HW Konfig - [SIMATIC 300(1) - leer (Konfiguration) -- Schulung]

Station Bearbeiten Einfügen Zielsystem Ansicht Extras Fenster Hilfe

(0) UR

1	PS 307 5A
2	CPU 314IFM
3	
4	
5	
6	
7	
8	
9	
10	
11	

Eigenschaften - CPU 314IFM - (R0/S2)

Alarme | Uhrzeitalarme | Weckalarme | Diagnose / Uhr

Allgemein | Anlauf | Integrierte Funktion | Zyklus / Taktmerker | Remanenz

Zyklus

OB1-Prozeßabbild zyklisch aktualisieren

Zyklusüberwachungszeit [ms]: 150

Mindestzykluszeit [ms]: 0

Zyklusbelastung durch Kommunikation [%]: 20

Größe des Prozeßabbilds:

OB85-Aufruf bei Peripheriezugriffsfehler: Kein OB85-Aufruf

Taktmerker

Taktmerker

Merkerbyte: 0

OK | Abbrechen | Hilfe

Suchen:

Profil: Standard

- PROFIBUS-DP
- PROFIBUS-PA
- PROFINET IO
- SIMATIC 300
- SIMATIC 400
- SIMATIC HMI Station
- SIMATIC PC Based Control 300/400
- SIMATIC PC Station

(0) UR

Steckplatz	Baugrupp...	B...	Fi...	M...	E...	A...	Kommer
1	PS 307 5A	6ES7					
2	CPU 314IFM	6ES7		2	124...	124...	
3							
4							
5							
6							
7							
8							
9							
10							
11							

PROFIBUS-DP-Slaves der SIMATIC S7, M7 und C7 (dezentraler Aufbau)

Drücken Sie F1, um Hilfe zu erhalten.

HW Konfig - [SIMATIC 300(1) - leer (Konfiguration) -- Schulung]

Station Bearbeiten Einfügen Zielsystem Ansicht Extras Fenster Hilfe

(0) UR

1	PS 307 5A
2	CPU 314IFM
3	
4	
5	
6	
7	
8	
9	
10	
11	

Suchen:

Profil: Standard

- PROFIBUS-DP
- PROFIBUS-PA
- PROFINET IO
- SIMATIC 300
- SIMATIC 400
- SIMATIC HMI Station
- SIMATIC PC Based Control 300/400
- SIMATIC PC Station

Eigenschaften - CPU 314IFM - (R0/S2)

Alarme | Uhrzeitalarme | Weckalarme | Diagnose / Uhr

Allgemein | Anlauf | Integrierte Funktion | Zyklus / Taktmerker | Remanenz

Zyklus

OB1-Prozeßabbild zyklisch aktualisieren

Zyklusüberwachungszeit [ms]: 150

Mindestzykluszeit [ms]: 0

Zyklusbelastung durch Kommunikation [%]: 20

Größe des Prozeßabbilds:

OB85-Aufruf bei Peripheriezugriffsfehler: Kein OB85-Aufruf

Taktmerker

Taktmerker

Merkerbyte: 1

OK | Abbrechen | Hilfe

(0) UR

Steckplatz	Baugrupp...	B...	Fi...	M...	E...	A...	Kommer
1	PS 307 5A	6ES7					
2	CPU 314IFM	6ES7		2	124...	124...	
3							
4							
5							
6							
7							
8							
9							
10							
11							

PROFIBUS-DP-Slaves der SIMATIC S7, M7 und C7 (dezentraler Aufbau)

Drücken Sie F1, um Hilfe zu erhalten.

Register "Zyklus / Taktmerker"

Im Register "Zyklus/Taktmerker" bestimmen Sie

- das zyklische Verhalten der CPUs
- Taktmerker

Sie können folgende Parameter einstellen:

OB1-Prozeßabbild zyklisch aktualisieren (Nicht relevant für S7-300 mit Ausnahme von CPU 318-2)

Zyklusüberwachungszeit

Mindestzykluszeit (Nicht relevant für S7-300)

Zyklusbelastung durch Kommunikation

Größe des Prozeßabbilds

OB 85-Aufruf bei Peripheriezugriffsfehler (PZF)

Taktmerker

Hinweise

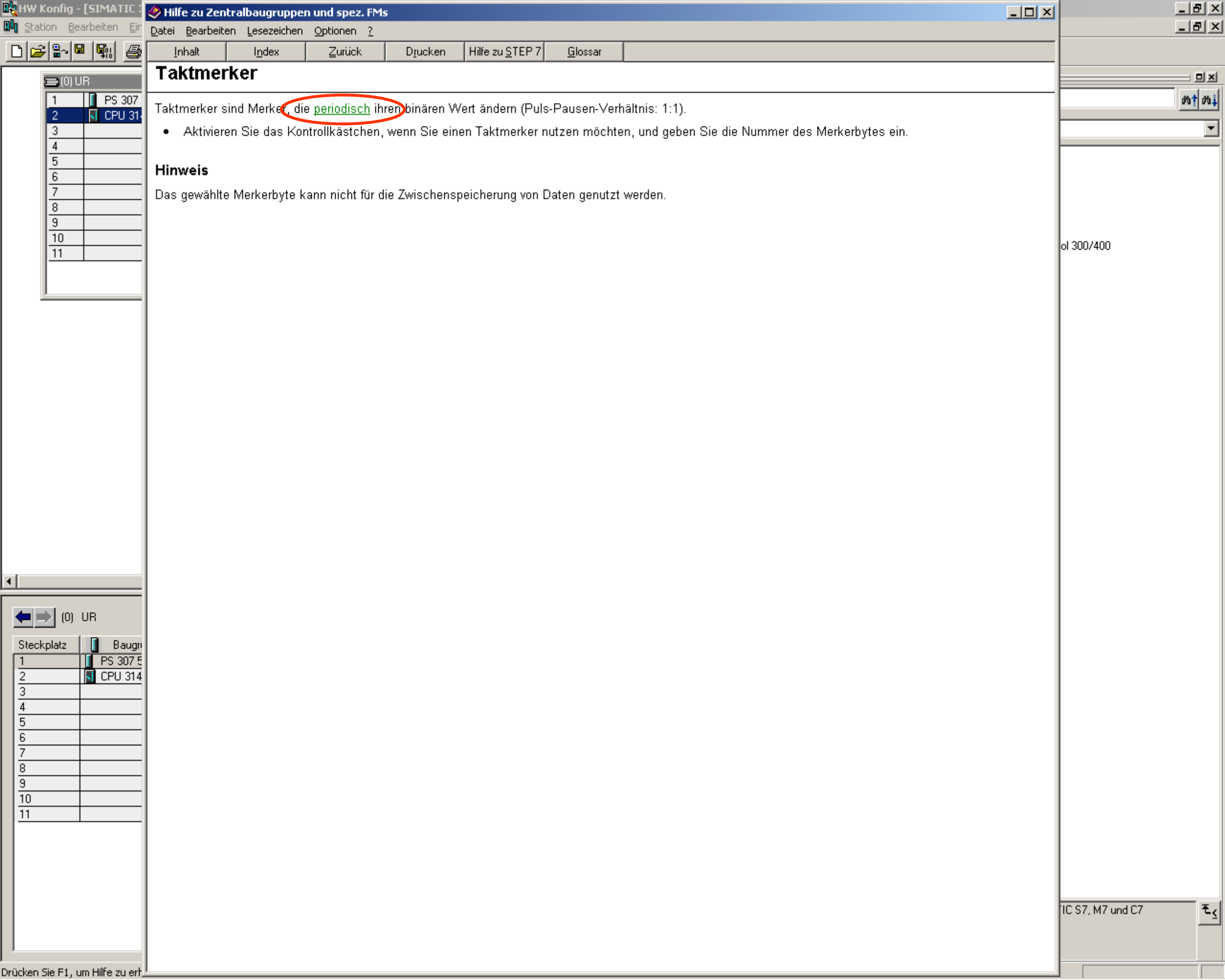
Bearbeiten der Parameter

Farblich hervorgehobene Parameter

Siehe auch:

Organisationsbaustein für zyklische Programmbearbeitung (OB 1)

Weitere Informationen und FAQs im Internet suchen...


HW Konfig - [SIMATIC
Station Bearbeiten
(0) UR
1 PS 307
2 CPU 31
3
4
5
6
7
8
9
10
11
Hilfe zu Zentralbaugruppen und spez. FMs
Datei Bearbeiten Lesezeichen Optionen ?
Inhalt Index Zurück Drucken Hilfe zu STEP 7 Glossar
Taktmerker
Taktmerker sind Merker, die periodisch ihren binären Wert ändern (Puls-Pausen-Verhältnis: 1:1).
• Aktivieren Sie das Kontrollkästchen, wenn Sie einen Taktmerker nutzen möchten, und geben Sie die Nummer des Merkerbytes ein.
Hinweis
Das gewählte Merkerbyte kann nicht für die Zwischenspeicherung von Daten genutzt werden.
ol 300/400
(0) UR
Steckplatz
1 PS 307 5
2 CPU 314
3
4
5
6
7
8
9
10
11
IC S7, M7 und C7
Drücken Sie F1, um Hilfe zu erh

Hilfe zu Zentralbaugruppen und spez. FMs

Datei Bearbeiten Lesezeichen Optionen ?

Inhalt | Index | Zurück | Drucken | Hilfe zu STEP 7 | Glossar

Periodendauer

Jedem Bit des Taktmerkerbytes ist eine Periodendauer/Frequenz zugeordnet:

Bit	7	6	5	4	3	2	1	0
Perioden-dauer (s):	2	1,6	1	0,8	0,5	0,4	0,2	0,1
Frequenz (Hz):	0,5	0,625	1	1,25	2	2,5	5	10

SIMATIC Manager - [Schulung -- g:\Schulung]

Datei Bearbeiten Einfügen Zielsystem Ansicht Extras Fenster Hilfe

< Kein Filter >

- Schulung
  - SIMATIC 300(1) - leer
    - CPU 314IFM
      - S7-Programm(1)
        - Quellen
        - Bausteine
  - SIMATIC 300(1) - projekt
    - CPU 314IFM
      - S7-Programm(1)
        - Quellen
        - Bausteine

Objektname	Symbolischer Name	Typ	Größe	Autor	Änderungsdatum	Kommentar
Quellen	---	Quellordner	---		04.12.2007 09:31:24	
Bausteine	---	Bausteinordner offline	---		05.12.2007 16:43:37	
Symbole	---	Symboltabelle	3788		05.12.2007 16:43:36	

Symboltabelle

Drücken Sie F1, um Hilfe zu erhalten.

TCP/IP -> 3Com EtherLink XL 10/1...

3788 Bytes

Symbol Editor - [S7-Programm(1) (Symbole) -- Schulung\SIMATIC 300(1) - leer\CPU 314IFM]

Tabelle Bearbeiten Einfügen Ansicht Extras Fenster Hilfe

Alle Symbole

	Status	Symbol	Adresse	Datentyp	Kommentar
1					

Drücken Sie F1, um Hilfe zu erhalten.

NUM

Symbol Editor - [S7-Programm(1) (Symbole) -- Schulung\SIMATIC 300(1) - leer\CPU 314IFM]

Tabelle Bearbeiten Einfügen Ansicht Extras Fenster Hilfe

Alle Symbole

	Status	Symbol	Adresse	Datentyp	Kommentar
1		Takt_10Hz	M 1.0	BOOL	Taktmerker 10Hz
2		Takt_5Hz	M 1.1	BOOL	Taktmerker 5Hz
3		Takt_2_5Hz	M 1.2	BOOL	Taktmerker 2,5Hz
4		Takt_2Hz	M 1.3	BOOL	Taktmerker 2Hz
5		Takt_1_25Hz	M 1.4	BOOL	Taktmerker 1,25Hz
6		Takt_1Hz	M 1.5	BOOL	Taktmerker 1Hz
7		Takt_0_625Hz	M 1.6	BOOL	Taktmerker 0,625 Hz
8		Takt_0_5Hz	M 1.7	BOOL	Taktmerker 0,5Hz
9					

Hilfe zu Zentralbaugruppen und spez. FMs

Datei Bearbeiten Lesezeichen Optionen ?

Inhalt | Index | Zurück | Drucken | Hilfe zu STEP 7 | Glossar

Periodendauer

Jedem Bit des Taktmerkerbytes ist eine Periodendauer/Frequenz zugeordnet:

Bit	7	6	5	4	3	2	1	0
Periodendauer (s):	2	1,6	1	0,8	0,5	0,4	0,2	0,1
Frequenz (Hz):	0,5	0,625	1	1,25	2	2,5	5	10

Drücken Sie F1, um Hilfe zu erhalten.

NUM

Symbol Editor - [S7-Programm(1) (Symbole) -- Schulung\SIMATIC 300(1) - leer\CPU 314IFM]

Tabelle Bearbeiten Einfügen Ansicht Extras Fenster Hilfe

Alle Symbole

	Status	Symbol	Adresse	Datentyp	Kommentar
1		Takt_10Hz	M 1.0	BOOL	Taktmerker 10Hz
2		Takt_5Hz	M 1.1	BOOL	Taktmerker 5Hz
3		Takt_2_5Hz	M 1.2	BOOL	Taktmerker 2,5Hz
4		Takt_2Hz	M 1.3	BOOL	Taktmerker 2Hz
5		Takt_1_25Hz	M 1.4	BOOL	Taktmerker 1,25Hz
6		Takt_1Hz	M 1.5	BOOL	Taktmerker 1Hz
7		Takt_0_625Hz	M 1.6	BOOL	Taktmerker 0,625 Hz
8		Takt_0_5Hz	M 1.7	BOOL	Taktmerker 0,5Hz
9		PFlanke_Takt_1Hz	M 2.0	BOOL	
10					

Drücken Sie F1, um Hilfe zu erhalten.

NUM

SIMATIC Manager - [Schulung -- g:\Schulung]

Datei Bearbeiten Einfügen Zielsystem Ansicht Extras Fenster Hilfe

< Kein Filter >

Schulung
- SIMATIC 300(1) - leer
  - CPU 314IFM
    - S7-Programm(1)
      - Quellen
      - Bausteine
- SIMATIC 300(1) - projekt

Objektname	Symbolischer Name	Typ	Größe	Autor	Änderungsdatum	Kommentar

Ausschneiden Ctrl+X
Kopieren Ctrl+C
Einfügen Ctrl+V
Löschen Del
Neues Objekt einfügen
Zielsystem
Objekteigenschaften... Alt+Return
Spezielle Objekteigenschaften

AWL-Quelle
SCL-Quelle
SCL-Übersetzungssteuerdatei
Externe Quelle...

Fügt SCL-Quelle an der Cursorposition ein.

SIMATIC Manager - [Schulung -- g:\Schulung]

Datei Bearbeiten Einfügen Zielsystem Ansicht Extras Fenster Hilfe

< Kein Filter >

- Schulung
  - SIMATIC 300(1) - leer
  - SIMATIC 300(1) - projekt
    - CPU 314IFM
      - S7-Programm(1)
        - Quellen
        - Bausteine

Objektname	Symbolischer Name	Typ	Größe	Autor	Änderungsdatum	Kommentar
DB_Eingaenge	---	SCL-Quelle	1556		05.12.2007 08:38:04	
FC_Abfrage	---	SCL-Quelle	2553		05.12.2007 16:38:08	
FC_Ausgang	---	SCL-Quelle	1968		05.12.2007 16:24:08	
OB_1	---	SCL-Quelle	523		05.12.2007 15:55:18	
UDT_Eingang	---	SCL-Quelle	254		04.12.2007 15:32:34	
Alles_Uebersetzen	---	SCL-Übersetzungssteu...	479		05.12.2007 16:41:16	

Drücken Sie F1, um Hilfe zu erhalten.

TCP/IP -> 3Com EtherLink XL 10/1...

SIMATIC Manager - [Schulung -- g:\Schulung]

Datei Bearbeiten Einfügen Zielsystem Ansicht Extras Fenster Hilfe

< Kein Filter >

- Schulung
  - SIMATIC 300(1) - leer
    - CPU 314IFM
      - S7-Programm(1)
        - Quellen
        - Bausteine
  - SIMATIC 300(1) - projekt
    - CPU 314IFM
      - S7-Programm(1)
        - Quellen
        - Bausteine

Objektname	Symbolischer Name	Typ	Größe	Autor	Änderungsdatum	Kommentar
DB_Eingaenge	---	SCL-Quelle	1556		05.12.2007 08:38:04	
FC_Abfrage	---	SCL-Quelle	2553		05.12.2007 16:38:08	
FC_Ausgang	---	SCL-Quelle	1968		05.12.2007 16:24:08	
OB_1	---	SCL-Quelle	523		05.12.2007 15:55:18	
UDT_Eingang	---	SCL-Quelle	254		04.12.2007 15:32:34	
Alles_Uebersetzen	---	SCL-Übersetzungssteuerdatei	479		05.12.2007 16:41:16	

SCL-Quelle

Drücken Sie F1, um Hilfe zu erhalten.

TCP/IP -> 3Com EtherLink XL 10/1...

254 Bytes

SCL - [UDT_Eingang -- Schulung\SIMATIC 300(1) - projekt\CPU 314IFM]

Datei Bearbeiten Einfügen Zielsystem Test Ansicht Extras Fenster Hilfe

Fehler Info

Drücken Sie F1, um Hilfe zu erhalten.

Ze 1 Sp 1 Einfg Änd

SCL - [UDT_Eingang -- Schulung\SIMATIC 300(1) - projekt\CPU 314IFM]

Datei Bearbeiten Einfügen Zielsystem Test Ansicht Extras Fenster Hilfe

Einfügen:
- Bausteinaufruf...
- Bausteinvorlage ▸
- Kontrollstruktur ▸

Bausteinvorlage:
- OB
- FB
- FC
- Kommentar
- Konstante
- Parameter
- DB
- Instanz DB
- DB aus UDT
- UDT

Fehler | Info

Fügt Textgerüst für neuen anwenderdefinierten Datentyp an der Cursorposition ein.

Ze 1 Sp 1 | Einfg | Änd

SCL - [UDT_Eingang -- Schulung\SIMATIC 300(1) - projekt\CPU 314IFM]

Datei Bearbeiten Einfügen Zielsystem Test Ansicht Extras Fenster Hilfe

```
TYPE UDTxxx
  STRUCT
  //  Typbeschreibung

  END_STRUCT
END_TYPE
```

Fehler Info

Drücken Sie F1, um Hilfe zu erhalten.

Ze 8 Sp 1 Einfg Änd

SCL - [UDT_Eingang -- Schulung\SIMATIC 300(1) - projekt\CPU 314IFM]

Datei Bearbeiten Einfügen Zielsystem Test Ansicht Extras Fenster Hilfe

```
TYPE UDT_Eingang
  STRUCT
    E_Byte   : INT;          //Eingangsbyte
    E_Bit    : INT;          //Eingangsbit
    Aktiv    : BOOL;         //Wird der Eingang verarbeitet?
    E_Name   : STRING[10];   //Wie heißt der Eingang?
  END_STRUCT
END_TYPE
```

Fehler | Info

Drücken Sie F1, um Hilfe zu erhalten.

Ze 9 Sp 1 | Einfg

SIMATIC Manager - [Schulung -- g:\Schulung]

Datei Bearbeiten Einfügen Zielsystem Ansicht Extras Fenster Hilfe

< Kein Filter >

- Schulung
  - SIMATIC 300(1) - leer
    - CPU 314IFM
      - S7-Programm(1)
        - Quellen
        - Bausteine
  - SIMATIC 300(1) - projekt
    - CPU 314IFM
      - S7-Programm(1)
        - Quellen
        - Bausteine

Objektname	Symbolischer Name	Typ	Größe	Autor	Änderungsdatum	Kommentar
DB_Eingaenge	---	SCL-Quelle	1556		05.12.2007 08:38:04	
FC_Abfrage	---	SCL-Quelle	2553		05.12.2007 16:38:08	
FC_Ausgang	---	SCL-Quelle	1968		05.12.2007 16:24:08	
OB_1	---	SCL-Quelle	523		05.12.2007 15:55:18	
UDT_Eingang	---	SCL-Quelle	254		04.12.2007 15:32:34	
Alles_Uebersetzen	---	SCL-Übersetzungssteuerdatei	479		05.12.2007 16:41:16	

SCL-Quelle

Drücken Sie F1, um Hilfe zu erhalten.

TCP/IP -> 3Com EtherLink XL 10/1...

1556 Bytes

SCL - [DB_Eingaenge -- Schulung\SIMATIC 300(1) - projekt\CPU 314IFM]

Datei Bearbeiten Einfügen Zielsystem Test Ansicht Extras Fenster Hilfe

Bausteinaufruf...
Bausteinvorlage
Kontrollstruktur

OB
FB
FC

Kommentar
Konstante
Parameter

DB
Instanz DB
DB aus UDT
UDT

Fehler Info

Fügt Textgerüst für neuen Datenbaustein an der Cursorposition ein.

Ze 2 Sp 1 Einfg Änd

```
DATA_BLOCK DBxxx
//
// Baustein-Kommentar ...
//
  STRUCT

  END_STRUCT
BEGIN

END_DATA_BLOCK
```

SCL - [DB_Eingaenge -- Schulung\SIMATIC 300(1) - projekt\CPU 314IFM]

Datei Bearbeiten Einfügen Zielsystem Test Ansicht Extras Fenster Hilfe

```
DATA_BLOCK Eingaenge
//
// Definiert, welche Eingänge abgefragt werden, schaltet sie aktiv und gibt ihnen Namen
//
  STRUCT
    Eingaenge       : ARRAY [1..8] OF UDT_Eingang;  //Feld für 8 Eingänge
    Erster_Eingang  : STRING[10];                   //Enthält den Namen des ersten aktiven Eingangs
    Zaehler         : INT;                          //Kontrollzähler, ob die Schleife arbeitet
  END_STRUCT
BEGIN

END_DATA_BLOCK
```

Fehler | Info

Drücken Sie F1, um Hilfe zu erhalten.

Ze 13 Sp 1 | Einfg | Änd

```
DATA_BLOCK Eingaenge
//
// Definiert, welche Eingänge abgefragt werden, schaltet sie aktiv und gibt ihnen Namen
//
  STRUCT
    Eingaenge       : ARRAY [1..8] OF UDT_Eingang;  //Feld für 8 Eingänge
    Erster_Eingang  : STRING[10];                   //Enthält den Namen des ersten aktiven Eingangs
    Zaehler         : INT;                          //Kontrollzähler, ob die Schleife arbeitet
  END_STRUCT
BEGIN
    Eingaenge[1].E_Byte := 124;
    Eingaenge[1].E_Bit  := 0;
    Eingaenge[2].E_Byte := 124;
    Eingaenge[2].E_Bit  := 1;
    Eingaenge[3].E_Byte := 124;
    Eingaenge[3].E_Bit  := 2;
    Eingaenge[4].E_Byte := 124;
    Eingaenge[4].E_Bit  := 3;
    Eingaenge[5].E_Byte := 124;
    Eingaenge[5].E_Bit  := 4;
    Eingaenge[6].E_Byte := 124;
    Eingaenge[6].E_Bit  := 5;
    Eingaenge[7].E_Byte := 124;
    Eingaenge[7].E_Bit  := 6;
    Eingaenge[8].E_Byte := 124;
    Eingaenge[8].E_Bit  := 7;

    Eingaenge[1].Aktiv  := true;
    Eingaenge[2].Aktiv  := true;
    Eingaenge[3].Aktiv  := true;
    Eingaenge[4].Aktiv  := true;
    Eingaenge[5].Aktiv  := true;
    Eingaenge[6].Aktiv  := true;
    Eingaenge[7].Aktiv  := true;
    Eingaenge[8].Aktiv  := true;

    Eingaenge[1].E_Name := 'Eingang 1';
    Eingaenge[2].E_Name := 'Eingang 2';
    Eingaenge[3].E_Name := 'Eingang 3';
    Eingaenge[4].E_Name := 'Eingang 4';
    Eingaenge[5].E_Name := 'Eingang 5';
    Eingaenge[6].E_Name := 'Eingang 6';
    Eingaenge[7].E_Name := 'Eingang 7';
    Eingaenge[8].E_Name := 'Eingang 8';
END_DATA_BLOCK
```

SIMATIC Manager - [Schulung -- g:\Schulung]

Datei Bearbeiten Einfügen Zielsystem Ansicht Extras Fenster Hilfe

< Kein Filter >

- Schulung
  - SIMATIC 300(1) - leer
    - CPU 314IFM
      - S7-Programm(1)
        - Quellen
        - Bausteine
  - SIMATIC 300(1) - projekt
    - CPU 314IFM
      - S7-Programm(1)
        - Quellen
        - Bausteine

Objektname	Symbolischer Name	Typ	Größe	Autor	Änderungsdatum	Kommentar
DB_Eingaenge	---	SCL-Quelle	1556		05.12.2007 08:38:04	
FC_Abfrage	---	SCL-Quelle	2553		05.12.2007 16:38:08	
FC_Ausgang	---	SCL-Quelle	1968		05.12.2007 16:24:08	
OB_1	---	SCL-Quelle	523		05.12.2007 15:55:18	
UDT_Eingang	---	SCL-Quelle	254		04.12.2007 15:32:34	
Alles_Uebersetzen	---	SCL-Übersetzungssteuerdatei	479		05.12.2007 16:41:16	

SCL-Quelle

Drücken Sie F1, um Hilfe zu erhalten.

TCP/IP -> 3Com EtherLink XL 10/1...

2553 Bytes

SCL - [FC_Abfrage -- Schulung\SIMATIC 300(1) - projekt\CPU 314IFM]

Datei Bearbeiten Einfügen Zielsystem Test Ansicht Extras Fenster Hilfe

Bausteinaufruf...
Bausteinvorlage ▸
Kontrollstruktur ▸

OB
FB
FC

Kommentar
Konstante
Parameter

DB
Instanz DB
DB aus UDT
UDT

Fehler Info

Fügt Textgerüst für neue Funktion an der Cursorposition ein.

Ze 1 Sp 1 Einfg Änd

SCL - [FC_Abfrage -- Schulung\SIMATIC 300(1) - projekt\CPU 314IFM]

Datei Bearbeiten Einfügen Zielsystem Test Ansicht Extras Fenster Hilfe

Einfügen:
- Bausteinaufruf...
- Bausteinvorlage ▸
  - OB
  - FB
  - FC
  - Kommentar
  - Konstante
  - Parameter
  - DB
  - Instanz DB
  - DB aus UDT
  - UDT
- Kontrollstruktur ▸

```
FUNCTION FCxxx : INT

VAR_TEMP
  // temporäre Variablen

END_VAR

  // Anweisungsteil
  ;
  FCxxx := 100;
END_FUNCTION
```

Fehler | Info

Fügt Textgerüst für neuen Baustein-Kommentar an der Cursorposition ein.

Ze 3 Sp 1 | Einfg | Änd

SCL - [FC_Abfrage -- Schulung\SIMATIC 300(1) - projekt\CPU 314IFM]

Datei Bearbeiten Einfügen Zielsystem Test Ansicht Extras Fenster Hilfe

Einfügen:
- Bausteinaufruf...
- Bausteinvorlage ▸
  - OB
  - FB
  - FC
  - Kommentar
  - Konstante
  - Parameter
  - DB
  - Instanz DB
  - DB aus UDT
  - UDT
- Kontrollstruktur ▸

```
FUNCTION FCxxx : INT

TITLE = 'Baustein-Überschrift'
//
// Baustein-Kommentar ...
//
VERSION : '1.0'
AUTHOR  : author
NAME    : name
FAMILY  : family

VAR_TEMP
  // temporäre Variablen

END_VAR

  // Anweisungsteil
  ;
  FCxxx := 100;
END_FUNCTION
```

Fehler | Info

Fügt Textgerüst für neuen Parameterblock an der Cursorposition ein.

Ze 12 Sp 1 | Einfg | Änd

```
FUNCTION FCxxx : INT

TITLE = 'Baustein-Überschrift'
//
// Baustein-Kommentar ...
//
VERSION : '1.0'
AUTHOR  : author
NAME    : name
FAMILY  : family

// Bausteinparameter
VAR_INPUT
  // Eingangsparameter

END_VAR

VAR_IN_OUT
  // Durchgangsparameter
END_VAR

VAR_OUTPUT
 // Ausgangsparameter

END_VAR

VAR_TEMP
  // temporäre Variablen

END_VAR

  // Anweisungsteil
  ;
  FCxxx := 100;
END_FUNCTION
```

```
FUNCTION Abfrage : BOOL

TITLE = 'Abfrage'
//
// Der Baustein überprüft Eingänge, die über einen DB vorgegeben werden und setzt entsprechend
// der gespeicherten Daten die Ausgänge
//
// Erstellt: 04.12.2007, Schürmann, HIT
// Geändert:
//
VERSION : '1.0'
AUTHOR  : Sc
NAME    : Abfrage
FAMILY  : FBs


// Bausteinparameter
VAR_INPUT
  // Eingangsparameter
  Pruefbedingung    : BOOL;
END_VAR

VAR_IN_OUT
  // Durchgangsparameter
END_VAR

VAR_OUTPUT
 // Ausgangsparameter

END_VAR

VAR_TEMP
  // temporäre Variablen
Zaehler     : INT;
END_VAR

CONST
  // Konstanten
Anzahl_Eingaenge := 8;
END_CONST
END_FUNCTION
```

```
  // temporäre Variablen
Zaehler     : INT;
END_VAR

CONST
  // Konstanten
Anzahl_Eingaenge := 8;
END_CONST
BEGIN

//Wenn die Prüfbedingung (hier der 1Hz-Taktmerker) stimmt, sollen einmalig (deshalb auf Flanke des Merkers geprüft)
//die Eingänge abgefragt werden.
IF(Pruefbedingung AND NOT PFlanke_Takt_1Hz) THEN
```

```
END_IF;

END_FUNCTION
```

```
  // temporäre Variablen
Zaehler     : INT;
END_VAR

CONST
  // Konstanten
Anzahl_Eingaenge := 8;
END_CONST
BEGIN

//Wenn die Prüfbedingung (hier der 1Hz-Taktmerker) stimmt, sollen einmalig (deshalb auf Flanke des Merkers geprüft)
//die Eingänge abgefragt werden.
IF(Pruefbedingung AND NOT PFlanke_Takt_1Hz) THEN
    //Das im DB gespeicherte Array durchlaufen
    FOR Zaehler := 1 TO Anzahl_Eingaenge BY 1 DO









    END_FOR;



END_IF;

END_FUNCTION
```

```
  // temporäre Variablen
Zaehler     : INT;
END_VAR

CONST
  // Konstanten
Anzahl_Eingaenge := 8;
END_CONST
BEGIN

//Wenn die Prüfbedingung (hier der 1Hz-Taktmerker) stimmt, sollen einmalig (deshalb auf Flanke des Merkers geprüft)
//die Eingänge abgefragt werden.
IF(Pruefbedingung AND NOT PFlanke_Takt_1Hz) THEN
    //Das im DB gespeicherte Array durchlaufen
    FOR Zaehler := 1 TO Anzahl_Eingaenge BY 1 DO
        //Eingang abfragen: indirekte Abfrage und nur prüfen, wenn das "Aktiv"-Bit gesetzt ist.
        //   komplett symbolisch adressiert       Teilweise absolut, teilw. symbolisch     komplett absolut Zugriff
        IF(                                                                                                    ) THEN



        END_IF;
    END_FOR;



END_IF;

END_FUNCTION
```

```
  // temporäre Variablen
Zaehler     : INT;
END_VAR

CONST
  // Konstanten
Anzahl_Eingaenge := 8;
END_CONST
BEGIN

//Wenn die Prüfbedingung (hier der 1Hz-Taktmerker) stimmt, sollen einmalig (deshalb auf Flanke des Merkers geprüft)
//die Eingänge abgefragt werden.
IF(Pruefbedingung AND NOT PFlanke_Takt_1Hz) THEN
    //Das im DB gespeicherte Array durchlaufen
    FOR Zaehler := 1 TO Anzahl_Eingaenge BY 1 DO
        //Eingang abfragen: indirekte Abfrage und nur prüfen, wenn das "Aktiv"-Bit gesetzt ist.
        //   komplett symbolisch adressiert       Teilweise absolut, teilw. symbolisch     komplett absolut Zugriff
        IF(E[                                ,                                  ] = true                                    ) THEN



        END_IF;
    END_FOR;



END_IF;

END_FUNCTION
```

```
  // temporäre Variablen
Zaehler     : INT;
END_VAR

CONST
  // Konstanten
Anzahl_Eingaenge := 8;
END_CONST
BEGIN

//Wenn die Prüfbedingung (hier der 1Hz-Taktmerker) stimmt, sollen einmalig (deshalb auf Flanke des Merkers geprüft)
//die Eingänge abgefragt werden.
IF(Pruefbedingung AND NOT PFlanke_Takt_1Hz) THEN
    //Das im DB gespeicherte Array durchlaufen
    FOR Zaehler := 1 TO Anzahl_Eingaenge BY 1 DO
        //Eingang abfragen: indirekte Abfrage und nur prüfen, wenn das "Aktiv"-Bit gesetzt ist.
        //   komplett symbolisch adressiert       Teilweise absolut, teilw. symbolisch     komplett absolut Zugriff
        IF(E[Eingaenge.Eingaenge[Zaehler].E_BYTE,                                ] = true                                        ) THEN




        END_IF;
    END_FOR;
```

```
END_IF;

END_FUNCTION
```

```
  // temporäre Variablen
Zaehler     : INT;
END_VAR

CONST
  // Konstanten
Anzahl_Eingaenge := 8;
END_CONST
BEGIN

//Wenn die Prüfbedingung (hier der 1Hz-Taktmerker) stimmt, sollen einmalig (deshalb auf Flanke des Merkers geprüft)
//die Eingänge abgefragt werden.
IF(Pruefbedingung AND NOT PFlanke_Takt_1Hz) THEN
    //Das im DB gespeicherte Array durchlaufen
    FOR Zaehler := 1 TO Anzahl_Eingaenge BY 1 DO
        //Eingang abfragen: indirekte Abfrage und nur prüfen, wenn das "Aktiv"-Bit gesetzt ist.
        //   komplett symbolisch adressiert       Teilweise absolut, teilw. symbolisch     komplett absolut Zugriff
        IF(E[                                , DB1.Eingaenge[Zaehler].E_Bit] = true                                    ) THEN




        END_IF;
    END_FOR;
```

```
END_IF;

END_FUNCTION
```

```
  // temporäre Variablen
Zaehler     : INT;
END_VAR

CONST
  // Konstanten
Anzahl_Eingaenge := 8;
END_CONST
BEGIN

//Wenn die Prüfbedingung (hier der 1Hz-Taktmerker) stimmt, sollen einmalig (deshalb auf Flanke des Merkers geprüft)
//die Eingänge abgefragt werden.
IF(Pruefbedingung AND NOT PFlanke_Takt_1Hz) THEN
    //Das im DB gespeicherte Array durchlaufen
    FOR Zaehler := 1 TO Anzahl_Eingaenge BY 1 DO
        //Eingang abfragen: indirekte Abfrage und nur prüfen, wenn das "Aktiv"-Bit gesetzt ist.
        //   komplett symbolisch adressiert       Teilweise absolut, teilw. symbolisch     komplett absolut Zugriff
        IF(E[Eingaenge.Eingaenge[Zaehler].E_BYTE, DB1.Eingaenge[Zaehler].E_Bit] = true                                    ) THEN




        END_IF;
    END_FOR;
```

```
END_IF;

END_FUNCTION
```

```
  // temporäre Variablen
Zaehler     : INT;
END_VAR

CONST
  // Konstanten
Anzahl_Eingaenge := 8;
END_CONST
BEGIN

//Wenn die Prüfbedingung (hier der 1Hz-Taktmerker) stimmt, sollen einmalig (deshalb auf Flanke des Merkers geprüft)
//die Eingänge abgefragt werden.
IF(Pruefbedingung AND NOT PFlanke_Takt_1Hz) THEN
    //Das im DB gespeicherte Array durchlaufen
    FOR Zaehler := 1 TO Anzahl_Eingaenge BY 1 DO
        //Eingang abfragen: indirekte Abfrage und nur prüfen, wenn das "Aktiv"-Bit gesetzt ist.
        //   komplett symbolisch adressiert       Teilweise absolut, teilw. symbolisch     komplett absolut Zugriff
        IF(E[Eingaenge.Eingaenge[Zaehler].E_BYTE, DB1.Eingaenge[Zaehler].E_Bit] = true AND                                ) THEN




        END_IF;
    END_FOR;
```

```
END_IF;

END_FUNCTION
```

```
  // temporäre Variablen
Zaehler     : INT;
END_VAR

CONST
  // Konstanten
Anzahl_Eingaenge := 8;
END_CONST
BEGIN

//Wenn die Prüfbedingung (hier der 1Hz-Taktmerker) stimmt, sollen einmalig (deshalb auf Flanke des Merkers geprüft)
//die Eingänge abgefragt werden.
IF(Pruefbedingung AND NOT PFlanke_Takt_1Hz) THEN
    //Das im DB gespeicherte Array durchlaufen
    FOR Zaehler := 1 TO Anzahl_Eingaenge BY 1 DO
        //Eingang abfragen: indirekte Abfrage und nur prüfen, wenn das "Aktiv"-Bit gesetzt ist.
        //   komplett symbolisch adressiert       Teilweise absolut, teilw. symbolisch     komplett absolut Zugriff
        IF(E[Eingaenge.Eingaenge[Zaehler].E_BYTE, DB1.Eingaenge[Zaehler].E_Bit] = true AND db1.dbx[(Zaehler-1)*18+4, 0] = 1) THEN




        END_IF;
    END_FOR;



END_IF;

END_FUNCTION
```

KOP/AWL/FUP - [DB1 -- "Eingaenge" -- Schulung\SIMATIC 300(1) - projekt\CPU 314IFM\...\DB1]

Datei Bearbeiten Einfügen Zielsystem Test Ansicht Extras Fenster Hilfe

Adresse	Name	Typ	Anfan	Aktualwert	Kommentar
0.0	Eingaenge[1].E_Byte	INT	0	124	Eingangsbyte
2.0	Eingaenge[1].E_Bit	INT	0	0	Eingangsbit
4.0	Eingaenge[1].Aktiv	BOOL	FALSE	TRUE	Wird der Eingang verarbeitet?
6.0	Eingaenge[1].E_Name	STRING [10]	''	'Eingang 1'	Wie heißt der Eingang?
18.0	Eingaenge[2].E_Byte	INT	0	124	Eingangsbyte
20.0	Eingaenge[2].E_Bit	INT	0	1	Eingangsbit
22.0	Eingaenge[2].Aktiv	BOOL	FALSE	TRUE	Wird der Eingang verarbeitet?
24.0	Eingaenge[2].E_Name	STRING [10]	''	'Eingang 2'	Wie heißt der Eingang?
36.0	Eingaenge[3].E_Byte	INT	0	124	Eingangsbyte
38.0	Eingaenge[3].E_Bit	INT	0	2	Eingangsbit
40.0	Eingaenge[3].Aktiv	BOOL	FALSE	TRUE	Wird der Eingang verarbeitet?
42.0	Eingaenge[3].E_Name	STRING [10]	''	'Eingang 3'	Wie heißt der Eingang?
54.0	Eingaenge[4].E_Byte	INT	0	124	Eingangsbyte
56.0	Eingaenge[4].E_Bit	INT	0	3	Eingangsbit
58.0	Eingaenge[4].Aktiv	BOOL	FALSE	TRUE	Wird der Eingang verarbeitet?
60.0	Eingaenge[4].E_Name	STRING [10]	''	'Eingang 4'	Wie heißt der Eingang?
72.0	Eingaenge[5].E_Byte	INT	0	124	Eingangsbyte
74.0	Eingaenge[5].E_Bit	INT	0	4	Eingangsbit
76.0	Eingaenge[5].Aktiv	BOOL	FALSE	TRUE	Wird der Eingang verarbeitet?
78.0	Eingaenge[5].E_Name	STRING [10]	''	'Eingang 5'	Wie heißt der Eingang?
90.0	Eingaenge[6].E_Byte	INT	0	124	Eingangsbyte
92.0	Eingaenge[6].E_Bit	INT	0	5	Eingangsbit
94.0	Eingaenge[6].Aktiv	BOOL	FALSE	TRUE	Wird der Eingang verarbeitet?
96.0	Eingaenge[6].E_Name	STRING [10]	''	'Eingang 6'	Wie heißt der Eingang?
108.0	Eingaenge[7].E_Byte	INT	0	124	Eingangsbyte
110.0	Eingaenge[7].E_Bit	INT	0	6	Eingangsbit
112.0	Eingaenge[7].Aktiv	BOOL	FALSE	TRUE	Wird der Eingang verarbeitet?
114.0	Eingaenge[7].E_Name	STRING [10]	''	'Eingang 7'	Wie heißt der Eingang?
126.0	Eingaenge[8].E_Byte	INT	0	124	Eingangsbyte
128.0	Eingaenge[8].E_Bit	INT	0	7	Eingangsbit
130.0	Eingaenge[8].Aktiv	BOOL	FALSE	TRUE	Wird der Eingang verarbeitet?
132.0	Eingaenge[8].E_Name	STRING [10]	''	'Eingang 8'	Wie heißt der Eingang?
144.0	Erster_Eingang	STRING [10]	''	''	Enthält den Namen des ersten aktiven Eingangs
156.0	Zaehler	INT	0	0	Kontrollzähler, ob die Schleife arbeitet

Bibliotheken

Programmelemente Aufrufstruktur

Drücken Sie F1, um Hilfe zu erhalten.

offline Abs < 5.2 Einfg

```
  // temporäre Variablen
Zaehler     : INT;
END_VAR

CONST
  // Konstanten
Anzahl_Eingaenge := 8;
END_CONST
BEGIN

//Wenn die Prüfbedingung (hier der 1Hz-Taktmerker) stimmt, sollen einmalig (deshalb auf Flanke des Merkers geprüft)
//die Eingänge abgefragt werden.
IF(Pruefbedingung AND NOT PFlanke_Takt_1Hz) THEN
    //Das im DB gespeicherte Array durchlaufen
    FOR Zaehler := 1 TO Anzahl_Eingaenge BY 1 DO
        //Eingang abfragen: indirekte Abfrage und nur prüfen, wenn das "Aktiv"-Bit gesetzt ist.
        //   komplett symbolisch adressiert       Teilweise absolut, teilw. symbolisch     komplett absolut Zugriff
        IF(E[Eingaenge.Eingaenge[Zaehler].E_BYTE, DB1.Eingaenge[Zaehler].E_Bit] = true AND db1.dbx[(Zaehler-1)*18+4, 0] = 1) THEN




        END_IF;
    END_FOR;



END_IF;

END_FUNCTION
```

```
  // temporäre Variablen
Zaehler     : INT;
END_VAR

CONST
  // Konstanten
Anzahl_Eingaenge := 8;
END_CONST
BEGIN

//Wenn die Prüfbedingung (hier der 1Hz-Taktmerker) stimmt, sollen einmalig (deshalb auf Flanke des Merkers geprüft)
//die Eingänge abgefragt werden.
IF(Pruefbedingung AND NOT PFlanke_Takt_1Hz) THEN
    //Das im DB gespeicherte Array durchlaufen
    FOR Zaehler := 1 TO Anzahl_Eingaenge BY 1 DO
        //Eingang abfragen: indirekte Abfrage und nur prüfen, wenn das "Aktiv"-Bit gesetzt ist.
        //   komplett symbolisch adressiert       Teilweise absolut, teilw. symbolisch     komplett absolut Zugriff
        IF(E[Eingaenge.Eingaenge[Zaehler].E_BYTE, DB1.Eingaenge[Zaehler].E_Bit] = true AND db1.dbx[(Zaehler-1)*18+4, 0] = 1) THEN
            //Den Namen des ersten aktiven Eingangs in den Speicher schreiben
            Eingaenge.Erster_Eingang := Eingaenge.Eingaenge[Zaehler].E_Name;


      END_IF;
    END_FOR;



END_IF;

END_FUNCTION
```

```
  // temporäre Variablen
Zaehler     : INT;
END_VAR

CONST
  // Konstanten
Anzahl_Eingaenge := 8;
END_CONST
BEGIN

//Wenn die Prüfbedingung (hier der 1Hz-Taktmerker) stimmt, sollen einmalig (deshalb auf Flanke des Merkers geprüft)
//die Eingänge abgefragt werden.
IF(Pruefbedingung AND NOT PFlanke_Takt_1Hz) THEN
    //Das im DB gespeicherte Array durchlaufen
    FOR Zaehler := 1 TO Anzahl_Eingaenge BY 1 DO
        //Eingang abfragen: indirekte Abfrage und nur prüfen, wenn das "Aktiv"-Bit gesetzt ist.
        //   komplett symbolisch adressiert       Teilweise absolut, teilw. symbolisch     komplett absolut Zugriff
        IF(E[Eingaenge.Eingaenge[Zaehler].E_BYTE, DB1.Eingaenge[Zaehler].E_Bit] = true AND dbl.dbx[(Zaehler-1)*18+4, 0] = 1) THEN
            //Den Namen des ersten aktiven Eingangs in den Speicher schreiben
            Eingaenge.Erster_Eingang := Eingaenge.Eingaenge[Zaehler].E_Name;
            //Da ein Wert gefunden wurde, nicht weitersuchen
            EXIT;
      END_IF;
    END_FOR;



END_IF;

END_FUNCTION
```

```
  // temporäre Variablen
Zaehler     : INT;
END_VAR

CONST
  // Konstanten
Anzahl_Eingaenge := 8;
END_CONST
BEGIN

//Wenn die Prüfbedingung (hier der 1Hz-Taktmerker) stimmt, sollen einmalig (deshalb auf Flanke des Merkers geprüft)
//die Eingänge abgefragt werden.
IF(Pruefbedingung AND NOT PFlanke_Takt_1Hz) THEN
    //Das im DB gespeicherte Array durchlaufen
    FOR Zaehler := 1 TO Anzahl_Eingaenge BY 1 DO
        //Eingang abfragen: indirekte Abfrage und nur prüfen, wenn das "Aktiv"-Bit gesetzt ist.
        //   komplett symbolisch adressiert       Teilweise absolut, teilw. symbolisch     komplett absolut Zugriff
        IF(E[Eingaenge.Eingaenge[Zaehler].E_Byte, DBl.Eingaenge[Zaehler].E_Bit] = true AND dbl.dbx[(Zaehler-1)*18+4, 0] = 1) THEN
            //Den Namen des ersten aktiven Eingangs in den Speicher schreiben
            Eingaenge.Erster_Eingang := Eingaenge.Eingaenge[Zaehler].E_Name;
            //Da ein Wert gefunden wurde, nicht weitersuchen
            EXIT;
        END_IF;
    END_FOR;

    //Kontrollzaehler mitlaufen lassen: Bei jedem Aufruf wird 1 hochgezählt,
    //damit die Variable nicht überläuft und Fehler verursacht, bei 10000 zurücksetzen.
    IF(Eingaenge.Zaehler<10000) THEN
        Eingaenge.Zaehler := Eingaenge.Zaehler+1;
    ELSE
        Eingaenge.Zaehler := 0;
    END_IF;
END_IF;

END_FUNCTION
```

```
  // temporäre Variablen
Zaehler     : INT;
END_VAR

CONST
  // Konstanten
Anzahl_Eingaenge := 8;
END_CONST
BEGIN

//Wenn die Prüfbedingung (hier der 1Hz-Taktmerker) stimmt, sollen einmalig (deshalb auf Flanke des Merkers geprüft)
//die Eingänge abgefragt werden.
IF(Pruefbedingung AND NOT PFlanke_Takt_1Hz) THEN
    //Das im DB gespeicherte Array durchlaufen
    FOR Zaehler := 1 TO Anzahl_Eingaenge BY 1 DO
        //Eingang abfragen: indirekte Abfrage und nur prüfen, wenn das "Aktiv"-Bit gesetzt ist.
        //   komplett symbolisch adressiert       Teilweise absolut, teilw. symbolisch     komplett absolut Zugriff
        IF(E[Eingaenge.Eingaenge[Zaehler].E_Byte, DBl.Eingaenge[Zaehler].E_Bit] = true AND dbl.dbx[(Zaehler-1)*18+4, 0] = 1) THEN
            //Den Namen des ersten aktiven Eingangs in den Speicher schreiben
            Eingaenge.Erster_Eingang := Eingaenge.Eingaenge[Zaehler].E_Name;
            //Da ein Wert gefunden wurde, nicht weitersuchen
            EXIT;
        END_IF;
    END_FOR;

    //Kontrollzaehler mitlaufen lassen: Bei jedem Aufruf wird 1 hochgezählt,
    //damit die Variable nicht überläuft und Fehler verursacht, bei 10000 zurücksetzen.
    IF(Eingaenge.Zaehler<10000) THEN
        Eingaenge.Zaehler := Eingaenge.Zaehler+1;
    ELSE
        Eingaenge.Zaehler := 0;
    END_IF;
END_IF;

//Flankenmerker setzen.
PFlanke_Takt_1Hz := Pruefbedingung;









END_FUNCTION
```

SCL - [FC_Abfrage -- Schulung\SIMATIC 300(1) - projekt\CPU 314IFM]

Datei Bearbeiten Einfügen Zielsystem Test Ansicht Extras Fenster Hilfe

```
  // temporäre Variablen
Zaehler     : INT;
END_VAR

CONST
  // Konstanten
Anzahl_Eingaenge := 8;
END_CONST
BEGIN

//Wenn die Prüfbedingung (hier der 1Hz-Taktmerker) stimmt, sollen einmalig (deshalb auf Flanke des Merkers geprüft)
//die Eingänge abgefragt werden.
IF(Pruefbedingung AND NOT PFlanke_Takt_1Hz) THEN
    //Das im DB gespeicherte Array durchlaufen
    FOR Zaehler := 1 TO Anzahl_Eingaenge BY 1 DO
        //Eingang abfragen: indirekte Abfrage und nur prüfen, wenn das "Aktiv"-Bit gesetzt ist.
        //   komplett symbolisch adressiert       Teilweise absolut, teilw. symbolisch     komplett absolut Zugriff
        IF(E[Eingaenge.Eingaenge[Zaehler].E_Byte, DBl.Eingaenge[Zaehler].E_Bit] = true AND dbl.dbx[(Zaehler-1)*18+4, 0] = 1) THEN
            //Den Namen des ersten aktiven Eingangs in den Speicher schreiben
            Eingaenge.Erster_Eingang := Eingaenge.Eingaenge[Zaehler].E_Name;
            //Da ein Wert gefunden wurde, nicht weitersuchen
            EXIT;
        END_IF;
    END_FOR;

    //Kontrollzaehler mitlaufen lassen: Bei jedem Aufruf wird 1 hochgezählt,
    //damit die Variable nicht überläuft und Fehler verursacht, bei 10000 zurücksetzen.
    IF(Eingaenge.Zaehler<10000) THEN
        Eingaenge.Zaehler := Eingaenge.Zaehler+1;
    ELSE
        Eingaenge.Zaehler := 0;
    END_IF;
END_IF;

//Flankenmerker setzen.
PFlanke_Takt_1Hz := Pruefbedingung;

//Bausteinrückgabewert: Wenn ein aktiver Eingang gefunden wurde, 1, sonst 0
IF(Zaehler <= 8) THEN
    Abfrage := 1;
ELSE
    Abfrage := false;
    //Wenn kein aktiver Eingang gefunden wurde, den String löschen.
    Eingaenge.Erster_Eingang := '          ';
END_IF;
END_FUNCTION
```

Fehler | Info

Drücken Sie F1, um Hilfe zu erhalten.

Ze 79 Sp 1 | Einfg | Änd

SIMATIC Manager - [Schulung -- g:\Schulung]

Datei Bearbeiten Einfügen Zielsystem Ansicht Extras Fenster Hilfe

< Kein Filter >

- Schulung
  - SIMATIC 300(1) - leer
    - CPU 314IFM
      - S7-Programm(1)
        - Quellen
        - Bausteine
  - SIMATIC 300(1) - projekt
    - CPU 314IFM
      - S7-Programm(1)
        - Quellen
        - Bausteine

Objektname	Symbolischer Name	Typ	Größe	Autor	Änderungsdatum	Kommentar
DB_Eingaenge	---	SCL-Quelle	1556		05.12.2007 08:38:04	
FC_Abfrage	---	SCL-Quelle	2553		05.12.2007 16:38:08	
FC_Ausgang	---	SCL-Quelle	1968		05.12.2007 16:24:08	
OB_1	---	SCL-Quelle	523		05.12.2007 15:55:18	
UDT_Eingang	---	SCL-Quelle	254		04.12.2007 15:32:34	
Alles_Uebersetzen	---	SCL-Übersetzungssteuerdatei	479		05.12.2007 16:41:16	

SCL-Quelle

Drücken Sie F1, um Hilfe zu erhalten.

TCP/IP -> 3Com EtherLink XL 10/1...

1968 Bytes

SCL - [FC_Ausgang -- Schulung\SIMATIC 300(1) - projekt\CPU 314IFM]

Datei Bearbeiten Einfügen Zielsystem Test Ansicht Extras Fenster Hilfe

Bausteinaufruf...
Bausteinvorlage ▸
Kontrollstruktur ▸

OB
FB
FC

Kommentar
Konstante
Parameter

DB
Instanz DB
DB aus UDT
UDT

Fehler Info

Fügt Textgerüst für neue Funktion an der Cursorposition ein.

Ze 1 Sp 1 Einfg Änd

SCL - [FC_Ausgang -- Schulung\SIMATIC 300(1) - projekt\CPU 314IFM]

Datei Bearbeiten Einfügen Zielsystem Test Ansicht Extras Fenster Hilfe

Einfügen:
- Bausteinaufruf...
- Bausteinvorlage ▸
  - OB
  - FB
  - FC
  - Kommentar
  - Konstante
  - Parameter
  - DB
  - Instanz DB
  - DB aus UDT
  - UDT
- Kontrollstruktur ▸

```
FUNCTION FCxxx : INT

VAR_TEMP
  // temporäre Variablen

END_VAR

  // Anweisungsteil
  ;
  FCxxx := 100;
END_FUNCTION
```

Fehler | Info

Fügt Textgerüst für neuen Baustein-Kommentar an der Cursorposition ein.

Ze 3 Sp 1 | Einfg | Änd

```
FUNCTION Ausgang : BOOL

TITLE = 'Ausgang'
//
// Baustein setzt anhand des gespeicherten "Ersten Eingangs" die Ausgänge
// Erstellt:    04.12.07, Schürmann, HIT
// Geändert:
//
VERSION : '1.0'
AUTHOR  : Sc
NAME    : Ausgang
FAMILY  : FCs

VAR_TEMP
  // temporäre Variablen
  Ausgang_Bit   : INT;
END_VAR
BEGIN

//Funktionsrückgabewert
  Ausgang := 1;
END_FUNCTION
```

```
FUNCTION Ausgang : BOOL

TITLE = 'Ausgang'
//
// Baustein setzt anhand des gespeicherten "Ersten Eingangs" die Ausgänge
// Erstellt:    04.12.07, Schürmann, HIT
// Geändert:
//
VERSION : '1.0'
AUTHOR  : Sc
NAME    : Ausgang
FAMILY  : FCs

VAR_TEMP
  // temporäre Variablen
  Ausgang_Bit   : INT;
END_VAR
BEGIN

//Falls kein entsprechender Eingang gesetzt ist, sicherheitshalber einen ungewöhnlichen Wert setzen,
//um den Fall abzufangen.
Ausgang_Bit := 8;


//Funktionsrückgabewert
  Ausgang := 1;
END_FUNCTION
```

```
FUNCTION Ausgang : BOOL

TITLE = 'Ausgang'
//
// Baustein setzt anhand des gespeicherten "Ersten Eingangs" die Ausgänge
// Erstellt:    04.12.07, Schürmann, HIT
// Geändert:
//
VERSION : '1.0'
AUTHOR  : Sc
NAME    : Ausgang
FAMILY  : FCs

VAR_TEMP
  // temporäre Variablen
  Ausgang_Bit   : INT;
END_VAR
BEGIN

//Falls kein entsprechender Eingang gesetzt ist, sicherheitshalber einen ungewöhnlichen Wert setzen,
//um den Fall abzufangen.
Ausgang_Bit := 8;

//Da in CASE nur INTs abgefragt werden können, werden hier die Strings verglichen und ein entsprechender
//INT-Wert gesetzt
IF(Eingaenge.Erster_Eingang = 'Eingang 1') THEN
    Ausgang_Bit := 0;
END_IF;

//Funktionsrückgabewert
  Ausgang := 1;
END_FUNCTION
```

```
FUNCTION Ausgang : BOOL

TITLE = 'Ausgang'
//
// Baustein setzt anhand des gespeicherten "Ersten Eingangs" die Ausgänge
// Erstellt:    04.12.07, Schürmann, HIT
// Geändert:
//
VERSION : '1.0'
AUTHOR  : Sc
NAME    : Ausgang
FAMILY  : FCs

VAR_TEMP
  // temporäre Variablen
  Ausgang_Bit   : INT;
END_VAR
BEGIN

//Falls kein entsprechender Eingang gesetzt ist, sicherheitshalber einen ungewöhnlichen Wert setzen,
//um den Fall abzufangen.
Ausgang_Bit := 8;

//Da in CASE nur INTs abgefragt werden können, werden hier die Strings verglichen und ein entsprechender
//INT-Wert gesetzt
IF(Eingaenge.Erster_Eingang = 'Eingang 1') THEN
    Ausgang_Bit := 0;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 2') THEN
    Ausgang_Bit := 1;
END_IF;

//Funktionsrückgabewert
  Ausgang := 1;
END_FUNCTION
```

```
FUNCTION Ausgang : BOOL

TITLE = 'Ausgang'
//
// Baustein setzt anhand des gespeicherten "Ersten Eingangs" die Ausgänge
// Erstellt:    04.12.07, Schürmann, HIT
// Geändert:
//
VERSION : '1.0'
AUTHOR  : Sc
NAME    : Ausgang
FAMILY  : FCs

VAR_TEMP
  // temporäre Variablen
  Ausgang_Bit   : INT;
END_VAR
BEGIN

//Falls kein entsprechender Eingang gesetzt ist, sicherheitshalber einen ungewöhnlichen Wert setzen,
//um den Fall abzufangen.
Ausgang_Bit := 8;

//Da in CASE nur INTs abgefragt werden können, werden hier die Strings verglichen und ein entsprechender
//INT-Wert gesetzt
IF(Eingaenge.Erster_Eingang = 'Eingang 1') THEN
    Ausgang_Bit := 0;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 2') THEN
    Ausgang_Bit := 1;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 3') THEN
    Ausgang_Bit := 2;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 4') THEN
    Ausgang_Bit := 3;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 5') THEN
    Ausgang_Bit := 4;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 6') THEN
    Ausgang_Bit := 5;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 7') THEN
    Ausgang_Bit := 6;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 8') THEN
    Ausgang_Bit := 7;
END_IF;


//Funktionsrückgabewert
  Ausgang := 1;
```

```
    Ausgang_Bit := 0;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 2') THEN
    Ausgang_Bit := 1;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 3') THEN
    Ausgang_Bit := 2;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 4') THEN
    Ausgang_Bit := 3;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 5') THEN
    Ausgang_Bit := 4;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 6') THEN
    Ausgang_Bit := 5;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 7') THEN
    Ausgang_Bit := 6;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 8') THEN
    Ausgang_Bit := 7;
END_IF;

//Erst einmal alle Ausgänge zurücksetzen
AB124 := B#16#0;









//Funktionsrückgabewert
  Ausgang := 1;
END_FUNCTION
```

SCL - [FC_Ausgang -- Schulung\SIMATIC 300(1) - projekt\CPU 314IFM]

Datei Bearbeiten Einfügen Zielsystem Test Ansicht Extras Fenster Hilfe

Einfügen:
- Bausteinaufruf...
- Bausteinvorlage ▸
- Kontrollstruktur ▸
  - IF
  - CASE
  - FOR
  - WHILE
  - REPEAT

```
    Ausgang_Bit := 0;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 2') THEN
    Ausgang_Bit := 1;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 3') THEN
    Ausgang_Bit := 2;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 4') THEN
    Ausgang_Bit := 3;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 5') THEN
    Ausgang_Bit := 4;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 6') THEN
    Ausgang_Bit := 5;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 7') THEN
    Ausgang_Bit := 6;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 8') THEN
    Ausgang_Bit := 7;
END_IF;

//Erst einmal alle Ausgänge zurücksetzen
AB124 := B#16#0;









//Funktionsrückgabewert
  Ausgang := 1;
END_FUNCTION
```

Fügt Textgerüst für CASE-Anweisung an der Cursorposition ein.

Ze 62 Sp 1 | Einfg | Änd

```
    Ausgang_Bit := 0;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 2') THEN
    Ausgang_Bit := 1;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 3') THEN
    Ausgang_Bit := 2;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 4') THEN
    Ausgang_Bit := 3;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 5') THEN
    Ausgang_Bit := 4;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 6') THEN
    Ausgang_Bit := 5;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 7') THEN
    Ausgang_Bit := 6;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 8') THEN
    Ausgang_Bit := 7;
END_IF;

//Erst einmal alle Ausgänge zurücksetzen
AB124 := B#16#0;


  CASE wert OF
    0..3 :
      // Anweisungen_0..3
      ;
    8 :
      // Anweisungen_8
      ;
  ELSE:
    // Anweisungen_ELSE
    ;
  END_CASE;




//Funktionsrückgabewert
  Ausgang := 1;
END_FUNCTION
```

```
    Ausgang_Bit := 0;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 2') THEN
    Ausgang_Bit := 1;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 3') THEN
    Ausgang_Bit := 2;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 4') THEN
    Ausgang_Bit := 3;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 5') THEN
    Ausgang_Bit := 4;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 6') THEN
    Ausgang_Bit := 5;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 7') THEN
    Ausgang_Bit := 6;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 8') THEN
    Ausgang_Bit := 7;
END_IF;

//Erst einmal alle Ausgänge zurücksetzen
AB124 := B#16#0;

//Abfrage des oben gesetzten INTs
  CASE Ausgang_Bit OF









  ELSE:
                //Falls ein nicht erwarteter Wert ankommt, alle Ausgänge setzen
                AB124 := B#16#FF;
  END_CASE;

//Funktionsrückgabewert
  Ausgang := 1;
END_FUNCTION
```

```
    Ausgang_Bit := 0;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 2') THEN
    Ausgang_Bit := 1;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 3') THEN
    Ausgang_Bit := 2;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 4') THEN
    Ausgang_Bit := 3;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 5') THEN
    Ausgang_Bit := 4;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 6') THEN
    Ausgang_Bit := 5;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 7') THEN
    Ausgang_Bit := 6;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 8') THEN
    Ausgang_Bit := 7;
END_IF;

//Erst einmal alle Ausgänge zurücksetzen
AB124 := B#16#0;

//Abfrage des oben gesetzten INTs
  CASE Ausgang_Bit OF
            0 :

            1 :

            2 :

         3..5 :



            6 :

            7 :

  ELSE:
                //Falls ein nicht erwarteter Wert ankommt, alle Ausgänge setzen
                AB124 := B#16#FF;
  END_CASE;

//Funktionsrückgabewert
  Ausgang := 1;
END_FUNCTION
```

```
    Ausgang_Bit := 0;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 2') THEN
    Ausgang_Bit := 1;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 3') THEN
    Ausgang_Bit := 2;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 4') THEN
    Ausgang_Bit := 3;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 5') THEN
    Ausgang_Bit := 4;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 6') THEN
    Ausgang_Bit := 5;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 7') THEN
    Ausgang_Bit := 6;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 8') THEN
    Ausgang_Bit := 7;
END_IF;

//Erst einmal alle Ausgänge zurücksetzen
AB124 := B#16#0;

//Abfrage des oben gesetzten INTs
  CASE Ausgang_Bit OF
            0 :
                 A124.0 := true;
            1 :

            2 :

          3..5 :



            6 :

            7 :

  ELSE:
                //Falls ein nicht erwarteter Wert ankommt, alle Ausgänge setzen
                AB124 := B#16#FF;
  END_CASE;

//Funktionsrückgabewert
  Ausgang := 1;
END_FUNCTION
```

```
    Ausgang_Bit := 0;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 2') THEN
    Ausgang_Bit := 1;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 3') THEN
    Ausgang_Bit := 2;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 4') THEN
    Ausgang_Bit := 3;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 5') THEN
    Ausgang_Bit := 4;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 6') THEN
    Ausgang_Bit := 5;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 7') THEN
    Ausgang_Bit := 6;
END_IF;

IF(Eingaenge.Erster_Eingang = 'Eingang 8') THEN
    Ausgang_Bit := 7;
END_IF;

//Erst einmal alle Ausgänge zurücksetzen
AB124 := B#16#0;

//Abfrage des oben gesetzten INTs
  CASE Ausgang_Bit OF
            0 :
                A124.0 := true;
            1 :
                A124.1 := true;
            2 :
                A124.2 := true;
         3..5 :
                A124.3 := true;
                A124.4 := true;
                A124.5 := true;
            6 :
                A124.6 := true;
            7 :
                A124.7 := true;
  ELSE:
                //Falls ein nicht erwarteter Wert ankommt, alle Ausgänge setzen
                AB124 := B#16#FF;
  END_CASE;

//Funktionsrückgabewert
  Ausgang := 1;
END_FUNCTION
```

SIMATIC Manager - [Schulung -- g:\Schulung]

Datei Bearbeiten Einfügen Zielsystem Ansicht Extras Fenster Hilfe

< Kein Filter >

- Schulung
  - SIMATIC 300(1) - leer
    - CPU 314IFM
      - S7-Programm(1)
        - Quellen
        - Bausteine
  - SIMATIC 300(1) - projekt
    - CPU 314IFM
      - S7-Programm(1)
        - Quellen
        - Bausteine

Objektname	Symbolischer Name	Typ	Größe	Autor	Änderungsdatum	Kommentar
DB_Eingaenge	---	SCL-Quelle	1556		05.12.2007 08:38:04	
FC_Abfrage	---	SCL-Quelle	2553		05.12.2007 16:38:08	
FC_Ausgang	---	SCL-Quelle	1968		05.12.2007 16:24:08	
OB_1	---	SCL-Quelle	523		05.12.2007 15:55:18	
UDT_Eingang	---	SCL-Quelle	254		04.12.2007 15:32:34	
Alles_Uebersetzen	---	SCL-Übersetzungssteuerdatei	479		05.12.2007 16:41:16	

SCL-Quelle

Drücken Sie F1, um Hilfe zu erhalten.

TCP/IP -> 3Com EtherLink XL 10/1...

523 Bytes

SCL - [OB_1 -- Schulung\SIMATIC 300(1) - projekt\CPU 314IFM]

Datei Bearbeiten Einfügen Zielsystem Test Ansicht Extras Fenster Hilfe

Bausteinaufruf...
Bausteinvorlage ▸
Kontrollstruktur ▸

OB
FB
FC

Kommentar
Konstante
Parameter

DB
Instanz DB
DB aus UDT
UDT

Fügt Textgerüst für neuen Organisationsbaustein an der Cursorposition ein.

Ze 1 Sp 1 Einfg Änd

SCL - [OB_1 -- Schulung\SIMATIC 300(1) - projekt\CPU 314IFM]

Datei Bearbeiten Einfügen Zielsystem Test Ansicht Extras Fenster Hilfe

Einfügen:
- Bausteinaufruf...
- Bausteinvorlage ▸
  - OB
  - FB
  - FC
  - Kommentar
  - Konstante
  - Parameter
  - DB
  - Instanz DB
  - DB aus UDT
  - UDT
- Kontrollstruktur ▸

```
ORGANIZATION_BLOCK OBxxx

VAR_TEMP
  // reserviert
  info : ARRAY[0..19] OF BYTE;
  // temporäre Variablen

END_VAR

  // Anweisungen

  ;
END_ORGANIZATION_BLOCK
```

Fügt Textgerüst für neuen Baustein-Kommentar an der Cursorposition ein.

Ze 3 Sp 1 | Einfg | Änd

SCL - [OB_1 -- Schulung\SIMATIC 300(1) - projekt\CPU 314IFM]

Datei Bearbeiten Einfügen Zielsystem Test Ansicht Extras Fenster Hilfe

```
ORGANIZATION_BLOCK OBxxx

TITLE = 'Baustein-Überschrift'
//
// Baustein-Kommentar ...
//
VERSION : '1.0'
AUTHOR  : author
NAME    : name
FAMILY  : family

VAR_TEMP
  // reserviert
  info : ARRAY[0..19] OF BYTE;
  // temporäre Variablen

END_VAR

  // Anweisungen

  ;
END_ORGANIZATION_BLOCK
```

Drücken Sie F1, um Hilfe zu erhalten.

Ze 12 Sp 1 Einfg Änd

```
ORGANIZATION_BLOCK OB1

TITLE = 'Hauptprogramm'
//
//  Hauptprogramm OB1
//
// Erstellt: 04.12.2007, Schürmann, HIT
// Geändert:
//
VERSION : '1.0'
AUTHOR  : Sc
NAME    : Main
FAMILY  : OBs

VAR_TEMP
  // reserviert
  info : ARRAY[0..19] OF BYTE;
  // temporäre Variablen

END_VAR

  // Anweisungen



END_ORGANIZATION_BLOCK
```

```
ORGANIZATION_BLOCK OB1

TITLE = 'Hauptprogramm'
//
//  Hauptprogramm OB1
//
// Erstellt: 04.12.2007, Schürmann, HIT
// Geändert:
//
VERSION : '1.0'
AUTHOR  : Sc
NAME    : Main
FAMILY  : OBs

VAR_TEMP
  // reserviert
  info : ARRAY[0..19] OF BYTE;
  // temporäre Variablen

END_VAR

  // Anweisungen

  M3.0 := Abfrage(Pruefbedingung := Takt_1Hz // IN: BOOL
          ); // BOOL



END_ORGANIZATION_BLOCK
```

SCL - [OB_1 -- Schulung\SIMATIC 300(1) - projekt\CPU 314IFM]

Datei Bearbeiten Einfügen Zielsystem Test Ansicht Extras Fenster Hilfe

Einfügen:
- Bausteinaufruf...
- Bausteinvorlage ▸
- Kontrollstruktur ▸
  - IF
  - CASE
  - FOR
  - WHILE
  - REPEAT

```
ORGANIZATION_BLOCK OB1

TITLE = 'Hauptprogramm'
//
//  Hauptprogramm OB1
//
// Erstellt: 04.12.2007, Schürmann, HIT
// Geändert:
//
VERSION : '1.0'
AUTHOR  : Sc
NAME    : Main
FAMILY  : OBs

VAR_TEMP
  // reserviert
  info : ARRAY[0..19] OF BYTE;
  // temporäre Variablen

END_VAR

  // Anweisungen

  M3.0 := Abfrage(Pruefbedingung := Takt_1Hz // IN: BOOL
          ); // BOOL




END_ORGANIZATION_BLOCK
```

Fügt Textgerüst für IF-Anweisung an der Cursorposition ein.

Ze 33 Sp 1 | Einfg | Änd

```
ORGANIZATION_BLOCK OB1

TITLE = 'Hauptprogramm'
//
//  Hauptprogramm OB1
//
// Erstellt: 04.12.2007, Schürmann, HIT
// Geändert:
//
VERSION : '1.0'
AUTHOR  : Sc
NAME    : Main
FAMILY  : OBs

VAR_TEMP
  // reserviert
  info : ARRAY[0..19] OF BYTE;
  // temporäre Variablen

END_VAR

  // Anweisungen

  M3.0 := Abfrage(Pruefbedingung := Takt_1Hz // IN: BOOL
          ); // BOOL

  IF a = b THEN
    // Anweisungsteil_IF
    ;
  ELSIF a = c THEN
    // Anweisungsteil_ELSIF
    ;
  ELSE
    // Anweisungsteil_ELSE
    ;
  END_IF;

END_ORGANIZATION_BLOCK
```

```
ORGANIZATION_BLOCK OB1

TITLE = 'Hauptprogramm'
//
//  Hauptprogramm OB1
//
// Erstellt: 04.12.2007, Schürmann, HIT
// Geändert:
//
VERSION : '1.0'
AUTHOR  : Sc
NAME    : Main
FAMILY  : OBs

VAR_TEMP
  // reserviert
  info : ARRAY[0..19] OF BYTE;
  // temporäre Variablen

END_VAR

  // Anweisungen

  M3.0 := Abfrage(Pruefbedingung := Takt_1Hz // IN: BOOL
          ); // BOOL

  IF a = b THEN

  ELSE

  END_IF;

END_ORGANIZATION_BLOCK
```

```
ORGANIZATION_BLOCK OB1

TITLE = 'Hauptprogramm'
//
//  Hauptprogramm OB1
//
// Erstellt: 04.12.2007, Schürmann, HIT
// Geändert:
//
VERSION : '1.0'
AUTHOR  : Sc
NAME    : Main
FAMILY  : OBs

VAR_TEMP
  // reserviert
  info : ARRAY[0..19] OF BYTE;
  // temporäre Variablen

END_VAR

  // Anweisungen

  M3.0 := Abfrage(Pruefbedingung := Takt_1Hz // IN: BOOL
          ); // BOOL

  IF(M3.0) THEN

  ELSE

  END_IF;

END_ORGANIZATION_BLOCK
```

```
ORGANIZATION_BLOCK OB1

TITLE = 'Hauptprogramm'
//
//  Hauptprogramm OB1
//
// Erstellt: 04.12.2007, Schürmann, HIT
// Geändert:
//
VERSION : '1.0'
AUTHOR  : Sc
NAME    : Main
FAMILY  : OBs

VAR_TEMP
  // reserviert
  info : ARRAY[0..19] OF BYTE;
  // temporäre Variablen

END_VAR

  // Anweisungen

  M3.0 := Abfrage(Pruefbedingung := Takt_1Hz // IN: BOOL
          ); // BOOL

  IF(M3.0) THEN
      M3.1 := Ausgang();
  ELSE

  END_IF;

END_ORGANIZATION_BLOCK
```

```
ORGANIZATION_BLOCK OB1

TITLE = 'Hauptprogramm'
//
//  Hauptprogramm OB1
//
// Erstellt: 04.12.2007, Schürmann, HIT
// Geändert:
//
VERSION : '1.0'
AUTHOR  : Sc
NAME    : Main
FAMILY  : OBs

VAR_TEMP
  // reserviert
  info : ARRAY[0..19] OF BYTE;
  // temporäre Variablen

END_VAR

  // Anweisungen

  M3.0 := Abfrage(Pruefbedingung := Takt_1Hz // IN: BOOL
          ); // BOOL

  IF(M3.0) THEN
      M3.1 := Ausgang();
  ELSE
      AB124 := B#16#0;
  END_IF;

END_ORGANIZATION_BLOCK
```

SIMATIC Manager - [Schulung -- g:\Schulung]

Datei Bearbeiten Einfügen Zielsystem Ansicht Extras Fenster Hilfe

< Kein Filter >

- Schulung
  - SIMATIC 300(1) - leer
    - CPU 314IFM
      - S7-Programm(1)
        - Quellen
        - Bausteine
  - SIMATIC 300(1) - projekt
    - CPU 314IFM
      - S7-Programm(1)
        - Quellen
        - Bausteine

Objektname	Symbolischer Name	Typ	Größe	Autor	Änderungsdatum	Kommentar
DB_Eingaenge	---	SCL-Quelle	1556		05.12.2007 08:38:04	
FC_Abfrage	---	SCL-Quelle	2553		05.12.2007 16:38:08	
FC_Ausgang	---	SCL-Quelle	1968		05.12.2007 16:24:08	
OB_1	---	SCL-Quelle	523		05.12.2007 15:55:18	
UDT_Eingang	---	SCL-Quelle	254		04.12.2007 15:32:34	
Alles_Uebersetzen	---	SCL-Übersetzungssteuerdatei	479		05.12.2007 16:41:16	

Drücken Sie F1, um Hilfe zu erhalten.

TCP/IP -> 3Com EtherLink XL 10/1...

479 Bytes

SCL - [Alles_Uebersetzen -- Schulung\SIMATIC 300(1) - projekt\CPU 314IFM]

Datei Bearbeiten Einfügen Zielsystem Test Ansicht Extras Fenster Hilfe

Fehler Info

Drücken Sie F1, um Hilfe zu erhalten.

Ze 1 Sp 1 Einfg Änd

```
//Die nachfolgenden Bausteine werden in der Reihenfolge übersetzt, wie sie hier aufgeführt sind.
//Somit wird bei einem komplexen Programm dem Benutzer ermöglicht, die Reihenfolge selbst zu bestimmt,
//ohne hundert Quellen einzeln aufzurufen und zu übersetzen.
//Wichtig ist, daß alle in einer Quelle benutzten Bausteine bereits übersetzt sind!
//OB1 muß demzufolge beispielsweise immer als letztes übersetzt werden!
```

```
//Die nachfolgenden Bausteine werden in der Reihenfolge übersetzt, wie sie hier aufgeführt sind.
//Somit wird bei einem komplexen Programm dem Benutzer ermöglicht, die Reihenfolge selbst zu bestimmt,
//ohne hundert Quellen einzeln aufzurufen und zu übersetzen.
//Wichtig ist, daß alle in einer Quelle benutzten Bausteine bereits übersetzt sind!
//OB1 muß demzufolge beispielsweise immer als letztes übersetzt werden!

UDT_Eingang
```

```
//Die nachfolgenden Bausteine werden in der Reihenfolge übersetzt, wie sie hier aufgeführt sind.
//Somit wird bei einem komplexen Programm dem Benutzer ermöglicht, die Reihenfolge selbst zu bestimmt,
//ohne hundert Quellen einzeln aufzurufen und zu übersetzen.
//Wichtig ist, daß alle in einer Quelle benutzten Bausteine bereits übersetzt sind!
//OB1 muß demzufolge beispielsweise immer als letztes übersetzt werden!

UDT_Eingang
DB_Eingaenge
```

SCL - [Alles_Uebersetzen -- Schulung\SIMATIC 300(1) - projekt\CPU 314IFM]

Datei Bearbeiten Einfügen Zielsystem Test Ansicht Extras Fenster Hilfe

```
//Die nachfolgenden Bausteine werden in der Reihenfolge übersetzt, wie sie hier aufgeführt sind.
//Somit wird bei einem komplexen Programm dem Benutzer ermöglicht, die Reihenfolge selbst zu bestimmt,
//ohne hundert Quellen einzeln aufzurufen und zu übersetzen.
//Wichtig ist, daß alle in einer Quelle benutzten Bausteine bereits übersetzt sind!
//OB1 muß demzufolge beispielsweise immer als letztes übersetzt werden!

UDT_Eingang
DB_Eingaenge
FC_Abfrage
```

Fehler | Info

Drücken Sie F1, um Hilfe zu erhalten.

Ze 10 Sp 1 | Einfg | Änd

```
//Die nachfolgenden Bausteine werden in der Reihenfolge übersetzt, wie sie hier aufgeführt sind.
//Somit wird bei einem komplexen Programm dem Benutzer ermöglicht, die Reihenfolge selbst zu bestimmt,
//ohne hundert Quellen einzeln aufzurufen und zu übersetzen.
//Wichtig ist, daß alle in einer Quelle benutzten Bausteine bereits übersetzt sind!
//OB1 muß demzufolge beispielsweise immer als letztes übersetzt werden!

UDT_Eingang
DB_Eingaenge
FC_Abfrage
FC_Ausgang
```

```
//Die nachfolgenden Bausteine werden in der Reihenfolge übersetzt, wie sie hier aufgeführt sind.
//Somit wird bei einem komplexen Programm dem Benutzer ermöglicht, die Reihenfolge selbst zu bestimmt,
//ohne hundert Quellen einzeln aufzurufen und zu übersetzen.
//Wichtig ist, daß alle in einer Quelle benutzten Bausteine bereits übersetzt sind!
//OB1 muß demzufolge beispielsweise immer als letztes übersetzt werden!

UDT_Eingang
DB_Eingaenge
FC_Abfrage
FC_Ausgang
OB_1
```

Symbol Editor - [S7-Programm(1) (Symbole) -- Schulung\SIMATIC 300(1) - projekt\CPU 314IFM]

Tabelle Bearbeiten Einfügen Ansicht Extras Fenster Hilfe

Alle Symbole

	Status	Symbol	Adresse	Datentyp	Kommentar
1		Eingaenge	DB 1	DB 1	
2		Abfrage	FC 1	FC 1	
3		Ausgang	FC 2	FC 2	
4		EQ_STRNG	FC 10	FC 10	Equal String
5		Takt_10Hz	M 1.0	BOOL	Taktmerker 10Hz
6		Takt_5Hz	M 1.1	BOOL	Taktmerker 5Hz
7		Takt_2_5Hz	M 1.2	BOOL	Taktmerker 2,5Hz
8		Takt_2Hz	M 1.3	BOOL	Taktmerker 2Hz
9		Takt_1_25Hz	M 1.4	BOOL	Taktmerker 1,25Hz
10		Takt_1Hz	M 1.5	BOOL	Taktmerker 1Hz
11		Takt_0_625Hz	M 1.6	BOOL	Taktmerker 0,625 Hz
12		Takt_0_5Hz	M 1.7	BOOL	Taktmerker 0,5Hz
13		PFlanke_Takt_1Hz	M 2.0	BOOL	
14		UDT_Eingang	UDT 1	UDT 1	
15					

Drücken Sie F1, um Hilfe zu erhalten.

NUM

SIMATIC Manager - [Schulung -- g:\Schulung]

Datei Bearbeiten Einfügen Zielsystem Ansicht Extras Fenster Hilfe

< Kein Filter >

- Schulung
  - SIMATIC 300(1) - leer
    - CPU 314IFM
      - S7-Programm(1)
        - Quellen
        - Bausteine
  - SIMATIC 300(1) - projekt
    - CPU 314IFM
      - S7-Programm(1)
        - Quellen
        - Bausteine

Objektname	Symbolischer Name	Erstellsprache	Größe im Arbeitsspei...	Typ	Version (Header)	Name (Header)	Unlinked	Autor	Nc
Systemdaten	---	---	---	SDB	---	---	---	---	---
OB1		SCL	100	Organisationsbaustein	1.0	Main	---	Sc	---
FC1	Abfrage	SCL	1112	Funktion	1.0	Abfrage	---	Sc	---
FC2	Ausgang	SCL	1152	Funktion	1.0	Ausgang	---	Sc	---
FC10	EQ_STRNG	AWL	152	Funktion	1.1	EQ_STRNG	---	SIMATIC	---
DB1	Eingaenge	DB	194	Datenbaustein	0.0		---		---
UDT1	UDT_Eingang	AWL	---	Datentyp	0.0		---		---

Drücken Sie F1, um Hilfe zu erhalten.

TCP/IP -> 3Com EtherLink XL 10/1...