. Sie werden in der Symbol-tabelle verwaltet. In IEC: Die Norm verwendet u.a. ebenfalls diese Datentypen. Die Details werden in späteren Folgen erarbeitet.
Parametrierbarer Baustein	Baustein, der mit formalen Operanden geschrieben wurde. Bei seinem Aufruf werden diesen Formaloperanden aktuelle Werte zugeordnet.