Technische Dokumentation

FMTool
und
Funktionsmodul

Handbuch

mit Einführung in die Programmierung

Vorwort

Das Konzept der Speicher-programmierbaren Steuerungen hat sich in der Automatisationstechnik seit der günstigen Verfügbarkeit leistungsfähiger Mikrocomputer und Mikrocontroller sehr schnell durchgesetzt. Hauptursache für diesen grossen Erfolg ist sicher die einfache Handhabbarkeit und die leichte Programmierbarkeit dieser Geräte. Spezielle, auf die Bedürfnisse des Anwenders in der Mess- Steuer- und Regelungstechnik ausgerichtete Programmiersprachen wie Anweisungslistenprogrammierung (AWL), Kontaktplanprogrammierung (KOP) oder Funktionsplanprogrammierung (FUPLA) wurden entwickelt und haben entscheidend zur schnellen Akzeptanz der Automatisierungssysteme in der Industrie beigetragen. Charakteristisch für die meisten solcher Systeme ist die Konzentration der Rechenleistung in einem zentralen System. Erst mit der Einführung des Buskonzeptes wird eine sinnvolle Verteilung der Intelligenz und ihre direkte Implementierung in den Peripheriegeräten möglich. Kosten, verfügbare Rechenleistung und Speicherkapazität sowie Energierestriktionen verunmöglichen häufig die Realisation komplexer, leistungsfähiger Steuer- und Regelprogramme in den peripheren Feldgeräten (Busteilnehmern) selbst.

Mit ihrem Funktionsmodul stellt die Firma Merten nun eine leistungsfähige programmierbare Steuereinheit für das EIB-System zur Verfügung. Mit dieser Steuerung ist es nun möglich, die oben genannten Restriktionen auch bei diesem Bussystem auf kostengünstige Weise zu sprengen.

Das vorliegende Handbuch behandelt im Detail die Programmierung des Funktionsmoduls mit Hilfe der eigens dazu entwickelten Funktionsplan-orientierten Programmiersprache FMTool. Mit ihr ist die Programmierung selbst komplexer, umfangreicher Aufgabenstellungen einfach und ausschliesslich grafisch möglich. Weiter sind im Handbuch auch die erforderlichen Informationen zum Gerät selbst enthalten.

Die Unterlagen richten sich an Planer, Installateure und andere, die das Funktionsmodul einsetzen und mit diesem Programmierwerkzeug die gewünschten Funktionen des Gerätes programmieren.

Detaillierte Fachkenntnisse des EIB-Systems werden zum Verständnis und für einen erfolgreichen Einsatz des Gerätes vorausgesetzt. Zum Programmieren mit FMTool sind zudem Kenntnisse im Umgang mit einem Personal Computer (PC) mit Windows erforderlich.

Wir wünschen Ihnen interessante Aufgabenstellungen und viel Erfolg bei der Anwendung des Funktionsmoduls.

Hinweise

Die in diesem Unterlagen enthaltenen Angaben, Daten, Werte usw. können ohne vorherige Ankündigung geändert werden. Ebenso sind die Abbildungen unverbindlich.

Ohne ausdrückliche schriftliche Erlaubnis von Merten darf kein Teil dieser Unterlagen für irgendwelche Zwecke vervielfältigt oder übertragen werden, unabhängig davon, auf welche Art und Weise und mit welchen Mitteln, elektronisch oder mechanisch, dies geschieht.

Alle im Handbuch verwendeten Produktbezeichnungen sind eingetragene Warenzeichen der jeweiligen Firmen.

Gebrüder Merten GmbH & Co. KG
Elektrotechnik • Elektronik
Fritz-Kotz-Str. 8
D-51674 Wiehl

Telefon 0 22 61 / 702-01
Telefax 0 22 61 / 702-284

1. Inhaltsverzeichnis

2. Installation von FMTool

2.1. Inhalt des Programmpaketes

Das Programmpaket FMTool umfasst:

- ein Handbuch
- zwei 3½" (1.44 MB) Installationsdisketten FMTool (Lizenzversion)
- einen Kopierschutzstecker (Software-Schlüssel, DB25-Steckermodul)
- einen Software-Lizenz und Nutzungsvertrag
- zwei 3½" (1.44 MB) Installationsdisketten FMDemo
- eine 3½" (1.44 MB) mit den Projektbeispielen des ‘Beispielkatalog für die Programmierung des Funktionsmoduls’

2.2. Hardware-Voraussetzungen

Der Personal Computer muss folgende Voraussetzungen erfüllen:

- IBM-PC kompatibler PC oder Laptop mit Prozessor 80386 oder höher
- mindestens 5 MB freie Kapazität auf der Festplatte
- Diskettenlaufwerk 1.44 MB (3,5“)
- eine parallele Schnittstelle
- eine serielle Schnittstelle und passendes Schnittstellenkabel
- VGA Grafikkarte 640/480 oder höher
- Maus, Trackball oder Trackpoint von Microsoft Windows unterstützt
- Drucker von Microsoft Windows unterstützt

2.3. Software-Voraussetzungen

- Betriebssystem MS-DOS 3.3 (oder IBM PC-DOS) oder höher und grafische Oberfläche Microsoft Windows 3.1 oder 3.11

oder

- Betriebssystem Microsoft Windows 95

Diese Systeme müssen ordnungsgemäss installiert worden sein, damit ein enwandfreier Betrieb der Tool-Software möglich ist.

- FMTool lässt sich, mit funktionalen Einschränkungen (kein Zugriff auf den Instabus EIB), grundsätzlich auch unter Windows NT 4.0 installieren und betreiben. Bitte wenden Sie sich für weitere Informationen an den Technischen Vertrieb.

2.4. Ausführen des Installationsprogramms

Nach dem Start des Rechners bzw. von Windows die Diskette ‘FMTool,’ Diskette 1 / 2 in 3,5“-Diskettenlaufwerk A bzw. B einlegen. Im Programm-Manager das Menu **Datei** wählen und **Ausführen...** anklicken (Windows 3.1 und 3.11) bzw. den **Start**-Knopf und dann das Menü **Ausführen...** Anklicken (Windows 95).

Bild 2.4.0.1.: Installation: Datei, Ausführen...

In der Befehlszeile ist **a:setup** oder **b:setup** einzugeben, je nach Bezeichnung des Diskettenlaufwerks. Danach ist der Befehl mit der Eingabetaste zu bestätigen oder mit der Maus die Schaltfläche „OK“ anzuklicken.

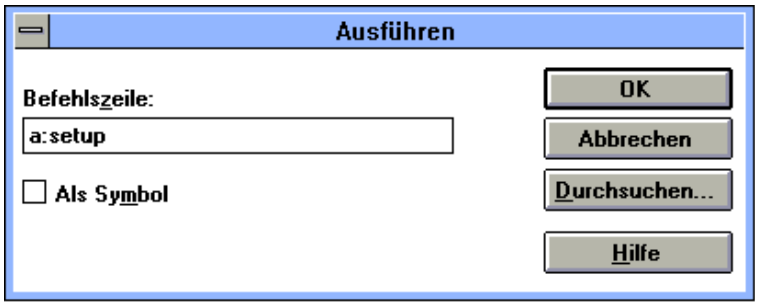


Bild 2.4.0.2.: Installation: Befehlszeile

Als erstes erscheint daraufhin der Dialog mit der Abfrage, ob FMTool in Deutsch oder Englisch installiert werden soll. Wählen Sie die gewünschte Spache an und betätigen anschliessend **OK**.

Bild 2.4.0.3.: Installation: Sprache / Language

Die Installation geht weiter, der Dialog ‘Verzeichnisse’ erscheint. Beim Zielverzeichnis wird als Standard ‘c:\eib\fmtool’ vorgeschlagen, das ggf. durch ein anderes ersetzt werden kann. Anschliessend ist durch anklicken von **Installieren** weiterzufahren.

Bild 2.4.0.4.: Installation: Verzeichnisse

Nachdem alle Dateien installiert wurden, erscheint der Dialog „Die Installation wurde erfolgreich durchgeführt.“. Bestätigen Sie dies mit **OK**. Das Installationsprogramm hat die Programmgruppe ‘FMTools’ neu angelegt und die Icons (Programmsymbole) ‘FMTool’ und ‘FMLoader’ erstellt. Sollte diese Programmgruppe bereits vorhanden sein, werden die beiden Icons zusätzlich zu den schon darin enthaltenen angelegt.

Bild 2.4.0.5.: Programmgruppe ‘Fmtools’

2.5. Auf dem Rechner installierte Software

Nach der Installation befinden sich die Programmdateien im Verzeichnis, das bei der Installation als Zielverzeichnis angegeben wurde (als Standard c:\eib\fmtool) Ein Unterverzeichnis mit dem Namen \LIBRARY wird in diesem Verzeichnis angelegt. LIBRARY enthält sämtliche Funktionsblockbibliotheken (siehe Anhang A) zu FMTool.

Bei allfälligen Aufdatierungen müssen neue Funktionsblockbibliotheken wiederum in diesem Verzeichnis abgespeichert werden. Wird das Update in einem gegenüber der bisherigen Version neuen Verzeichnis installiert, sind die Funktionsblockbibilotheken der bisherigen Version in das Verzeichnis \LIBRARY der neu installierten Version von FMTool zu kopieren. Nur so können die bisherigen Funktionsblockbibilotheken zusammen mit dem neu installierten Update auch weiterhin verwendet werden.

Um die ganze Konfiguration festzuhalten, generiert sich FMTool eine eigene INI-Datei (Datei für die Initialisierung von FMTool) im Systemverzeichnis von Windows (meist C:\WINDOWS\SYSTEM).

2.6. Software-Schlüssel (Dongle)

Vor dem Start von FMTool ist auf der parallelen Schnittstelle des Rechners (LPT1 oder LPT2) der mitgelieferte Software-Schlüssel (Dongle oder auch Kopierschutzstecker genannt) anzubringen. Das Druckerkabel kann hinten am Stecker wieder eingesteckt werden.

Beim Start und während des Betriebs überprüft das Programm, ob der Schlüssel angebracht ist. Andernfalls erscheint der nachstehende Dialog auf dem Bildschirm:

Bild 2.6.0.1.: Dialog bei fehlendem Software-Schlüssel

Achtung: Beim Anschluss gewisser Drucker am Software-Schlüssel kann dieser nur erkannt werden, wenn der Drucker eingeschaltet ist. Andernfalls erscheint derselbe Dialog wie oben.

3. Das EIB-System

3.1. Allgemeine Informationen zu EIB

EIB steht für Europäischer Installations-Bus. Die Technologie des EIB ist eine bei CENELEC TC105 standardisierte Übertragungs-Technologie. Sie wurde speziell für die Bedürfnisse in der Hausinstallationstechnik entwickelt. Dieses System ist ein dezentrales, ereignisgesteuertes Bussystem mit serieller Datenübertragung zum steuern, überwachen und melden von Betriebsfunktionen (Beleuchtung, Jalousien, Rolladen, Heizung usw. im Bereich von Gebäuden).

Der 'Bus' ist dabei das Übertragungsmedium (beim EIB ein verdrilltes Adernpaar), das jedem Apparat (Busteilnehmer) zu jedem Zeitpunkt und unabhängig von seinem Ort die gleichen Informationen zur Verfügung stellt (logischer Aspekt des Busses). Die Apparate besitzen eine einheitliche Schnittstelle zum Bus. Somit können sie über den Bus untereinander kommunizieren.

Sensoren, z.B. Taster nehmen die Informationen von aussen auf und senden sie als Datentelegramme auf den Bus. Aktoren nehmen diese Telegramme auf und setzen sie um, z.B. Schalten von Licht. Systemgeräte wie die Spannungsversorgungen sorgen für die Grundfunktionen. Weitere Funktionen werden durch Steuergeräte, Anzeigen usw. übernommen.

Die Projektierung und Inbetriebnahme einer Instabus-EIB-Anlage erfolgt mit einem PC und der ETS (EIBA-Tool Software). Die ETS greift auf die Produktdatenbanken der Gerätehersteller zurück. Diese enthalten die herstellerspezifischen Informationen bzw. die Anwenderprogramme zu den jeweiligen Instabus-EIB-Geräten. Die ETS ist ein Produkt der EIBA.

EIBA
Um EIB als einheitlichen Standard einzuführen und gezielt zu fördern, haben sich führende europäische Hersteller der Elektroinstallationstechnik zur „European Installation Bus Association“ (EIBA) zusammengeschlossen. Die EIBA ist eine Gesellschaft nach belgischem Recht mit Sitz in Brüssel.

Die Mitgliedsfirmen entwickeln die EIB Busgeräte nach den Richtlinien der EIBA, welche den Standard sicherstellt und die Geräte zertifiziert. Dadurch wird gewährleistet, dass EIB Geräte verschiedener Hersteller miteinander kommunizieren können, d.h., untereinander kompatibel sind.

Beispiel : Bussystem

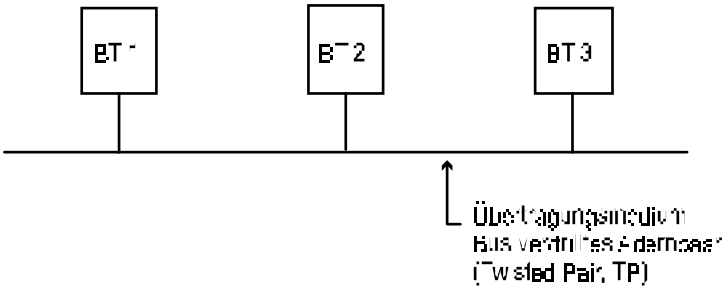


Bild 3.1.0.1.: Bussystem mit drei Teilnehmern

3.1.1. Physikalische Grundstruktur des EIB

Die gesamte Busstruktur kann beim EIB-System mittels Kopplungselementen (Linien- und Bereichskoppler) auch wesentlich komplexer gestaltet und physisch zu einer Baumstruktur erweitert werden. Die Baumstruktur ist physikalisch gesehen keine Busstruktur mehr. Trotzdem hören alle Teilnehmer zu jedem Zeitpunkt an jedem Ort das gleiche. Damit ist auch diese Busstruktur logisch noch ein Bus. Beim Maximalausbau eines EIB-Systems ergeben sich folgende Strukturen mit Grenzwerten.

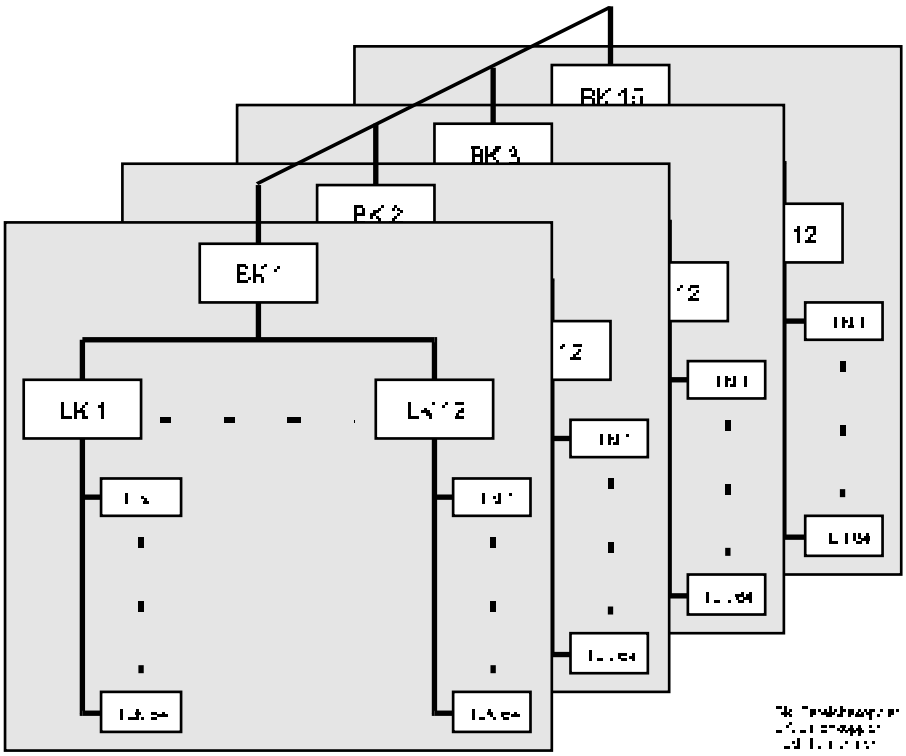


Bild 3.1.1.1.: Maximale Ausbaustruktur Instabus-EIB-Systems

3.1.2. Busteilnehmer

Die Geräte, die am Bus angeschlossen werden, bezeichnet man als Busteilnehmer, z.B. Tastsensoren zum Auslösen eines Vorgangs, Schaltaktoren (Relais, Schützen) zum ausführen des Schaltvorganges, Dimmaktoren, Ventilstellantriebe usw. Beim EIB-System können diese Busteilnehmer sowohl ausschliesslich über den Bus selbst mit Energie versorgt werden (z.B. Tastsensoren, Bewegungsmelder usw.), als auch über eine separate Spannungsversorgung, z.B. Schaltaktoren mit 230 V AC Versorgung verfügen.

Jeder Busteilnehmer besteht im wesentlichen aus einem Busankoppler (z.B. Unterputz-Busankoppler zu einem Tastsensor) und einem Applikationsmodul (z.B. Tastereinheit), das die eigentliche Aufgabe löst. Das Applikationsmodul wird über die sogenannte Anwenderschnittstelle (AST oder PEI = physical external interface) an den Busankoppler angeschlossen.

Der Busankoppler ist die (einheitliche) Schnittstelle der Module zum Bus. Damit werden die Applikationsmodule über den Bus mit anderen Applikationsmodulen kommunikationsfähig.

Busankoppler und Applikationsmodul können auch in einem einzigen Gerät zusammengefasst sein, z.B. REG-Aktor.

3.1.3. Physikalische Adresse

Jedem Busteilnehmer wird eine physikalische Adresse zugewiesen. Diese Adresse wird eineindeutig vergeben. Durch diese Adresse kann damit genau ein Gerät bzw. Busankoppler angesprochen werden. Im gesamten System darf eine physikalische Adresse nur einmal zugewiesen werden. Diese physikalische Adresse wird bei Serviceaufgaben verwendet, um beispielsweise den so adressierten Busankoppler und / oder das daran angeschlossene Applikationsmodul zu programmieren.

Am Gerät bzw. auf dem Busankoppler befindet sich eine Programmiertaste. Ein Druck auf diese Taste versetzt den Busankoppler in den Programmiermodus. Eine leuchtende LED markiert den aktivierten Programmiermodus. Über das EIB-System kann anschliessend die physikalische Adresse diesem Busankoppler zugewiesen werden.

3.1.4. Gruppenadressen

Gruppenadressen stellen die logischen Verbindungen zwischen einzelnen Busteilnehmern her. Sie werden verwendet, um ganze Gruppen von Busteilnehmern gleichzeitig anzusprechen. Im Gegensatz zu den physikalischen Adressen sind die Gruppenadressen nicht mehr eineindeutig, sondern nur noch eindeutig, d.h., eine Gruppenadresse darf im gleichen EIB-System mehrfach verwendet werden.

3.1.5. Bustelegramme

Die Information zwischen Busteilnehmern wird durch Telegramme übertragen. Ein Telegramm ist in verschiedene Felder strukturiert. Der Inhalt unterschiedlicher Felder hat unterschiedliche Bedeutung.

3.2. Vertiefte Informationen im Zusammenhang mit dem Einsatz des Funktionsmoduls

3.2.1. Telegrammaufbau

Die Struktur eines EIB-Telegramms ist wie folgt festgelegt:

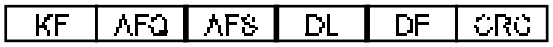


Bild 3.2.1.1.: Telegrammstruktur

KF : Im Kontrollfeld wird neben anderen Informationen die Priorität und die Art des Telegramms festgelegt.

AFQ : Das Adressfeld der Quelle ist 16 Bit lang. Es bestimmt den Absender des Telegramms und enthält immer eine physikalische Adresse.

AFS : Im Adressfeld der Senke steht die Empfängeradresse. Sie ist ebenfalls 16 Bit lang. Sie bestimmt den oder die Empfänger des Telegramms. Es kann sowohl eine physikalische Adresse (für Servicezwecke) wie auch eine Gruppenadresse (im Normalbetrieb) enthalten.

DL : Im Datenlängenfeld wird als Hauptinformation die Telegrammlänge bekannt gegeben. Daneben wird mit einen weiteren Bit zwischen physikalischer Adressierung und Gruppenadressierung im AFS Feld unterschieden.

DF : Im Datenfeld werden die eigentlichen Nutzdaten (zum Beispiel Steuer- oder Statusinformationen) übertragen. Die Datendarstellung innerhalb dieses Feldes ist standardisiert (EIS Standard: EIBA Interworking Standard). Diese Standardisierung ermöglicht die eindeutige Interpretation dieser übermittelten Information. Sie stellt das korrekte Zusammenarbeiten von Apparaten (Busteilnehmern) unterschiedlicher Lieferanten sicher.

CRC : Die Information in diesem Feld ermöglicht die Überprüfung der Korrektheit des Telegramms (CRC: Cyclic redundancy check). Im Fehlerfall fordert das Empfangsgerät eine Wiederholung des Telegramms an (Empfangsgerät sendet NAK = Not Acknowlege).

3.2.2. Gruppentelegramm-Typen

Value-Write	Das Value-Write-Telegramm ist das meist verwendete Gruppentelegramm. Mit ihm werden Werte über den Bus gesendet, die für alle Geräte mit einem Objekt, das die entsprechende Gruppenadresse hat, bestimmt sind.
Value-Read	Will ein Gerät den Objektwert von anderen Gerätes abfragen, kann es ein Value-Read-Telegramm mit der entsprechenden Gruppenadresse senden. Die Geräte, die das Read-Flag gesetzt haben, antworten mit einem Value-Response Telegramm. Das Value-Read-Telegramm enthält nur die Gruppenadresse, aber keine Daten. Es macht nur Sinn, Objekte abzufragen, bei denen nur der Zustand wichtig ist (1. Gruppe von oben). Bei den anderen Objekten könnte das Abfragen eine ungewollte Funktion auslösen.
Value-Response	Das Value-Response-Telegramm hat die gleiche Wirkung wie Value-Write-Telegramm. Es wird also von allen Geräten empfangen, welche die entsprechende Gruppenadresse empfangen können. Dieses Telegramm wird nur als Antwort auf ein Value-Read-Telegram gesendet.

3.2.3. Datentypen bzw. EIS-Standards

Die nachfolgende Tabelle zeigt die Funktionen definierter EIS (EIBA Interworking Standards):

EIS Nr.	EIB - Funktion	Beschreibung, Anwendung
EIS 1	switching	Schaltfunktionen : ein / aus für Licht, Boiler usw.
EIS 2	dimming	Dimmerfunktion : ein / aus, heller, dunkler, setzen auf festen Wert
EIS 3	time	Sekunden, Minuten, Stunden, Wochentag
EIS 4	date	Tag, Monat, Jahr
EIS 5	value	Fliesskomazahl für Werte wie Temperatur, Helligkeit, Durchflussmenge usw. (Bereich : ca. ± 670'000.00)
EIS 6	scaling	Wert zwischen 0% und 100% für diverse Anwendungen
EIS 7	drive control	Motorsteuerungen : Halt, Auf, Ab, Schritt z.B. für Storen

EIS 8	priority	Prioritätsfunktion
EIS 9	float value	Fliesskomazahl für sämtliche physikalischen Grössen (Auflösung nach IEEE 754)
EIS 10	counter value	16 / 32 Bit , signed / unsigned Integer für Zählerwerte

3.2.4. Objekte

Objekte sind die Träger von Werten in einem Gerät. Die Objekt-Werte können über den Bus übermittelt werden. Von der Funktion her kann zwischen zwei Gruppen von Objekten unterschieden werden.

1. Objekte bei denen nur eine Änderung des Wertes eine Funktion auslöst:
   Bei diesen Objekten ist nur der Zustand wichtig.
   - Schalten: EIS1, EIS2-Position
   - Dimmen: EIS2-Value (Absolutwert)
   - Zeit: EIS3
   - Datum: EIS4
   - Werte: EIS5 (Value)
   - Skalierung: EIS6 (8 Bit) (0..100%)
   - Priorität: EIS8
   - Werte: EIS9 (Fliesskommazahl)
   - Werte: EIS10 (16 oder 32 Bit)
2. Objekte bei denen auch nur das Empfangen eines Telegrammes eine Funktion auslöst:
   Bei diesen Objekten hat sowohl der Wert, als auch das Empfangen eines Telegrammes Einfluss auf die Funktion, z.B. schrittweises Herauffahren einer Jalousie wird durch mehrmaliges Senden eines 0-Telegrammes ausgelöst.
   - Sicherheitsfunktionen: EIS1 (z.B. Lebenszeichen)
   - Dimmen: EIS2-Control
   - Bewegen: EIS7 (Jalousien)

3.2.5. Kommunikationsflags von Objekten

Im ETS können für jedes Objekt vier Flags gesetzt werden.

Communication (Kommunikation)	Mit diesem Flag kann eingestellt werden, ob das Objekt überhaupt mit dem Bus kommunizieren kann oder nicht. In welche Richtung die Kommunikation freigegeben ist, wird mit den anderen Flags eingestellt. Dieses Flag ist meistens eingeschaltet. Es kann ausgeschaltet werden, wenn die Funktion des Objektes gesperrt werden soll. gelöscht = keine Kommunikation möglich gesetzt = Kommunikation möglich
Read (Lesen)	Dieses Flag stellt ein, ob das Objekt gelesen werden kann. Ist es gesetzt, wird auf ein Value-Read-Telegramm (Anfrage-Telegramm) ein Value-Response-Telegramm (Antwort-Telegramm) gesendet. Dieses Flag soll meistens gelöscht sein. In einem Bussystem soll pro Gruppenadresse immer nur ein Gerät das Read-Flag gesetzt haben (typischerweise der Aktor, der den Zustand zurückmeldet). Durch das Antwort-Telegramm werden die Objekt-Werte aller Geräte neu gesetzt gelöscht = kein Lesen möglich gesetzt = Lesen möglich
Write (Schreiben)	Mit diesem Flag wird eingestellt, ob das Objekt vom Bus beschrieben werden kann. Ist dieses Flag gesetzt, empfängt das Objekt ein Value-Write- oder ein Value-Restponse-Telegramm. Dieses Flag soll meistens gesetzt sein. Bei Aktoren muss es gesetzt sein und bei Sensoren auch, damit bei mehreren Sensoren alle den aktuellen Wert empfangen. gelöscht = kein Schreiben möglich gesetzt = Schreiben möglich
Transmit (Übertragen)	Dieses Flag stellt ein, ob ein Objekt auf den Bus gesendet werden kann. Ist es gesetzt, kann ein Value-Write-Telegramm gesendet werden. Dieses Flag muss bei Sensoren gesetzt sein und bei Aktoren soll es gelöscht sein. Auch wenn dieses Flag gelöscht ist, kann bei gesetztem Read-Flag ein Value-Response-Telegramm gesendet werden. gelöscht = kein Senden möglich gesetzt = Senden möglich

3.2.6. Mehrere Gruppenadressen auf ein Objekt

Werden auf ein Objekt mehrere Gruppenadressen vergeben, wird immer nur die erste Gruppenadresse gesendet. Die anderen Gruppenadressen werden nur empfangen.

3.3. Realisierung von Funktionen im EIB-System

Die Intelligenz des Systems (Programme des Systems) ist beim EIB-System konzeptionell auf die einzelnen Busteilnehmer verteilt. Sie befindet sich bei einfachen Modulen im Busankoppler, bei komplexeren Geräten sowohl im Busankoppler als auch im Applikationsmodul. Mit der Applikation wird die Buskommunikation als auch die Funktionalität des Applikationsmodules verwirklicht. Rechenkapazität und Speicherkapazität der Busankoppler sind äusserst limitiert. Die Möglichkeiten eines Bussystems bleiben bei dieser Konzeption der vollständigen Dezentralisation beschränkt. Sie können nicht voll ausgenützt werden.

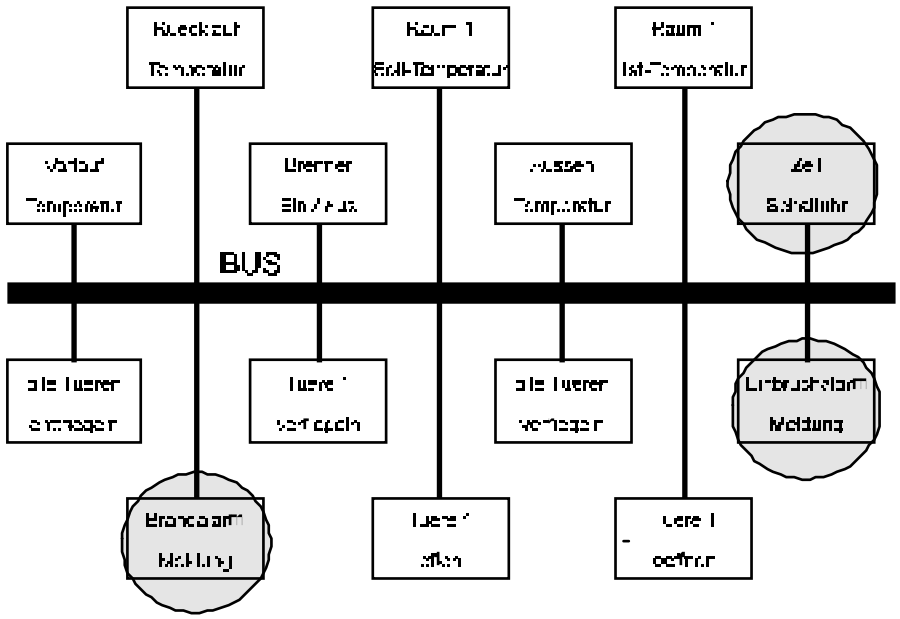


Bild 3.3.0.1.: Vollständig dezentrales Bussystem

Die Funktionalität des einzelnen Busteilnehmers wird durch seinen Hardware-aufbau (Aktor, Sensor, usw.) sowie sein unveränderbares Programm bestimmt. Durch die ETS (EIBA-Tool Software) kann das Programm nur noch parametriert werden. Es wird erst bei der Inbetriebsetzung mittels ETS in den Busteilnehmer geladen (Programm download).

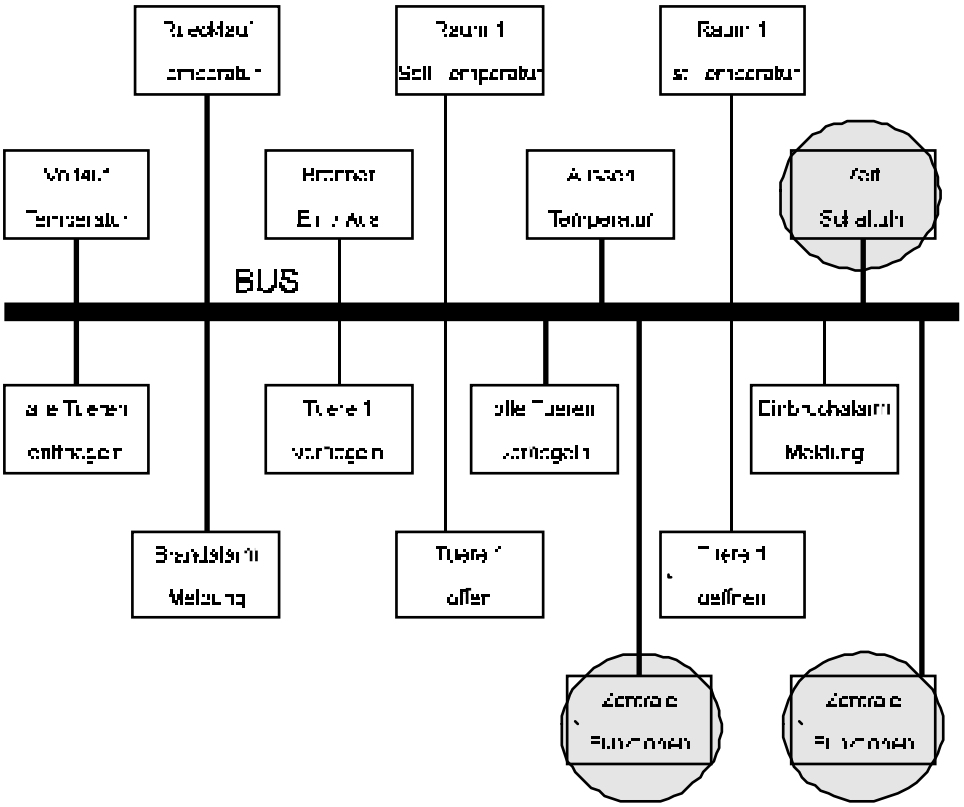


Bild 3.3.0.2.: Ansatz ‘Dezentrales Bussystem kombiniert mit zentralen Funktionen’ (mittels Funktionsmodul)

Bei Verwendung eines Funktionsmoduls kann die Flexibilität und Funktionalität des EIB-Systems wesentlich gesteigert werden. Es stellt massiv mehr Rechenleistung und Speicherkapazität (512 kByte) als ein Busankoppler zur Verfügung. Die verfügbare Rechenleistung entspricht mit 1,1 MIPS der Leistungsfähigkeit eines Standard IBM AT Personal Computers. Konzeptionell entspricht das Funktionsmodul einer speicherprogrammierbaren Steuerung. Es ist grafisch programmierbar mittels der Toolsoftware ‘FMTool’.

Mit dem Funktionsmodul können logische Zentralfunktionen realisiert werden. Sie übernehmen applikationsübergreifende oder applikationsspezifische Funktionen. Die Verwendung von Funktionsmodulen ermöglicht die Trennung von Verantwortlichkeiten verschiedener Anwendungen im gleichen Gebäude. Mit Hilfe des Funktionsmoduls können Funktionalitäten optimiert oder überhaupt erst realisiert werden. Bei entsprechendem Einsatz lassen sich auch die Kosten optimieren.

4. Informationen zum Funktionsmodul

4.1. Aufbau des Geräts

Das Funktionsmodul ist eine frei programmierbare Steuereinheit mit einem Echtzeit-Multitasking-Betriebssystem, speziell für das EIB-System. Die Funktion des Gerätes ist abhängig von der Anwendungs-Software. Die integrierte Funktionsblockbibliothek mit über 60 Funktionsblöcken ermöglicht das Erstellen universeller und bedürfnisgerechter Steuerungen, Regelungen und Sonderfunktionen, welche nur mit einfachen Sensoren und Aktoren nicht realisiert werden können.

Besonderes Gewicht wurde auf Sicherheit, Zuverlässigkeit, Robustheit und Wartungsfreiheit (selbstheilender Überspannungs-und Überstromschutz, Stromausfallsicherheit, kein Batteriewechsel usw.) gelegt. Hardware und Software arbeiten als hartes Echtzeitsystem (feste, beim Programmieren vorgegebene Taskzykluszeiten). Dies garantiert nicht nur die Korrektheit des errechneten Resultates, sondern auch seine rechtzeitige Verfügbarkeit. Bedingt durch die hohe Rechengeschwindigkeit und die grosse Speicherkapazität können umfangreiche und komplexe Steuerungs- und Regelungsaufgaben gelöst werden.

Das Funktionsmodul besteht im wesentlichen aus folgenden Komponenten:

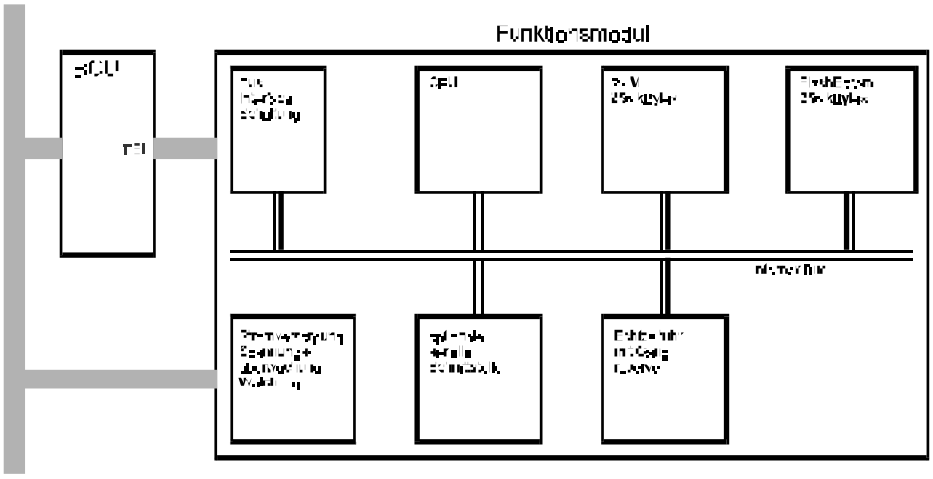


Bild 4.1.0.1.: Aufbau des Funktionsmoduls

4.2. Arbeitsweise

Die ‘Central processing unit’ (CPU) ist die verarbeitende Einheit. Sie wird durch das Programm gesteuert. Es ist im sogenannten Flash-EEPROM (Electrically erasable programmable read only memory) gespeichert. Damit bleibt es auch bei Stromausfall erhalten.

Eine Businterface-Schaltung empfängt die auf dem EIB übermittelten Telegramme als Daten. Diese werden von der CPU übernommen und zeitgerecht im Datenspeicher (RAM) abgelegt. Bei Bedarf werden sie von der CPU aus dem RAM (Random accessable memory) in die CPU gelesen. Dort werden sie interpretiert, logisch miteinander verknüpft oder es werden damit mathematische Operationen ausgeführt. Danach werden die Resultate wieder im RAM abgespeichert.

Zum Übermitteln der Resultate über den EIB werden die Daten von der CPU wieder aus dem RAM gelesen und der Businterfaceschaltung zum Versand bereitgestellt.

Die Arbeitsweise dieses Computers kann in die drei folgenden Hauptphasen eingeteilt werden:
- Input (Einlesen der Daten vom EIB),
- Process (Verarbeiten der vorhandenen Information) und
- Output (Ausgabe der Daten an den EIB).

In ähnlicher Art wie die Businterfaceschaltung wird auch die serielle Schnittstelle bedient. Die CPU bedient so auch die Echtzeituhr. Diese ist für Datum- und Uhrzeitführung verantwortlich.

4.3. Bearbeitungsorganisation und Tasktypen

Die Bearbeitung aller auszuführenden Aufgaben wird durch das Betriebssystem gesteuert. Dies ist eine Software, die im Funktionsmodul schon enthalten ist. Durch das Betriebssystem ist die CPU in der Lage, einzelne Programmteile mit unterschiedlichen Strategien abzuarbeiten.

Einen Programmteil, der unter einer bestimmten Strategie abgearbeitet wird, nennt man **Task**. Im Normalbetrieb wird die Arbeitsreihenfolge streng periodisch mit einer vorgegebenen Zykluszeit durchlaufen. Die kleinste mögliche Zykluszeit beträgt 20ms. Zyklisch abgearbeitete Programmteile sind damit zyklische Tasks.

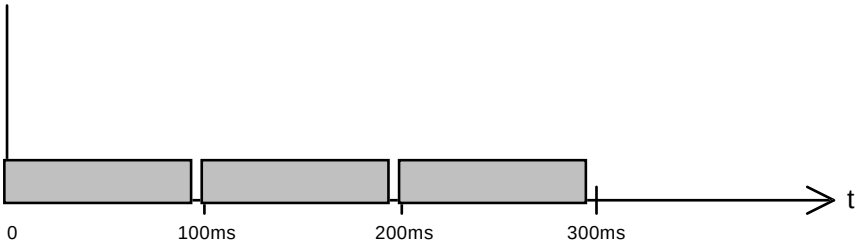


Bild 4.3.0.1.: zyklischer Task mit Taskzykluszeit von 100 ms, Verarbeitungsdauer < 100ms

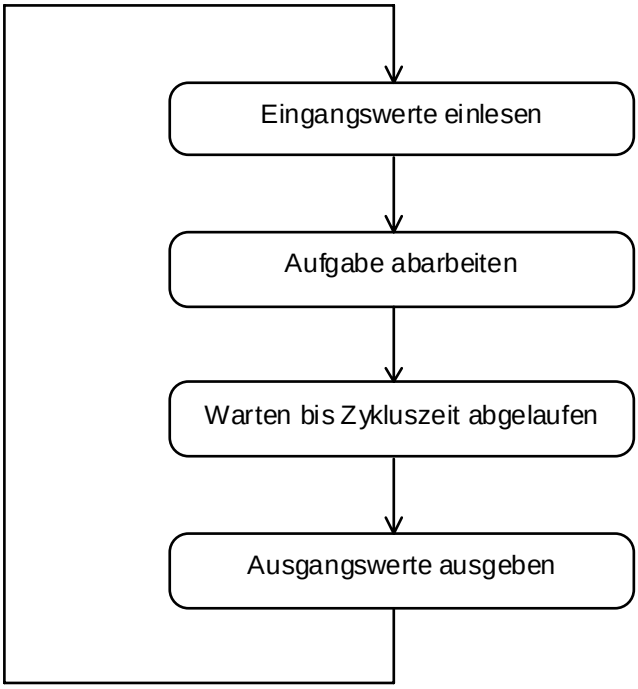


Bild 4.3.0.2.: Hauptschritte der zyklischen Tasks beim Funktionsmodul

Eine andere Abarbeitungsstrategie wird beim Initialisierungstask verwendet. Er wird nur einmal beim Einschalten des Funktionsmoduls aktiv. Damit können Aktionen durchgeführt werden, die ausschliesslich für die Initialisierung notwendig sind, z.B. Eingangswerte für Funktionsblöcke, initialisieren externer Geräte oder anderen Tasks Konstanten als Eingangswerte zuweisen.

Auch bei Betriebsspannungsverlust (Stromausfall) können letztmalig bestimmte Vorgänge durchgeführt werden. Diese Programmteile gehören in den Powerfail-task (Stromausfalltask). Es besteht jedoch keine Gewähr, dass das gesendete Telegramm bei seinem Empfänger ankommt. Wird die Unterspannung zum Beispiel durch einen Kurzschluss auf dem Bus ausgelöst, ist es unmöglich, Telegramme zu senden.

Tasktyp	**Beschreibung**	**Priorität**	**vom Anwender benützbar**	**Aufgabe**
Initialisierungs-task	einmalig aktiv, nur bei Power Up (Start)	höchste	ja, max. 1	Initialisierung von Werten und Geräten
Power Fail Task	einmalig aktiv, nur bei Power Down (Spannungsausfall)	zweithöchste	ja, max. 1	Abspeichern von Werten, Sichern von Zuständen
Zyklische Tasks	periodisch aktiv, Bearbeitung innerhalb Zykluszeit vollständig	Entsprechend den Scheduling-Regeln	ja, mehrere möglich, max. 15	Ausführen von Funktionen / Berechnungen.

4.4. Einsatz mehrer Tasks / Multitasking

4.4.1. Taskzykluszeiten

Zur effizienteren Nutzung der CPU können im Normalfall mehrere zyklische Tasks mit unterschiedlichen Aufgaben und unterschiedlichen Zykluszeiten definiert werden. Eine langsamere Zykluszeit (z.B. 300 ms) beansprucht die CPU weniger als eine schnelle (z.B. 20 ms). Die Zykluszeit jedes langsameren Tasks muss immer in einem ganzzahligen Verhältnis zur Zykluszeit eines jeden schnelleren Tasks stehen.

Beispiel 1:

Task 1 20 ms

$$n_{21} = \frac{80}{20} = 4 \Rightarrow \text{ganzzahlig}$$

Task 2 80 ms

$$n_{32} = \frac{160}{80} = 2 \Rightarrow \text{ganzzahlig}$$

$$n_{31} = \frac{160}{20} = 8 \Rightarrow \text{ganzzahlig}$$

Task 3 160 ms

Da Task 3 zu Task 2 im ganzzahligen Verhältnis (n_{32}) steht und Task 2 zu Task 1 ebenfalls (n_{21}), ist auch das Verhältnis (n_{31}) von Task 3 zu Task 1 ganzzahlig. Damit sind diese Taskbeziehungen gültig.

Beispiel 2:

Task 1 20 ms

$$n_{21} = \frac{60}{20} = 3 \Rightarrow \text{ganzzahlig}$$

Task 2 60 ms

$$n_{32} = \frac{160}{60} = 2.66 \Rightarrow \text{nicht ganzzahlig}$$

$$n_{31} = \frac{160}{20} = 8 \Rightarrow \text{ganzzahlig}$$

Task 3 160 ms

Dadurch, dass Task 3 zu Task 2 nicht in einem ganzzahligen Verhältnis (n_{32}) steht, ist eine solche Taskbeziehung ungültig.

Das Programmierwerkzeug FMTool stellt selbst sicher, dass diese Regel eingehalten wird.

4.4.2. Tasklaufzeiten

Jeder Task benötigt für seine Bearbeitung eine bestimmte Zeit. Sie wird Laufzeit genannt. Die Laufzeit (τ_T) eines Tasks ist durch die Komplexität und seine Grösse bestimmt. Zusammen mit ihrer Taskzykluszeit bestimmen die Tasks die Auslastung der CPU. Das Abschätzen und auch genaue Berechnen der Laufzeiten wird im Kapitel 'Auslastung des Funktionsmoduls, Laufzeiten' erklärt.

Die Taskzykluszeiten sind durch den Programmierer festzulegen. Sie sind beeinflusst durch:
- die maximal zulässige Auslastung der CPU,
- das geforderte Regelungsverhalten für einen Prozess,
- die geforderte, maximal zulässige Reaktionszeit,
- die durch die Busübertragungsrate begrenzte, maximal vom Funktionsmodul auf den EIB absetzbare, Anzahl Telegramme pro Zeiteinheit.

Faustregel:

> Grundsätzlich ist eine Aufgabe zyklisch so schnell (häufig) wie nötig und so langsam (selten) wie möglich auszuführen.

Regel zum Bestimmen des Auslastungsgrades:

$$\sum_{i=1}^{n} \frac{\tau_{Ti}}{T_{Zi}} \overset{!}{<} 1$$

i : Laufvariable
n : Anzahl zyklischer Tasks
T_{Zi} : Zykluszeit des i-ten Task
τ_{Ti} : Laufzeit des i-ten Task

Es wird für jeden Task das Verhältnis seiner Laufzeit zu seiner Zykluszeit bestimmt. Diese Verhältnisse werden addiert. Das Resultat (Summe) muss kleiner sein als 1.

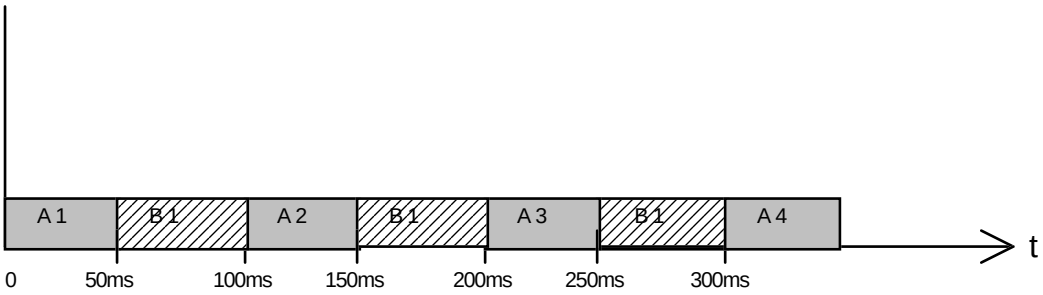
Ist die oben genannte Regel nicht erfüllt, so ist die CPU überlastet und das Funktionsmodul kann die Resultate nicht mehr zeitgerecht berechnen.

Eine ungünstige Wahl der Taskzykluszeiten wird vom Funktionsmodul im Betrieb erkannt und durch die Fehler-LED (unterste LED) signalisiert.

Beispiel 1:

2 Tasks, 100% Auslastung

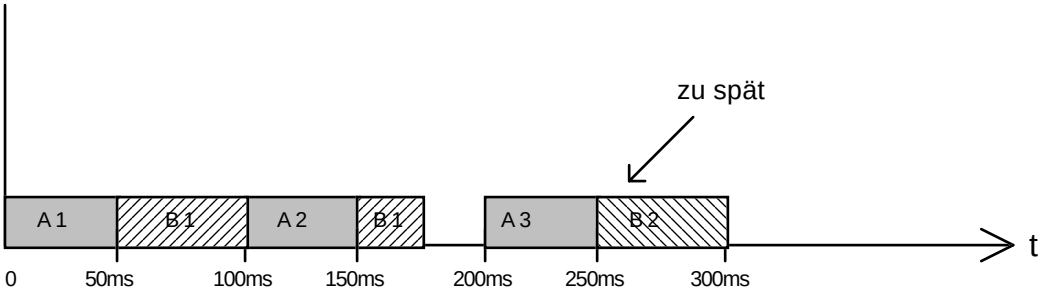
Task A:		Task B:	
Taskzykluszeit	100 ms	Taskzykluszeit	300 ms
Verarbeitungsdauer	50 ms	Verarbeitungsdauer	150 ms



Beispiel 2:

2 Tasks, Überlast

Task A:		Task B:	
Taskzykluszeit	100 ms	Taskzykluszeit	150 ms
Verarbeitungsdauer	50 ms	Verarbeitungsdauer	75 ms



4.4.3. Verfügbarkeit der Daten zwischen den Funktionsplan-Tasks und Datenkonsistenz

Um Datenkonsistenz zu erreichen, muss sichergestellt werden, dass jeweils nur Daten miteinander verarbeitet werden, die zum gleichen Zeitpunkt gültig waren. Beim Funktionsmodul wird dies durch das Betriebssystem (OS) sichergestellt.

Es sorgt dafür, dass jeder Satz von Ausgangssignalen auschliesslich auf einem zeitlich konsistenten Eingangssignalsatz, der zu einem bestimmten Zeitpunkt erhalten wurde, basiert.

4.5. Auslastung des Funktionsmoduls

4.5.1. Applikationsgrösse

Die Grösse einer Applikation ist durch die Grösse des Speicherplatzes im Funktionsmodul beschränkt. Dabei wird zwischen Arbeits- und Programmspeicher unterschieden.

- RAM (Arbeitsspeicher): 65536 Byte
- Flash-EPROM (Programmspeicher): 65536 Byte

4.5.2. RAM-Bedarf

Wieviel RAM eine Applikation benötigt, hängt von der Anzahl Funktionsblöcke und von der Anzahl der externen Signale (Gruppenadressen) ab. Um den RAM-Bedarf einer Applikation zu berechnen, kann mit folgenden Werten gerechnet werden:

- Pro Funktionsblock: 15 Byte
- Pro externem Eingang (Inport-Signal): 22 Byte
- Pro externem Ausgang (Outport-Signal): 44 Byte

Werden viele Funktionsblöcke mit mehr als 6 Ein- und Ausgängen verwendet, steigt der RAM-Bedarf um 2 Byte pro Anschluss.
Der RAM-Bedarf wird wie folgt berechnet:

RAM-Bedarf	=	Anzahl Funtionsblöcke · 15
		+ Anzahl Eingänge · 22
		+ Anzahl Ausgänge · 44

Ist der RAM-Bedarf kleiner als 65536 Byte hat die Applikation im Funktionsmodul Platz.

4.5.3. Programmspeicher-Bedarf

Zum Berechnen wieviel Programmspeicher eine Applikation benötigt, können folgende Werte verwendet werden:

- Pro Funktionsblock: 20 Byte
- Pro externem Eingang: 6 Byte
- Pro externem Ausgang: 10 Byte
- Pro Konstante: 2 Byte (je nach Datentyp auch mehr)

Werden viele Funktionsblöcke mit mehr als 6 Ein- und Ausgängen verwendet, steigt der Programmspeicher-Bedarf um 2 Byte pro Anschluss.

Der EPROM-Bedarf wird wie folgt berechnet:

EPROM-Bedarf =	Anzahl Funtionsblöcke · 20 + Anzahl Eingänge · 6 + Anzahl Ausgänge · 10 + Anzahl Konstanten · 2

Ist der EPROM-Bedarf kleiner als 65536 Byte hat die Applikation im Funktionsmodul Platz.

4.5.4. Kapazität des Funktionsmodules – Beispiele

Aus den obigen Angaben sind Applikationen mit folgender Zusammensetzung möglich:

	Applikation 1	Applikation 2	Applikation 3
Anzahl Funktionsblöcke	2700	2000	1200
Anzahl Eingänge (Gruppenadressen)	200	500	700
Anzahl Ausgänge (Gruppenadressen)	200	500	700

In der Applikation 1 ist das EPROM erschöpft, weil viele Funktionsblöcke vorhanden. In den beiden anderen Applikationen ist das RAM erschöpft, da die Ein- und Ausgänge sehr Speicherintensiv sind.

Anmerkungen:
- Die Anzahl der Segmente und Cluster in einem Projekt hat keinen Einfluss auf die Grösse der Applikation.
- Benötigt die Applikation zuviel RAM oder EPROM, wird der Kompiler eine Fehlermeldung generieren und kein HEX-File erzeugen (siehe Fehlermeldungen des Kompilers, Anhang B).

4.5.5. Laufzeiten

Abschätzen der Laufzeiten:

Das Bearbeiten eines durchschnittlichen Funktionsblockes dauert weniger als ca. 50 µs.

Damit werden 400 Funktionsblöcke in 20 ms abgearbeitet. Diese Anzahl überlastet die CPU noch nicht.

Die Laufzeit eines Tasks (τ_T) berechnet sich aus der Anzahl Funktionsblöcke aller Segmente dieses Tasks, multipliziert mit der durchnittlichen Laufzeit (τ_F) eines Funktionsblockes.

$$\tau_T = n \bullet \tau_F$$

Berechnen der Laufzeiten:

Um die Taskzykluszeiten festzulegen, oder um festzustellen, wieviele Funktionsblöcke in einer bestimmten Taskzykluszeit abgearbeitet werden, können folgende Angaben gemacht werden:

- Die meisten Funktionsblöcke haben eine Verarbeitungszeit von 20µs. Werden nur solche Blöcke verwendet, können also 1000 Blöcke in einem 20ms-Task abgearbeitet werden.
- 'Mittlere' Blöcke (DIV[1)], EXP, INTE[1)], KLICK, MUL[1)], PACD, PACT, SUHR, VERZ, ZUER) haben eine etwas höhere Verarbeitungszeit von 50µs.
- Blöcke, die komplizierte Berechnungen machen, haben eine Verarbeitungszeit von bis zu 600µs (DIV[2)], DT1, FKTG, INTE[2)], LOG, MUL[2)], PI, PT1).

[1)] nur für Datenypen Word und Sint
[2)] nur für Datenyp Value

Wenn bei Blöcken mit variabler Anzahl Ein- und Ausgänge viele Ein- und Ausgänge gewählt sind, kann die Verarbeitungsdauer etwas grösser sein als hier angegeben.

Anmerkung:

Ist die Verarbeitungszeit eines Tasks länger als die Taskzykluszeit, kann dies erst nach dem Laden der Applikation ins Funktionsmodul festgestellt werden. Das Funktionsmodul erzeugt in diesem Falle eine Fehlermeldung mit der Identifikations-Nr. 99.

4.5.6. Telegramm-Durchsatz

Jenachdem, wieviele Daten übermittelt werden, belegt ein Telegramm den EIB für ca. 20 bis 40 ms. Wird nur ein Bit-Signal übertragen, ist der Bus 20 ms belegt, wird das Maximum von 14 Byte übertragen, benötigt dies 40 ms.

Telegramme empfangen
Empfängt die Busankopplung des Funktionsmoduls ein Telegramm, wird dieses unmittelbar an das Funktionsmodul weitergegeben. Es können also maximal 50 Telegramme pro Sekunde empfangen werden.

Das Funktionsmodul speichert zuerst alle empfangenen Gruppentelegramme in einem Empfangspuffer. Vor jedem Taskzyklus der Applikation wird dieser Puffer geleert und die Daten der Telegramme werden den Eingangssignalen zugeordnet.

Ist die Taskzykluszeit der Applikation sehr langsam (Einstellung unter ‘Task erzeugen, Tasks definieren’), kann der Empfangspuffer überlaufen. In diesem Fall gehen Telegramme verloren und das Funktionsmodul erzeugt die Fehlermeldung mit Identifkations-Nr. 0.

Telegramme senden
Will das Funktionsmodul ein Telegramm senden, muss es dieses zuerst an die Busankopplung weiterleiten. Danach sendet diese das Telegramm auf den Bus und gibt anschliessend eine Bestätigung an das Funktionsmodul zurück. Durch diesen Mechanismus dauert es zwischen 55 und 105 ms, bis ein Telegramm gesendet ist (je nach der Grösse der Daten). Es können also nicht mehr als 18 Telegramme pro Sekunde gesendet werden. Werden in der Zwischenzeit noch Telegramme empfangen, wird das Senden zusätzlich verzögert.

Die Telegramme, die von der Applikation gesendet werden (Ausgangssignale), werden zuerst in einem Ausgangspuffer gespeichert. Danach leert das Funktionsmodul diesen Puffer laufend und sendet die Telegramme an die Busankopplung.

Erzeugt die Applikation mehr Telegramme, als auf den Bus gesendet werden können, überläuft der Ausgangspuffer. Die ‘überschüssigen’ Telegramme werden nicht mehr gesendet und es wird die Fehlermeldung mit Identifkations-Nr. 1 generiert.

Der Ausgangspuffer kann 100 Telegramme zwischenspeichern. Die Applikation darf also nicht mehr als 100 Telegramme gleichzeitig erzeugen.

4.6. Funktionsmodul und ETS

Damit ETS die Filtertabellen vollständig erstellen kann, steht für das Funktionsmodul in der EIB Produktdatenbank bzw. im Produktkatalog (ETS2) eine Dummy-Applikationen zur Verfügung.

Die Dummy-Applikation muss nur konfiguriert werden, um die im Funktionsmodul verwendeten Gruppenadressen im ETS bekannt zu machen, damit ETS die Filtertabellen für die Bereichs- und Linienkoppler erstellen kann.

Werden keine Bereichs- oder Linienkoppler verwendet, oder sind die Filtertabellen ausgeschaltet, muss die Dummy-Applikation nicht konfiguriert werden.

Die Dummy-Applikation darf *nicht* in die Busankopplung des Funktions-moduls geladen werden. Dies kann Funktionsstörungen in der Busankopplung des FM zur Folge haben kann.

Die Busankopplung des Gerätes benötigt nur eine physikalische Adresse, damit diese zum Laden des Programms ins FM angesprochen werden kann (Programmierung der physikalischen Adresse mittels ETS).

Wurde die Dummy-Applikation fälschlicherweise geladen und treten dadurch Probleme auf, kann dies durch erneutes Laden einer anderen Applikation behoben werden. Es kann zum Beispiel die Applikation des Schaltaktors geladen werden.

4.7. Hardware des Funktionsmoduls

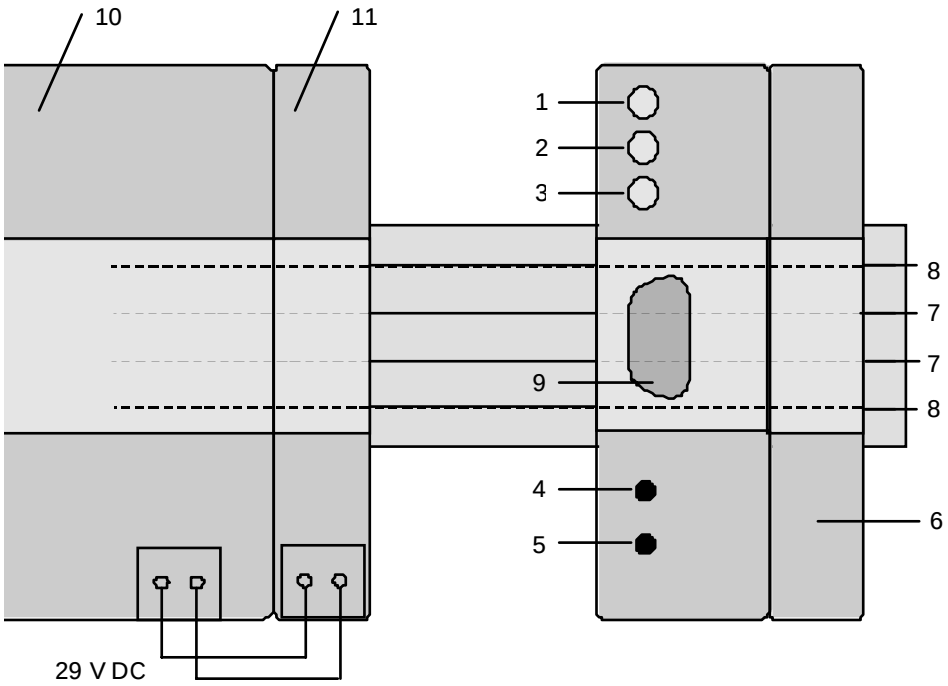


Bild 4.7.0.1.: Funktionsmodul

1 bis 5 siehe Kapitel ‘Bedienungselemente und Funktionsanzeigen’
6 Funktionsmodul mit Busankopplung, fest verbunden
7 Instabus Datenschiene, Busleitungen
8 Instabus Datenschiene, äussere Leiterbahnen zur Spannungsversorgung
9 serielle Schnittstelle
10 EIB Spannungsversorgung, Art.-Nr. 6803 02
11 Verbinder 4-fach, Art.-Nr. 6806 02

4.7.1. Bedienungselemente und Funktionsanzeigen

Das Gerät verfügt über drei LED und zwei Servicetasten (siehe Zeichnung).

1 Service-LED; blinkt, wenn das Betriebssystem läuft (1 x pro Sekunde), blinkt schnell (2 x pro Sekunde), wenn Notladebetrieb (OS-Nukleus) läuft
2 Applikations-LED; leuchtet, wenn ein Applikations-Programm läuft
3 Fehler-LED; leuchtet, wenn das Funktionsmodul einen Fehler festgestellt hat; Fehler können mittels FMLoader abgefragt werden
4 Reset-Knopf; kann mit einem dünnen Gegenstand, z.B. Draht bis 1,5 mm^2 betätigt werden
5 Notladebetriebs-Knopf; wird während einem Reset diese Taste gedrückt gehalten, startet das Funktionsmodul den Notladebetrieb

4.7.2. Montage und Installationshinweise

Das Funktionsmodul als Reiheneinbaugerät wird auf eine Hutprofilschiene 35 x 7,5 mm (nach EN 50 022), in die eine EIB Datenschiene eingeklebt ist, montiert. Die Spannungsversorgung (29 V DC) erfolgt direkt über die äusseren beiden Leiterbahnen der Datenschiene mittels EIB Spannungsversorgung und einen Verbinder 4-fach, siehe Zeichnung.

Planung, Programmierung und Inbetriebnahme erfolgen ausschliesslich mit der Programmiersoftware FMTool für das Funktionsmodul.

4.7.3. Technische Daten

- EIB-Anschluss und Spannungsversorgung über Druckkontakte ab EIB-Datenschiene
- zulässige Betriebsspannung 20...30 V DC
- eigene interne Spannungsaufbereitung
- Stromverbrauch:
  typ. 40 mA im Normalbetrieb @ 29 V DC
  < 100 mA beim Programmieren @ 21 V DC
- Goldkondensator für Gangreserve der internen Echtzeituhr, Pufferung für min. 24 h

Schutzfunktionen
- verpolsicher
- überspannungsgeschützt
- elektronischer Überstromschutz
- Überwachung der 5 V DC

'Power down'-Funktionen
- durch BCU initialisiert
- bei Ausfall der Spannungsversorgung
- Reset

Serielle Schnittstelle
Das Funktionsmodul verfügt über eine 9-polige RS232-Schnittstelle. Diese kann für das Laden der Applikations-Software ins Funktionsmodul sowie zum Auslesen der Fehlermeldungen benutzt werden.

5. Programmieren mit FMTool allgemeine Informationen

Das Funktionsmodul wird mit Hilfe der Tool-Software 'FMTool' (lauffähig auf PC unter Windows) vollständig grafisch programmiert, d.h., mit dem grafischen Programmeditor von FMTool wird das Applikationsprogramm für das Funktionsmodul erstellt. Dieses legt (basierend auf den Vorgaben des Pflichtenhefts) das Verhalten des Gerätes im Betrieb eindeutig fest.

5.1. Teile von FMTool

FMTool besteht aus den Komponenten:

- **Grafikeditor**: Damit werden die Applikations-Programme für das Funktionsmodul grafisch erstellt und dargestellt (siehe Kapitel Segmentaufbau, Funktionsblöcke, Verbindungen usw.).
- **Taskbuilder**: Nach dem Erstellen der Programmteile muss festgelegt werden, wie diese im Funktionsmodul bearbeitet ('abgearbeitet') werden sollen. Mit dem Taskbulider werden die Abarbeitungsstrategien von Tasks festgelegt und die einzelnen Segmente den Tasks zugewiesen.
- **Kompiler**: Dieser übersetzt das grafisch dargestellte Programm damit es durch das Funktionsmodul interpretiert werden kann.
- **FMLoader**: Dieses Service-Programm ermöglicht das Kopieren des übersetzten (kompilierten) Applikations-Programms in den (nichtflüchtigen) Speicher des Funktionsmoduls. Danach ist das zuvor erstellte und kompilierte Programm im Funktionsmodul ausführbar, womit das Gerät die ihm bestimmte Funktion ausführen kann.
- Im Hauptmenu von FMTool, unter 'Projekt',
  'Drucken Projekt' lassen sich als Zusammenstellungen und Listen die Struktur, Signale und Tasks usw. drucken, unter
  'Drucken Segmente...' können die der Segmente grafisch ausgedruckt werden

5.2. Projekthierarchie

Das Applikationsprogramm wird hierarchisch strukturiert. Diese Projekthierarchie spiegelt sich direkt in der Verzeichnisstruktur wider. Die Struktur ist wie folgt organisiert:

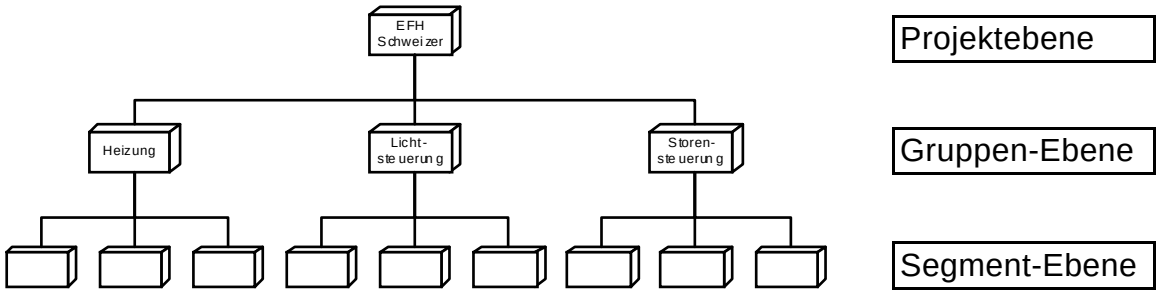


Bild 5.2.0.1.: Projekthierarchie

Ein Projekt umfasst alle, von einem Funktionsmodul auszuführenden Aufgaben. Pro Funktionsmodul besteht genau ein Projekt, d.h., 1 Projekt → 1 FM. Das Projekt ist die oberste Organisationsebene.

Die Gruppen-Ebene ist eine reine Strukturierungsstufe. Diese dient dem Unterteilen von Hauptaufgaben- und / oder -funktionen. Eine Gruppe besteht aus einem oder mehreren Segmenten.

Ausschliesslich auf Segment-Ebene wird programmiert. Die Segmente enthalten die programmierte Logik, die der gewünschten Abarbeitungsstrategie entsprechend einem oder mehreren Tasks zugewiesen wird.

Die Segmente enthalten den Code, d.h., die Instruktionen in Form grafisch dargestellter, miteinander verbundener Funktionsblöcke. Ein Segment hat einen DOS-kompatiblen Namen und wird als Datei im Verzeichnis der übergeordneten Gruppe abgespeichert. Es ist die kleinste Einheit, die einem Task zugewiesen werden kann. Es hat damit eine eigenständige Abarbeitungsstrategie. Die Gesamtheit aller Segmente (und aller Gruppen) ist das Applikationsprogramm.

5.3. Segmentaufbau

Ein Segment ist wie folgt aufgebaut:

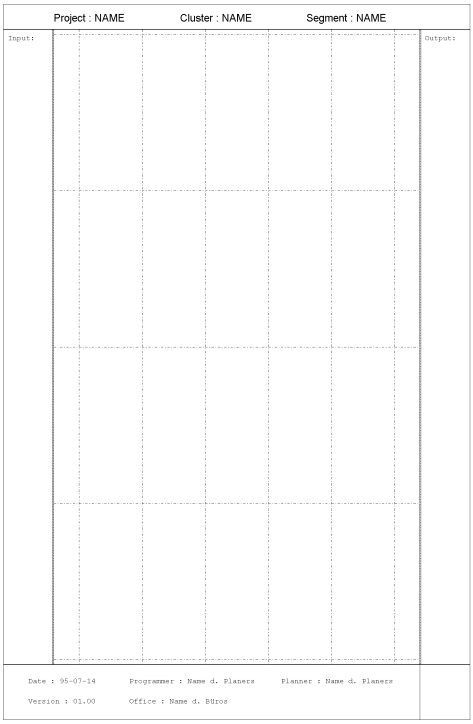


Bild 5.3.0.1.: Segmentaufbau

Das Segment hat das Format einer DIN A4 Seite. Es besteht aus einem Kopfteil, einem Hauptteil und einem Fussteil. Im Kopfteil wird der Projektname, der Gruppenname und der Segmentname bekanntgegeben. Der Fussteil informiert über Programmierer, Planer, Planungsfirma, Erstellungsdatum und Version.

Der Hauptteil besteht aus einem Eingangssignalfeld (Input), einem Funktionsplanfeld und einem Ausgangssignalfeld (Output). Diese Anordnung spiegelt die Strukturierung in Input, Process, Output wieder.

Im Eingangssignalfeld werden die Namen von Signalen oder Werte von Konstanten, die für den Programmteil (Funktionsplan) des Segments benötigt werden, bekanntgegeben.

Im Funktionsplanfeld werden Funktionsblöcke gezeichnet und miteinander verbunden. Damit wird die Verarbeitung der Eingangssignale bzw. die Ausgangssignale festgelegt. Es ist Teil des eigentlichen Programmes. Das Funktionsplanfeld ist mit einem Gitter horizontaler und vertikaler Linien versehen. Auf jeden Kreuzungspunkt dieses Gitters kann ein Funktionsblock (mit seiner linken oberen Ecke) plaziert werden.

Die Plazierung der Funktionsblöcke auf dem Gitternetz bestimmt die Reihenfolge ihrer Bearbeitung (‘Abarbeitung’) bei der Ausführung des Programms im Funktionsmodul.

Es gilt die **Bearbeitungsregel**:

'von oben links nach unten rechts',

d.h., der am meisten oben und am meisten links liegende Funktionsblock wird zuerst ausgeführt. Alle weiteren Funktionsblöcke werden zeilenweise von links nach rechts und von oben nach unten bearbeitet. Der in einem Segment zuletzt aufgerufene Funktionsblock liegt damit auf der tiefst liegenden Zeile am meisten rechts (auf dem Blatt unten rechts).

Im Ausgangssignalfeld werden die Resultate des Programms bzw. der Signalverarbeitung zur Verfügung gestellt. Jedem Resultat wird ein Signalname zugewiesen.

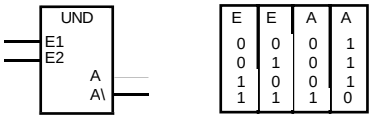
5.4. Funktionsblöcke

Innerhalb eines Segmentes beschreiben die Funktionsblöcke und ihre Verbindungen die Bearbeitung. Ein Funktionsblock selbst entspricht einer nicht teilbaren Sequenz von Instruktionen.
Die Funktionsblockbibliothek umfasst alle diese Funktionsblöcke und ist damit der Instruktionssatz dieser grafischen Programmiersprache. In **Anhang A** zu diesem Handbuch sind alle Blöcke beschrieben.

Jeder Funktionsblock wird grafisch als rechteckiges Symbol mit Anschlusspunkten links und rechts dargestellt. Er hat bei seiner aktuellen Verwendung eine im voraus bestimmte Anzahl Eingänge und Ausgänge. Bei jedem Funktionsblock sind die Eingänge auf der linken Seite und die Ausgänge auf der rechten Seite angeordnet. Beim Bearbeiten des Funktionsblockes werden seine Eingänge (von oben nach unten) gelesen, verarbeitet und die Ergebnisse (von oben nach unten) an seine Ausgänge geschrieben. In zyklischen Tasks wird jeder Funktionsblock des Segmentes pro Zyklus einmal bearbeitet. Innerhalb eines Segments und damit auch Programms kann jeder Funktionsblock beliebig oft verwendet werden.

Beispiel eines einfachen Funktionsblocks ist der UND-Baustein. Seine einfachste Ausführung besitzt zwei Eingänge (E, E) und zwei Ausgänge (A, A\). Als Eingangssignale werden zwei binäre Signale erwartet. Ihr Datentyp (siehe Kapitel Datentypen) ist BIT (EIS 1). Sie werden entsprechend der Regel für die logische UND-Verknüpfung zu den binären Ausgangssignalen A, A\ verknüpft. Bei jeder Bearbeitung des Funktionsblockes werden die Signale an den Eingängen E1, E2

gelesen und die Ausgänge A und A\ entsprechend der Wertetabelle neu zugewiesen.



E	E	A	A
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

Bild 5.4.0.1.: UND-Funktionsblock mit Wertetabelle

Zeitabhängige Funktionsblöcke

Diese sind eine besondere Klasse von Funktionsblöcken. Auch ein zeitabhängiger Funktionsblock ist in einem Segment enthalten.

Dieses Segment muss *zwingend einer zyklischen Task zugeordnet* werden. Zeitabhängige Funktionsblöcke beziehen ihre Zeitangaben auf diese *Taskzykluszeit*.

Beispiele zeitabhängiger Funktionsblöcke sind: Oszillator, monostabile FF, Integratoren, Verzögerung usw.

5.5. Verbindungen

Verbindungen dienen dazu, Informationen vom Eingangssignalfeld zu den Funktionsblockeingängen resp. von den Funktionsblockausgängen zu anderen Funktionsblockeingängen oder dem Ausgangssignalfeld zu führen.

Jeder Verbindung ist exklusiv ein Speicherplatz zur Zwischenspeicherung der durch sie repräsentierten Information zugewiesen. Daten werden Informationen zwischen verbundenen Funktionsblöcken wie folgt ausgetauscht:

Der Wert des Funktionsblockausganges wird in diesen Speicherplatz geschrieben. Alle Funktionsblöcke die mit diesem Ausgang verbunden sind, lesen diesen Speicherplatz. Aus Konsistenzgründen darf nur ein Ausgang auf einen entsprechenden Platz schreiben. Ausgangswerte müssen vor ihrer Verwendung berechnet sein. Daher muss ein Funktionsblock, der einen Ausgangswert generiert vor dem ersten Funktionsblock, der diesen Wert verwendet, bearbeitet werden.

Daraus ergeben sich für mögliche Verbindungen folgende Regeln:

- Mehrere Eingänge dürfen mit einem Ausgang verbunden sein.
- Ein Eingang darf und kann nicht mit mehreren Ausgängen verbunden sein.

Damit fliessen auch alle Signale innerhalb eines Segmentes von oben links nach unten rechts.

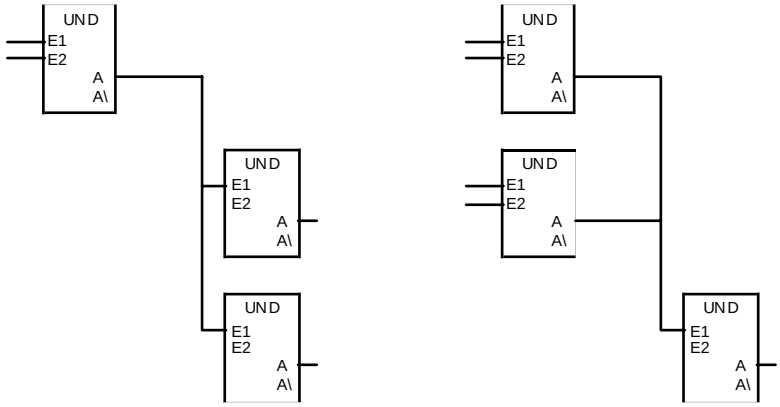


Bild 5.5.0.1.: Mögliche Verbindung und unmögliche Verbindung (rechts)

FMTool verhindert das Zeichnen nicht zulässiger Verbindungen. Das Zeichnen der Verbindung wird durch die Bekanntgabe des Start- und Zielpunktes automatisch ausgeführt. Der Eingang eines Funktionsblockes kann nur mit einem Ausgang verbunden werden, wenn der erwartete Datentyp beim Eingang mit dem generierten Datentyp beim Ausgang übereinstimmt.

5.6. Datentypen

Der sogenannte Datentyp bestimmt, wie die Information einer Verbindung interpretiert wird. Er beschreibt den Wertebereich und die auf diese Information zulässigen, anwendbaren Operationen. Die nachfolgende Tabelle zeigt die implementierten Datentypen und ihre Anwendung auf die EIS-Standards.

Name	Erläuterungen	Wertebereich	Verwendung für
Bit	kleinste Darstellungseinheit, Schalten (Zustände Ein/Aus, 1/0, high/low)	0 / 1	EIS 1, EIS 2a (switch), EIS 7a, EIS 7b, EIS 8a
Byte	8 Bit-Feld, nicht interpretiert oder als vorzeichenlose Ganzzahl; es kann eingestellt werden, wieviele Bits auf den Bus gesendet oder empfangen werden sollen	0 ... 255	EIS 2c (value), EIS 6
Word	2 Byte-Feld, nicht interpretiert oder als vorzeichenlose Ganzzahl	0 ... 65535	EIS 10 (Code 10000)
Ganzzahl (Sint)	16 Bit-Zahl, Ganzzahl mit Vorzeichen (signed integer)	-32768 ... +32767	EIS 10 (Code 10001)
Fliess-komma-zahl (Value)	2 Byte Feld, Darstellung von Fliess- oder auch Gleitkommazahlen genannt	-671088,64... ....670760,96 Auflösung: 0,01	EIS 5
Zeit (Time)	3 Byte-Feld, Zeitdarstellung (24 Stunden und Wochentag)	Sekunden, Minuten, Stunden, Wochentag	EIS 3
Datum (Date)	3 Byte-Feld, Datum	Tag, Monat, Jahr	EIS 4
Priorität (Pcontrol)	2 Bit Feld, nur in EIB-Systemen gebraucht; Priorität, zur Interpretation anderer Felder verwendet	contol / no contol, enable / disable	EIS 8b
Steuerung (Control)	4 Bit Feld, Dimmen relativ, das erste Bit heisst Up/Down, die weitern drei Bit geben die Schrittweite an	Inc /dec , Step: 0...7	EIS 2b (control)
MAX (ASCII)	14 Byte-Feld, es kann eingestellt werden, wieviele Bytes auf den Bus gesandt oder empfangen werden sollen	14 Bytes	EIS 11

Hinweise zum Programmieren im Zusammenhang mit den Datentypen:

- Es können nur Ein- und Ausgänge von Funktionsblöcken verbunden werden, die den gleichen Datentyp haben.
- Datentyp-Umwandlungen können nur mit den speziell dafür bestimmten Blöcken vorgenommen werden (PAC- bzw. TEIL-Blöcke).
- Tip: Durch Klick auf einen Funktionsblock wird dessen Datentyp angezeigt.

5.7. Signale

Die Verbindungen auf einem Segment werden auch Signale genannt. Es wird zwischen vier Gruppen von Signalen unterschieden:

- Konstanten
- lokale Signale
- interne Signale
- externe Signale (Eingangs- und Ausgangsignale, Kontroll-Signale)

Signale sind zeitlich veränderliche Grössen,
Konstanten sind zeitlich unveränderliche Werte.

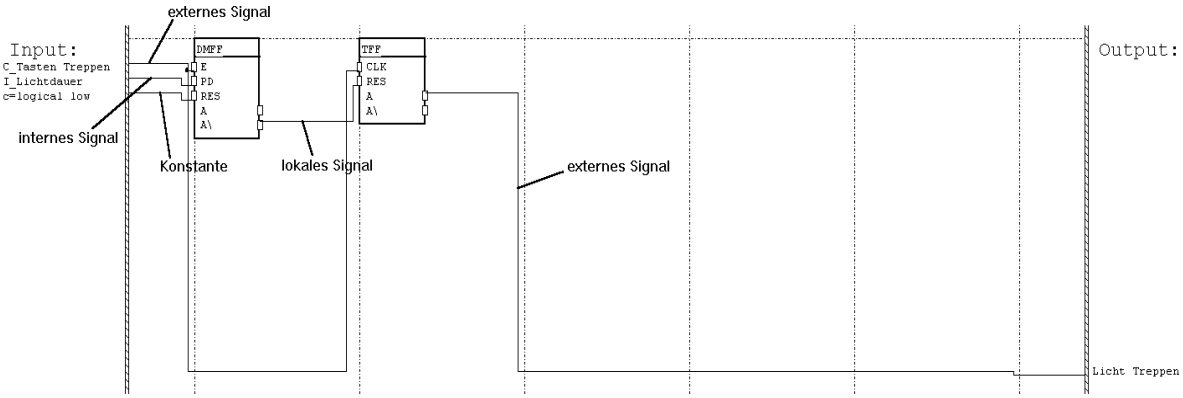


Bild 5.7.0.1.: Signale

5.7.1. Konstanten

Konstanten können nur an die Eingänge der Funktionsblöcke angeschlossen werden. Ihr Wert wird im Eingangssignalfeld angegeben und er ändert sich nie. Konstanten haben keine Namen.

5.7.2. Lokale Signale

Lokale Signale sind die Verbindungen zwischen den Funktionsblöcken. Sie haben keinen Namen und können nicht ausserhalb eines Segmentes verwendet werden. Sie können auch nicht initialisiert werden, da nach einem Reset der Wert des Signals immer durch den Ausgang eines Funktionsblockes bestimmt wird.

5.7.3. Interne Signale

Interne Signale funktionieren gleich wie die lokalen Signale und verbinden Funktionsblöcke miteinander. Doch werden sie zum Eingangs- oder Ausgangssignalfeld gezeichnet und haben einen Namen. Dadurch können sie die Funktionsblöcke über mehrere Segmente hinweg miteinander verbinden. Die internen Signale können auch initialisiert werden, d.h., sie erhalten unmittelbar nach einem Reset einen programmierten Wert. Dieser Wert wird erst überschrieben, wenn derjenige Funktionsblock abgearbeitet wird, welcher den Ausgang mit diesem Signal verbunden hat. Wird das Signal nicht initialisiert, hat es nach einem Reset immer den Wert 0.

Hinweis:
Beim Festlegen der Abarbeitungsreihenfolge der Segmente mit dem Taskbuilder, ist darauf zu achten, dass *zuerst dasjenige Segment abgearbeitet wird, in welchem der Signalwert durch einen Funktionsblock definiert wird*. Sonst rechnen andere Funktionsblöcke mit einem Signalwert, der nicht akuell ist.

Damit in den Segmenten die internen Signale besser von den externen Signalen unterschieden werden können, empfiehlt es sich, sie besonders zu kennzeichnen (z.B. durch anfügen eines ‘I_’ zu Beginn des Namens).

5.7.4. Externe Signale

Externe Signale sind die Verbindungen zum Bus. Ein externes Signal kann entweder ein Eingang (Input) vom Bus oder ein Ausgang (Output) sein. Externe Signale sind mit den Objekten der Busankoppler-Applikationen im ETS zu vergleichen. Sie haben also auch eine Gruppenadresse.

Externe Signale können allerdings **nur eine Gruppenadresse** haben und können **nur** entweder empfangen **oder** senden.

Nach Möglichkeit sollen die Namen der externen Signalen gleich gewählt werden wie die Namen der Gruppenadressen im ETS. So ist das Zusammenspiel zwischen Funktionsmodul und ETS besser ersichtlich.

Eingangssignale sind externe Signale, die ins Eingangssignalfeld führen. Sie empfangen die Telegramme, die von anderen Busteilnehmern abgesandt werden. Der Wert der Eingangssignale bleibt immer erhalten, bis ein neues Telegramm empfangen wurde, das einen neuen Wert hat.
Zu einem Eingangssignal gehört immer auch ein *Kontrollsignal*. Dieses ist vom Datentyp Bit und zeigt mit einem Puls (Dauer ein Taskzyklus) an, wenn ein Telegramm auf diese Gruppenadresse gesendet wurde. Es entspricht somit einem Update-Flag. Bei vielen Anwendungen wird dieses Signal gebraucht, da oft nicht der empfangene Wert wichtig ist, sondern das Vorhandensein eines Telegramms selber. Die Kontrollsignale sind durch ‘C_’ vor dem Signalnamen gekennzeichnet. Die Kontrollsignale werden automatisch erzeugt wenn ein neues externes Signal erzeugt wird.

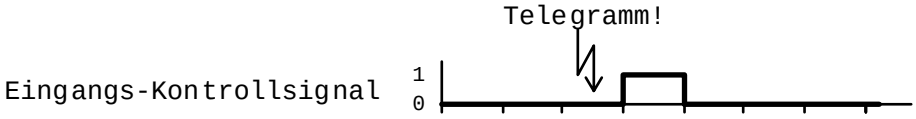


Bild 5.7.4.1.: Eingangs-Kontrollsignal

Eingangssignale können initialisiert werden. Nach einem Reset hat das Signal solange diesen Initialisierungswert, bis ein Telegramm mit einem anderen Wert empfangen wird. Nicht initialisierte Signale haben nach einem Reset immer den Wert 0.
Sowohl das Eingangssignal als auch das Eingangs-Kontrollsignal kann in mehreren Segmenten verwendet werden.

Ausgangsignale sind externe Signale, die ins Ausgangssignalfeld führen. Sie senden Telegramme, welche die anderen Busteilnehmer empfangen können, auf den Bus. Immer wenn sich der Wert des Ausgangssignales ändert, wird ein Telegramm auf den Bus gesandt. Dies kann nur durch Funktionsblockschaltungen verhindert werden.

Auch zu den Ausgangssignalen gehört immer ein *Kontrollsignal*. Dieses kann das *Senden eines Telegrammes erzwingen*. Wird ein Puls auf das Kontrollsignal gegeben, wird ein Telegramm mit dem aktuellen Signalwert gesendet, egal ob sich dieser Wert geändert hat oder nicht, d.h., durch das Setzen eines Ausgangs-Kontrollsignals wird ein Ausgangstelegramm erzwungen. Die Kontrollsignale sind durch ‘C_’ vor dem Signalnamen gekennzeichnet. Die Kontrollsignale werden automatisch erzeugt wenn ein neues externes Signal erzeugt wird.

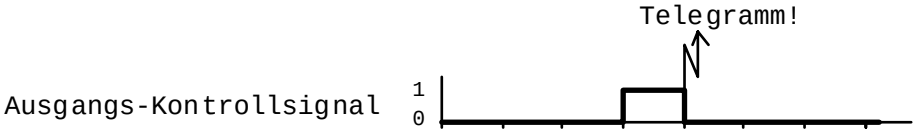


Bild 5.7.4.2.: Ausgangs-Kontrollsignal

Ausgangssignale können initialisiert werden. Werden sie mit einem Wert ungleich 0 initialisiert, wird sofort nach einem Reset ein Telegramm mit dem Initialisierungswert auf den Bus gesendet. Nicht initialisierte Signale haben immer den Wert 0. Auch das Ausgangs-Kontrollsignal kann auf 1 initialisiert werden. Dadurch wird das Telegramm sofort nach dem Reset gesendet. In diesem Fall wird das Kontrollsignal automatisch wieder auf 0 gesetzt, wenn es nicht von einem Funktionsblock wieder gesetzt wird.
Sowohl das Ausgangssignal als auch das Ausgangs-Kontrollsignal kann nur einmal verwendet werden.

Die Namen der internen und externen Signale dürfen nicht gleich sein. Sie unterscheiden sich also immer durch ihren Namen. Es kann auch jede Gruppenadresse nur einem Signal zugewiesen werden.

6. Vorgehen beim Programmieren

Nachstehend wird kurz auf die Methodik des Programmierens mit FMTool eingegangen. Das daran anschliessende Kapitel zeigt Schritt für Schritt, wie ein Projekt erstellt und anschliessend auch überarbeitet bzw. ergänzt wird.

6.1. Analyse des Projektes bzw. der Aufgabenstellung

Das Pflichtenheft muss verstanden sein. Es ist auf Vollständigkeit, Eindeutigkeit und Widerspruchsfreiheit zu überprüfen. Damit wird erkannt, **was** realisiert werden muss.

Das bedeutet:

- Das ganze EIB-System ist im Überblick bekannt.
- Die durch das Funktionsmodul zu lösende Aufgabe (Process) ist im Detail verstanden.
- Welche Informationen werden für die Lösung der Aufgabe benötigt oder / und sind verfügbar (Input)?
  - Von welchen Komponenten werden sie abgesandt?
  - Aufgrund welcher Auslöse-Kriterien werden sie übermittelt?
  - Welche Prioritätsverhältnisse liegen vor?
  - An welche Gruppenadressen werden sie gesandt?
  - In welcher Form werden sie übermittelt?
- Unter welchen Bedingungen und bei welcher Vorgeschichte werden sie übermittelt (Zustands-Transitionen-Modell)?
- Welche Informationen sind als Lösung der Aufgabe zu generieren (Output)?
  - An welche Komponenten sind sie zu senden?
  - Aufgrund welcher Auslöse-Kriterien müssen sie übermittelt werden?
  - Welche Prioritätsverhältnisse sind zu berücksichtigten?
  - Wie lauten die Gruppenadressen, an die sie gesandt werden?
  - In welcher Form ist sie zu übermitteln?
- Unter welchen Bedingungen und bei welcher Vorgeschichte muss sie übermittelt werden (Zustands-Transitionen-Modell)?
- Auslöse-Kriterien sind dabei:
  - periodische Werteübermittlung
  - Übermittlung nur auf spezielle Anfrage
  - Übermittlung bei jeder Änderung des Wertes
- Als Prioritätsverhältnis ist die Dominanz eines Befehles gegenüber eines anderen zu verstehen, z.B. soll eine Automatik gegenüber manueller Steuerung dominant sein oder umgekehrt.

Als Form wird dabei die Datendarstellung im EIB-Telegramm verstanden. In welchem EIS Standard werden die Daten übermittelt (evtl. auch Datentyp).

Als Bedingungen und Vorgeschichte versteht man die Abhängigkeit der Informationsübermittlung von vorangegangenen Ereignissen und aktuell gegebenen anderen Informationen oder Informationswerten (Zustands-Transitionen-Modell).

6.2. Strukturierung der Ausgabenstellung

Unter Aufgaben-Strukturierung wird die funktionale Zerlegung und die Bestimmung der Abarbeitungsstrategien verstanden.
Mit der funktionalen Zerlegung soll die Gesamtaufgabe stufenweise in voneinander möglichst unabhängige, in sich abgeschlossene Teilaufgaben und Teile zerlegt werden. Die Gesamtaufgabe wird so in einfachere, handhabbare Teile aufgegliedert. Teilaufgaben sind dann in sich abgeschlossen und unabhängig, wenn die Anzahl ihrer benötigten und generierten Informationsflüsse minimal wird.

Das Kriterium der minimalen Informationsflüsse führt meist auch in eine Zerlegung nach Teilfunktionen. wie z.B. Heizungssteuerung, Licht, Storen usw. Das stufenweise Zerlegen wird auch ‘Top down design’ genannt.

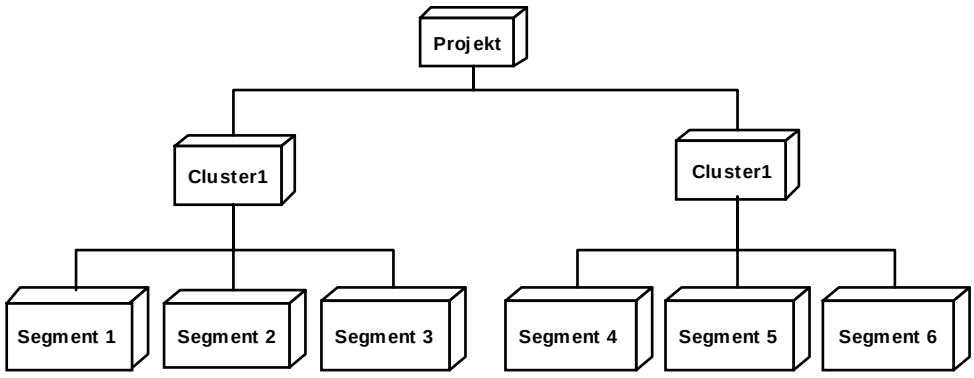


Bild 6.2.0.1.: Projektstruktur

Die Teilfunktionen (Teilaufgaben) werden den Gruppen (englisch Clustern) zugewiesen. Es werden anschliessend die einzelnen Gruppen aufgeteilt und die erhaltenen Teile den Segmenten zugewiesen. So werden die Segmente unter dem funktionalen Aspekt gesehen. Ausschliesslich die Segmente enthalten die auszuführenden Instruktionen. Auch die den Gruppen zugewiesenen Segmente sollen inhaltlich so gewählt werden, dass die Anzahl ihrer Signale über die Segment-grenzen möglichst gering wird.

Segmente sind aber auch diejenigen Teile, die einem Task zugewiesen werden. Sie unterliegen damit einer durch den Task definierten Abarbeitungsstrategie.

Somit müssen Teile von Aufgaben, die mit unterschiedlicher Strategie abgearbeitet werden auch in unterschiedliche Segmente zu liegen kommen (Segmentaspekt der Abarbeitungsstrategie).

Daher sind die Teile für die Initialisierung in ein oder mehrere separate Segmente zu schreiben, ebenso wie die zyklisch zu bearbeitenden Teile oder solche die nur bei Stromausfall ausgeführt werden müssen.
FMTool stellt pro Task eine Segmentliste zur Verfügung. Segmente werden dort entsprechend der Reihenfolge ihrer Bearbeitung eingetragen.

Die Zuweisungsmatrix ermöglicht die übersichtliche Darstellung des abarbeitungstechnischen und des funktionellen Aspektes.

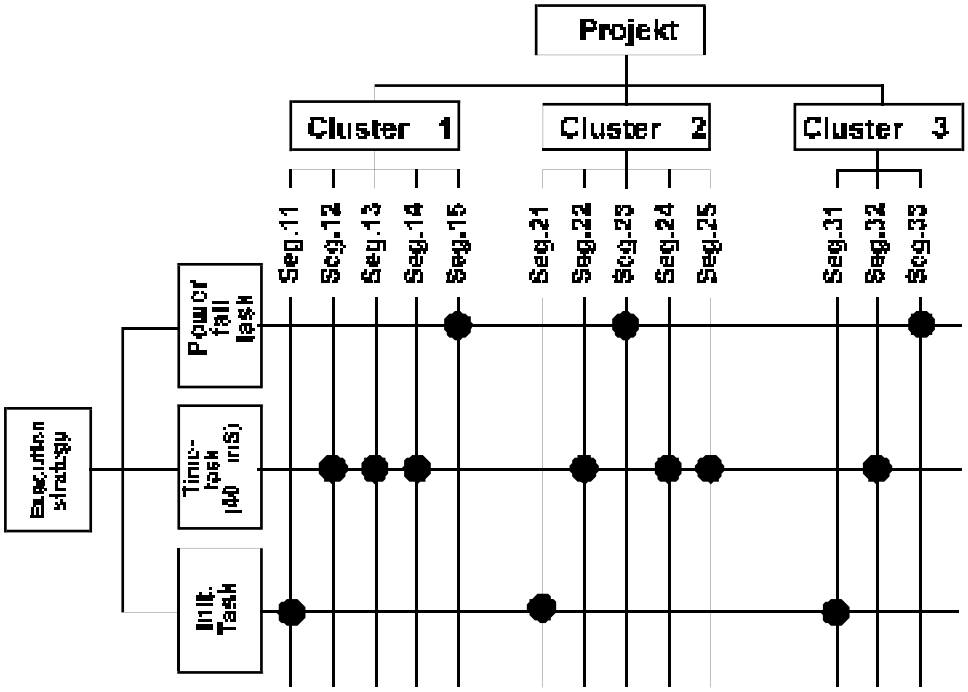


Bild 6.2.0.2.: Zuweisungsmatrix

Ein Punkt markiert die Zuordnung eines Segmentes sowohl zur Gruppe (Funktion) als auch zum Task (Bearbeitungs- oder ‘Abarbeitungsstrategie’).
Ein Segment kann nur einem Task zugewiesen werden.

Nochmals zusammenfassend das schrittweise Vorgehen:
- festlegen des Projektnamens
- Zerlegen des Projektes in unabhängige Hauptteilaufgaben. Dies ist die erste Strukturierungsebene des Projektes.
- zuordnen der Hauptteilaufgaben in Gruppen
- festlegen der Gruppennamen
- Aufteilen der einzelnen Gruppen und zuweisen dieser Teile in Segmente. Dies ist die zweite Strukturierungsebene des Projektes.
- Abarbeitungsstrategie berücksichtigen.
- Festlegen der Segmentnamen

6.3. Synthese der Lösung

Bei der Synthese wird der Ansatz (das Modell), wie eine Teilaufgabe gelöst werden soll, bestimmt (Lösungsidee).

Zur schnellen und effizienten Implementation des Lösungsansatzes sind gute Kenntnisse der Funktionsblockbibliothek (siehe Anhang A) notwendig. Mit Hilfe der verfügbaren Funktionsblöcke aus der Funktionsblockbibliothek wird nun dieser Ansatz realisiert und im Detail aufgezeichnet.

Die Signalnamen und Konstanten werden bestimmt.

Für jedes Segment wird die Abarbeitungsstrategie festgelegt. Gegebenenfalls ist es von Vorteil, wenn die Segmente zuerst vollständig auf Papier entwickelt und gezeichnet werden. Alle Verbindungen und Funktionsblöcke werden gezeichnet.

Danach ist im Detail bekannt, wie die Aufgabe gelöst wird. Die eigentliche Programmieraufgabe ist damit vollständig bearbeitet. Dies ist das Resultat der Synthese.

6.4. Erfassen des Programmes

Aus organisatorischen Gründen ist es durchaus sinnvoll, vor dem Programmieren auf dem PC ein Verzeichnis einzurichten, in welchem alle FMTool-Projekte abgelegt werden.

Beim Eröffnen eines Projektes
- ist der Projektname einzugeben,
- die Version der zu verwendenden Funktionsblockbibliothek festzulegen,
- weitere Angaben sollten ebenfalls gemacht werden (Name des Planers, Name des Planungsbüros).

Als Projektverzeichnis soll das vorhin erwähnte Verzeichnis angwählt werden. Nach dem Eröffnen erstellt FMTool ein eigenes Unterverzeichnis für das neue Projekt.

Entsprechend der Zerlegung des Projektes sind danach die Gruppen ebenfalls zu eröffnen. Darin können die Segmente definiert und mit dem Grafikeditor programmiert werden. Nachdem alle Gruppen und Segmente erfasst sind, kann mit dem Taskbuilder die Abarbeitungsstrategie definiert und die Segmente den Tasks zugewiesen werden.

6.5. Übersetzen des erfassten Programmes (kompilieren)

Das Übersetzen, auch kompilieren genannt, setzt die grafisch erfasste Information automatisch in eine Aufgabenbeschreibung um (Programmcode), die vom Funktionsmodul interpretiert werden kann. Das Übersetzen umfasst hier auch das Linken, welches verschiedene Programmteile (Segmente) miteinander verbindet und gleichnamigen Signalen eine gemeinsame Referenz zuweist.

6.6. Laden des Projektes bzw. des Programms ins Funktionsmodul

Mit Hilfe des FMLoader wird das übersetzte Programm über ein serielles Verbindungskabel vom PC zum Funktionsmodul übermittelt. Dort wird es nicht flüchtig gespeichert, d.h., es bleibt im Funktionsmodul gespeichert, auch wenn dieses von der Spannungsversorgung getrennt wird.

Nach der Rückstellung – dem Reset – des Funktionsmodules startet das Programm und steuert das Funktionsmodul.

6.7. Inbetriebnahme und Programmtest

Die Inbetriebnahme soll systematisch und schrittweise durchgeführt werden, um ein *sicherheitsorientiertes* Vorgehen zu garantieren.

Bei komplexen Programmen sollte jede Gruppe einzeln in Betrieb genommen werden. Dazu werden nur die in der zu testenden Gruppe enthaltenen und benötigten Segmente einem Task zugewiesen, dann übersetzt und anschliessend in das Funktionsmodul geladen.

Der Test des Programmes soll Abweichungen zwischen Vorgaben aus dem Pflichtenheft und der erstellten Lösung in der Realität aufzeigen. Diese sind gegebenenfalls durch Programmänderungen zu korrigieren.

6.8. Dokumentation

Mit Hilfe der im Hauptmenu unter ‘Projekt’ vorhandenen Menüpunkte können alle erfassten Informationsarten in strukturierter und gegliederter Form zur späteren Fehlersuche und Archivierung ausgedruckt werden. Die in den Segmenten grafisch erfassten funktionalen Zusammenhänge lassen sich wieder grafisch gleich ausdrucken.

Eine vollständige Dokumentation besteht aus:

- Titelblatt
- Inhaltsverzeichnis
- Projektstrukturdarstellung
- Signalbeschreibungsliste
- Signallokalisierungsliste
- Taskdefinitionsliste
- Kompilerfehler-Meldungsliste.
- Segmentausdrucken

7. Einführung in die Programmierung mit FMTool anhand eines Beispiels

Im folgenden wird nun Schritt für Schritt gezeigt, wie mit FMTool programmiert wird. Anhand von zwei einfachen Aufgaben werden die wichtigsten Funktionen und Möglichkeiten von FMTool aufgezeigt.

Die komplette Beschreibung aller Funktionen, Dialoge und Menüleisten von FMTool finden Sie in Kapitel 8, diejenigen von FMLoader in Kapitel 9.

7.1. Aufgabenstellung Treppenhausautomat

Herr Meier hat ein Einfamilienhaus. Er möchte, dass das Licht im Treppenhaus immer nur für 3 Minuten eingeschaltet ist, wenn jemand auf eine Taste drückt. Es sind drei Tasten und ein Aktor vorhanden.

Das folgende Bild zeigt das Projekt in der ETS.

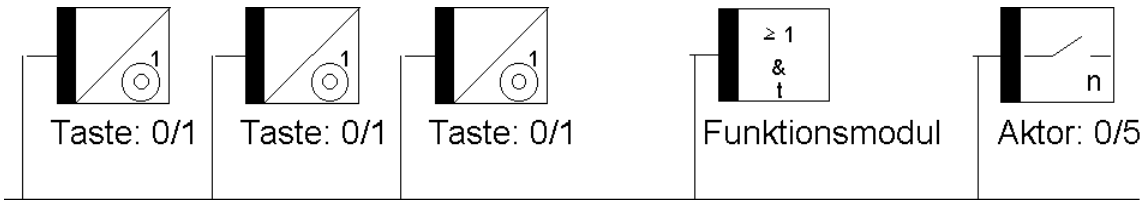


Bild 7.1.0.1.: Aufgabenstellung ‘Treppenhausautomat’

Das Funktionsmodul soll den Aktor für drei Minuten einschalten, wenn eine der Tasten gedrückt wird. Die Tasten sind so konfiguriert, dass sie beim Drücken der Taste ein Einschalttelegramm auf den Bus senden und beim Loslassen der Taste ein Ausschalttelegramm senden. Die Gruppenadresse der Tasten ist 0/1, die des Aktors 0/5. Die physikalischen Adressen können beliebig gewählt werden.

7.2. Programm erstellen

7.2.1. Projekt erstellen

Die Programme für das Funktionsmodul werden im FMTool als Projekte bezeichnet. Ein Projekt wird am Schluss, wenn es fertig programmiert ist, kompiliert und in das Funktionsmodul geladen. Bevor Sie mit dem Programmieren beginnen, müssen Sie ein neues Projekt erstellen. Nennen Sie das Projekt „MEIER“.

1. Starten Sie FMTool mit einem Doppelklick auf das Startsymbol von FMTool (Windows 3.1 oder 3.11) oder durch Auswahl des entsprechenden Menupunktes von Windows 95. Es kann nur gestartet werden, wenn es ordnungsgemäss installiert wurde und der Software-Schlüssel am Druckeranschluss (z.B. LPT1) eingesteckt ist.
2. Wählen Sie aus dem Menü **Projekt** den Befehl **Neu**. Es erscheint der Dialog „Neues Projekt“.
3. Geben Sie den Projektnamen „MEIER“ ein. Es können bis zu acht Buchstaben eingegeben werden.
4. Wählen Sie den Pfad, wo Sie Ihr Projekt ablegen möchten, z.B. C:\PROJEKTE.
5. Falls mehrere FBL-Versionen (Funktionsblockbibliotheken) zur Verfügung stehen, können Sie die gewünschte Version anwählen. Wenn Sie keine Version auswählen, verwendet das FMTool automatisch die neuste Version.
6. Geben Sie im Feld „Planer“ Ihren Namen ein und schreiben Sie den Namen Ihrer Firma in das unterste Feld. Diese Angaben werden später auf den verschiedenen Dokumenten ausgedruckt. Sie können diese später auch wieder ändern. Bestätigen Sie daraufhin mit **OK**.

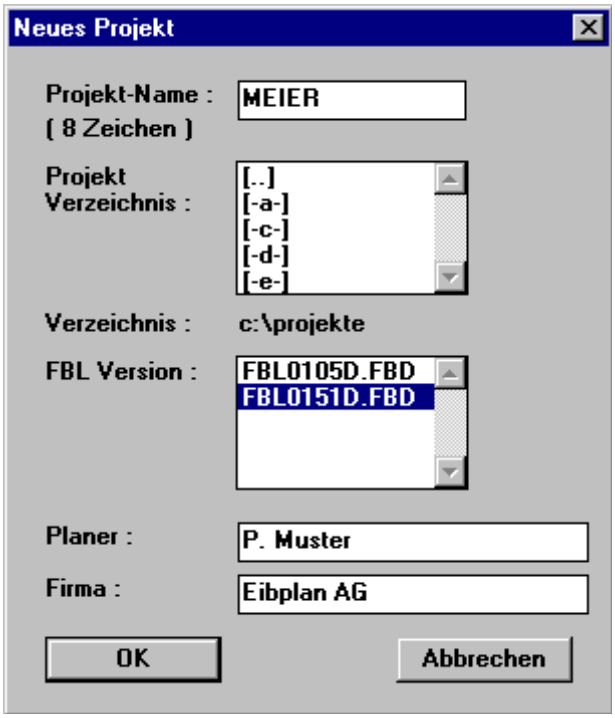


Bild 7.2.1.1.: Dialog „Neues Projekt“

7. Wählen Sie im Menu **Projekt** den Menupunkt **Optionen**. Dort können Sie 2-stufige oder 3-stufige Gruppenadressen einstellen. Wählen Sie für dieses Beispiel **2-stufige Gruppenadressen**.

7.2.2. Gruppe erstellen

Ein Projekt kann in verschiedene Teile, sogenannte Gruppen, aufgeteilt werden. Diese Aufteilung soll helfen, ein Projekt zu strukturieren. Dies ist vor allem bei grösseren Projekten wichtig, damit beim programmieren die einzelnen Teile schneller gefunden werden können.
Erstellen Sie nun eine neue Gruppe mit dem Namen „TREPPE“.

1. Wählen Sie aus dem Menü **Gruppe** den Befehl **Neu**. Es erscheint der Dialog „Neue Gruppe“.
2. Geben Sie den Namen „TREPPE“ ein. Es können bis zu acht Buchstaben eingegeben werden.
3. Schliessen Sie den Dialog mit **OK**.

Bild 7.2.2.1.: Dialog „Neue Gruppe“

7.2.3. Segment erstellen

Ein Segment ist wie ein Blatt Papier, auf dem gezeichnet werden kann. In den Segmenten wird das Programm erstellt.
Erstellen Sie ein neues Segment mit dem Namen „LICHT“.

1. Wählen Sie aus dem Menü **Segment** den Befehl **Neu**. Es erscheint der Dialog „Neues Segment“.
2. Geben Sie den Segmentnamen „LICHT“ ein. Es können bis zu acht Buchstaben eingegeben werden.
3. Schliessen Sie den Dialog mit **OK**.

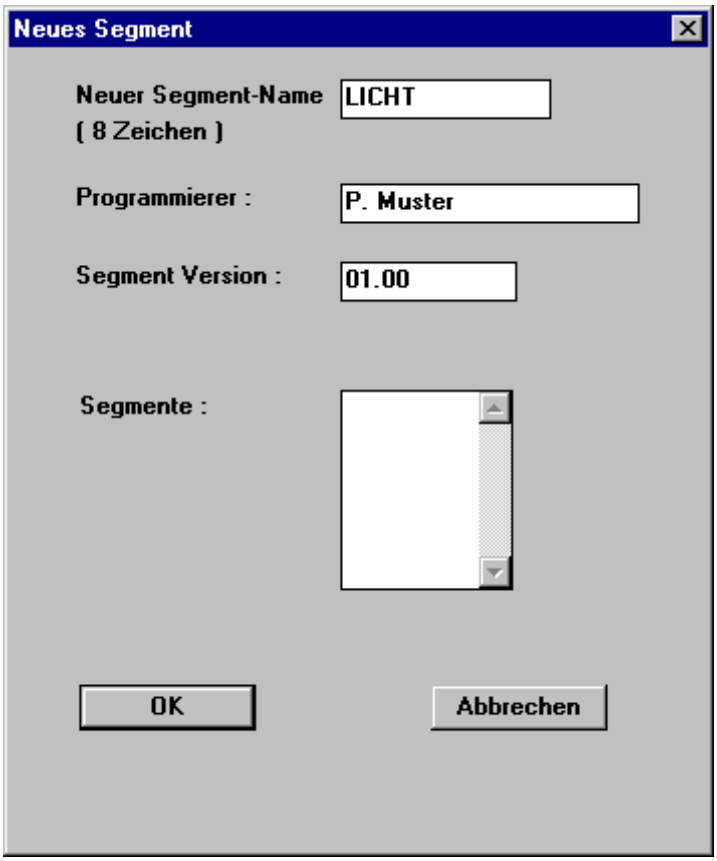


Bild 7.2.3.1.: Dialog „Neues Segment“

4. Es erscheint das leere Segment, das wie folgt aussieht:

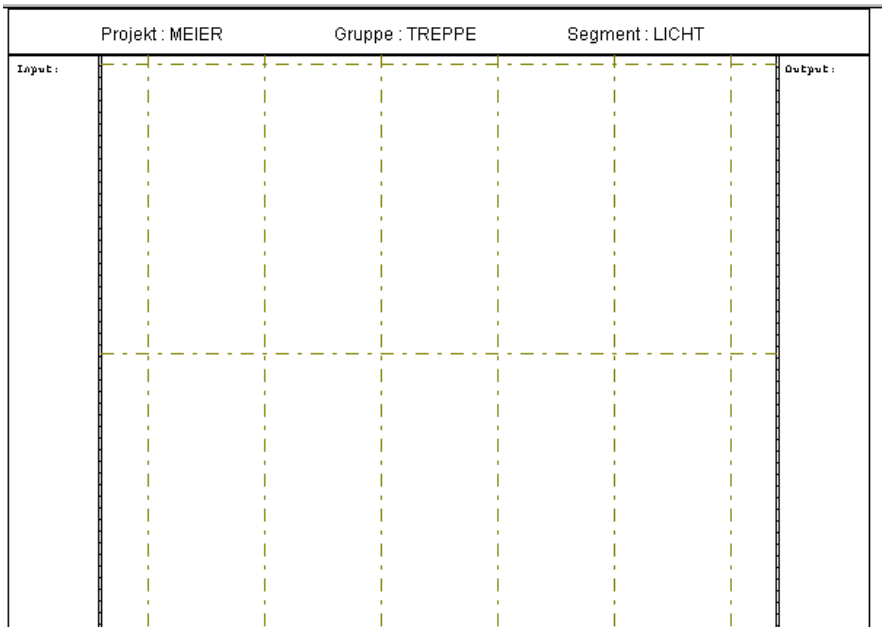


Bild 7.2.3.2.: leeres Segment

7.2.4. Funktionsblock setzen

Für unseren Treppenhausautomaten können wir den DMFF-Block verwenden. Die Beschreibung dieses Blockes ist Anhang A zu finden. Fügen Sie nun den Funktionsblock in das linke obere Feld im Segment ein.

1. Klicken Sie mit der Maus in das linke obere Feld im Segment. Es erscheint der Dialog „Funktionsblock auswählen“.
2. Wählen Sie in der Liste den DMFF-Block aus und drücken Sie **OK**. Bei diesem Block können Sie keine anderen Einstellungen vornehmen. Nun ist der Block im Segment eingefügt.

So sieht das Segment mit dem eingefügten Funktionsblock aus:

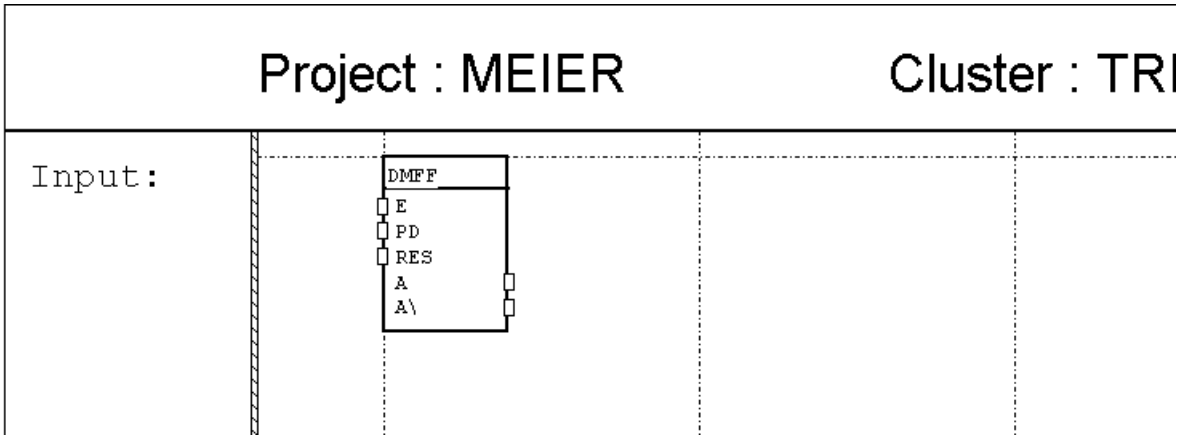


Bild 7.2.4.1.: Segment mit eingefügtem Funktionsblock

Anmerkung:
Einfache Funktionen können meist mit einem einzigen Funktionsblock programmiert werden. Oft müssen aber auch mehrere Blöcke zusammengehängt werden.
Der Beispielkatalog (Anhang zu diesem Handbuch) zeigt viele Beispiele, wie oft verwendete Funktionen mit dem Funktionsmodul programmiert werden können.

7.2.5. Zoom

Falls das Segment auf dem Bildschirm zu klein erscheint und der Text nicht lesbar ist, können Sie die Darstellung vergrössern.

1. Klicken Sie auf das Vergrösserungs-Symbol oder drücken sie die rechte Maustaste, um das Segment zu vergrössern.
2. Klicken Sie auf das Verkleinerungs-Symbol oder drücken Sie die rechte Maustaste, um das Segment zu verkleinern.

7.2.6. Linien zeichnen

Der Zustand der Tasten (gedrückt oder nicht gedrückt) soll auf den ersten Eingang des DMFF-Blockes gehen. Dazu muss der Eingang des Blockes mit dem Eingangssignalfeld (linker Segmentrand) verbunden werden. Gehen Sie wie folgt vor:

1. Klicken Sie auf das kleine Rechteck beim E-Eingang.
2. Klicken Sie auf den schraffierten Rand des Eingangssignalfeldes. Die Verbindung ist nun gezeichnet und es erscheint ein schraffierter Querbalken im Eingangssignalfeld.

So sieht das Signal nun aus:

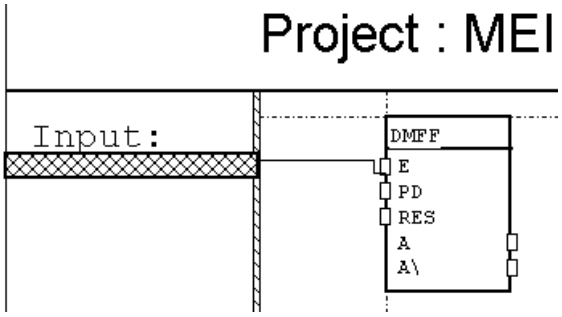


Bild 7.2.6.1.: Eingangssignal, noch nicht definiert

Alle Linien werden auf diese Weise gezeichnet: Zuerst den einen Verbindungspunkt anklicken, danach den anderen Verbindungspunkt anklicken. Es kommt dabei nicht darauf an, welchen Punkt Sie zuerst anklicken.

Tip:

Mittels Doppelklick auf das kleine Rechteck (Klemme) eines Funktionsblock-Eingangs wird dieser direkt mit dem Eingangssignalfeld verbunden.

7.2.7. Signale definieren

Der E-Eingang soll immer den Zustand der Tasten (gedrückt oder nicht gedrückt) haben. Deshalb weisen wir dem schraffierten Balken im Eingangssignalfeld ein Eingangssignal zu. Das Eingangssignal muss die Gruppenadresse 0/1 haben. Führen Sie die folgenden Schritte aus, um das Signal zu definieren:

1. Klicken Sie auf den schraffierten Querbalken. Der Dialog „Funktionsblock-parameter“ erscheint.
2. Drücken Sie den Knopf **Signal erzeugen**. Der Dialog „Signal erzeugen“ öffnet sich.

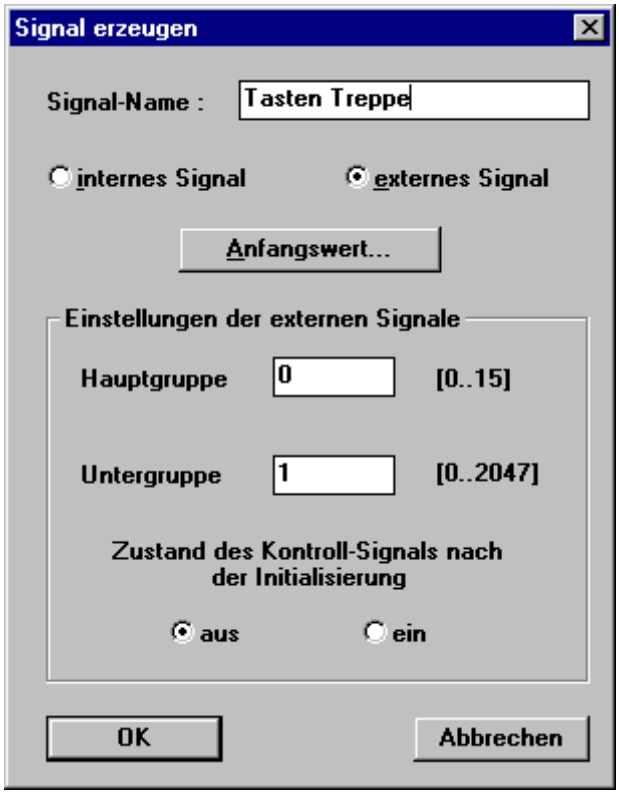


Bild 7.2.7.1.: Dialog „Neue Gruppe“

3. Geben Sie einen passenden Namen für das Signal ein, z.B. „Tasten Treppe“.
4. Normalerweise sind die folgenden Angaben schon so vorhanden, prüfen Sie jedoch zur Sicherheit, ob sie wirklich stimmen:
   - „externes Signal“ muss gewählt sein. Es handelt sich hier um ein externes Signal (vom Bus).
   - „Hauptgruppe“ muss 0 sein. Dies ist der erste Teil der Gruppenadresse.
   - „Untergruppe“ muss 1 sein. Dies ist der zweite Teil der Gruppenadresse.
   - „Zustand des Kontroll-Signals nach der Initialisierung“ muss „aus“ sein.

   Die genaue Bedeutung dieser Einstellungen können Sie im Kapitel „FMTool“ nachlesen.
5. Drücken Sie **OK**.

So sieht das Signal nun aus:

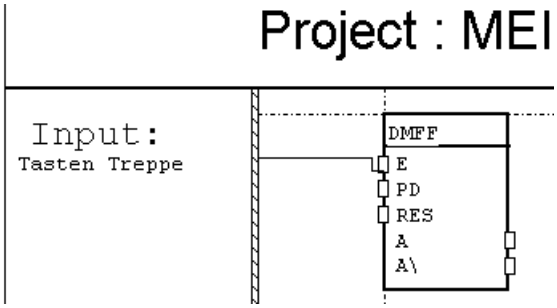


Bild 7.2.7.2.: Eingangssignal, definiert

Der Wert des A-Ausganges vom DMFF-Block muss zum Aktor gesendet werden. Dafür muss dieser Ausgang mit dem Ausgangssignalfeld (rechter Segmentrand) verbunden werden und es muss das Ausgangssignal mit der Gruppenadresse 0/5 definiert werden. Führen Sie folgende Schritte aus, um das Ausgangssignal zu zeichnen.

6. Klicken Sie auf das kleine Rechteck am DMFF-Block, das mit „A“ gekennzeichnet ist.
7. Klicken Sie auf den schraffierten Rand des Ausgangssignalfeldes. Die Verbindung ist nun gezeichnet. (Doppelklick auf das kleine Rechteck ergibt diese Verbindung direkt)
8. Klicken Sie auf den schraffierten Querbalken.
9. Der Dialog „Signal erzeugen“ erscheint.

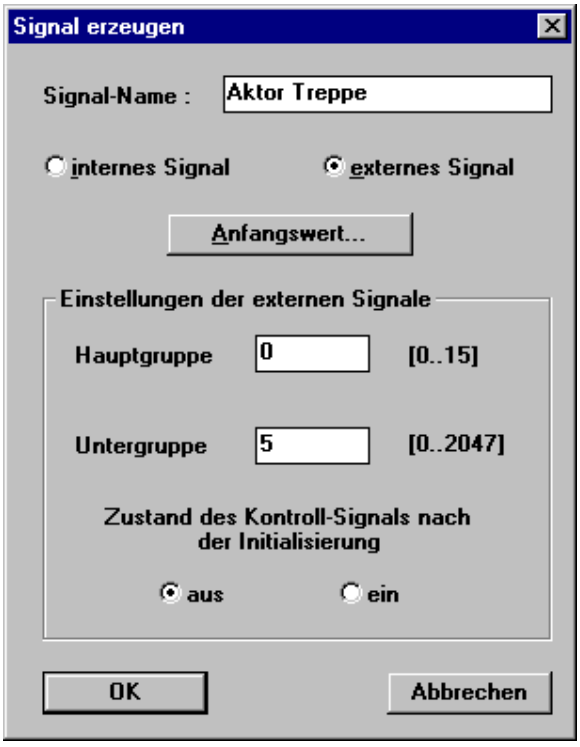


Bild 7.2.7.3.: Dialog „Signal erzeugen“

10. Geben Sie einen passenden Namen für das Signal ein, z.B. „Aktor Treppe“.
11. Überprüfen Sie oder ändern Sie die folgenden Einstellungen:
    - „externes Signal“ muss gewählt sein.
    - „Hauptgruppe“ muss 0 sein.
    - „Untergruppe“ muss 5 sein.
    - „Zustand des Kontroll-Signals nach der Initialisierung“ muss „aus“ sein.
12. Drücken Sie **OK**.

Das Segment sieht nun so aus:

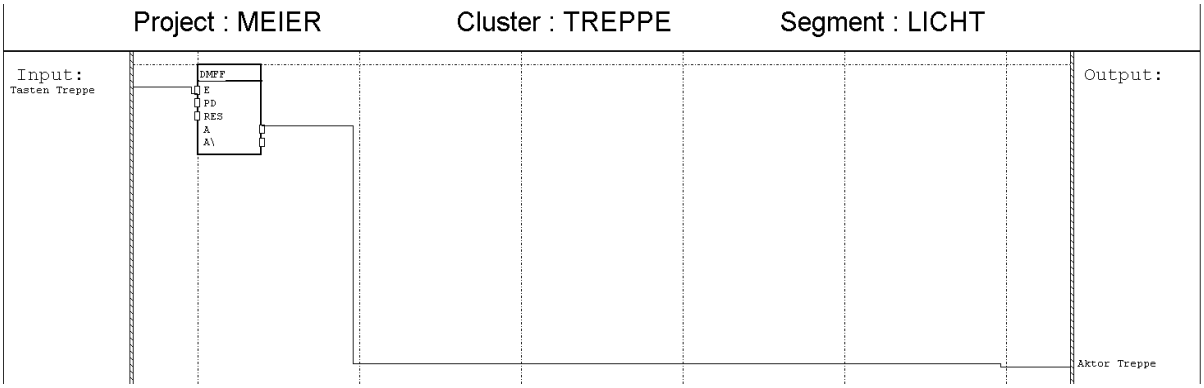


Bild 7.2.7.4.: Segment ‘LICHT’

7.2.8. Konstante setzen

Die beiden Eingänge „PD“ und „RES“ des DMFF-Blockes sollen dauernd den gleichen Wert haben. Der PD-Eingang (Pulsdauer) soll den Wert 1800 haben und der RES-Eingang (Reset) soll Null sein. Es müssen also Festwerte – sogenannte Konstanten – an die Eingänge angeschlossen werden. Verbinden Sie beide Eingänge mit dem Eingangssignalfeld und definieren Sie die Konstanten.

1. Verbinden Sie den PD-Eingang mit dem Eingangssignalfeld durch Klicken auf den Eingang und auf den schraffierten Rand des Eingangssignalfeldes.
2. Verbinden Sie den RES-Eingang mit dem Eingangssignalfeld durch Klicken auf den Eingang und auf den schraffierten Rand des Eingangs-signalfeldes.
3. Klicken Sie auf den schraffierten Querbalken der mit dem PD-Eingang verbunden ist.
4. Wählen Sie „Konstante eingeben“. Der Dialog „Word“ erscheint.

5. Geben Sie die Zahl 1800 ein und drücken Sie **OK**. Dieser Wert entspricht den 3 Minuten für den Treppenhausautomaten. Da wir die Taskzykluszeit weiter unten auf 100 ms setzen werden (Millisekunden = 1/1000 Sekunden), wird die Ausschaltverzögerung 1800 · 100 ms = 180 Sekunden betragen.
6. Klicken Sie auf den schraffierten Querbalken der mit dem RES-Eingang verbunden ist.
7. Wählen Sie „Konstante eingeben“. Der Dialog „Bit“ erscheint.
8. Wählen Sie den Wert 0 und drücken Sie **OK**.

Die Eingänge sehen nun so aus:

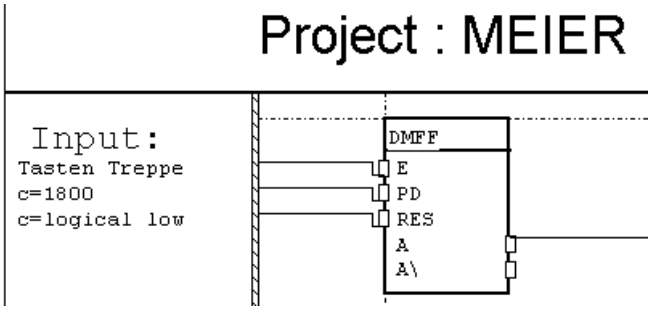


Bild 7.2.8.1.: Segment ‘LICHT’

7.2.9. Segment speichern

Unser Segment für den Treppenhausautomaten ist jetzt fertig. Nun muss es noch gespeichert werden, damit diese Daten nicht verloren gehen.

1. Klicken Sie auf den Speicherknopf oder wählen Sie im Menü **Segment** die Funktion **Speichern**.

7.2.10. Segment schliessen

Schliessen Sie das Segment wie folgt:

1. Drücken Sie auf den Schliessen-Knopf oder wählen Sie im Menü **Segment** die Funktion **Schliessen**.

7.2.11. Task definieren

Das Segment „LICHT“ muss einem Task zugewiesen werden, damit das Funktionsmodul weiss, wann es dieses Segment ausführen muss. Zuerst müssen Sie einen neuen Task definieren:

1. Wählen Sie im Hauptmenü **Task erzeugen**. Der Dialog „Task erzeugen“ erscheint.
2. Drücken Sie **Tasks definieren**. Der Dialog „Tasks festlegen“ erscheint.
3. Setzen Sie den Textcursor in das oberste Eingabefeld in der Spalte „Namen“ und geben Sie den Tasknamen „Task 1“ ein.
4. Klicken Sie 5 mal auf den Knopf [+] in der Spalte „Anpass“. Die Zeit in der Spalte „Neue Zeit“ muss jetzt 100 ms anzeigen.

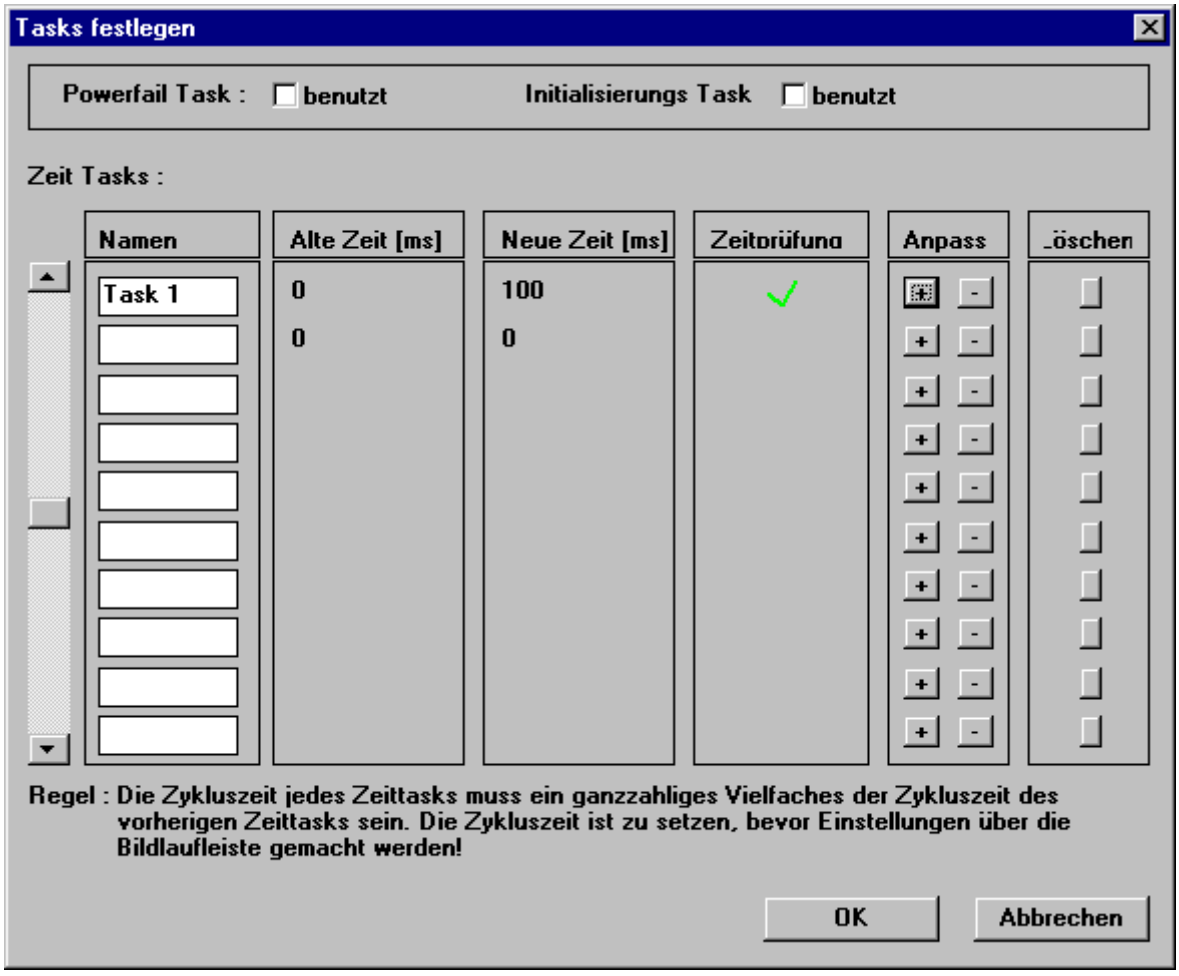


Bild 7.2.11.1.: Dialog „Task festlegen“

5. Drücken Sie **OK**. Der neue Task ist nun definiert.

6. Nun weisen Sie das Segment „LICHT“ dem Task „Task 1“ zu. Dazu drücken Sie den Knopf [->], um das Segment in den Task zu verschieben.

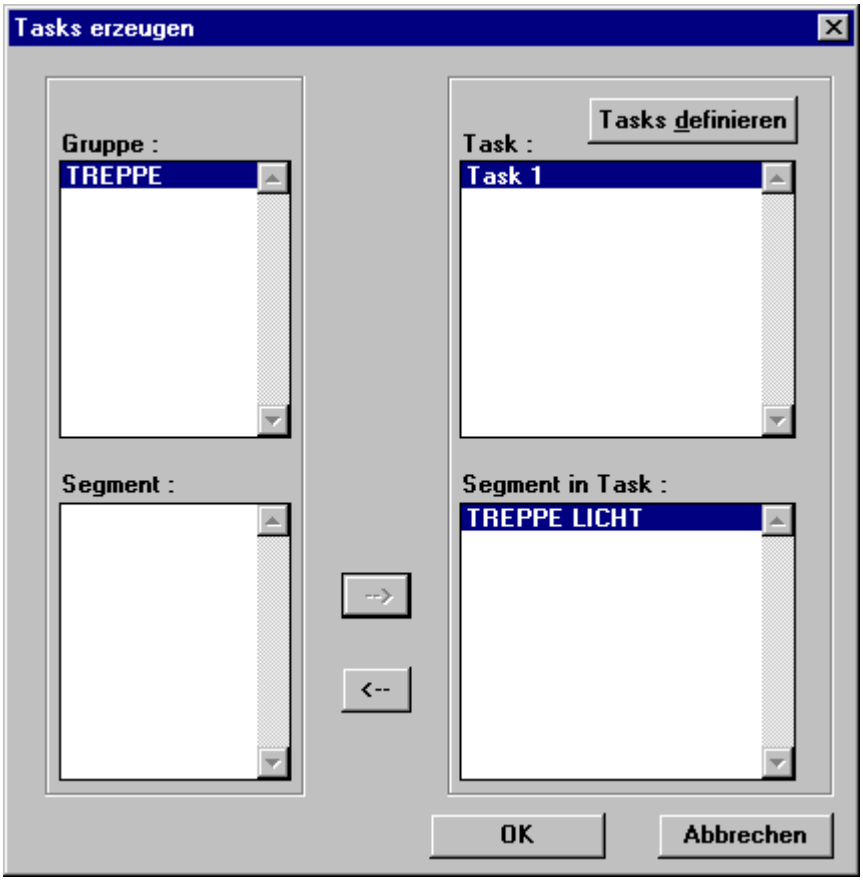


Bild 7.2.11.2.: Dialog „Tasks festlegen“

7. Bestätigen Sie mit **OK**, um den Taskbuilder zu verlassen.

7.2.12. Projekt kompilieren

Damit das Projekt auf dem Funktionsmodul ausgeführt werden kann, muss es in Code umgewandelt werden, den das Funktionsmodul versteht. Dieser Vorgang wird kompilieren genannt. Der Kompiler wandelt die grafischen Projektdaten um und erzeugt danach ein sogenanntes Hex-File.
Um das Projekt zu kompilieren gehen Sie wie folgt vor:

1. Wählen Sie im Hauptmenü **Kompiler**. Der Dialog „Kompiler“ erscheint.
2. Drücken Sie **Start** um den Kompiliervorgang zu starten.
3. Prüfen Sie, ob das Projekt erfolgreich kompiliert werden konnte. Achten Sie auf folgende Punkte:
   - Alle acht Schritte sind mit einem Häckchen ✓ bestätigt.
   - Die letzte Zeile der Kompilermeldungen meldet keine „Fatale Fehler“ und keine „Fehler“.

   In unserer Aufgabe sieht dies wie folgt aus:

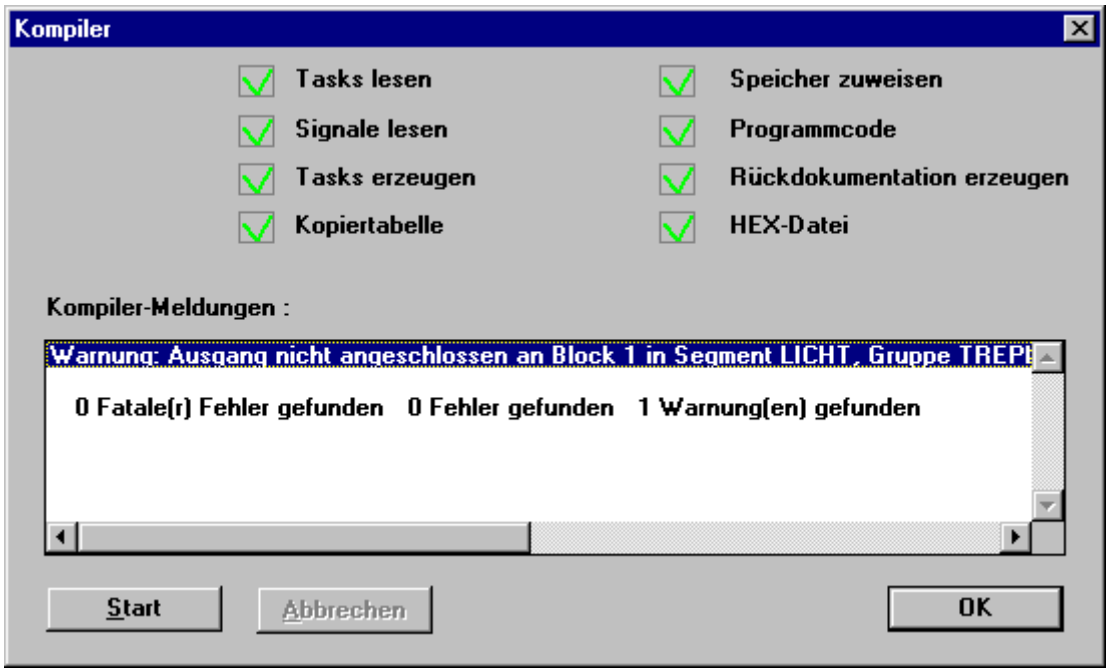


Bild 7.2.12.1.: Dialog „Kompiler“

   Stimmen die Meldungen in Ihrer Aufgabe nicht mit den hier gezeigten überein, prüfen Sie nochmals alle ausgeführten Schritte, bis Ihr Projekt fehlerfrei kompiliert werden kann.
4. Um den Kompiler wieder zu verlassen, drücken Sie **OK**.

7.2.13. Projekt ins Funktionsmodul laden

Das kompilierte Projekt kann nun in ein Funktionsmodul geladen werden. Falls Sie einen Testaufbau zur Verfügung haben, der mindestens einen Taster, einen Aktor, eine EIB-Schnittstelle (RS232) und ein Funktionsmodul beinhaltet, können Sie die Aufgabe ausprobieren.
Konfigurieren Sie mittels ETS die Taster und den Aktor wie in der Aufgabenstellung beschrieben. Laden Sie die physikalische Adresse in die im Funktionsmodul integrierte Busankopplung.
Jetzt können Sie mittels FMLoader das Projekt in das Funktionsmodul laden:

1. Stellen Sie sicher, dass Ihr Computer richtig mit der EIB-Schnittstelle verbunden ist. Das Kabel muss genau gleich eingesteckt werden wie für das Programmieren mit der ETS-Inbetriebnahme.
2. Prüfen Sie, ob das Funktionsmodul richtig mit Strom versorgt ist. Wenn die oberste der drei LED's blinkt und die unterste LED nicht leuchtet, ist die Stromversorgung in Ordnung.
3. Wählen Sie im Hauptmenü **Service** die Funktion **FM Loader**. Der FMLoader wird nun gestartet.
4. Wählen Sie im Hauptmenü **Kommunikations-Einstellungen**. Es erscheint der Dialog „Kommunikationseinstellungen".
5. Wählen Sie die serielle Schnittstelle, an der Sie das Kabel zu der EIB-Schnittstelle angeschlossen haben (COM1 oder COM2).
6. Geben Sie die physikalische Adresse ein, welche Sie dem Funktionsmodul (mittels ETS) zugewiesen haben. Tippen Sie z.B. „1.0.4". Drücken Sie **OK**, um den Dialog zu schliessen.
7. Wählen Sie im Hauptmenü **Laden in FM** die Funktion **Applikation**. Der Dialog „Applikation in FM laden" erscheint.
8. In der Zeile „Dateiname:" ist automatisch das Hex-File Ihres Projektes eingetragen (z.B. C:\PROJEKTE\MEIER\MEIER.hex). Drücken Sie nun den Knopf **Ladevorgang starten**. Der gelbe Balken wird Ihnen den Fortschritt des Ladevorgangs anzeigen. Am Schluss erscheint die Meldung „Applikation erfolgreich in das FM geladen. Bitte führen Sie einen 'Reset' aus, um die Applikation zu starten". Drücken Sie **OK**, um den Ladevorgang zu beenden.
   Sollte bei Ihnen ein Fehler auftreten, schliessen Sie den FMLoader, überprüfen Sie nochmals die Verbindung zwischen Ihrem Komputer und der Instabus-EIB-Schnittstelle und führen Sie die obigen Schritte erneut aus.
9. Wählen Sie im Hauptmenü **Reset FM** und bestätigen Sie die Frage „Möchten Sie einen 'Reset' am Funktionsmodul ausführen?" mit **OK**.
10. Die Meldung „'Reset' am Funktionsmodul ist erfolgt." zeigt an, dass das Funktionsmodul neu gestartet wurde. Drücken Sie **OK**.

In diesem Zustand muss die oberste der drei LEDs blinken, die mittlere muss leuchten und die unterste LED darf nicht leuchten.

7.2.14. FMLoader und FMTool schliessen

Schliessen Sie den FMLoader und das FMTool durch einen Doppelklick auf den Knopf  in der linken oberen Ecke (Windows 3.1 oder 3.11), mit dem Knopf  in der rechten oberen Ecke (Windows 95) oder so, wie Sie sich gewohnt sind, andere Windows-Applikationen zu schliessen.

7.2.15. Projekt testen

Testen Sie, ob die Schaltung richtig funktioniert. Drücken Sie auf die Taste. Der Aktor muss nun für 3 Minuten eingeschaltet bleiben.

In der folgenden Aufgabe stellen wir die Zeit des Treppenhausautomaten etwas kürzer ein, damit wir zum Testen nicht immer so lange warten müssen. Wir werden die Funktion auch etwas ändern und dabei weitere wichtige Funktionen von FMTool kennenlernen.

7.3. Aufgabenstellung Treppenhausautomat mit Ökoschaltung

Herr Meier möchte die Funktion der Lichtsteuerung in seinem Treppenhaus erweitern. Er möchte, dass das Licht im Treppenhaus nach 3 Minuten automatisch löscht, doch will er das Licht auch von Hand wieder einschalten können (sogenannte Ökoschaltung).

Um die Aufgabe noch etwas anspruchsvoller zu machen, gehen wir davon aus, dass die drei Tasten alle eine andere Gruppenadresse haben. Zusätzlich konfigurieren wir die Schalter so, dass sie bei jedem Tastendruck (entweder beim Drücken oder beim Loslassen der Taste, jedoch *nicht* beim drücken und loslassen) abwechslungsweise ein Ein- und ein Ausschalttelegramm senden (Umschaltfunktion).

Das folgende Bild zeigt das Projekt in der ETS mit den Gruppenadressen der Busteilnehmer. Die physikalischen Adressen können beliebig gewählt werden.

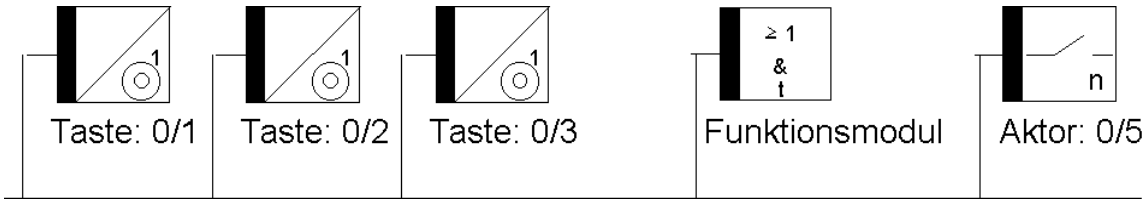


Bild 7.3.0.1.: Aufgabenstellung ‘Treppenhausautomat mit Ökoschaltung’

7.3.1. Die Lösung

Die Aufgabe kann mit folgender Schaltung gelöst werden:

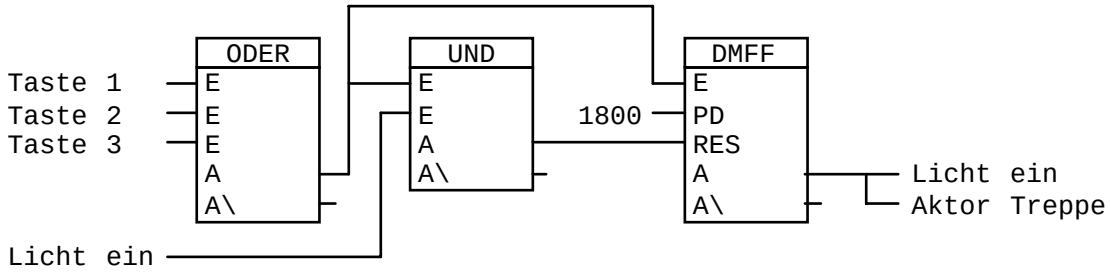


Bild 7.3.1.1.: Lösung ‘Treppenhausautomat mit Ökoschaltung’

Der ODER-Block stellt fest, ob einer der drei Tasten gedrückt wurde. Da die drei Tasten abwechsungsweise ein Ein- und ein Ausschalttelegramm senden, müssen wir sogenannte Kontrollsignale an den ODER-Block anschliessen.

Die Kontrollsignale haben kurz den Wert 1, wenn ein Telegramm von einem Taster empfangen wurde. Sonst haben sie immer den Wert 0. Sie geben also nicht an, ob ein Telegramm ein- oder ausschaltet, sondern nur, ob ein Telegramm empfangen wurde oder nicht.

Wird eine Taste gedrückt, schaltet der A-Ausgang des ODER-Blockes kurz auf 1. Dadurch wird der DMFF-Block eingeschaltet und der Aktor wird ebenfalls eingeschaltet. Das Signal „Licht ein“ hat den gleichen Wert wie der Aktor. Ist das Licht eingeschaltet und wird nochmals eine Taste gedrückt, so setzt der UND-Block den DMFF-Block auf 0 und schaltet somit das Licht wieder aus.

Sehen Sie sich die Umsetztung dieser Schaltung in den folgenden Schritten genauer an.

7.4. Lösung in ein FM-Programm umsetzen

7.4.1. Projekt öffnen

Starten Sie FMTool erneut und öffnen Sie das Projekt „MEIER“. Führen Sie folgende Schritte durch:

1. Starten Sie FMTool durch einen Doppelklick auf das Startsymbol von FMTool (Windows 3.1 oder 3.11) oder durch Auswahl des entsprechenden Menupunktes von Windows 95.
2. Starten Sie den Browser durch Drücken auf die Browser-Taste 
(ggf. erscheint der Dialog **Gruppe / Segment öffnen**, in welchem Sie die Taste **Suchen...** drücken können)
oder im Menü **Projekt** mit der Funktion **Öffnen**.
Der Dialog „Datei öffnen“ erscheint.

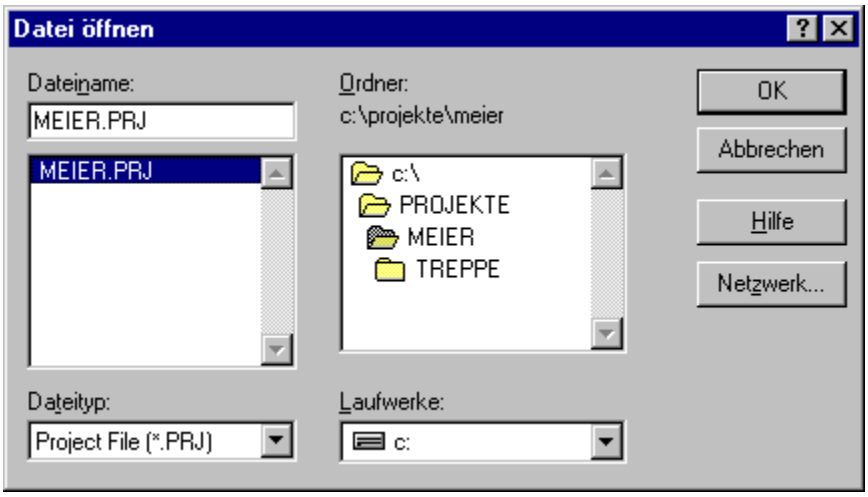


Bild 7.4.1.1.: Dialog „Datei öffnen“

3. Wechseln Sie in das Verzeichnis, in welchem Sie das Projekt „MEIER“ abgelegt haben (z.B. C:\PROJEKTE\MEIER). Dort finden Sie das Projektfile „MEiER.PRJ“. Wählen Sie dieses aus und drücken Sie die Taste **OK**.
4. Der Dialog **Gruppe / Segment öffnen** erscheint: (nächste Seite)

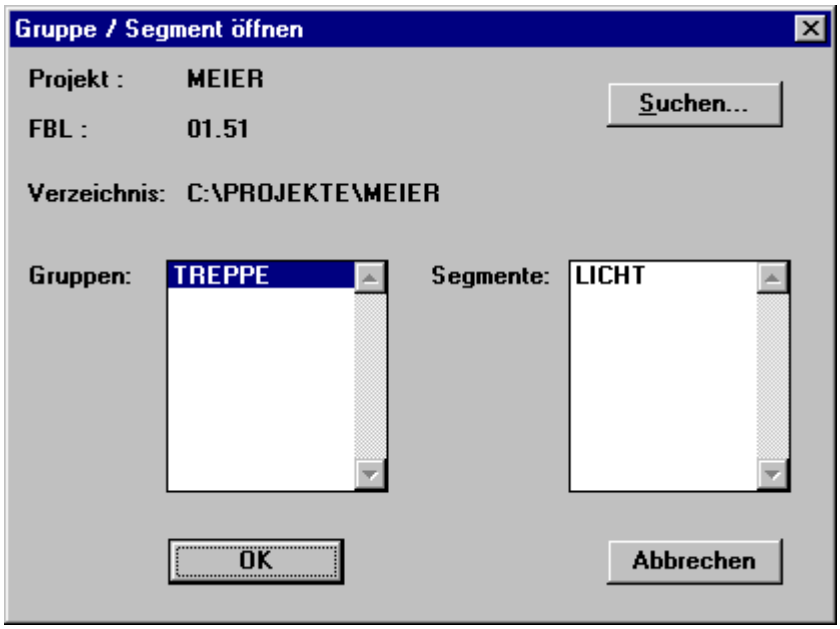


Bild 7.4.1.2.: Dialog „Gruppe / Segment öffnen“

7.4.2. Gruppe öffnen

Öffenen Sie die Gruppe „TREPPE“, indem Sie in der Liste „Gruppen“ auf diesen Namen klicken.

7.4.3. Segment öffnen

Öffnen Sie das Segment „LICHT“ durch folgende Tätigkeit:

Wählen Sie das Segment „LICHT“ in der Liste „Segmente“ und drücken Sie **OK**. Sie können auch auf das Segment doppelklicken, um es schneller zu öffnen.

7.4.4. Signal löschen

Löschen Sie die Verbindung am E-Eingang des DMFF-Blockes. Sie müssen dazu den Editor in den Löschmodus umschalten. Gehen Sie wie folgt vor:

1. Drücken Sie die Taste für den Löschmodus  .
2. Klicken Sie in das kleine Rechteck am E-Eingang vom DMFF-Block. Das Signal verschwindet.
3. Drücken Sie die Taste für den Einfügemodus  um, wieder normal arbeiten zu können.

Diesen Schritt können Sie auch auslassen, da dieses Signal ohnehin im nächsten Schritt gelöscht würde.

Anmerkung:
Wenn Sie eine Verbindung löschen, die mit einem Eingangssignal verbunden ist, löschen Sie wirklich nur die Verbindung. Das Signal „Tasten Treppe“ bleibt immer noch bestehen, doch ist es nicht mehr mit dem Block-Eingang verknüpft. Wir werden es später wieder mit einem anderen Blockeingang verknüpfen.

7.4.5. Funktionsblock löschen

Für die neue Schaltung brauchen wir den DMFF-Block an dieser Stelle nicht mehr. Befolgen Sie die nachfolgenden Schritte, um den Funktionsblock und die daran verbundenen Linien zu löschen:

1. Klicken Sie in die Mitte des DMFF-Blockes. Der Dialog „Funktionsblock-Parameter“ erscheint.
2. Drücken Sie **Löschen**.
3. Bestätigen Sie die Frage „Wollen Sie den Funktionsblock und alle Verbindungen wirklich löschen?“ mit **OK**. Das Segment ist nun ganz leer.

Anmerkung:
Durch das Löschen des Blockes werden alle Verbindungen zu diesem Block gelöscht. Signale die einen Namen haben, gehen jedoch nicht verloren. Konstanten werden hingegen gelöscht.

7.4.6. Funktionsblöcke einfügen

Fügen Sie die vier Funktionsblöcke so in das Segment ein, dass es gleich aussieht wie im untenstehenden Bild. Achten Sie darauf, dass der ODER-Block drei Eingänge hat.

Anmerkung:
Da Sie einen Funktionsblock-Ausgang nicht zweimal mit dem Ausgangssignalfeld verbinden können, d.h., nicht auf zwei Ausgangssignale führen können, haben Sie zwischen den DMFF-Block und das Ausgangssignalfeld einen TRA-Block mit zwei Eingängen / Ausgängen einzufügen. Achten Sie darauf, dass dieser den Datentyp Bit hat.

1. Klicken Sie in das erste Feld. Wählen Sie den ODER-Block aus. Wählen Sie in der Tabelle „Anzahl Pins:“ die Zahl 3. Drücken Sie **OK**.
2. Klicken Sie in das zweite Feld. Wählen Sie den UND-Block aus. Drücken Sie **OK**.
3. Klicken Sie in das dritte Feld. Wählen Sie den DMFF-Block aus. Drücken Sie **OK**.
4. Klicken Sie in das vierte Feld. Wählen Sie den TRA-Block aus. Wählen Sie in der Tabelle „Anzahl Pins:“ die Zahl 2. Drücken Sie **OK**.

Das Segment sieht nun so aus:

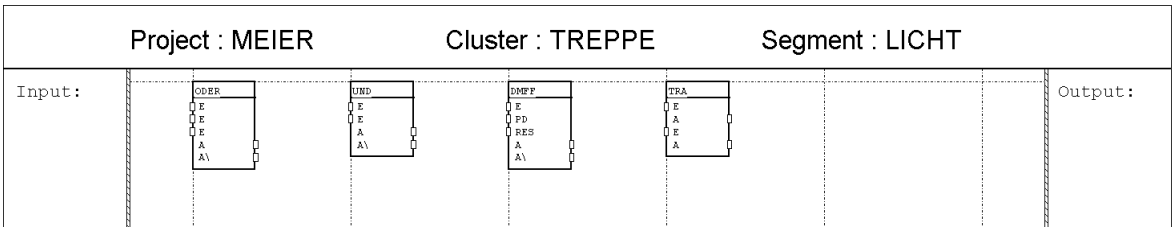


Bild 7.4.6.1.: Segment ‘LICHT’ mit Funktionsblöcken

7.4.7. Das erste Kontrollsignal einfügen

Verbinden Sie den ersten Eingang des ODER-Blockes mit dem Eingangssignalfeld und weisen Sie diesem Eingang das Eingangskontrollsignal der Taste mit der Gruppenadresse 0/1 zu.

1. Verbinden Sie den ersten Eingang des ODER-Blockes mit dem Eingangssignalfeld.
2. Klicken Sie auf den schraffierten Querbalken. und wählen Sie **Signal auswählen**. Der Dialog „Signal auswählen“ erscheint.
3. Sie haben in der Signalliste zwei Signale zur Auswahl. Das Signal „Tasten Treppe“ ist das normale Eingangssignal, wie wir es in der vorherigen Aufgabe verwendet haben. Das Signal „C_Tasten Treppe“ ist

das Kontrollsignal für dieselbe Gruppenadresse. Es wurde beim definieren des ersten Signales automatisch erzeugt. Wählen Sie das Kontrollsignal und drücken Sie **OK**.

Der Eingang sieht nun so aus:

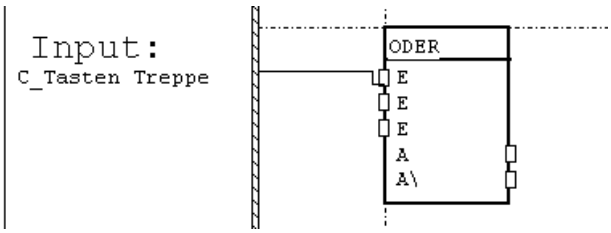


Bild 7.4.7.1.: Kontroll-Signal C_Tasten Treppe als Eingang

7.4.8. Signalnamen ändern

Da wir nun drei verschiedene Eingänge für die Tasten haben, wollen wir das erste Signal „Taste1 Treppe“ nennen. Ändern Sie den Namen des Signales wie folgt:

1. Klicken Sie auf den Signalnamen. Der Dialog „Signal löschen / umbenennen“ erscheint.
2. Drücken Sie **Umbenennen**. Der Dialog „Externes Signal umbenennen“ erscheint.
3. Ändern Sie den Namen des Signals auf „Taste1 Treppe“.
4. Drücken Sie **OK**.

7.4.9. Weitere Kontrollsignale einfügen

Da Kontrollsignale nur erzeugt werden, wenn ein normales Signal neu definiert wird, müssen wir zuerst ein normales Signal einfügen. Schliessen Sie das Signal „Taste2 Treppe“ an den zweiten Eingang des ODER-Blockes und ersetzen Sie es anschliessend durch das Kontrollsignal.

1. Verbinden Sie den zweiten Eingang des ODER-Blockes mit dem Eingangssignalfeld.
2. Klicken Sie auf den schraffierten Querbalken. Wählen im erscheinenden Dialog **Signal erzeugen**. Der Dialog „Signal erzeugen“ erscheint.
3. Geben Sie den Signalnamen „Taste2 Treppe“ ein und ändern Sie die Gruppenadresse auf 0/2. Drücken Sie **OK**.
4. Klicken Sie auf das neu eingefügte Signal. Der Dialog „Signal löschen / umbenennen“ erscheint.
5. Drücken Sie **Wechseln**. Der Dialog „Signal auswählen“ erscheint.

6. Wählen Sie das Kontrollsignal „C_Taste2 Treppe“ und drücken Sie **OK**.

Schliessen Sie nun auf die gleiche Weise das Signal „C_Taste3 Treppe“ an den dritten Eingang des ODER-Blockes. Das Segment sieht nun so aus:

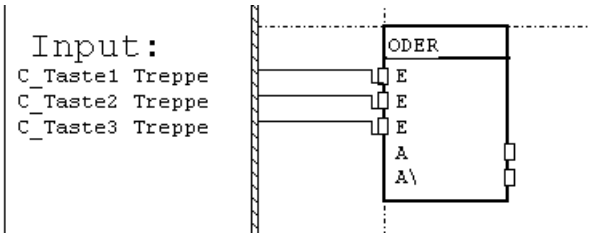


Bild 7.4.9.1.: ODER-Block mit drei angeschlossenen Eingängen

7.4.10. Verbindungen zeichnen

Zeichnen Sie nun alle nötigen Verbindungen, bis Ihr Segment wie in der untenstehenden Abbildung aussieht.

1. Verbinden Sie den A-Ausgang des ODER-Blocks mit dem ersten Eingang des UND-Blocks.
2. Verbinden Sie den A-Ausgang des ODER-Blocks mit dem E-Eingang des DMFF-Blocks.
3. Verbinden Sie den zweiten Eingang des UND-Blocks mit dem Eingangs-signalfeld.
4. Verbinden Sie den PD-Eingang des DMFF-Blocks mit dem Eingangs-signalfeld.
5. Verbinden Sie den A-Ausgang des UND-Blocks mit dem RES-Eingang des DMFF-Blocks.
6. Verbinden Sie den A-Ausgang des DMFF-Blocks zuerst mit dem ersten Eingang des TRA-Blocks und danach mit dem zweiten Eingang dieses Blockes.
7. Verbinden Sie die beiden Ausgänge des TRA-Blocks mit dem Ausgangs-signalfeld.

Das Segment sieht nun so aus:

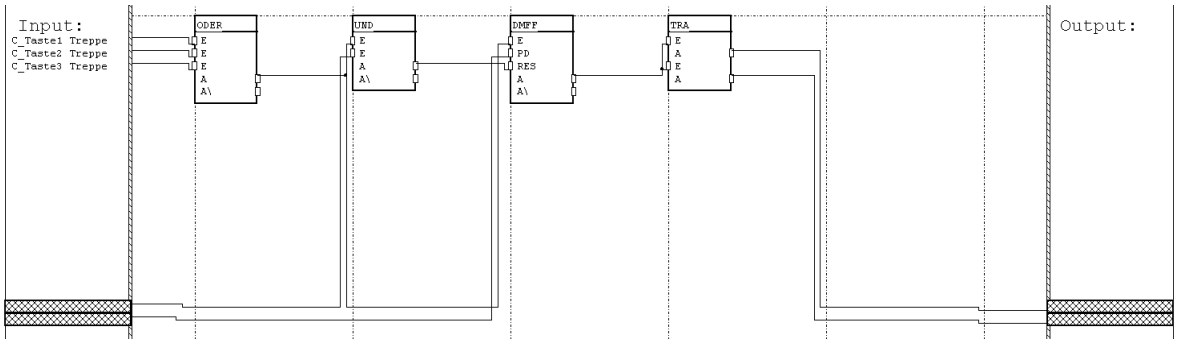


Bild 7.4.10.1.: Aufgabenstellung 'Treppenhausautomat mit Ökoschaltung'

7.4.11. Internes Signal einfügen

Neben den Eingangs- und Ausgangssignalen (sogenannte externe Signale) gibt es noch interne Signale. Diese verbinden die Funktionsblöcke miteinander, wie die Verbindungen zwischen den Blöcken (lokale Signale). Sie haben jedoch einen Namen und können auch in anderen Segmenten verwendet werden. Zudem ist es mit den internen Signalen möglich, Signalrückführungen zu machen, so wie wir es hier benötigen.
Definieren Sie nun ein Internes Signal am ersten Ausgang und nennen Sie es „I_Licht ein".

1. Klicken Sie auf den oberen schraffierten Querbalken im Ausgangssignalfeld. Der Dialog „Signal auswählen" erscheint.
2. Drücken Sie **Neues Signal**. Der Dialog „Signal erzeugen" erscheint.
3. Geben Sie den Signalnamen „I_Licht ein".
4. Klicken Sie auf den „Punkt" **internes Signal**.
5. Drücken Sie **OK**.

Der Ausgang sieht nun so aus:

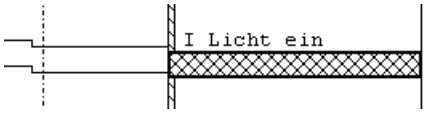


Bild 7.4.11.1.: Ausgangssignale

Anmerkung:
Es lohnt sich, interne Signale speziell zu kennzeichnen. So werden sie beim Betrachten der Segmente sofort als interne Signale erkannt. Eine Möglichkeit, wie dies gemacht werden kann: alle Namen der internen Signale mit „I_" beginnen.

7.4.12. Ein internes Signal weiterverbinden

Das soeben neu definierte interne Signal soll nun mit dem zweiten Eingang des UND-Blockes verbunden werden. Verbinden Sie das Signal „I_Licht ein" mit diesem Eingang.

1. Klicken Sie auf den oberen schraffierten Querbalken im Eingangssignalfeld. Der Dialog „Funktionsblock-Parameter“ erscheint.
2. Drücken Sie **Signal auswählen**. Der Dialog „Signal auswählen“ erscheint.
3. Wählen Sie das Signal „I_Licht ein“ und drücken Sie **OK**.

Der Eingang sieht nun so aus:

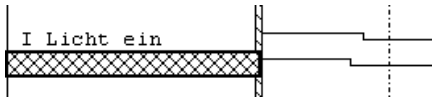


Bild 7.4.12.1.: interne Signale

7.4.13. Zeit für den Treppenhausautomaten einstellen

Wie in der ersten Aufgabe wird die Abfallverzögerung für den Treppenhausautomaten am PD-Eingang des DMFF-Blockes eingestellt. Damit wir zum Testen der Schaltung nicht immer drei Minuten warten müssen, setzten wir diese Zeit nun auf 7 Sekunden ein. Gemäss Formel für den PD-Eingang, die wir von der Funktionsblockbeschreibung haben, muss die Konstante den Wert 70 haben (7 Sek. / 0.1 Sek. = 70). Fügen Sie die Konstante ein.

1. Klicken Sie auf den verbleibenden schraffierten Querbalken im Eingangssignalfeld. Der Dialog „Funktionsblock-Parameter“ erscheint.
2. Drücken Sie **Konstante eingeben**. Der Dialog „Word“ erscheint.
3. Geben Sie den Wert 70 ein und drücken Sie **OK**.

7.4.14. Ausgangssignal verbinden

Der zweite Ausgang des TRA-Blockes muss mit dem Ausgangssignal verbunden werden, das die Gruppenadresse 0/5 hat und den Aktor schaltet. Das Signal „Aktor Treppe“ von der Aufgabe 1 ist immer noch vorhanden. Weisen Sie dieses Signal dem Ausgang des TRA-Blockes zu.

1. Klicken Sie auf den verbleibenden schraffierten Querbalken im Ausgangssignalfeld. Der Dialog „Signal auswählen“ erscheint.
2. Wählen Sie das Signal „Aktor Treppe“ aus und drücken Sie **OK**.

Das fertige Segment sieht nun wie folgt aus:

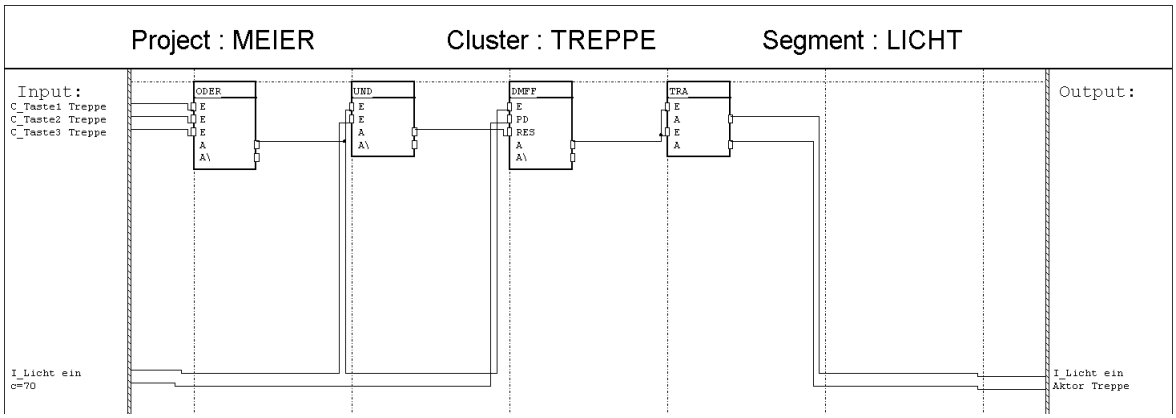


Bild 7.4.14.1.: Segment ‘LICHT’, fertig erstellt

7.4.15. Speichern, Schliessen

Das Segment ist nun fertig. Speichern und schliessen Sie es.

1. Drücken Sie auf den Speichern-Knopf  .
2. Drücken Sie auf den Schliessen-Knopf  .

7.4.16. Projekt kompilieren

Das Projekt ist nun fertig programmiert. Im Taskbuilder („Task erzeugen“ im Hauptmenü), muss nichts geändert werden. Somit muss das Projekt nur noch neu kompiliert werden. Starten Sie den Kompiler.

1. Wählen Sie **Kompiler** im Hauptmenu.
2. Drücken Sie **Start** und überprüfen Sie die Kompilermeldungen. Falls Sie keine „Fatale Fehler“, keine „Fehler“ und drei „Warnungen“ haben, konnte das Projekt erfolgreich kompiliert werden.
3. Drücken Sie **OK**.

7.4.17. Projekt ins Funktionsmodul laden

Konfigurieren Sie die Taster und den Aktor auf Ihrem Testaufbau wie in der Aufgabenstellung beschrieben. Laden Sie anschliessend die Applikation in das Funktionsmodul, so wie Sie es in der Aufgabe 1 gemacht haben. Nach dem Reset muss die mittlere LED leuchten und die unterste LED muss dunkel sein.

7.4.18. Projekt testen

Testen Sie nun, ob das Treppenhauslicht richtig funktioniert. Es muss von jedem der drei Tasten ein- und ausgeschaltet werden können. Das Licht bleibt aber höchstens für sieben Sekunden eingeschaltet.

7.4.19. Projekt dokumentieren

Von jedem Projekt kann eine Dokumentation ausgedruckt werden. Diese besteht aus:

1. Projektstruktur → Gruppen und zugehörende Segmente
2. Signal-Beschreibungsliste → Beschreibung der Signale
3. Signal-Positionstabelle → Orte der Verwendung der Signale

4. Task-Definitionsliste → Zusammenstellung der Tasks
5. Kompiler-Protokoll → Auflistung der Meldungen des Kompilers
6. Segmente → grafischer Ausdruck der Segmente

Die unter 1. bis 5. aufgelisteten Dokumente können Sie wie folgt ausdrucken:

Wählen Sie im Hauptmenu **Projekt** den Befehl **Drucken Projekt**. Es erscheint der Dialog „Drucken“. Durch Eingabe von **OK** werden alle oben erwähnten Dokumente ausgedruckt. Es besteht jedoch auch die Möglichkeit, einzelne Seiten auszudrucken.

Die unter 6. aufgeführten Segmente sind separat auszudrucken:

1. Wählen Sie im Hauptmenu **Projekt** den Befehl **Drucken Segmente...** . Es erscheint der Dialog „Segmente drucken“. Hier können Sie in der linken Liste die zu druckenden Segmente auswählen und mit den Pfeiltasten ins Feld „Ausgewählte Segmente“ verschieben. Durch die Taste **Alle** ⇉ lassen sich alle Segmente aus allen Gruppen gleichzeitig verschieben.
2. Verlassen Sie den Dialog durch Drücken der Taste **Drucken...** . Die ausgewählten Segmente werden gedruckt.

8. FMTool

8.1. Einleitung

8.1.1. Hauptmenü im Überblick

Bild 8.1.1.1.: Menüzeile (Hauptmenü von FMTool)

Projekt: Neu...
- Eröffnet ein neues Projekt
- Festlegen von Projektnamen und Auswahl der Funktionsblockbibliothek - Eingabe der Projekt-Infodaten (Planer, Firma)
- Generiert ein Unterverzeichnis für das Projekt mit dem Projektnamen

Öffnen...
- Öffnet ein bereits bestehendes Projekt

Schliessen
- Schliesst das aktuell geöffnete Projekt
- Speichert die Projektdaten

Speichern unter...
- Speichert das aktuell geöffnete Projekt unter einem andern Namen ab

Version der FBL...
- ‘Update’ des Projektes auf neuere Funktionsblockbibliothek

Projekt-Info...
- Verändern der Infodaten ‘Planer’, ‘Datum’, ‘Firma’

Optionen
- Auswahl 2- oder 3-stufige Gruppenadressen

Suchen...
- Suche und Öffnen einer Datei in beliebigem Verzeichnis

Drucken Projekt
- Druckt die Dokumentation des Projektes aus (ohne Segmente)

Drucken Segmente
- Ausdrucken von Segmenten (grafischer Ausdruck)

Beenden
Beendet FMTool

Gruppe: Neu...
- Eröffnet eine neue Gruppe im Projekt
- Generiert ein Unterverzeichnis für die neue Gruppe

Öffnen...
- Öffnet eine bereits vorhandene Gruppe

Schliessen
- Schliesst die aktuelle Gruppe

Löschen...
- Löscht die aktuelle Gruppe
  - inklusive den globalen Verbindungen zur Gruppe,
  - sowie die Segmente der Gruppe
  - und das Unterverzeichnis

Segment:

Neu...
- Eröffnet ein neues Segment in aktuellen Gruppe
- Eingeben des Namens, des Prgrammierers und der Segment-Version
- Anzeigen des Segments im Grafikeditor

Öffnen...
- Öffnet ein bereits vorhandenes Segment

Löschen...
- Löschen eines Segments mit Verbindungen aus der aktuellen Gruppe

Task erzeugen: (Taskbuiler)

- Definieren von Tasks
- Zuordnen der Segmente zu den Tasks
  (= Festlegen der Abarbeitungsstrategie)
- definieren der Task-Optionen

Kompiler:

- Übersetzt das Projekt (generiert den ‘Maschinencode’ für das Funktionsmodul

Service: FM Loader
Startet das Programm ‘FM Loader’

8.1.2. Werkzeugleiste

Bild 8.1.2.1.: Werkzeugleiste

Speichern des aktuell geöffneten Segments (im Grafikeditor)

Einfüge- bzw. Löschmodus für Funktionsblöcke und Verbindungen im Grafikeditor

Normalansicht bzw. vergrösserte Ansicht des Segments im Grafikeditor

Aufruf des ‘Browsers’:
- Dialog „Gruppe / Segment öffnen“
- Suchen eines Projektes in beliebigem Verzeichnis („Datei öffnen“)

Segment schliessen (im Grafikeditor)

8.1.3. Menü des Grafikeditors im Überblick

Bild 8.1.3.1.: Menüzeile des Grafikeditors

Segment:

Schliessen
- Schliesst das aktuelle Segment und wechselt zum Hauptmenü

Speichern
- Speichert das Segment

Segment-Info...
- Verändern der Segment-Infodaten ‘Programmierer’, ‘Datum’, ‘Version’

Drucken...
- Druckt das Segment

Seitenansicht
- Zeigt das ganze Segment (Druckvorschau)

Importieren...
- Importiert den Inhalt (Funktionsblöcke mit den Verbindungen, auch zu externen Signalen, jedoch ohne die vergebenen Gruppenadressen) eines andern Segments

Modus: Einfügemodus
- Schaltet in den Einfügemodus um und erlaubt das Einfügen von entsprechenden Elementen (Funktionsblöcke, Verbindungen)

Löschmodus
- Schaltet in den Löschmodus um und erlaubt das Löschen von Elementen auf dem Segment

Ansicht: ZOOM1
- Verkleinerte Ansicht des Segments

ZOOM2
- Vergrösserte Ansicht (Ausschnitt aus dem Segment)

Verbinden: Nach Mausklick auf eine Klemme eines Funktionsblock → Erstellen der Verbindung zur Eingangs- bzw. Ausgangssignalleiste

8.2. Beschreibung des Hauptmenüs

8.2.1. Projekt, Neu...

Unter diesem Befehl können Sie ein neues Projekt angelegen. FMTool erstellt für dieses Projekt ein eigenes, neues Unterverzeichnis. In diesem Verzeichnis sind alle Dateien des Projektes enthalten.

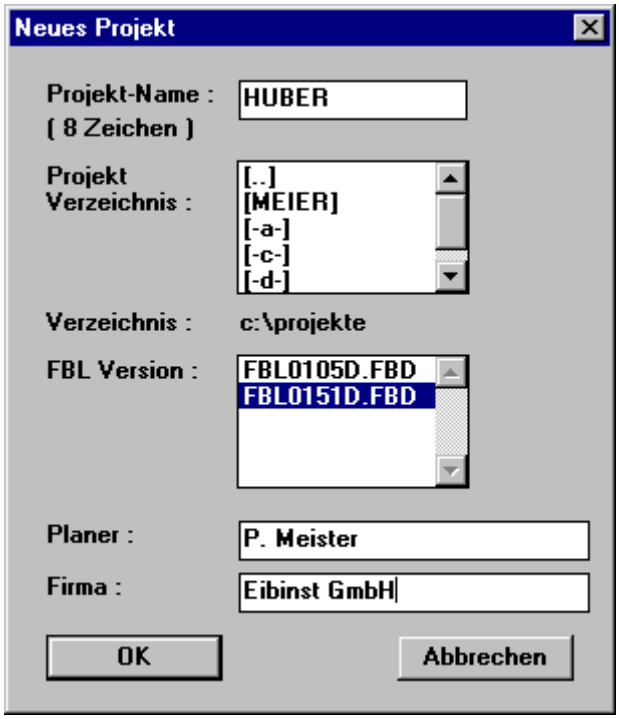


Bild 8.2.1.1.: Dialog „Neues Projekt“

Im Eingabefeld 'Projekt-Name' können Sie den Namen des Projektes eingeben. Dieser darf **max. 8 Zeichen** lang sein und nur die allgemein üblichen Zeichen für Dateinamen enthalten.

Wählen Sie mittels Doppelklicken in der Liste 'Projekt-Verzeichnis' ein Verzeichnis aus, in welchem das Projekt gespeichert werden soll. Der Eintrag 'Verzeichnis' zeigt den vollständigen Verzeichnisnamen an. Wählen Sie ebenfalls mittels Doppelklick in der Liste 'FBL Version' die gewünschte Funktionsblockbibliothek an. FMTool wählt als Standard immer die neueste, aktuellste und somit höchste Versionnummer an.

In den Eingabefeldern 'Planer' und 'Firma' können Sie den Namen des Planers sowie den Namen der Firma, z.B. des Planungsbüros eingetragen. Diese Angaben sind für das gesamte Projekt gültig und werden später in den Fusszeilen der Segmente angezeigt.

Mit dem Betätigen von OK erzeugt FMTool das Projektverzeichnis und die Projektdateien. Nach dem Erstellen erscheint in der Statuszeile der Projektname.

Bild 8.2.1.2.: Statuszeile

8.2.2. Projekt, Öffnen...

Unter diesem Befehl können Sie ein bestehendes Projekt öffnen.

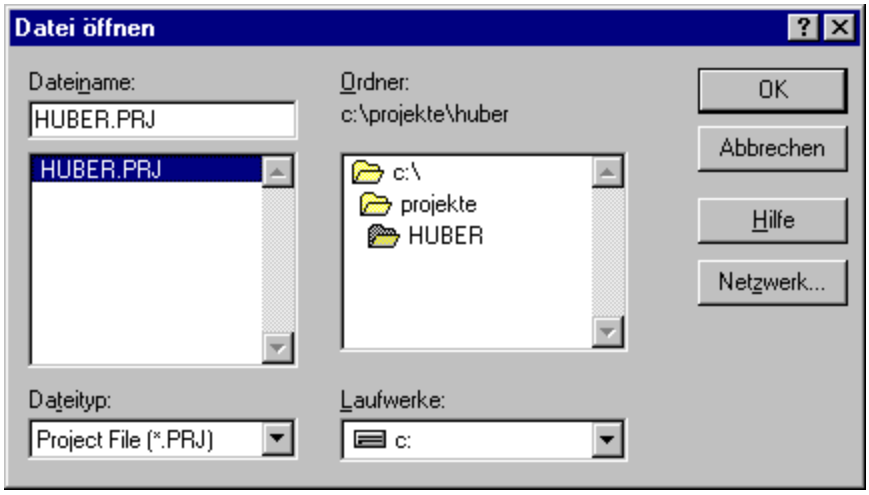


Bild 8.2.2.1.: Dialog „Datei öffnen“

Wählen Sie in der Verzeichnisliste zuerst das Projektverzeichnis an. Dieses Verzeichnis ist gleich bezeichnet wie das Projekt selbst. Danach öffnen Sie die Projektdatei. Die Projektdatei ist mit der Namenserweiterung *.PRJ im Projektverzeichnis abgespeichert.

Mit dem Betätigen von OK wird das Projekt geöffnet. Nach dem Öffnen erscheint in der Statuszeile der Projektname.

8.2.3. Projekt, Schliessen

Mit diesem Befehl wird ein aktuell geöffnetes Projekt geschlossen. Dabei speichert FMTool alle Änderungen in der Datenbank.

In der Statuszeile wird nach dem Betätigen der OK Taste kein Projektname eingeblendet.

8.2.4. Projekt, Speichern unter...

Bild 8.2.4.1.: Dialog „Speichern unter“

Ein geöffnetes Projekt lässt sich als ganzes unter einem andern Namen speichern und somit duplizieren bzw. vervielfachen. Geben Sie dazu im Eingabefeld ‘Projekt-Name’ denjenigen Namen ein, unter dem das Projekt gespeichert werden soll (Achtung: max. 8 Zeichen).

Im Feld ‘Projekt-Verzeichnis’ lässt sich das Verzeichnis anwählen, in dem das Projekt gespeichert werden soll. Das aktuelle Verzeichnis ist aus der darunter liegenden Zeile ersichtlich.

8.2.5. Version der FBL (Funktionsblockbibliothek)

Unter diesem Menüpunkt können Sie die Funktionsblockbibliothek des aktuell geöffneten Projektes ändern.

Hinweis:
Die Funktionsblockbibliothek muss dieselbe Hauptversionsnummer (zweite Ziffer nach der ersten Null) enthalten wie die bisherige und die Subversion muss grösser sein als die bisherige.

Wählen Sie in der Liste unter Berücksichtigung des Hinweises die neue Funktionsblockbibliothek. Die Änderungen werden beim Beenden des Dialoges durch OK übernommen.

8.2.6. Projekt-Info...

Nach dem Anwählen dieses Menüpunktes erscheint der Dialog „Projekt Info“. Dieser zeigt die aktuellen Informationen der Info-Felder ‘Projekt Name’, ‘Planer’, (Erstellungs-) ‘Datum’ und ‘Firma’. Ausser dem Projekt-Namen können alle Daten verändert werden. Beim Verlassen des Dialogs mit OK werden die gegebenenfalls geänderten Daten übernommen.

8.2.7. Projekt, Optionen

An dieser Stelle können Sie wie in ETS2 zwischen 2- und 3-stufigen Gruppenadressen auswählen. Entsprechend der gewählten Darstellungsart erscheinen im Grafikeditor in den Dialogen „Signal erzeugen“ und „Signal löschen / umbenennen“ die Eingabe- und Darstellungsfelder für die Gruppenadressen für 2- oder 3-stufige Gruppenadressen.

Die Darstellungsart kann jederzeit gewechselt werden. Sämtliche Gruppenadressen werden bei einem Wechsel der Darstellungsart umgerechnet.

8.2.8. Projekt, Suchen...

Wenn Sie diesen Punkt anwählen erscheint der Dialog „Gruppe / Segment öffnen". Dieser ermöglicht, die gewünschte Gruppe und das Segment durch Anwählen in den Listen direkt zu öffnen und in den Grafikeditor zu wechseln.

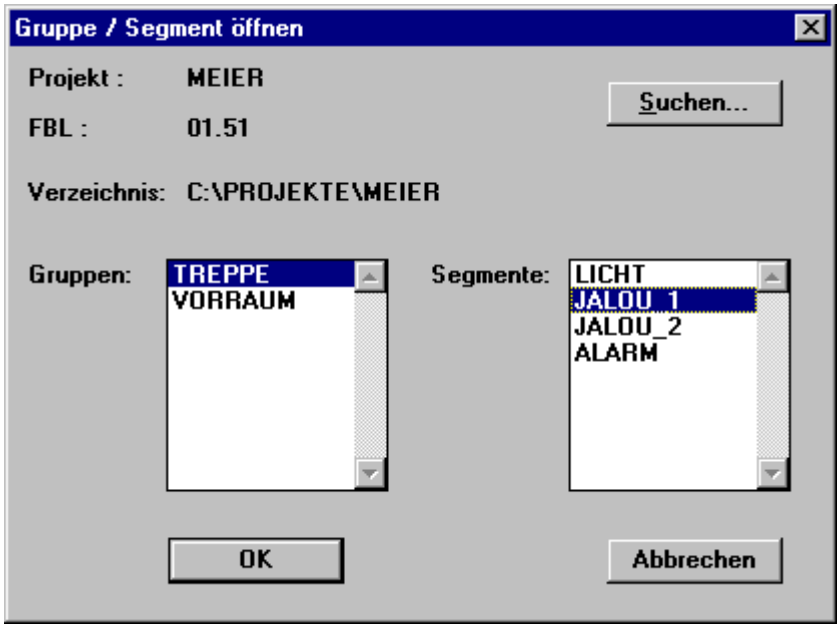


Bild 8.2.8.1.: Dialog „Gruppe / Segment öffnen“ (Browser)

Wird die Taste ‘Suchen...’ betätigt, erscheint der Dialog „Datei öffnen“ (siehe Kapitel ‚Datei, Öffnen’). In diesem Dialog haben Sie die Möglichkeit, ein anderes Projekt zu suchen und durch Eingabe von OK auch zu öffnen. Dabei wird das bisher aktive Projekt geschlossen und das neu angewählte geöffnet.

 Dieselbe Funktion wie der beschriebene Menüpunkt hat die Taste des Browsers auf der Werkzeugleiste.

8.2.9. Drucken Projekt

An dieser Stelle können Sie eine Dokumentation von jedem Projekt ausdrucken lassen. Diese besteht aus folgenden Listen und Tabellen:

1. Projektstruktur → Gruppen und zugehörende Segmente
2. Signal-Beschreibungsliste → Beschreibung der Signale
3. Signal-Positionstabelle → Orte der Verwendung der Signale
4. Task-Definitionsliste → Zusammenstellung der Taske
5. Kompiler-Protokoll → Auflistung der Meldungen des Kompilers

Nach dem Anwählen von ‘Drucken Projekt’ erscheint der Standard-Druckdialog „Drucken“. Wollen Sie alle unter 1. bis 5. aufgelisteten Dokumente ausdrucken, wählen Sie im Feld ‘Druckbereich’ den Punkte ‘Alle’ an und bestätigen die Eingabe mit OK. Durch Anwählen des Punktes ‘Seiten’ lassen sich auch einzelne Seiten drucken.

8.2.10. Drucken Segmente...

Zusätzlich zu den Listen können Sie zur vollständigen Dokumentation der Projekte auch die Segmente ausdrucken. Nach dem Anklicken von ‘Drucken Segemente...’ erscheint der nachfolgend abgebildete Dialog „Segmente drucken“ (siehe nächste Seite).

Links oben sind die Gruppen aufgelistet, darunter die Segmente, die zur oben markierten Gruppe gehören. Durch betätigen der Pfeiltaste oben können alle Segmente einer Gruppe gleichzeitig nach verschoben werden, die untere Pfeiltaste dient dem Verschieben eines einzelnen Segmentes.

Den Dialog verlassen Sie mit der Taste ‘Drucken...’. Nachher werden die ausgewählten Segmente gedruckt.

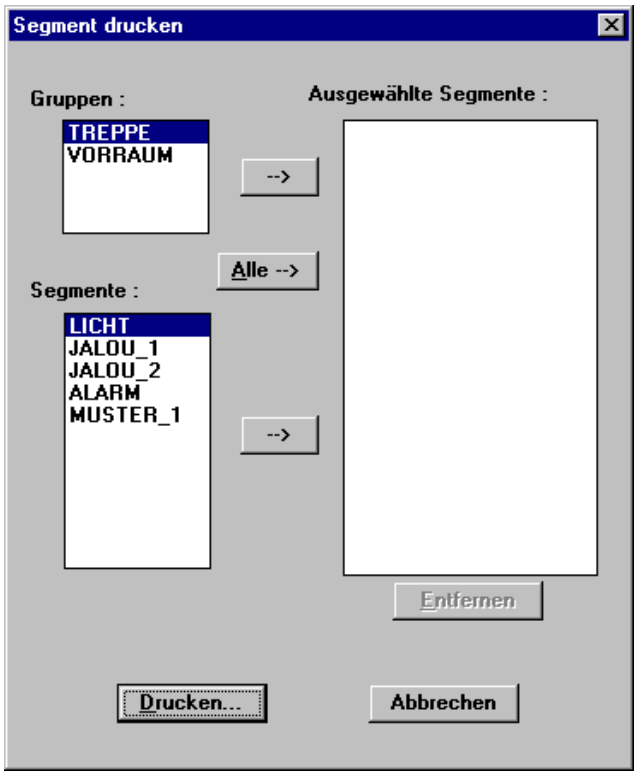


Bild 8.2.10.1.: Dialog „Segment drucken“

8.2.11. Projekt, Beenden

Mit ‘Beenden’ wird das Projekt geschlossen und ‘FMTool’ verlassen.

8.2.12. Gruppe, Neu...

Unter diesem Befehl können Sie eine neue Gruppe anlegen. FMTool erstellt für jede Gruppe im Projektverzeichnis ein eigenes Unterverzeichnis, in welchem alle Segmente der entsprechenden Gruppe abgelegt werden.

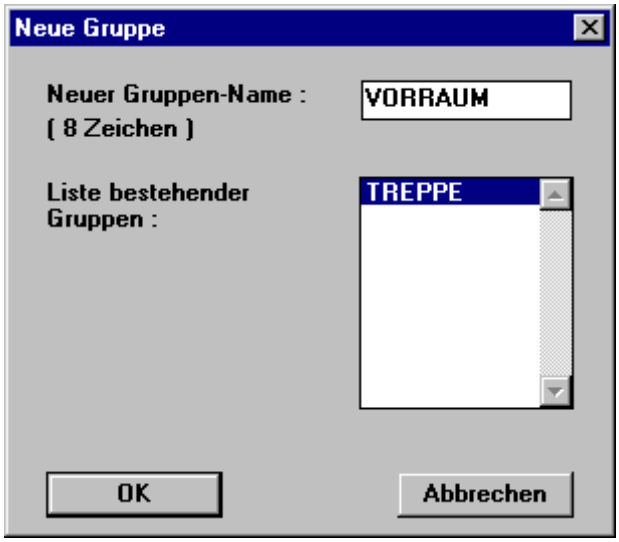


Bild 8.2.12.1.: Dialog „Neue Gruppe“

Geben Sie den neuen Gruppennamen ins Eingabefeld ein. Er darf **max. 8 Zeichen** lang sein und nur die allgemein üblichen Zeichen für Dateinamen enthalten.

8.2.13. Gruppe, Öffnen

Mit diesem Befehl können Sie eine bestehende Gruppe öffnen. Es lässt sich nur eine Gruppe gleichzeitig öffnen.

In der Liste, die unter diesem Menüpunkt eingeblendet wird, sind alle verfügbaren Gruppen aufgelistet. Wählen Sie die entsprechend gewünschte an und betätigen die OK-Taste.

8.2.14. Gruppe, Schliessen

Mit diesem Befehl wird die aktuell geöffnete Gruppe geschlossen. Dabei speichert FMTool alle Änderungen in der Datenbank des Projektes.

In der Statuszeile erlischt nach dem Betätigen der OK Taste der Gruppenname.

8.2.15. Gruppe, Löschen...

Unter diesen Menüpunkt können Sie eine Gruppe löschen. FMTool entfernt dabei alle Segmente der entsprechenden Gruppe sowie das Verzeichnis für diese Gruppe. Zusätzlich werden in der Projektdatenbank alle Verbindungen zu den Segmenten dieser Gruppe ausgetragen. Signale, welche exklusiv in dieser Gruppe Verwendung fanden, werden ebenfalls gelöscht in der Datenbank.

Bild 8.2.15.1.: Dialog „Gruppe löschen“

Die Liste zeigt alle im Projekt vorhandenen Gruppen an. Wählen Sie mit dem Balken die zu löschende Gruppe an. Nach dem Betätigen der Taste ‘Löschen’ erscheint der Dialog „Wollen Sie wirklich ...“. Wird dieser mit OK bestätigt, wird die entsprechende Gruppe **definitiv gelöscht**.

8.2.16. Segment, Neu...

Mit diesem können Sie ein neues Segment erstellen. Dieses Segment wird in einer Datei im aktuellen Gruppenverzeichnis des Projektes abgespeichert.

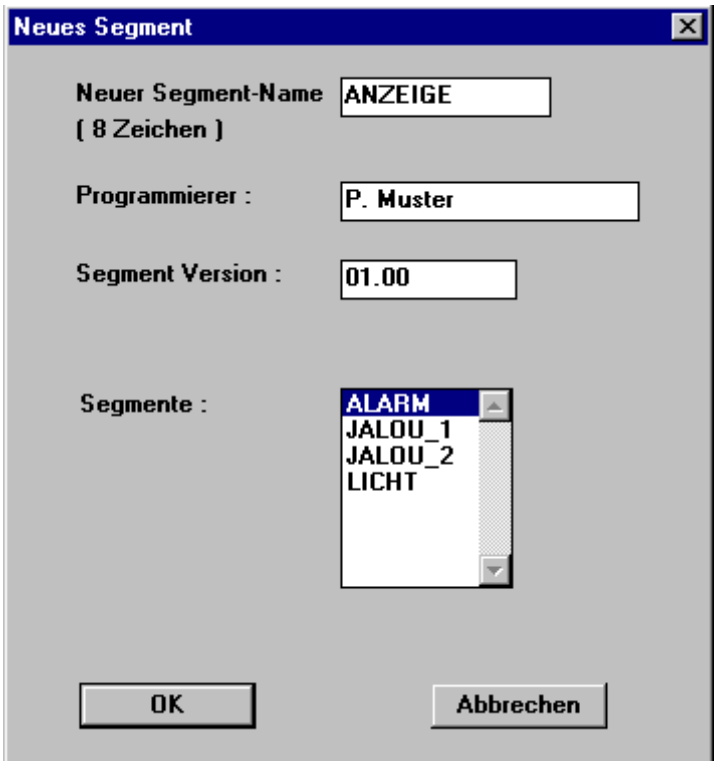


Bild 8.2.16.1.: Dialog „Neues Segment“

Geben Sie den neuen Segmentnamen ins Eingabefeld ein, **max. 8 Zeichen** lang und nur mit allgemein üblichen Zeichen für Dateinamen.

In der Liste sind alle bestehenden Segmente der aktuellen Gruppe aufgeführt. Wählen Sie für das Segment einen noch nicht verwendeten Namen.

In den Eingabefeldern ‘Programmierer’ und ‘Segment Version’ können Sie beliebige Einträge machen. Diese Angaben sind nur für das neue Segment gültig und werden später in den Fusszeilen des Segmentes angezeigt. Als Standard trägt FMTool den Eintrag des ‘Planers’ vom Projekt ein, genauso wie 01.00 als ‘Version’.

Nach dem Bestätigen mit OK wird der Grafikeditor mit dem neuen Segment geöffnet.

8.2.17. Segment, Öffnen...

Mit diesem Befehl können Sie ein beliebiges Segment der aktuell geöffneten Gruppe öffnen.

Wählen Sie dazu das gewünschte Segment in der Liste mittels Balken an. Nach dem Betätigen der OK-Taste wird der Grafikeditor mit dem aktuellen Segment geöffnet.

8.2.18. Segment, Löschen

Unter diesem Menüpunkt können Sie ein Segment löschen. In der Projektdatenbank werden alle Verbindungen zu diesem Segment ausgetragen.

Die Liste, die als Dialog nach dem Anwählen dieses Menüpunktes erscheint, zeigt alle in der aktuell geöffneten Gruppe vorhandenen Segmente an. Wählen Sie das zu löschende Segment an und betätigen die Taste ‘Löschen’. Danach erscheint der Dialog „Wollen Sie wirklich ...“. Wird dieser mit OK bestätigt, wird das entsprechende Segment **definitiv gelöscht**.

8.2.19. Task erzeugen (Taskbuilder)

Unter diesem Menüpunkt können Sie die Abarbeitungsstrategie Ihres Projektes festlegen, d.h., Sie bestimmen, wie oft und in welcher Reihenfolge die einzelnen Segmente (= Programmteile) im Funktionsmodul bearbeitet (abgearbeitet) werden.

Der Dialog „Tasks erzeugen“ (siehe nächste Seite) besteht aus mehreren Listen. Auf der linken Seite sind oben alle Gruppen, unten die der angewählten Gruppe zugeordneten Segmente dargestellt. Diese Segmente (Liste ‘Segment:’ sind noch keinem Task zugewiesen.

Die Liste rechts oben zeigt die verfügbaren Tasks, unten sind die Segmente aufgelistet, die dem angewählten Task zugewiesen sind.

Beim erstmaligen Öffnen dieses Taskbuilders sind die Listen ‘Task:’ und ‘Segment in Task:’ leer. Bevor Sie die Segmente der Liste links den Tasks zuordnen können, müssen Sie die Tasks definieren. Drücken Sie dazu die Taste ‘Tasks definieren’. Es erscheint der Dialog „Tasks festlegen“ (siehe nächste Seite).

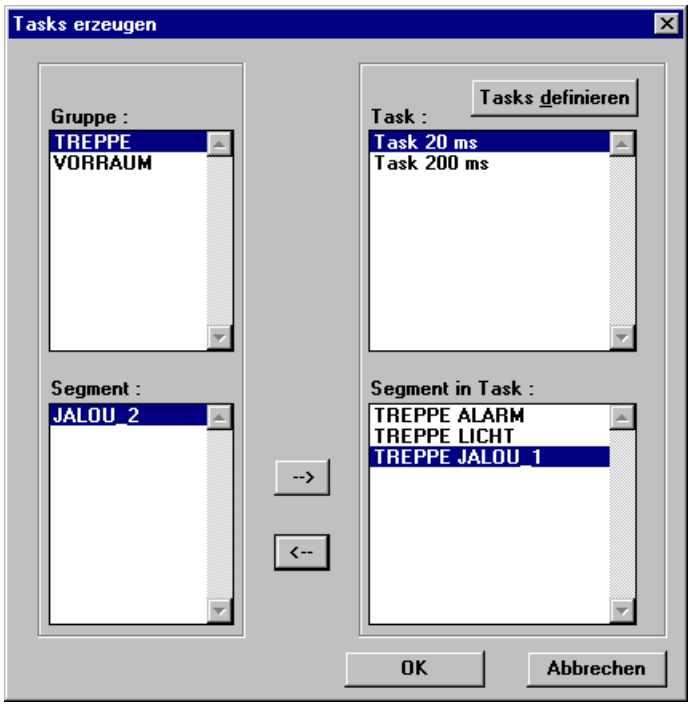


Bild 8.2.19.1.: Dialog „Tasks erzeugen“

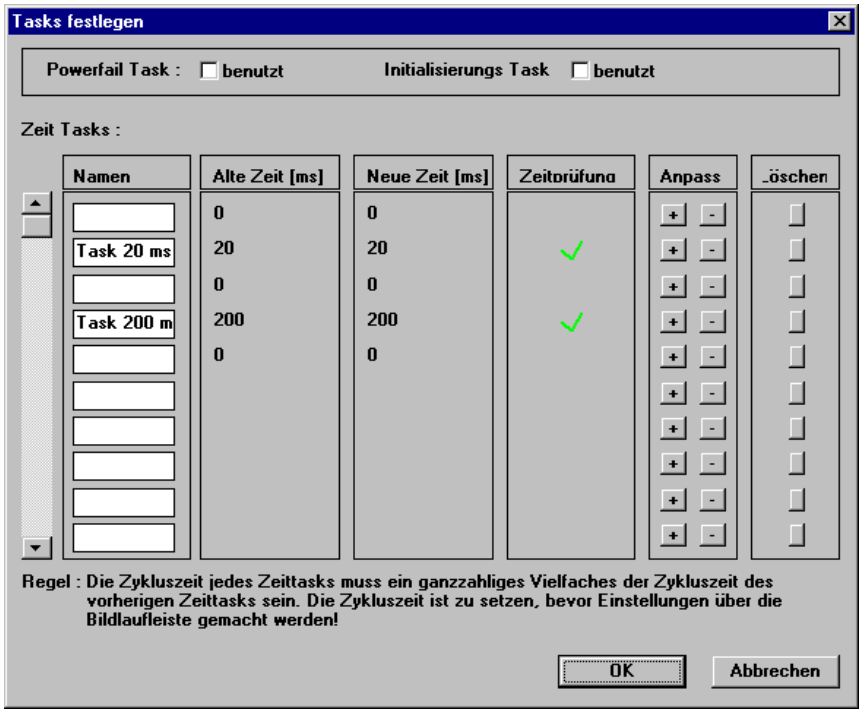


Bild 8.2.19.2.: Dialog „Tasks festlegen“

Zeittasks sind zyklische Tasks, bei denen Sie die Zykluszeit eingeben müssen. Es können mehrere dieser Tasks definiert werden.

Hinweis:
Die Liste der Zeittasks ist den Zeiten nach geordnet. Oben sind die schnelleren, unten die langsameren Tasks eingetragen. Die Zykluszeit eines jeden Tasks muss ein ganzzahliges Verhältnis zur Zykluszeit des nächst schnelleren Tasks haben.

Beim Einfügen eines neuen Zeittasks, ist zuerst der Name des Tasks in der Tabelle ‘Namen’ einzutragen. Danach stellen Sie die Zykluszeit des Tasks durch klicken auf die Taste  oder  ein. Die aktuelle Zykluszeit des Taskes wird in der Spalte ‘Neue Zeit’ eingetragen. In der Spalte ‘Zeitprüfung’ erscheint für eine gültige Zykluszeit ein grünes  Zeichen. Falls ein rotes  Zeichen erscheint, ist die Zykluszeit undültig. Sie können diesen Dialog erst dann mit der OK-Taste erfassen bzw. abschliessen, wenn alle Zeiten gültig sind. Korrigieren bei Sie bei einer ungültigen Zeit mit den Tasten – oder +. Beim nochmaligen Öffnen dieses Dialoges stehen aus rein informativen Gründen die vorgängig gewählten Zykluszeiten in der Tabelle ‘Alte Zeit’.

Sie haben zusätzlich zu den Zeittasks die Möglichkeit, einen Initialisierungs-Task und / oder einen Powerfail Task (Spannungsausfall) zu definieren.

Der **Initialisierungs-Task** wird beim Aufstarten bzw. nach dem ‘Reset’ des Funktionsmodules vor dem ersten zyklischen Task (Zeittask) einmalig ausgeführt.

Der **Powerfail Task** wird beim Ausfallen der Speisespannung vom Funktionsmodul ausgelöst.

Falls Sie einen Initialisierungs-Task oder einen Powerfail Task benötigen, markieren Sie das entsprechende Feld.

Beenden Sie nach dem Definieren der Tasks diesen Dialog mit der OK-Taste. Im Dialog „Tasks erzeugen“ sind die definierten Tasks nun mit ihrem Namen eingetragen.

Das markierte Segment in der Liste links kann mittels Taste → dem in der Liste rechts oben markierten Task zugeordnet werden. Jedes neu zugeordnete Segment wird am Schluss der Liste ‘Segment in Task’ (rechts unten) angefügt.

Die Reihenfolge der Segmente in der Liste rechts entspricht der zeitlichen Abarbeitungsreihenfolge im Funktionsmodul..

Mit der Taste ← kann ein Segment, das in der Liste rechts markiert ist, zurück nach links verschoben werden.

Beenden Sie nach dem Bestimmen der Abarbeitungsstrategie, d.h. dem Zuordnen der Segment zu den Tasks, diesen Dialog mit OK.

8.2.20. Kompiler

Unter diesem Menüpunkt können Sie das aktuell geöffnete Projekt in die für das Funktionsmodul notwendige Maschinensprache übersetzen.

Hinweis:
Bevor Sie den Kompiler mit der Taste ‘Start’ aktivieren, müssen Sie die Abarbeitungsstrategie definiert haben (Tasks definieren usw.).

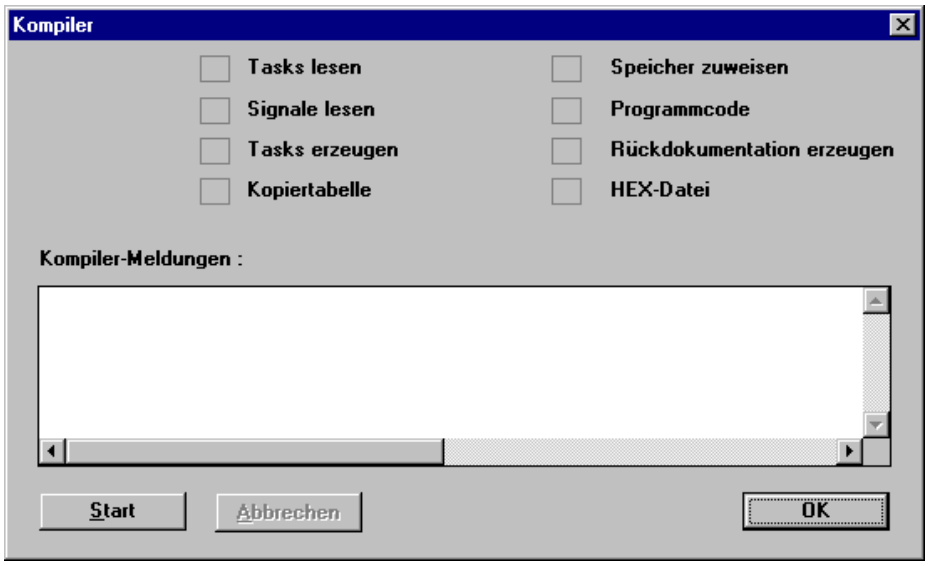


Bild 8.2.20.1.: Dialog „Kompiler“

Im oberen Teil des Dialoges sind die acht Teilschritte eines Übersetzungsvorganges dargestellt. Hat der Kompiler einen Teilschritt fehlerfrei abgearbeitet, so wird ein ✓ Zeichen eingeblendet. Beim Auftreten eines Fehlers wird statt dessen ein ✗ angezeigt.

In der Liste ‘Kompiler-Meldungen’ werden alle Meldungen des Kompilers eingetragen. Diese Liste wird beim Ausdrucken einer Projektdokumentation ebenfalls mit ausgedruckt.

Mit der Taste ‘Abbrechen’ können Sie den laufenden Übersetzungsprozess jederzeit unterbrechen. Nachdem der Kompiler mit dem Übersetzen fertig ist, können Sie diesen Dialog mit OK verlassen.

8.2.21. Service, FM Loader

Mit diesem Menüpunkt können Sie das Programm FMLoader aus FMTool heraus starten. Weitere Einzelheiten siehe Kapitel FMLoader.

8.3. Beschreibung des Grafikeditor-Menüs

Der Grafikeditor mit den nachstehend beschriebenen Menüpunkten erscheint, nachdem entweder ein neues Segment angelegt oder ein bestehendes geöffnet wurde.

8.3.1. Segment, Schliessen

Mit diesem Befehl können Sie ein geöffnetes Segment schliessen. Damit wird auch der Grafikeditor geschlossen und das Programm wechselt ins Hauptmenü. Bei allenfalls noch nicht gesicherten Änderungen im Segment, fragt FMTool, ob diese gespeichert werden sollten.

8.3.2. Segment, Speichern

Mit diesem Befehl können Sie das aktuelle Segment speichern. Dieser Befehl hat dieselbe Funktion wie der Knopf, der sich auf der Werkzeugleiste befindet.

8.3.3. Segment-Info...

Nach dem Anwählen dieses Menüpunktes erscheint der Dialog „Segment-Info“. Dieser zeigt die aktuellen Informationen der zu jedem Segment gehörenden Info-Felder (Segment-) ‘Name’, ‘Programmierer’, (Erstellungs-) ‘Datum’ und ‘Version’. Bis auf den Segment-Namen können alle Daten verändert werden. Beim Verlassen des Dialogs mit OK werden die geänderten Daten übernommen.

8.3.4. Segment, Drucken...

Unter diesem Menüpunkt können Sie das aktuelle Segment ausdrucken. Beim Anwählen erscheint ein Standard-Druckdialog. Dieser ermöglicht, den Drucker auszuwählen und einzustellen.

Nach dem Betätigen der OK-Taste wird der Druckauftrag ausgeführt.

8.3.5. Segment, Seitenansicht

Die ‘Seitenansicht’ zeigt eine Druckvorschau des Segments. Aus diesem Dialog heraus kann das Segment direkt ausgedruckt werden.

8.3.6. Segment, Importieren...

FMTool bietet die Möglichkeit, bereits einmal programmierte Segmente zu ‘vervielfältigen’ (ohne externen Signalen bereits zugeordnete Gruppenadressen und Namen). Durch Wählen von ‘Importieren...’ erscheint der Browser-Dialog „Segment importieren“.

Es kann ein Segment aus dem geöffneten Projekt, aus einer beliebigen Gruppe ausgewählt und als ganzes importiert werden. Diese Segmente können in den Listen des Browsers direkt angewählt werden.

Es ist auch möglich, ein Segment eines andern (fremden, abgelegten) Projekts zu importieren. Um das gewünschte Projekt zu suchen und anzuwählen, wird durch Betätigen der Taste ‘Suchen...’ der Dialog „Datei öffnen“ aufgerufen. Darin können Sie das gewünschte Projekt auswählen. Anschliessend erscheinen in den Listen des Browsers die Gruppen und Segmente des entsprechenden Projektes. Wie oben beschrieben markieren Sie das gesuchte Segment und bestätigen den Import mit OK.

Ist das Segment, in das Sie importieren wollen, leer, erscheint der untenstehende Dialog. Mit OK übernehmen Sie das zuvor angewählte Segment als ganzes. Externe Signale im Signaleingangs- und -ausgangsfeld werden ohne zugeordnete Namen und Gruppenadressen übernommen, d.h., sie erscheinen schraffiert.

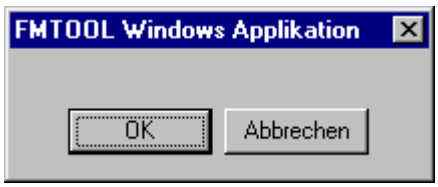


Bild 8.3.6.1.: Dialog zum Bestätigen eines Segment-Imports

Enthält das Segment, in das Sie importieren wollen, bereits Daten, werden Sie vor dem Import darauf hingewiesen, dass beim Importieren alle bestehenden Daten des Segments gelöscht werden.

8.3.7. Modus

In diesem Menüpunkt können Sie den aktuellen Betriebsmodus (Einfügen oder Löschen) einstellen. Dieselbe Funktion erfüllen die Tasten + und – in der Werkzeugleiste.

Einfügemodus Setzt den Grafikeditor in den Einfügemodus. In diesem Modus sind die Funktionen *Funktionsblock* und *Verbindungen einfügen* aktiv.

Löschmodus Setzt den Grafikeditor in den Löschmodus. In diesem Modus sind die Funktionen *Funktionsblock* und *Verbindungen löschen* aktiv. Signale können unabhängig vom aktuellen Modus eingesetzt oder gelöscht werden. Die Auswertung, einsetzen oder löschen, erfolgt anhand des aktuellen Signaleintrages. Nähere Informationen siehe unter Kapitel Signal einfügen bzw. löschen.

8.3.8. Ansicht

In diesem Menüpunkt können Sie die aktuelle Ansicht einstellen.

Bild 8.3.8.1.: Menü ‘Ansicht’ und Werkzeugleiste

Zoom 1 Normalansicht des Segmentes.

Zoom 2 Vergrösserte Ansicht des Segmentes.

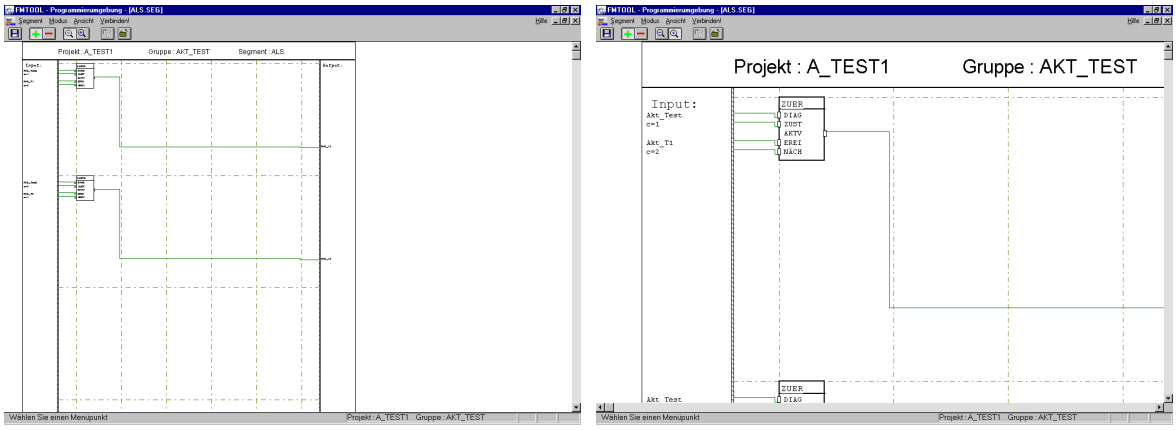


Bild 8.3.8.2.: Ansicht in Zoom 1 und Zoom 2

Die Umschaltfunktion der Ansicht wird sehr oft gebraucht. Daher gibt es zwei weitere Möglichkeiten, den Ansichtsmodus umzuschalten:

1. In der Werkzeugleiste mit den Zoomtasten

2. Mit der rechten Maustaste

   Da der Grafikeditor nur zwei Ansichtsmöglichkeiten unterscheidet, kann ein Wechsel in den anderen Modus mittels der rechten Maustaste ausgelöst wer-

den. Beim Vergrössern nimmt der Grafikeditor die Position des Mauszeigers als Zentrum für die vergrösserte Ansicht.

8.3.9. Verbinden!

Um einen Anschluss (eine Klemme) eines Funktionsblocks mit der Signaleingangs- oder -ausgangsleiste zu verbinden, können Sie mit der Maus zuerst auf die Klemme und anschliessend auf den Menüpunkt ‘Verbinden!’ klicken.

8.4. Funktionen des Grafikeditors

Bei der Entwicklung des Grafikeditors ist besonders auf die einfache und ergonomische Bedienung Gewicht gelegt worden. Die aktuell auszuführende Funktion ist im wesentlichen abhängig vom Betriebsmodus und vom Ort, auf welchem mit dem Mauszeiger etwas angeklickt wurde. Dies bedeutet, dass die „Intelligenz“ der Funktionsauswahl im Grafikeditor implementiert ist. Für den Bediener entfällt somit das mühsame und zeitraubende umschalten bzw. holen von anderen Werkzeugen, Funktionen usw. Dies fördert nicht nur die Effizienz im Zeichenprozess, sondern zwingt den Benutzer auch zu einem systematischeren und überlegten Arbeiten.

8.4.1. Funktionen verwerfen

Beim Drücken der Maustaste an einem beliebigen Ort wird die aktuell auszuführende Funktion in der Statuszeile angezeigt. Erst beim Loslassen der Taste wird diese dann ausgeführt.

Währenddem Sie die Maustaste gedrückt halten und die Funktion eingeblendet ist, können Sie diese verwerfen. Zu diesem Zweck fahren Sie mit gedrückter Maustaste zu einem anderen Funktionsblock bzw. möglichen Ort für einen Funktionsblock und lassen dann die Maustaste wieder los. In der Statuszeile wird das Verwerfen der Funktion mit der Anzeige 'Funktion verworfen' quittiert.

Bild 8.4.1.1.: Statuszeile mit Anzeige Funktionsanzeige ‘Funktionsblock einfügen’ bzw. 'Funktion verworfen’

8.4.2. Funktionsblock einfügen

Vorgehen :
- Setzen Sie den Grafikeditor in den Einfügemodus

- Klicken Sie im aktuellen Segment mit der linken Maustaste auf ein freies Feld, in das der neue Funktionsblock eingefügt werden soll.
  Beim Drücken der Maustaste erscheint in der Statuszeile 'Funktionsblock einfügen'.

Der Grafikeditor öffnet dann den Funktionsblock-Auswahldialog „Funktionsblock einfügen“.

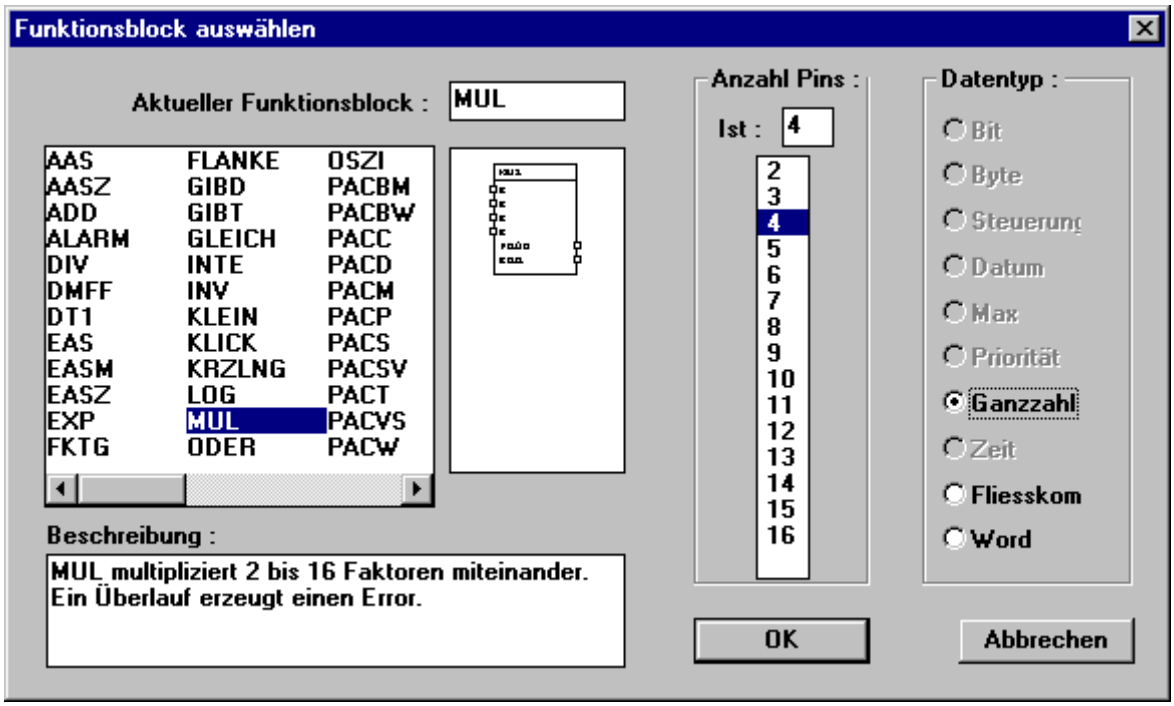


Bild 8.4.2.1.: Dialog „Funktionsblock einfügen“

Im Auswahlfeld wählen Sie den von Ihnen verlangten Funktionsblock mit der Maus oder den Cursortasten an.

Das Feld 'Beschreibung' zeigt immer eine kurze Beschreibung des aktuell markierten Funktionsblockes. Eine ausführlichere Beschreibung zu jedem Funktionsblock finden Sie im Anhang A. Der Name des entsprechenden Blocks wird zusätzlich im Feld 'Aktueller Funktionsblock' eingeblendet. Das Feld darunter zeigt die Vorschau des Funktionsblockes an.

Im Feld 'Anzahl Pins' können Sie bei gewissen Funktionsblöcken die Anzahl der Ein- bzw. Ausgänge einstellen, z.B. der logische ODER-Baustein mit variabler Anzahl Eingängen. In der zugehörenden Liste werden alle zur Auswahl stehenden Konfigurationen aufgelistet. Die mit der Maus angewählte Anzahl der Eingänge wird jeweils im Feld 'Ist' angezeigt.

Haben Sie einen Funktionsblock ausgewählt, welcher mehrere Datentypen unterstützt, so können Sie im Feld 'Datentyp' den entsprechend benötigten auswählen.

Mit dem Drücken der OK-Taste wird der aktuelle FB mit den entsprechenden Konfigurationen im Segment eingefügt.

8.4.3. Funktionsblock verschieben

Funktionsblöcke können innerhalb der Zeile (oder Reihe), in der sie sich befinden, nach rechts oder links verschoben werden. Bestehen bereits Verbindungen zum Block, bleiben diese bestehen.

Klicken Sie dazu mit der Maus unterhalb des Namensfeldes auf den Block und halten Sie die Taste gedrückt. Ziehen Sie den Block mit der Maus in das gewünschte neue Feld und lassen die Maustaste dort wieder los. (siehe Beispiel).

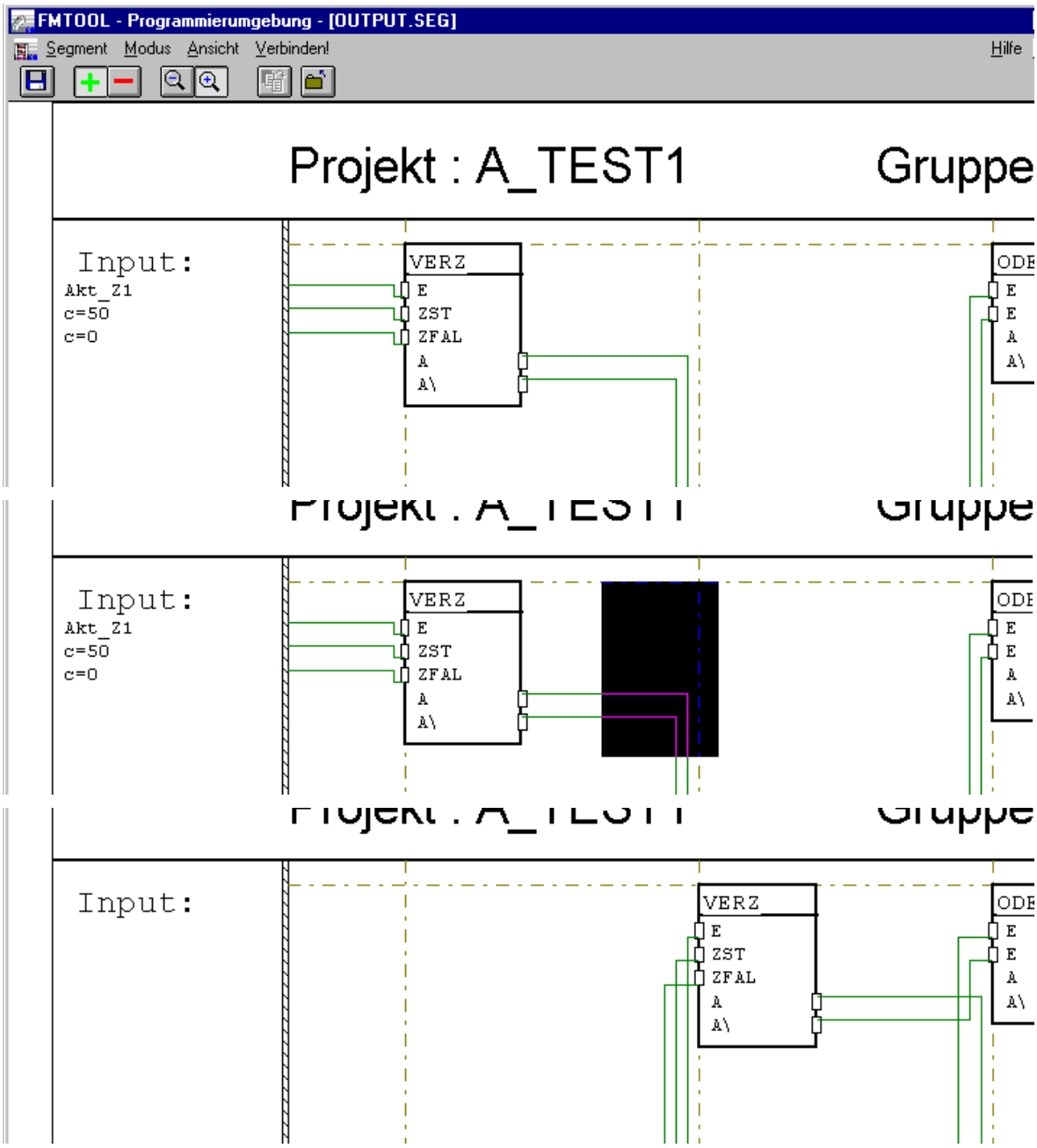


Bild 8.4.3.1.: Funktionsblock verschieben

8.4.4. Funktionsblock, Datentyp anzeigen

Sie können den Datentyp eines gesetzten Funktionsblocks in Erfahrung bringen, indem Sie mit der Maus unterhalb des Namensfeldes auf den Block klicken. Es öffnet sich der Dialog „Funktionsblock-Parameter“, der den Namen und den Datentyp des Blocks anzeigt. Mit ‘Schliessen’ verlassen Sie den Dialog wieder.

Dieser Dialog bietet zusätzlich die Möglichkeit, den Block zu löschen (Taste ‘Löschen’).

8.4.5. Funktionsblock löschen

Vorgehen :

- Setzen Sie den Grafikeditor in den Löschmodus.

- Klicken Sie in das Namenfeld (oberer Teils des Blocks) des zu löschenden Funktionsblocks.
  Beim Drücken der Maustaste erscheint in der Statuszeile 'Löscht diesen Funktionsblock'.

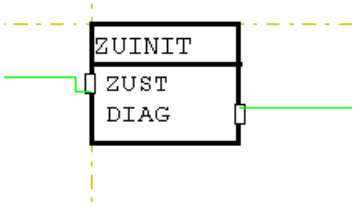


Bild 8.4.5.1.: Funktionsblock

Existieren beim angewählten Funktionsblock Verbindungen zu Signalen oder anderen Funktionsblöcken, so erscheint der folgende Dialog. Ansonsten wird der Funktionsblock kommentarlos gelöscht.

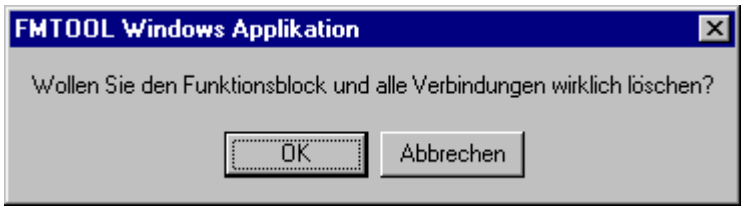


Bild 8.4.5.2.: Dialog „Wollen Sie den Funktionsblock und alle Verbindungen wirklich löschen?“

Wenn Sie den Funktionsblock inklusive seinen Verbindungen löschen wollen, betätigen Sie die OK-Taste. Der Funktionsblock wird dann mit allen Verbindungen gelöscht.

8.4.6. Verbindungen einführen

Vorgehen :
- Setzen Sie den Grafikeditor in den Einfügemodus

- Klicken Sie auf den Startpunkt bzw. den gewünschten Pin des Funktionsblocks oder auf die Signaleingangs- bzw. -ausgangsleiste.
  Beim Drücken der Maustaste erscheint in der Statuszeile ‘Funktionsblock einfügen'.

- Klicken Sie anschliessend auf den Endpunkt der gewünschten Verbindung (Pin eines Funktionsblocks oder Signalein- / ausgangsleiste).
  Beim Drücken der Maustaste erscheint in der Statuszeile 'Neue Verbindung zu Funktionsblock’.

Bild 8.4.6.1.: Segmentausschnitt mit Funktionsblöcken

Nach dem Bestimmen des Start- und Endpunktes der Verbindung, überprüft der Grafikeditor, ob diese auch wirklich gemacht werden darf. Falls die Prüfung erfolgreich abläuft, zeichnet der Autorouter die Verbindung selbsttätig.

8.4.7. Fehlermeldungen bei ungültigen Verbindungen

Bei ungültigen Verbindungen meldet der Grafikeditor folgende Fehler:

- **Verschiedene Datentypen**

  Sie versuchen, zwei Pins mit verschiedenen Datentypen miteinander zu verbinden.

  Mögliche Fehlerbehebung:

  Ist ein Funktionsblock, von dem oder zu dem die neue Verbindung hätte gehen sollen, mit einem anderen Datentyp verfügbar, sollte dieser Funktionsblock gelöscht und mit dem entsprechenden neuen Datentyp wieder eingefügt werden. Die Verbindung ist danach nochmals zu zeichnen.

  Ansonsten ist das Problem schaltungstechnisch anders zu lösen, damit diese Verbindung umgangen werden kann.

- **Gleicher Anschluss-Pin**

  Sie versuchen, zwei Eingänge oder zwei Ausgänge miteinander zu verbinden.

- **Falscher Weg des Signals**

  Sie haben versucht, eine Verbindung einzuführen, deren Signalfluss entgegengesetzt der Abarbeitungsreihenfolge (von oben links nach unten rechts; siehe Kapitel Segmentaufbau) eines Segmentes läuft.

  Mögliche Fehlerbehebung:

  Führen Sie anstatt der geplanten Verbindung ein Internes Signal ein. Verbinden Sie mit Hilfe dieses Signals die ursprünglich vorgesehenen Pins.

- **Zweite Eingangsverbindung (Zweite Verbindung muss auf einen Eingang)**

  Sie versuchen, mehrere Signale auf einen Eingang eines Funktionsblocks oder auf ein Ausgangssignal (Signalausgangsleiste) zu ziehen.

  Dies ist im Funktionsmodul „schaltungstechnisch“ nicht möglich. Es sind nur Verbindungen „1:n“ d.h., von einem Ausgangspin zu mehreren Eingängen erlaubt. Umgekehrt nicht!

- **Mischen von Signalen (Sie können lokale Signale nicht mit internen oder externen Signalen mischen. ...)**

Sie versuchen, von einem Ausgangspin aus, der schon mit einem andern Funktionsblock (FB) verbunden ist (lokales Signal), eine Signalverbindung (FB zu
Signal) zu erstellen (internes oder externes Signal). Diese verschiedenartigen Verbindungen zusammen sind nicht zugelassen.

Mögliche Fehlerbehebung:

Umgehen Sie die direkte Signalverbindung. Führen Sie dazu einen Transferblock (TRA) ein und ziehen die Signalverbindung vom Ausgang dieses Blockes. Verbinden Sie dann den ursprünglichen Ausgang mit dem Transferblock (siehe Zeichnung).

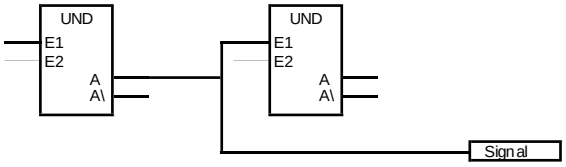


Bild 8.4.7.1.: Signalmix-Fehlersituation

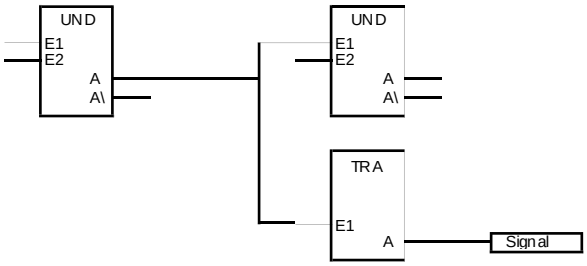


Bild 8.4.7.2.: Lösung des Signalmixes

- **Zuwenig Platz, um Verbindungen zu zeichnen.**

  Diese Meldung erscheint, wenn der Autorouter beim Zeichnen ein unlösbares Platzproblem erkennt.

  Die geplante Verbindung ist bezüglich Signalfluss und Datentyp wohl korrekt, aber der Platz zum Durchziehen der Verbindung zwischen den Funktionsblöcken ist an einer Stelle nicht ausreichend:

  Mögliche Fehlerbehandlungen:

  Versuchen Sie durch Verschieben von Funktionsblöcken den Engpass zu lösen.

  Splitten Sie die Schaltung auf mehrere Zeilen (Reihen von nebeneinander liegenden Funktionsblöcke) oder ggf. auf mehrere Segmente auf. Dadurch können Sie dem Router mehr Platz schaffen (siehe Zeichnung).

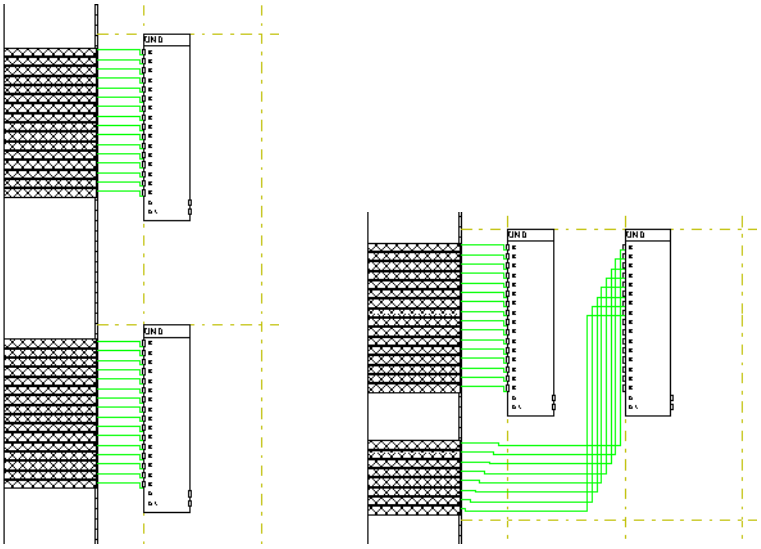


Bild 8.4.7.3.: Lösungsvariante für ‘Zuwenig Platz, Verbindungen zu zeichnen’

8.4.8. Verbindungen zu den Signalleisten

Nebst den oben beschriebenen Möglichkeiten, Verbindungen vorzunehmen, haben Sie für Verbindungen von einem Funktionsblock zur Signaleingangs- bzw. -ausgangsleiste (Signalverbindungen) weitere Möglichkeiten:

1. Sie machen auf den Pin, den Sie mit einer der Signalleisten verbinden wollen, mit der linken Maustaste einen Doppelklick.
2. Sie klicken den gewünschten Pin mit der Maus an und klicken anschliessend auf den Menüpunkt ‘Verbinden!’
3. Sie klicken den gewünschten Pin mit der Maus an und verwenden danach den „Shortcut“ Alt-V.

8.4.9. Verbindungen löschen

Vorgehen :
- Setzen Sie den Grafikeditor in den Löschmodus

- Klicken Sie die zu löschende Verbindung beim Eingangs- oder Ausgangspin an. Die Verbindung wird dann kommentarlos gelöscht.
  Beim Drücken der Maustaste erscheint in der Statuszeile 'Löscht diese Verbindung zum Pin'.

Bemerkung: Wird ein Ausgangspin einer Verbindung angeklickt, so werden alle Verbindungen, welche von diesem Pin ausgehen, gelöscht. Beim Anklicken eines Eingangspins wird nur die eine Verbindung, welche zu diesem führt, gelöscht.

8.4.10. Signal einführen

Vorgehen :
- Es besteht eine Verbindung von einem Funktionsblock zur Signaleingangsleiste bzw. Signalausgangsleiste. Das noch nicht definierte Signal wird als schraffiertes Feld angezeigt (siehe Bild).

- Klicken Sie auf das schraffierte Feld.
  Beim Drücken der Maustaste erscheint in der Statuszeile die Anzeige 'Einfügen / löschen / verschieben eines Signaleingangs' bzw. '.... eines Signalausgangs'.

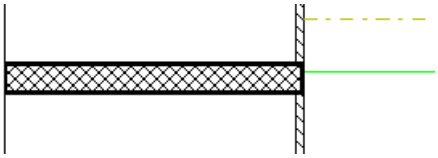


Bild 8.4.10.1.: undefinierter Signalanschluss

Bei einem Eingangssignal haben Sie grundsätzlich die Möglichkeit, eine Konstante oder ein Signal anzuschliessen. Der Grafikeditor öffnet im Falle eines Eingangs den Dialog „Funktionsblock-Parameter“.

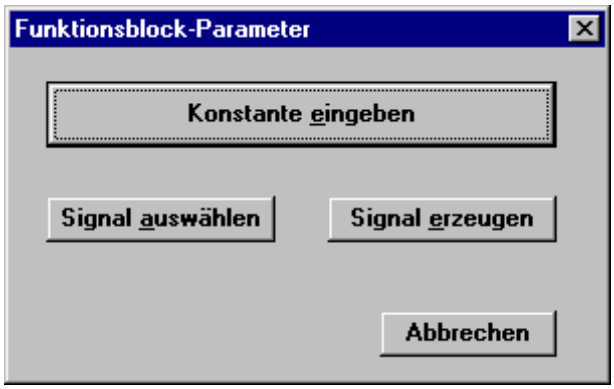


Bild 8.4.10.2.: Dialog „Funktionsblock-Parameter“

Sie haben hier verschiedene Möglichkeiten zum weiterfahren:
- Mit ‘Konstante eingeben definieren Sie das eingeführte Signal als Konstante.
- Existiert im Projekt bereits ein Signal, das Sie an diesem Pin anschliessen wollen, können Sie mit ‘Signal auswählen’ fortfahren.
- Wollen Sie ein neues Signal definieren, betätigen Sie die Taste ‘Signal erzeugen’.

In den nächsten Kapiteln wird auf die verschiedenen Möglichkeiten eingegangen.

8.4.11. Konstante eingeben

Vorgehen :
- Es besteht eine Verbindung von einem Funktionsblock zur Signaleingangsleiste. Das noch nicht definierte Konstante wird als schraffiertes Feld (siehe Bild) angezeigt.

- Klicken Sie auf das schraffierte Feld.
  Beim Drücken der Maustaste erscheint in der Statuszeile 'Einfügen / löschen / verschieben eines Signaleingangs’.

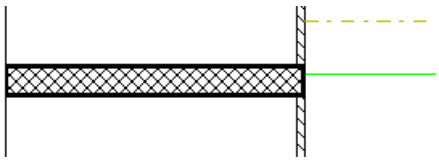


Bild 8.4.11.1.: undefinierter Signalanschluss

Der Grafikeditor öffnet den Dialog „Funktionsblock-Parameter“ (siehe oben). Betätigen Sie die Taste ‘Konstante eingeben’.

Abhängig vom Datentyp des entsprechenden Anschlusspunktes, wird einer der nachfolgend aufgezählten Dialoge geöffnet:

- Bit
- Byte
- Steuerung
- Datum
- Max
- Priorität
- Ganzzahl mit Vorzeichen
- Zeit
- Wert (Fliesskommazahl)
- Word

Die Dialoge sind unterschiedlich und enthalten den Datentypen entsprechende Eingabefelder, in die Sie die gewünschten Werte für die Konstanten eingeben können. Nachfolgend sind einige Dialoge als Beispiel angefügt:

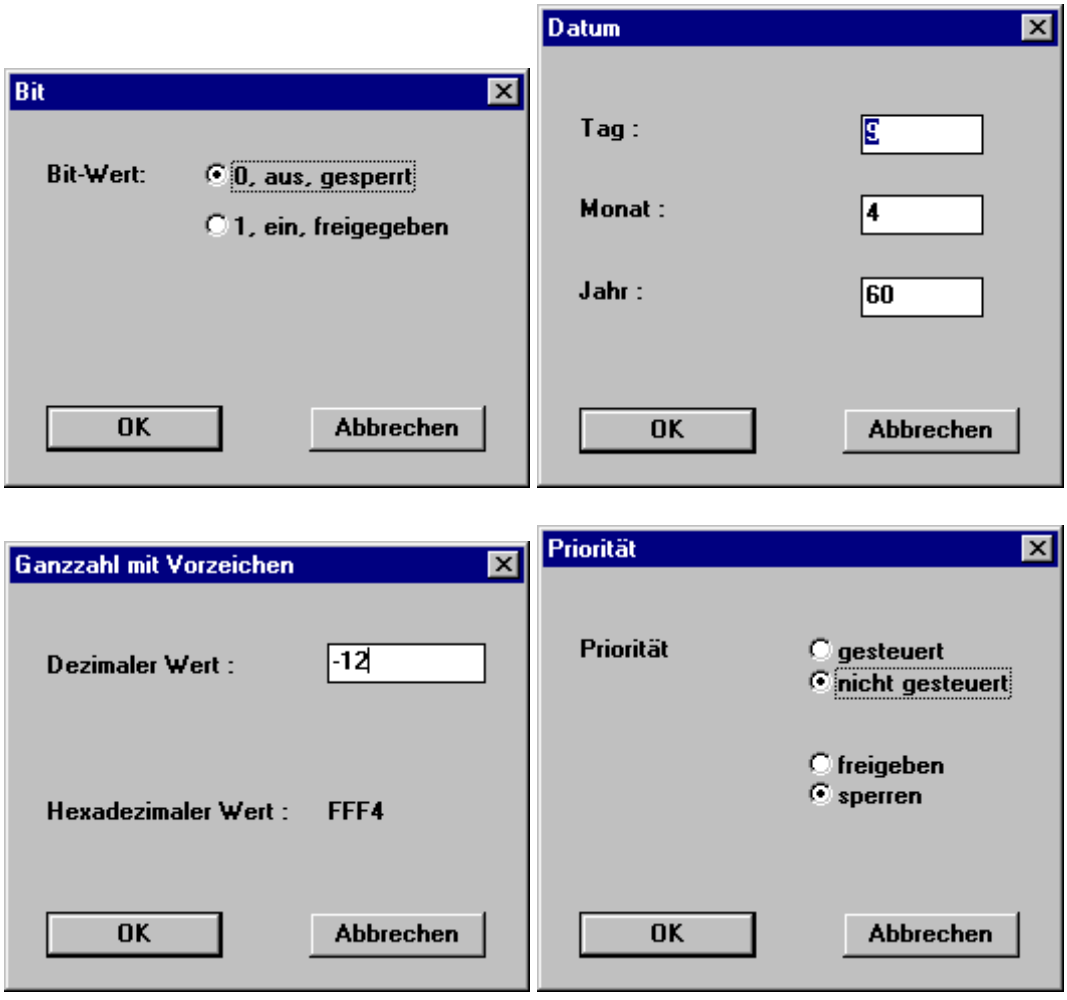


Bild 8.4.11.2.: Dialoge „Konstante eingeben“ für verschiedene Datentypen

8.4.12. Konstante ändern / löschen

Vorgehen :

- Es besteht eine Verbindung von einem Funktionsblock zur Signaleingangsleiste. Die Konstante ist auf einen bestimmten Wert definiert (siehe Bild).

- Klicken Sie auf die Konstante.
  Beim Drücken der Maustaste erscheint in der Statuszeile 'Einfügen / löschen / verschieben eines Signaleingangs'

Bild 8.4.12.1.: definierte Konstante

Nach dem Anklicken der Konstante erscheint der folgende Dialog:

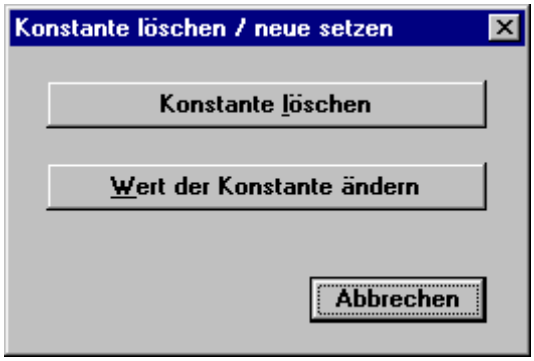


Bild 8.4.12.2.: Dialog „Konstante löschen / neue setzen“

Konstante löschen:
Beim Anwählen dieses Knopfes wird die Konstante kommentarlos gelöscht.

Wert der Konstante ändern:
Nach dem Anwählen dieses Knopfes wird der Dialog zum Eingeben einer Konstante geöffnet, entsprechend dem Datentyp des Anschlusspunktes (siehe Kapitel ‘Konstante eingeben’).

8.4.13. Signal auswählen

In diesem Dialog bietet Ihnen das System eine Auswahl von Signalen an. Die aufgelisteten Signale sind vorselektiert d.h., es werden nur mögliche Signale bezüglich Datentyp, Eingang oder Ausgang usw. angezeigt.

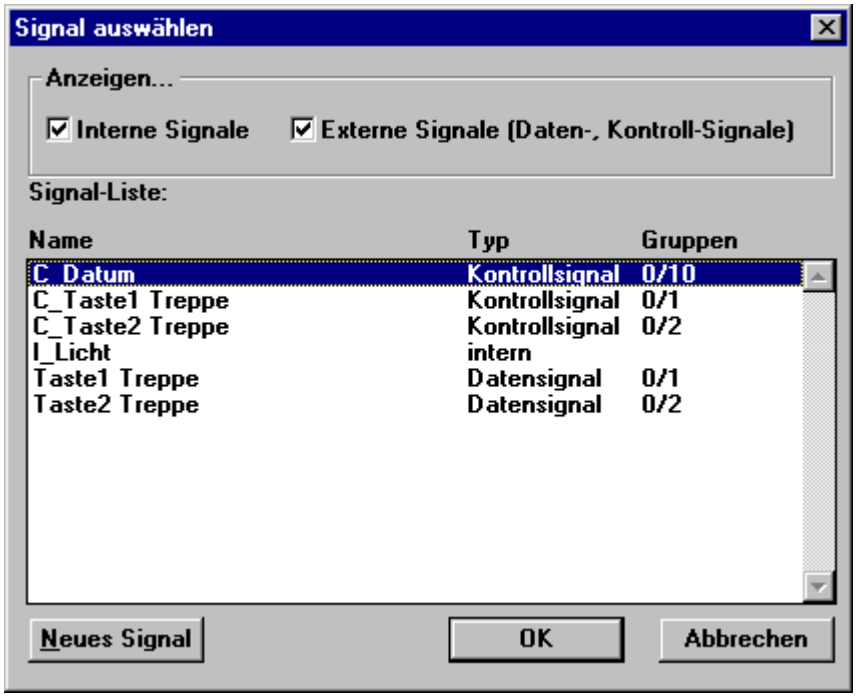


Bild 8.4.13.1.: Dialog „Signal auswählen“

Im Feld 'Anzeigen’ können Sie auswählen, ob externe Signale und / oder interne Signale aufgelistet werden sollen.

Selektieren Sie das gewünschte Signal in der Liste und schliessen den Dialog mit OK ab.

Wollen Sie ein neues Signal definieren, so klicken Sie auf die Taste 'Neues Signal'. Der Grafikeditor öffnet daraufhin dann den Dialog 'Signal erzeugen'.

8.4.14. Signal erzeugen

Dieser Dialog gibt Ihnen die Möglichkeit, ein neues Signal zu erzeugen. Sie können hier sowohl interne als auch externe Signale generieren.

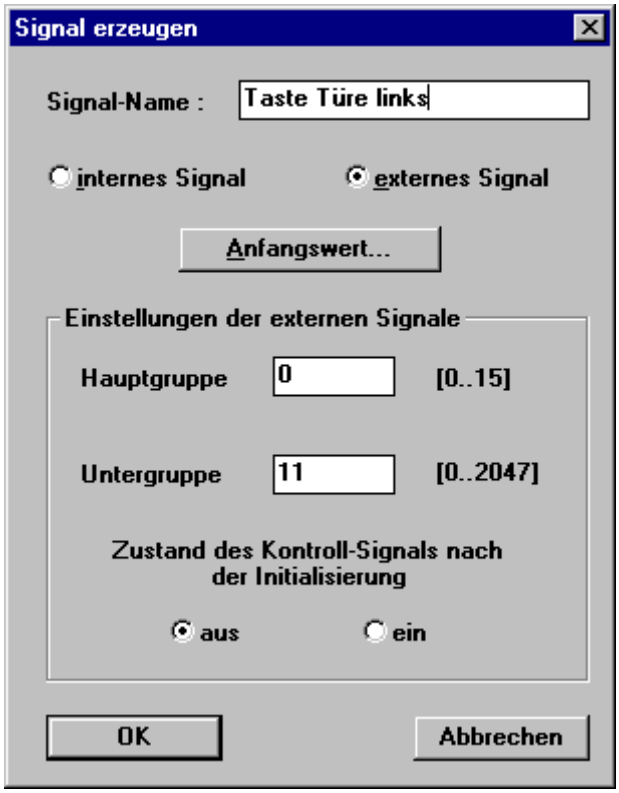


Bild 8.4.14.1.: Dialog „Signal erzeugen“

internes Signal erstellen:
Wählen Sie den Auswahlknopf ‘internes Signal’. Tragen Sie den gewünschten Namen in das Feld 'Signal-Name' ein.

externes Signal erstellen:
Wählen Sie den Auswahlknopf externes Signal’. Den gewünschten Namen des Signals setzen Sie in das Feld 'Signal-Name'.

Bitte Beachten Sie:

Beim Erstellen eines externen Signals werden immer zwei Teilsignale erstellt (siehe dazu auch Kapitel ‘Signale’).

Das Kontrollsignal (Update-Flag) ist das eine Signal. Dieses wird gebraucht, um einen Telegrammempfang oder eine Sendeaufforderung für ein Telegramm zu markieren. Das Kontrollsignal ist vom Datentyp BIT und durch die Namenserweiterung "C_" eindeutig vom Datensignal unterscheidbar. Zudem

haben Sie die Möglichkeit, den Zustand des Kontrollsignals nach der Initialisierung festzulegen. Dafür dienen die Auswahlknöpfe LOW und HIGH.

Beim Datensignal werden keine Namenserweiterungen gemacht. Dieses erscheint exakt mit dem Namen, den Sie eingegeben haben. Das Datensignal erhält automatisch den Datentyp des Signalpins (an einem Funktionsblock), mit dem es verbunden ist.

Die Gruppenadresse ist in den Eingabefeldern 'Hauptgruppe', ggf. 'Mittelgruppe' und 'Untergruppe' einzugeben.

Anfangswert:
Drücken Sie dazu den Knopf 'Anfangswert...'. Wie beim Festlegen der Konstanten wird ein dem Datentyp entsprechender Dialog zum Festlegen des Anfangswertes des Signals geöffnet.

Durch beenden mit OK wird das neue Signal in die Projekt-Datenbank aufgenommen.

8.4.15. Signal ändern / Signal-Informationen

Vorgehen :
- Es besteht eine Verbindung von einem Funktionsblock zur Signaleingangsleiste bzw. Signalausgangsleiste. Das Signal ist mit dem Namen angezeigt (siehe Bild).

- Klicken Sie auf den Signalnamen.
  Beim Drücken der Maustaste erscheint in der Statuszeile 'Einfügen / löschen / verschieben eines Signaleingangs'.

Bild 8.4.15.1.: definiertes Signal

Der Grafikeditor öffnet den Dialog „Signal löschen / umbenennen“ (siehe Bild). Im oberen Teil des Dialoges werden die Signalattribute angezeigt. Dabei erscheint zu oberst der Name des Signals. Der Signaltyp gibt Aufschluss darüber, ob es sich um ein internes Signal, ein externes Eingangs- oder Ausgangssignal handelt. Bei externen Signalen wird auch die Gruppenadresse eingeblendet. Unter 'Verwendet in Segment:' werden alle Segmente aufgelistet, die das Signal enthalten.

Bild 8.4.15.2.: Dialog „Signal löschen / umbenennen“

Signal wechseln:

Das Betätigen von ‘Wechseln’ bewirkt ein Öffnen des Dialogs „Signal auswählen“. Dort haben Sie die Möglichkeit, ein anderes bestehendes oder ein neues Signal anstelle des vorher bestandenen anzuwählen.

Signal umbenennen:

Mit dem Betätigen des Knopfs ‘Umbenennen’ öffnet sich

- bei einem internen Signal der Dialog „Internes Signal umbenennen“.
  Sie können dabei den Signalnamen und den Anfangswert ändern.

  Bitte beachten Sie:

  Ein internes Signal kann nicht in ein externes Signal umgewandelt werden. Wird dies jedoch trotzdem erwünscht, ist ‘Signal trennen’ vorzunehmen und anschliessend ein anderes Signal auszuwählen oder neu zu definieren.

- bei einem externen Signal der Dialog „Externes Signal umbenennen“.
  Dabei haben Sie die Möglichkeit, in den jeweiligen Feldern den Signalnamen, die Gruppenadresse, den Anfangswert des Signals und den Zustand des Kontroll-Signals nach der Initialisierung zu ändern.

  Bitte beachten Sie:

  Ein externes Signal kann nicht in ein internes Signal umgewandelt werden. Wird dies jedoch trotzdem erwünscht, ist ‘Signal trennen’ vorzunehmen und anschliessend ein anderes Signal auszuwählen oder neu zu definieren.

Durch das Beenden mit der OK-Taste werden die Änderungen übernommen.

Signal trennen:
Betätigen des Knopfs ‘Trennen’ bewirkt, dass das an der Signalleiste angewählte Signal wieder in den undefinierten Zustand zurückgesetzt wird. Die Signaldaten bleiben (in der Projekt-Datenbank) erhalten und stehen im Dialog „Signal auswählen“ wieder bzw. weiterhin zur Verfügung.

Signal entfernen:
Der Knopf ‘Entfernen’ bewirkt das Entfernen bzw. Löschen des Signals aus der Projekt-Datenbank. Die Funktion Signal entfernen ist nur aktiv, wenn das Signal beim letzten Verwendungsort aller Segmente angewählt wurde.

9. FMLoader

9.1. Einleitung

FMLoader ist ein Service-Programm zum Funktionsmodul. Es ermöglicht das Laden der Applikationen (auch Anwendungsprogramme oder Funktionsmodul-Projekte genannt) in das Funktionsmodul.

Ebenso kann damit auch ein neues Betriebssystem (OS) mit der zugehörenden Funktionsblockbibliothek oder ein neuer Betriebssystem-Nukleus (OS-Nukleus) ins Gerät geladen werden. Somit kann der Anwender eigenständig neue Versionen der Firmware in das Funktionsmodul laden.

Die Verbindung vom PC zum Funktionsmodul kann auf zwei verschiedene Arten erfolgen:

- über den EIB Bus, d.h., von einer seriellen Schnittstelle des PC über eine Datenschnittstelle des EIB und von dort mittels EIB zum Funktionsmodul
- vom PC direkt zum Funktionsmodul. In diesem Fall wird eine serielle Schnittstelle des PC über ein 9-poliges Datenkabel (1:1 Verbindung) direkt mit der Schnittstelle am Funktionsmodul verbunden.
  Diese Verbindungsart zum Funktionsmodul ist ca. 15 mal schneller als die andere, d.h., eine Applikation kann auf diese Art 15 mal schneller ins Funktionsmodul geladen werden.

Es bestehen zwei Möglichkeiten zum Laden eigener Applikationen in das Gerät.

- Wird das Serviceprogramm FMLoader aus FMTool heraus aufgerufen, so übergibt FMTool die Applikation, d.h., das aktuell geöffnete Projekt an FMLoader.
- Beim direkten Starten von FMLoader vom Windows-Menü bzw. dem Programm-Manager aus, fragt FMLoader nach der Applikationsdatei, die in das Funktionsmodul geladen werden soll.

Diese Datei, die vom FMTool-Kompiler in das entsprechende Hauptverzeichniss eines Projektes geschrieben wird, trägt die Namenserweiterung ‘.HEX’. Sie beinhaltet das lauffähige Programm für das Funktionsmodul.

Weiter kann bei einem im Normalbetrieb laufenden Funktionsmodul die Version des Betriebssystems, der Name der Applikation usw. abgefragt werden.

Alle beim Betrieb des Funktionsmoduls anfallenden Störungen (Fehlermeldungen) werden im Gerät selbst registriert und mit der Fehler-LED angezeigt. Mittels FMLoader können diese registrierten Meldungen bzw. die aufgetretenen Störungen nacheinander alle abgefragt werden.

Zudem kann mit dem FMLoader ein beliebiges Funktionsmodul via EIB neu gestartet werden (Reset).

9.2. Hauptmenü im Überblick

Bild 9.2.0.1.: Menüzeile von FMLoader

Datei: Beenden
Beendet das Programm FMLoader und schliesst das entsprechende Fenster

Laden in FM: Applikation
Lädt eine Applikationsdatei in das Funktionsmodul.

Betriebssystem (OS)
Lädt eine Betriebssystem-Datei in das Funktionsmodul.

OS Nukleus
Lädt eine Betriebssystem-Nukleus-Datei in das Funktionsmodul.

Version anzeigen!
zeigt Versionen und Namen von Firm- und Software, die sich im Funktionsmodul befinden
- Version Betriebssystem (OS)
- Version Funktionsblock-Bibliothek (FBL)
- Name der Applikation (Anwendungsprogramm)
- FMTool-Version, mit der die Applikation erstellt wurde
- die zum Ausführen der Applikation erforderliche OS-Version
- die zum Ausführen der Applikation erforderliche FBL-Version
- den aktuellem Betriebszustand des Funktionsmoduls

Reset FM ermöglicht das Neustarten (‘Reseten’) eines Funktionsmodules

Fehlermeldungen anzeigen!
zeigt von der nächst aktuellen registrierten Störung folgende Punkte an:
- fortlaufende Nummer aller Störungen seit dem letzten ‘Reset’
- Indentifikations-Nr. mit Fehlerbeschreibung
- Parameter 1
- Parameter 2
- Zeitpunkt der Störung, ab letztem ‘Reset’
- Angaben zum Ort der Störung Ort im Applikationsprogramm

Kommunikations-Einstellungen

Setzt die Parameter für die Datenübertragung
- physikalische Adresse der Busankopplung des Funktionsmoduls,
- Verbindungsart zum Funktionsmodul (EIB Bus oder lokal),
- Portnummer der seriellen Schnittstelle am PC (COM1 oder COM2)

9.3. Programm Beenden

Beim Anwählen dieses Menüpunktes wird das Programm FMLoader umgehend verlassen.

9.4. Kommunikations-Einstellungen

Durch Anwählen von ‘Kommunikations-Einstellungen’ können Sie die für die Verbindung zum Funktionsmodul notwendigen Parameter einstellen.

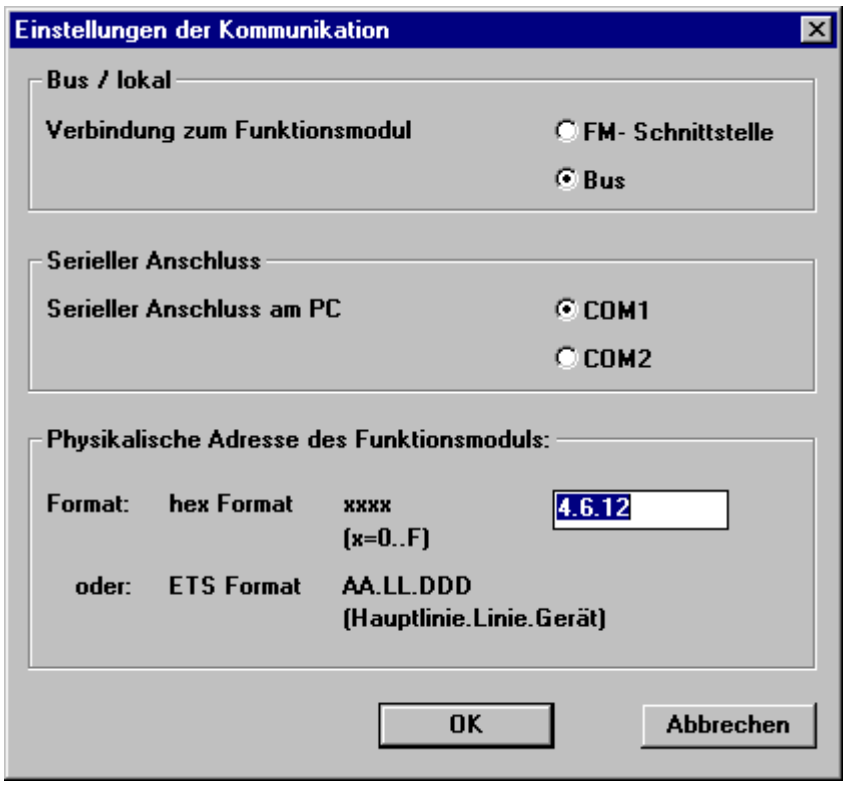


Bild 9.4.0.1.: Dialog „Einstellungen der Kommunikation“

Unter ‘Bus / lokal’ wird die Übertragungsart zum Funktionsmodul ausgewählt. Entweder kann die Verbindung via EIB über eine EIB Datenschnittstelle oder über die lokale Schnittstelle am Funktionsmodul erfolgen.

Im Feld 'Serieller Anschluss' können Sie die serielle Schnittstelle des Computers auswählen, möglich sind die Schnittstellen ‘COM 1’ und ‘COM 2’.
Im Eingabefeld 'Physikalische Adresse des Funktionsmoduls’ muss die physikalische Adresse des Funktionsmoduls im ETS-Format oder in hexadezimaler Schreibweise eingegeben werden.

9.5. Applikation laden

Unter diesem Menüpunkt können Sie eine Applikation in das Funktionsmodul laden.

FMLoader unterscheidet, ob Sie das Programm aus FMTool heraus aufgerufen haben oder ob es direkt vom Windows-Menü bzw. dem Programm-Manager aus gestartet wurde. Im Zweiten Fall kennt FMLoader die zu ladende Applikationsdatei nicht von selbst. Daher erscheint der nachfolgende Dialog um eine Datei anzuwählen.

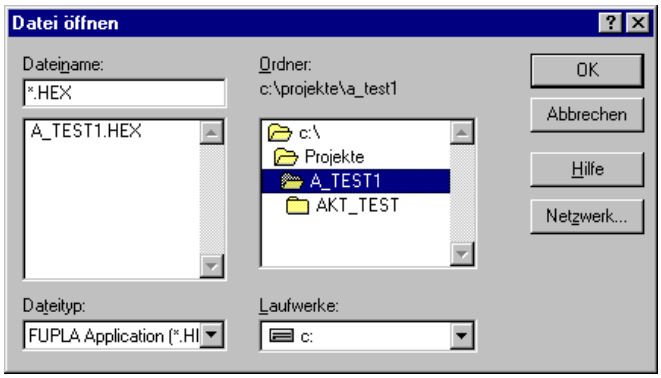


Bild 9.5.0.1.: Dialog „Datei öffnen“

Nach dem Öffnen der Datei erscheint der Dialog „Applikation in FM laden“.

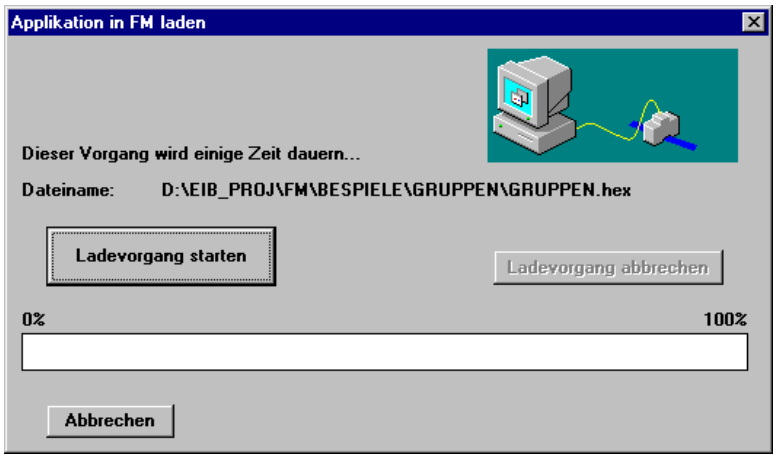


Bild 9.5.0.2.: Dialog „Applikation in FM laden“

Durch ‘Ladevorgang starten’ beginnt FMLoader mit dem Laden. Der Ballken zwischen 0% und 100% zeigt den Stand des Vorgangs laufend an. Mit der Taste 'Ladevorgang abbrechen’ kann dieser auch abgebrochen werden. Nach dem Abbrechen ist jedoch keine gültige Applikation im Funktionsmodul.

Nach dem ordentlichen Abschluss, d.h., dem vollständigen Laden erscheint der Dialog mit der Meldung:

Applikation erfolgreich in das FM geladen!

Bitte führen Sie einen ‘Reset’ am FM aus, um die Applikation zu starten.

Diese ist mit OK zu bestätigen.

Führen Sie einen Reset aus, entweder direkt am Gerät mit der Reset-Taste oder von FMLoader aus, durch Anwählen des Menüpunktes ‘Reset FM’.

9.6. Betriebssystem (OS) laden

Nachdem Sie dazu die Menüpunkte ‘Laden in FM’ und ‘Betriebssystem (OS)’ angewählt haben, erfolgt alles weitere grundsätzlich gleich wie eine ‘Applikation laden’.

FMLoader überprüft vor dem Laden, ob Sie bei der Dateiauswahl wirklich eine Betriebssystem-Datei angegeben haben. Sollte dies nicht der Fall sein, erscheint eine Fehlermeldung.

Sollte der Ladevorgang abgebrochen werden, bleibt weiterhin das bisherige Betriebssystem funktionstüchtig im FM.

Hinweis: Ganz am Schluss des Vorgangs, wenn der Balken 100% anzeigt, leuchten für einige Sekunden alle 3 LED am Funktionsmodul. Dabei wird das beim Ladevorgang ins RAM geschriebene neue Betriebssystem ins EPROM kopiert. Sollte es während dieser Operation beim FM zu einem Spannungsunterbruch kommen, verfügt das FM daraufhin über kein funktionstüchtiges Betriebssystem, d.h., es ist betriebsuntauglich. In diesem Fall ist im Notladebetrieb das Betriebssystem erneut ins FM zu laden (siehe Kapitel ‘Notladebetrieb’).

9.7. Notladebetrieb

Neben dem normalen Betriebssystem (OS) gibt es im Funktionsmodul einen Betriebsystem-Nukleus (OS-Nukleus). Dieser hat gegenüber dem normalen Betriebssystem eine eingeschränkte Funktionalität und kann nur für das Laden von Applikationen oder Betriebssystemen verwendet werden. Ist der OS-Nukleus in Betrieb, wird keine Applikation gestartet.

Der Notladebetrieb kann erforderlich sein, wenn durch eine fehlerhafte Applikation das Funktionsmodul mit FMLoader nicht mehr angesprochen werden kann und das normale Laden einer neuen Applikation unmöglich ist. Der OS-Nukleus startet die fehlerhafte Applikation nicht. In diesem Fall wird der OS-Nukleus für den Notladebetrieb genutzt.

Ebenso kann dieser Betrieb im Falle eines nicht (mehr) funktionstüchtigen Betriebssystems zum Laden eines neuen Betriebssystems verwendet werden.

Für den Notladebetrieb ist der OS-Nukleus von Hand zu starten. Dazu muss während einem Reset der Notladebetriebs-Knopf (untere Taste am FM) gedrückt werden. Anschliessend ist mit FMLoader zum Laden einer Applikation oder eines Betriebssystems wie gewohnt weiterzufahren.

9.8. OS-Nukleus (Betriebssystem-Nukleus) laden

Das für den Notladebetrieb im Funktionsmodul vorhandene Programm ‘Betriebssystem-Nukleus’, oder auch ‘OS-Nukleus’ genannt, kann wie das Betriebssystem selbst ‘upgedatet’ werden. Dazu sind die Menüpunkte ‘Laden in FM’ und ‘OS-Nukleus’ anzuwählen. Anschliessend können Sie gleich wie bei einer ‘Applikation laden’ bzw. bei ‘Betriebssystem laden’ weiterfahren.

9.9. Versionsabfrage (Version anzeigen)

Unter diesem Menüpunkt können Sie die Versionsdaten von Betriebssystem und Applikation abfragen. Weiter wird ebenfalls der aktuelle Betriebsmodus des Funktionsmodul gemeldet.

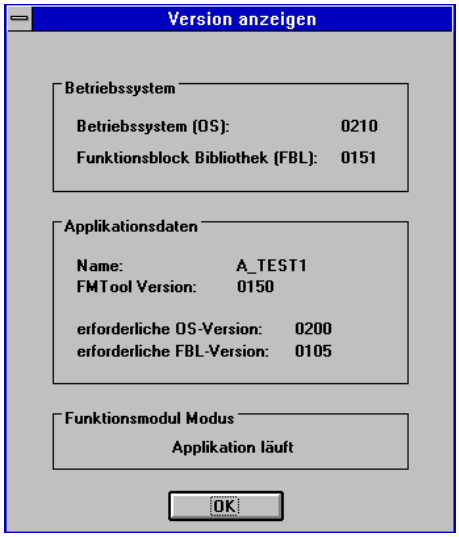


Bild 9.9.0.1.: Dialog „Version anzeigen“

Im Feld 'Betriebssystem' werden die Versionsnummer des Betriebssystemes (OS) und der Funktionsblockbibliothek (FBL) als Zahl mit vier Ziffern angezeigt (z.B. 0210 bedeutet Version 2.10).

Unter 'Applikationsdaten' ist
- der Name der aktuell geladenen Applikation,
- die Versionsnummer von FMTool, mit der sie erstellt wurde,
- die zum Betrieb der Applikation im FM benötigte Version des OS (minimal erforderliche Version oder höhere),
- die zum Betrieb der Applikation im FM benötigte Version der FBL (minimal erforderliche Version oder höhere)

angezeigt.

Folgende Betriebsmodi können unter 'Funktionsmodul-Modus' aufgeführt werden:

Applikation läuft: Das Funktionsmodul befindet sich in diesem Betriebszustand, nachdem es mit einer gültigen Applikation gestartet wurde. Die Applikation läuft.

Applikation in FM laden:

Das Funktionsmodul befindet sich in diesem Betriebsmodus, nachdem eine Applikation geladen aber noch nicht gestartet wurde. Wenn sich im Funktionsmodul eine ungültige Applikation befindet, ist ebenfalls dieser Betriebsmodus aktiv.

Betriebssystem (OS) in FM laden:

Das Funktionsmodul befindet sich in diesem Betriebszustand, nachdem ein Betriebssystem (OS) geladen wurde. Falls das Funktionsmodul mit dem Notladebetrieb (OS-Nukleus) gestartet wurde, so ist automatisch dieser Modus aktiv.

9.10. Funktionsmodul starten / rücksetzen (‘Reset’ FM)

Sie können ein Funktionsmodul von FMLoader aus neu starten, d.h., einen ‘Reset’ am FM ausführen.

Nach dem Anwählen des Menüpunktes ‘Reset FM’ erscheint der Dialog „Reset FM“, der mit OK zu bestätigen ist.

Bild 9.10.0.1.: Dialog „Reset FM“

Ein ausgeführter ‘Reset’ wird wieder am Bildschirm gemeldet, genauso wie wenn der Befehl nicht ausgeführt werden konnte.

9.11. Störungsabfrage (Fehlermeldungen anzeigen)

Unter diesem Menüpunkt können Sie beim Betrieb des Funktionsmoduls aufgetretene Störungen bzw. Fehler abfragen.

Alle Störungen die im Funktionsmodul während dem Betrieb auftreten, werden nacheinander in einer FIFO (first in first out) gespeichert. Somit erscheinen die Fehlermeldungen beim Abfragen in der Reihenfolge des Auftretens.

Mit der ersten Ausnahmemeldung wird am Funktionsmodul die Fehler-LED (unterste LED) eingeschaltet. Nach Abfrage aller Fehlermeldungen oder nach einem ‘Reset’ schaltet die LED wieder aus.

Nach einer Abfrage erscheint der Dialog „Fehlermeldung anzeigen“.

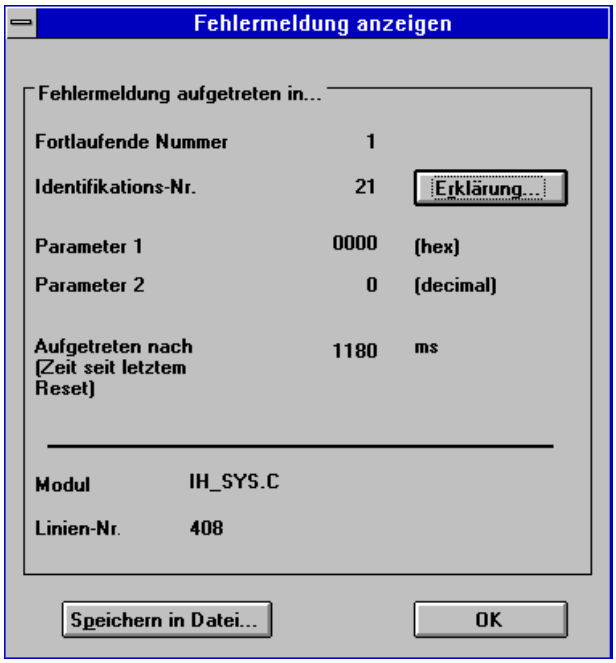


Bild 9.11.0.1.: Dialog „Fehlermeldung anzeigen“

Folgende Informationen sind im Dialog aufgelistet:

Fortlaufende Nummer	jede Fehlermeldung erhält eine Nummer, die erste 1, danach fortlaufend 2, 3 usw.
Identifikations-Nr.	gibt den Fehlertyp an; mit dem Knopf ‘Erklärung’ erscheinen die dazugehörenden weiteren Informationen (Beschreibung des Fehlers, Beschreibung der Werte von Parameter 1 und 2, Kommentar zum Fehler und Angaben über mögliche Korrekturen
Parameter 1 und 2	dienen der genaueren Umschreibung des Fehlers (weitere Informationen nach Betätigen von ‘Erklärung’
Aufgetreten nach (Zeit seit letztem Reset)	vergangene Zeit in Millisekunden seit den letzten Neustart (‘Reset’) des Funktionsmoduls bis zum Auftreten der Störung
Modul	umschreibt die Programmcode-Sequenz, in welcher der Fehler aufgetreten ist
Linien-Nr.	zeigt die Programmcode-Zeile des aufgetretenen Fehlers

Die aufgetretenen Fehlermeldungen lassen sich in einer Datei speichern. Betätigen Sie dazu den Knopf ‘Speichern in Datei...’. Sie können dann der Datei den von Ihnen gewünschten Namen geben und den Pfad, in dem die Datei gespeichert werden soll, festlegen.

Die verschiedenen Fehlertypen sind auch im Anhang B zu diesem Handbuch zusammengestellt.

10. Literaturverzeichnis

[1] u.Tietze, Ch. Schenk Halbleiter Schaltungstechnik, 10. Auflage, Springer Verlag, Berlin, ISBN3-540-56184-6, 1993

Technische Dokumentation

FMTool
und
Funktionsmodul

Anhang A

Beschreibung der Funktionsblöcke
Funktionsblock-Bibliothek (FBL) Version 1.51

Hinweise

Die in diesem Unterlagen enthaltenen Angaben, Daten, Werte usw. können ohne vorherige Ankündigung geändert werden. Ebenso sind die Abbildungen unverbindlich.

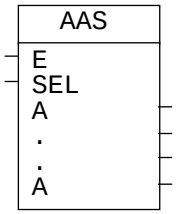
Gebrüder Merten GmbH & Co. KG
Elektrotechnik • Elektronik
Fritz-Kotz-Str. 8
D-51674 Wiehl

Telefon 0 22 61 / 702-01
Telefax 0 22 61 / 702-284

Beschreibung der Funktionsblöcke

AAS

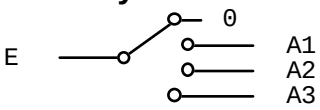
Ausgangsauswahlschalter



Funktion

Der AAS-Block schaltet den Eingang auf den selektierten Ausgang. Der ausgewählte Ausgang erhält also den Wert des Einganges. Die anderen Ausgänge verändern ihren Wert nicht. Die Anzahl der Ausgänge ist variabel zwischen 1 und 16. Der erste Ausgang ist die Nummer 1, der zweite die Nummer 2 usw. Hat SEL keinen gültigen Wert, so verändert sich kein Ausgang.
Nach einem Reset am Funktionsmodul haben alle nicht selektierten Ausgänge den Wert 0.

Schaltsymbol



Datentypen

Alle Datentypen können verwendet werden.

Ein-/Ausgänge

E Eingangssignal
SEL Selektion, wählt den Ausgang (Word)
A Ausgangssignale

Tips

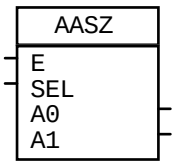
- Die Ausgänge dieses Blockes können als Signalwertspeicher verwendet werden. Wird der Ausgang angewählt, kann sein Wert verändert werden, sonst behält er seinen Wert.

Siehe auch Block:

AASZ, EAS, EASZ, EASM

AASZ

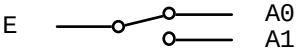
Ausgangsauswahlschalter zweifach



Funktion

Der AASZ-Block schaltet den Eingang auf einen der zwei Ausgänge. Der Selektions-Eingang ist vom Datentyp Bit. Ist SEL 0, wird der Ausgang A0 ausgewählt, sonst A1. Der nicht ausgewählte Ausgang ändert seinen Wert nicht. Nach einem Reset am Funktionsmodul hat der nicht selektierte Ausgang den Wert 0.

Schaltsymbol



Datentypen

Alle Datentypen können verwendet werden.

Ein-/Ausgänge

E	Eingangssignal
SEL	Selektion, wählt den Ausgang (Bit)
A0	Ausgang 0
A1	Ausgang 1

Tips

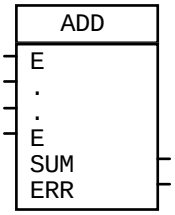
- Die Ausgänge dieses Blockes können als Signalwertspeicher verwendet werden. Wird der Ausgang angewählt kann sein Wert verändert werden, sonst behält er seinen Wert.
- Ein Signal kann unterbrochen werden, damit es seinen Wert nicht mehr ändert. So kann zum Beispiel verhindert werden, dass ein Telegramm gesendet wird.

Siehe auch Block:

AAS, EASZ, EAS, EASM

ADD

Addierer



Funktion

Der ADD-Block addiert mehrere Eingangswerte zusammen. Die Anzahl der Eingänge ist variabel zwischen 2 und 16. Geht die Summe oder ein Zwischenresultat über den Bereich des Datentypen hinaus, ist das Überlaufbit 1.

Datentypen

Word-, Sint- oder Valuewerte können addiert werden.

Ein-/Ausgänge

E Summande
SUM Summe
ERR Überlauf (Bit)

Formeln

Blockfunktion:

$$SUM = E1 + E2 + \ldots + En$$

Tips

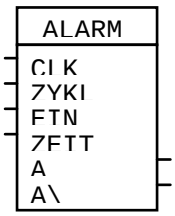
- Sollen grössere ganzahlige Zahlen als 65535 addiert werden (z.B. EIS11), kann der ADD-Block mit Datentyp Word verwendet werden. Bei diesem Block entspricht das Überlaufbit dem Carry-Flag. Es kann also als Bit 16 zu dem höherwertigen Word dazuaddiert werden. Der A-Ausgang ist dann das niederwertige Word.

Siehe auch Block:

SUB, MUL, DIV

ALARM

Alarm



Funktion

Der ALARM-Block ist wie ein Wecker. Bei einer steigenden Flanke am CLK-Eingang wird ein Zähler auf den Wert gesetzt, welcher am ZEIT-Eingang vorhanden ist. Gleichzeitig beginnt der Zähler rückwärts zu zählen. Hat der Zähler Null erreicht, wird der Ausgang wärend einem Taskzyklus auf 1 gesetzt (kurzer Puls).

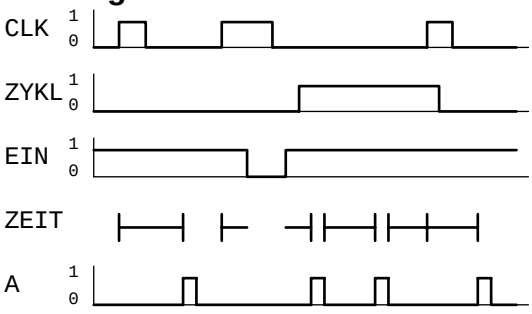
Ist der ZYKL-Eingang nach dem Puls 1, wird der Zähler erneut gestartet. Es entsteht so ein zyklischer Ausgang, wobei die Zykluszeit jedesmal vom ZEIT-Eingang abhängig ist.

Der Zählerstand wird eingefroren, wenn der EIN-Eingang 0 ist.

Wird eine zweite steigende Flanke auf den CLK-Eingang gegeben, bevor der Zähler Null wird, wird der Zähler wieder auf den Wert am ZEIT-Eingang gesetzt (retriggerbar).

Ist der CLK-Eingang auf den Wert 1 initialisiert, wird dies nach einem Reset des Funktionsmoduls als steigende Flanke interpretiert und der ALARM-Block wird gestartet.

Zeitdiagramm



Ein-/Ausgänge

CLK	flankengesteuerter Eingang (Bit)
ZYKL	zyklisches Ausgangssignal (Bit)
EIN	Ein-/Ausschalten (Bit)
ZEIT	Zeit in Taskzyklen (Word)
A	nichtinvertierter Ausgang (Bit)
A\	invertierter Ausgang (Bit)

Formeln

Der Eingangswert für ZEIT kann folgendermassen berechnet werden:

$$\text{ZEIT} = \frac{\text{Alarmzeit}[s]}{\text{Taskzykluszeit}[s]}$$

Tips

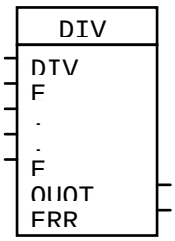
- Wird der A-Ausgang auf ein Ausgangskontrollsignal gegeben, kann der ALARM-Block dazu verwendet werden, Telegramme zyklisch zu senden.
- Er kann auch zum Verzögern von Pulsen verwendet werden.

Siehe auch Block:

OSZI

DIV

Dividierer



Funktion

Der DIV-Block dividiert eine Zahl (Dividend) durch eine oder mehrere Zahlen (Divisoren). Die Anzahl der Divisoren ist variabel zwischen 1 und 16.

Geht das Resultat oder ein Zwischenresultat über den Bereich des Datentypen hinaus, oder wird durch Null dividiert, ist der Ausgang undefiniert und das Überlaufbit ist 1.

Wird mit den Datentypen Word oder Sint dividiert, ist das Resultat immer auf die nächste ganze Zahl abgerundet.

Datentypen

Word, Sint und Value können voneinander dividiert werden.

Ein-/Ausgänge

DIV Dividend
E Divisoren
QUOT Quotient (Resultat)
ERR Divisionsfehler (Bit)

Formeln

Blockfunktion:

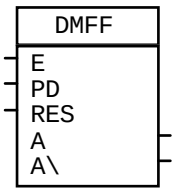
QUOT = DIV / E1 / E2 / ... / En

Siehe auch Block:

MUL, ADD, SUB

DMFF

Dynamisches, monostabiles Flip-Flop, retriggerbar



Funktion

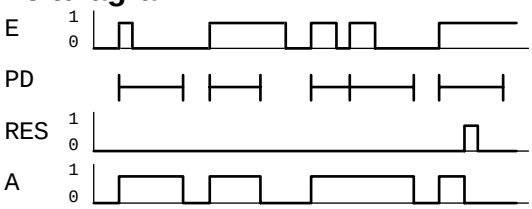
Der DMFF-Block ist ein monostabiles Flip-Flop. Bei einer steigenden Flanke am Eingang wird am Ausgang ein Puls generiert. Die Dauer des Pulses wird am PD-Eingang eingestellt. Gibt es während des Pulses erneut eine steigende Flanke am Eingang, wird die Pulsdauer neu gesetzt und der Puls wird verlängert (retriggerbar).

Der Wert am PD-Eingang hat nur während der steigenden Flanke am E-Eingang einen Einfluss auf die Pulsdauer. Ist PD Null bleibt auch der Ausgang Null.

Der RES-Eingang setzt das Ausgangssignal jederzeit auf Null.

Ist der E-Eingang auf den Wert 1 initialisiert, wird dies nach einem Reset des Funktionsmoduls als steigende Flanke interpretiert und der Puls am Ausgang wird gestartet.

Zeitdiagramm



Ein-/Ausgänge

E	flankengesteuerter Eingang (Bit)
PD	Pulsdauer in Taskzyklen (Word)
RES	Reset
A	nichtinvertiertes Ausgangssignal (Bit)
A\	invertiertes Ausgangssignal (Bit)

Formeln

Der Eingangswert für PD kann folgendermassen berechnet werden:

$$PD = \frac{\text{Pulsdauer}[s]}{\text{Taskzykluszeit}[s]}$$

Tips

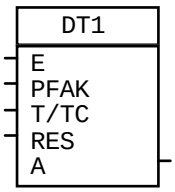
- Der DMFF-Block kann z. B. das Licht in einem Treppenhaus steuern (Treppenhausautomat).

Siehe auch Block:

SMFF

DT1

DT1-Glied (Hochpass ersten Grades)



Funktion

Dies ist ein DT1-Glied für Regelaufgaben. Es basiert auf dem Näherungsalgorithmus von Gauss. Reset setzt den Ausgang auf Null.

Damit eine vernünftige Rechengenauigkeit erreicht wird, soll die Taskzykluszeit mindestens 10 mal kleiner sein als die Periodendauer der kritischen Grenzfrequenz des Regelsystems.

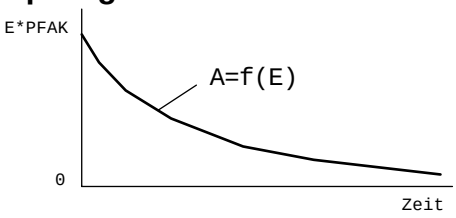
Der Ausgangswert ist auf den Wertebereich des Datentypen Value begrenzt. Ist T/TC negativ, oder ist PFAK negativ, so verändert sich der Ausgangswert nicht.

Die Auflösung des Ausgangssignals ist beschränkt. Ist die theoretische Änderung des Ausgangssignals pro Taskzyklus kleiner als 0.000’313, ändert sich der Ausgangswert nicht. Der Wert am T/TC-Eingang darf also nicht zu hoch gewählt werden.

Die Werte am PFAK- und am T/TC-Eingang dürfen zu jedem beliebigen Zeitpunkt geändert werden.

Ist der E-Eingang auf einen Wert ungleich 0 initialisiert, wird dies nach einem Reset des Funktionsmoduls als Signaländerung interpretiert und am Ausgang entsteht der entsprechende Sprung. Andernfalls bleibt der Ausgang 0.

Sprungantwort



Ein-/Ausgänge

E	Eingangssignal (Value)
PFAK	Proportionalfaktor (Value)
T/TC	Zeitkonstante (Value)
RES	Reset (Bit)
A	Ausgangssignal (Value)

Formeln

Blockfunktion:

A = ((PFAK * (E - Ealt) + Aalt) * (T/TC)) / (1 + (T/TC))
alt bedeutet: Wert aus dem letzten Taskzyklus.

Der Eingangswert für T/TC kann folgendermassen berechnet werden:

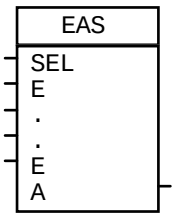
$$T / TC = \frac{\text{Zeitkonstante}[s]}{\text{Taskzykluszeit}[s]}$$

Siehe auch Block:

INTE, PI, PT1

EAS

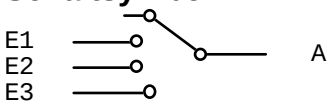
Eingangsauswahlschalter



Funktion

Der EAS-Block schaltet den selektierten Eingang auf den Ausgang. Der Ausgang erhält also den Wert des ausgewählten Eingangs. Die Anzahl der Eingänge ist variabel zwischen 1 und 16. Der erste Eingang ist die Nummer 1, der zweite die Nummer 2 usw. Hat SEL keinen gültigen Wert, so beibt der Ausgang unverändert.

Schaltsymbol



Datentypen

Alle Datentypen können verwendet werden.

Ein-/Ausgänge

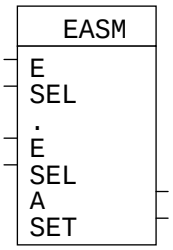
SEL Selektion, wählt den Eingang (Word)
E Eingangssignale
A Ausgangssignal

Siehe auch Block:

EASZ, EASM, AAS, AASZ

EASM

Eingangsauswahlschalter mehrfach



Funktion

Der EASM-Block schaltet den selektierten Eingang auf den Ausgang. Dieser erhält somit den Wert des ausgewählten Eingangs, d.h., des Eingangs, dessen SEL-Bit gesetzt ist. Das Ausgangssignal SET (Bit) wird gesetzt, wenn (mindestens) eines der Selektionsbits aktiv ist.

Ist mehr als ein Selektionsbit aktiv, wird der Wert des untersten selektierten Eingangs auf den Ausgang geschaltet.

Die Anzahl der Eingänge ist variabel zwischen 1 und 8.

Datentypen

Alle Datentypen können verwendet werden.

Ein-/Ausgänge

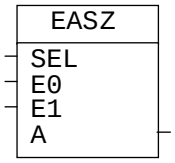
E	Eingangssignale
SEL	Selektion, wählt den Eingang (Bit)
A	Ausgangssignal
SET	Ausgang (Bit) = Oder-Verknüpfung aller SEL (Bit)

Siehe auch Block:

EAS, EASZ, AAS; AASZ

EASZ

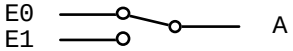
Eingangsauswahlschalter zweifach



Funktion

Der EASZ-Block schaltet den selektierten Eingang auf den Ausgang. Der Ausgang erhält also den Wert des ausgewählten Einganges. Ist SEL 0, wird der Eingang E0 ausgewählt, sonst E1.

Schaltsymbol



Datentypen

Alle Datentypen können verwendet werden.

Ein-/Ausgänge

SEL	Selektion, wählt den Eingang (Bit)
E0	Eingangssignal 0
E1	Eingangssignal 1
A	Ausgangssignal

Tips

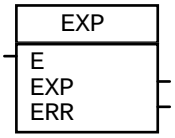
- Der EASZ-Block kann zur Realisierung von IF THEN ELSE verwendet werden. Beispiel: IF (SEL == 1) THEN A = E1 ELSE A = E0.

Siehe auch Block:

EAS, EASM, AAS, AASZ

EXP

Exponentialfunktion



Funktion

Des EXP-Block berechnet die Exponentialfunktion mit der Basis e (e = 2.71828). Überschreitet das Resultat den Wertebereich des Datentypen Value, ist der Ausgangswert unbestimmt und das Überlaufbit wird gesetzt.

Ein-/Ausgänge

E Eingangssignal (Value)
EXP Ausgangssignal (Value)
ERR Überlauf (Bit)

Formeln

Blockfunktion:

$$EXP = e^{E}$$

Rechengenauigkeit

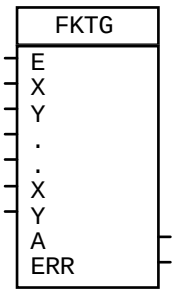
Die Auflösung des Datentypen Value ist beschränkt auf 0.000’313. Darum kann der Ausgangswert des EXP-Blockes bei kleinen Zahlen am Eingang, hohe relative Fehler aufweisen.

Siehe auch Block:

LOG

FKTG

Funktionsgeber



Funktion

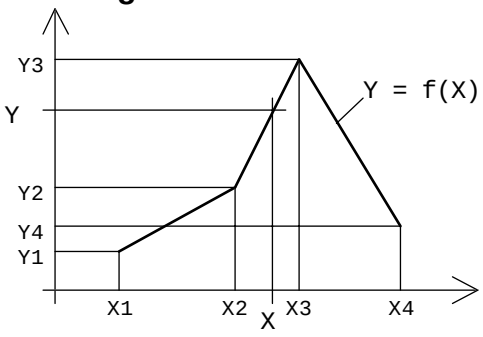
Der FKTG-Block bildet beliebige Funktionen nach. Die Funktion Y=f(X) wird anhand von 2 bis 8 X-Y-Koordinaten definiert. Die Bereiche zwischen diesen Punkten werden durch eine Gerade verbunden (linear approximiert).

Ist der Eingangswert ausserhalb der definierten Funktion, ist das Ausgangassignal unbestimmt und das Fehlerbit wird gesetzt.

Die X-Koordinaten müssen in aufsteigender Reihenfolge sortiert sein, sonst kann ein falsches Ausgangssignal resultieren. D.h. die Bedingung $X1 \le X2 \le K \le Xn$ muss erfüllt sein.

Haben zwei aufeinanderfolgende Koordinaten den gleichen X-Wert, so wird für diesen X-Wert der Y-Wert der ersten Koordinate auf den Ausgang gegeben. (z.B. ist X1=X2, so wird für diesen X-Wert Y1 auf den Ausgang gegeben.)

X-Y-Diagramm

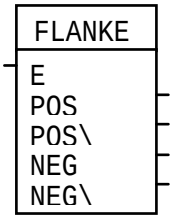


Ein-/Ausgänge

E Eingangsignal (Value)
X X-Koordinate (Value)
Y Y-Koordinate (Value)
A Ausgangssignal (Value)
ERR Fehler (Bit)

FLANKE

Flankendedektor



Funktion

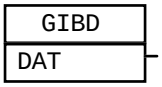
Der FLANKE-Block erkennt, ob es sich bei einem binären Eingangssignal um eine steigende oder um eine fallende Flanke handelt, d.h., ob der Wechsel am Eingang von 0 auf 1 oder umgekehrt erfolgt. Der entsprechende Ausgang wird danach für genau einen Taskzyklus gesetzt.

Ein-/Ausgänge

E Eingangssignal
POS nichtinvertierter Ausgang für steigende Flanke (Bit)
POS\ invertierter Ausgang für steigende Flanke (Bit)
NEG nichtinvertierter Ausgang für fallende Flanke (Bit)
NEG\ invertierter Ausgang für fallende Flanke (Bit)

GIBD

Datumgeber



Funktion

Der GIBD-Block gibt das Systemdatum des Funktionsmoduls. Das Datum enthält: Tag, Monat und Jahr. Das Jahr hat die Werte 00 bis 99.

Ein-/Ausgänge

DAT Datum (Date)

Einschränkungen

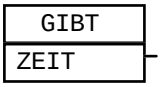
- Werden mehrere Tasks verwendet, soll dieser Block nur im schnellsten Task verwendet werden. Das Datum kann aber mit einem internen Signal auch in den anderen Tasks verwendet werden.

Siehe auch Block:

GIBT, SETD

GIBT

Zeitgeber



Funktion

Der GIBT-Block gibt die Systemzeit des Funktionsmoduls. Die Zeit enthält den Wochentag (Montag, Dienstag, usw.), die Stunden, Minuten und Sekunden.

Ein-/Ausgänge

ZEIT Zeit (Time)

Einschränkungen

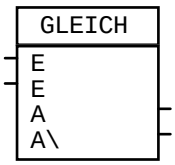
- Werden mehrere Tasks verwendet, soll dieser Block nur im schnellsten Task verwendet werden. Die Systemzeit kann aber mit einem internen Signal auch in den anderen Tasks verwendet werden.

Siehe auch Block:

GIBD, SETT

GLEICH

Vergleicher



Funktion

Der GLEICH-Block vergleicht die beiden Eingänge. Sind sie gleich, ist der A-Ausgang 1.

Datentypen

Alle Datentypen

Ein-/Ausgänge

E zu vergleichende Eingänge
A nichtinvertierter Ausgang (Bit)
A\ invertierter Ausgang (Bit)

Tips

- Der GLEICH-Block kann auch als XOR-Funktion verwendet werden, wenn der Datentyp Bit gewählt wird (siehe Beispiel, Wahrheitstabelle).

Wahrheitstabelle

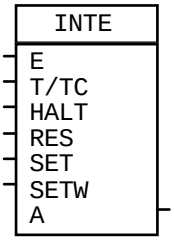
E	E	A	A\
0	0	1	0
0	1	0	1
1	0	0	1
1	1	1	0

Siehe auch Block:

KLEIN

INTE

Integrator



Funktion

Der INTE-Block ist ein Integrator-Glied mit einer Integrationskonstante T/TC. Die Integration basiert auf dem Näherungsalgorithmus von Gauss.

Mit dem HALT-Eingang kann die Integration angehalten werden, d.h. der Ausgangswert verändert sich nicht mehr. RES setzt den Ausgang auf Null.

Wenn SET 1 ist, wird der Ausgang auf den Wert am SETW-Eingang gesetzt.

RES hat die höhere Priorität als SET und SET hat die höhere Priorität als HALT.

Erreicht der Ausgang den Wertebereich des Datentypen, so wird dieser Maximal- bzw. Minimalwert beibehalten bis die Integrationsrichtung wechselt.

Ist T/TC negativ oder Null, wird der Ausgang nicht verändert.

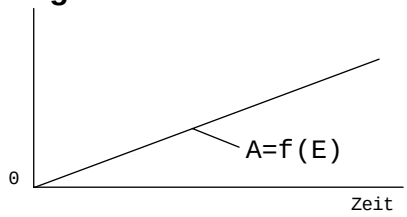
Damit eine vernünftige Rechengenauigkeit erreicht wird, soll die Taskzykluszeit mindestens 10 mal kleiner sein als die Periodendauer der kritischen Grenzfrequenz des Regelsystems.

Der Wert am T/TC-Eingang darf zu jedem beliebigen Zeitpunkt geändert werden.

Die Auflösung des Ausgangssignals ist beschränkt. Ist die theoretische Änderung des Ausgangssignals pro Taskzyklus kleiner als der kleinste darstellbare Wert, ändert sich der Ausgangswert nicht. Der Wert am T/TC-Eingang darf also nicht zu hoch gewählt werden. Die Auflösung beim Datentypen Value ist 0.000’313, bei Word und Sint 1.

Nach einem Reset am Funktionsmodul startet der Integrationswert bei 0.

Sprungantwort



Datentypen
Word, Sint, Value

Ein-/Ausgänge

E	Eingangssignal
T/TC	Zeitkonstante
HALT	Integration anhalten (Bit)
RES	Reset (Bit)
SET	Ausgangssignal setzen (Bit)
SETW	Wert der gesetzt wird
A	Ausgangssignal

Formeln
Blockfunktion:

$A = A_{alt} + E/(T/TC)$

A_{alt} bedeutet: Ausgangswert aus dem letzten Taskzyklus.

Der Eingangswert für T/TC kann folgendermassen berechnet werden:

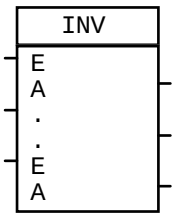
$$T/TC = \frac{\text{Zeitkonstante}[s]}{\text{Taskzykluszeit}[s]}$$

Tips
- Sind die Integrationsschritte pro Taskzyklus zu klein (weil der Wert am T/TC-Eingang zu gross ist), kann die Taskzykluszeit künstlich vergrössert werden indem der HALT-Eingang nur alle paar Taskzyklen einmal auf 0 gesetzt und sonst auf 1 belassen wird. Dies kann mit dem OSZI-Block realisiert werden, wobei die ‘künstliche’ Taskzykluszeit der Zykluszeit des Oszillators entspricht.

Siehe auch Block:
DT1, PI, PT1, ZÄHLER

INV

Logischer Inverter



Funktion

Der INV-Block invertiert ein binäres Signal. Die Anzahl der Ein- bzw. Ausgänge ist variabel zwischen 1 und 8.

Wahrheitstabelle

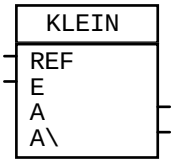
E	A
0	1
1	0

Ein-/Ausgänge

E Eingangssignal (Bit)
A Ausgangassignal (Bit)

KLEIN

Kleiner als (vergleichen)



Funktion

Der KLEIN-Block vergleicht den Eingang E mit einer Referenz. Ist E kleiner als REF, ist der nichtinvertierte Ausgang 1.

Verwendung

Die untenstehende Tabelle zeigt, wie die Signale für die verschiedenen Vergleiche an den KLEIN-Block angeschlossen werden müssen, zur Verwendung als KLEIN- bzw. als 'Grösser'-Block.

Vergleich	Signal A	Signal B	Signal C
C = A < B	E	REF	A
C = A ≤ B	REF	E	A\
C = A > B	REF	E	A
C = A ≥ B	E	REF	A\

Datentypen

Byte, Date, Sint, Time, Value, Word

Ein-/Ausgänge

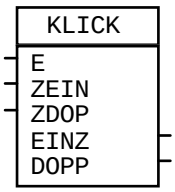
REF Referenzwert
E zu vergleichendes Signal
A nichtinvertierter Ausgang (Bit)
A\ invertierter Ausgang (Bit)

Siehe auch Block:

GLEICH

KLICK

Erkennen von Klick und Doppelklick



Funktion

Der KLICK-Block erkennt, ob auf einem binären Eingangssignal ein einzelner Puls, oder zwei aufeinanderfolgende Pulse auftreten (einzelner Klick, oder Doppelklick).

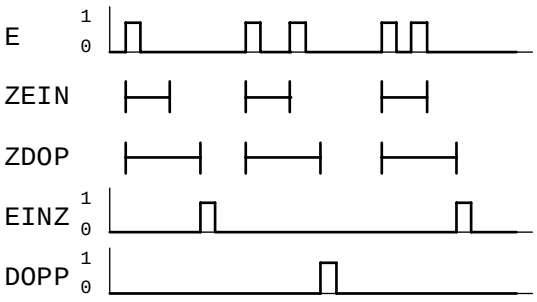
Ein Puls darf nur einen Taskzyklus lang dauern. Meistens wird der E-Eingang mit einem Eingangskontrollsignal verbunden, auf welches eine Taste sendet.

Wenn während der ganzen Zeit ZDOP nur ein Puls auftritt, wird der EINZ-Ausgang einen Taskzyklus lang auf 1 gesetzt. Tritt *nach* der Zeit ZEIN ein zweiter Puls auf, wird der DOPP-Ausgang einen Taskzyklus lang auf 1 gesetzt.

Treten zwei Pulse *während* der Zeit ZEIN auf, wird dies als *einen* Puls gewertet. Dieser Mechanismus kann zum entprellen verwendet werden. Meistens wird aber ZEIN auf 0 gesetzt.

Die Werte am ZEIN- und ZDOP-Eingang werden jeweils beim ersten Puls übernommen. Danach hat eine Änderung dieser Werte, bis die Zeit am ZDOP-Eingang abgelaufen ist, keinen Einfluss auf die Funktion des KLICK-Blockes.

Zeitdiagramm



Ein-/Ausgänge

E Eingang (Bit)
ZEIN Zeit in Taskzyklen, in der zwei Pulse als einzelner Klick gelten (Word)
ZDOP Zeit in Taskzyklen, nach der die Ausgänge gesetzt werden (Word)
EINZ Ausgang: Zeigt einen einzelnen Klick an (Bit)
DOPP Ausgang: Zeigt einen Doppelklick an (Bit)

Formeln

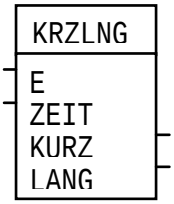
Der Eingangswert für ZEIN und ZDOP kann folgendermassen berechnet werden:

$$\text{ZEIN} = \frac{\text{Zeit}[s]}{\text{Taskzykluszeit}[s]}$$

$$\text{ZDOP} = \frac{\text{Zeit}[s]}{\text{Taskzykluszeit}[s]}$$

KRZLNG

Erkennen von kurzen und langen Signalen



Funktion

Der KRZLNG-Block erkennt, ob ein Eingangssignal kürzer oder länger als eine spezifische Zeit gesetzt ist. Damit können z.B. kurze oder lange Signale von Tastsensoren unterschieden werden.

Liegt das Eingangssignal E länger an als ZEIT, wird der Ausgang LANG für genau einen Taskzyklus gesetzt; liegt es kürzer an als ZEIT, wird KURZ gesetzt.

Ein-/Ausgänge

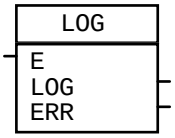
E	Eingang (Bit)
ZEIT	Zeit (Word)
KURZ	Ausgang: Zeigt ein kurzes Eingangssignal an (Bit)
LANG	Ausgang: Zeigt ein langes Eingangssignal an (Bit)

Formeln

$$\text{ZEIT} = \frac{\text{Zeit}[\text{s}]}{\text{Taskzykluszeit}[\text{s}]}$$

LOG

Natürlicher Logarithmus



Funktion

Der LOG-Block berechnet den natürlichen Logarithmus mit der Basis e (e = 2.71828).
Ist das Eingangssignal negativ, ist der Ausgangswert unbestimmt und das Fehlerbit wird gesetzt.

Ein-/Ausgänge

E Eingangssignal (Value)
LOG Ausgangssignal (Value)
ERR Fehler

Formeln

Blockfunktion:

$$LOG = \ln(E)$$

Rechengenauigkeit

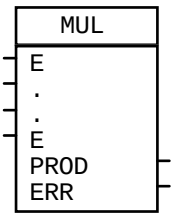
Die Auflösung des Datentypen Value ist beschränkt auf 0.000’313. Darum kann der Ausgangswert des LOG-Blockes bei kleinen Zahlen am Eingang, hohe relative Fehler aufweisen.

Siehe auch Block:

EXP

MUL

Multiplizierer



Funktion

Der MUL-Block multipliziert mehrere Faktoren miteinander. Die Anzahl der Eingänge ist variabel zwischen 2 und 16. Geht das Produkt oder ein Zwischenresultat über den Bereich des Datentypes hinaus, ist der Ausgang undefiniert und das Überlaufbit wird gesetzt.

Datentypen

Sint, Value, Word

Ein-/Ausgänge

E	Faktoren
PROD	Produkt
ERR	Überlauf (Bit)

Formeln

Blockfunktion:

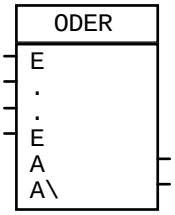
PROD = E1 · E2 · ... · En

Siehe auch Block:

DIV, ADD, SUB

ODER

Logisches ODER



Funktion

Der ODER-Block macht eine logische ODER-Verknüpfung mit mehreren Eingängen. Ist mindestens ein Eingang 1, ist der A-Ausgang 1. Die Anzahl der Eingänge ist variabel zwischen 2 und 16.

Wahrheitstabelle

E	E	A	A\
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

Ein-/Ausgänge

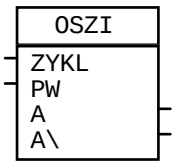
E Eingangssignale (Bit)
A nichtinvertierter Ausgang (Bit)
A\ invertierter Ausgang (Bit)

Siehe auch Block:

UND

OSZI

Oszillator



Funktion

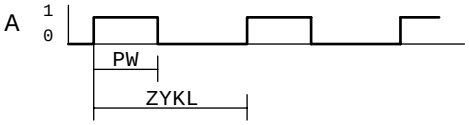
Der OSZI-Block generiert periodische Rechtecksignale. Die Zykluszeit und die Pulsweite sind programmierbar.

Eine Änderung des Eingangswertes am ZYKL- oder am PW-Eingang, wirkt sich erst zu Beginn der nächsten Periode aus (bei steigender Flanke am A-Ausgang).

Nach einem Reset des Funktionsmoduls startet der OSZI-Block sofort mit einer steigenden Flanke am A-Ausgang.

Ist der PW-Eingang 0, bleibt der A-Ausgang auf 0. Ist der Wert am PW-Eingang grösser als der Wert am ZYKL-Eingang, oder ist der ZYKL-Eingang 0, so bleibt der A-Ausgang 1.

Zeitdiagramm



Ein-/Ausgänge

ZYKL Zykluszeit in Taskzyklen (Word)
PW Pulsweite in Taskzyklen (Word)
A nichtinvertierter Ausgang (Bit)
A\ invertierter Ausgang (Bit)

Formeln

Der Eingangswert für ZYKL und PW kann folgendermassen berechnet werden:

$$\mathrm{ZYKL} = \frac{\mathrm{Zykluszeit}[s]}{\mathrm{Taskzykluszeit}[s]}$$

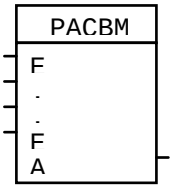
$$\mathrm{PW} = \frac{\mathrm{Pulsweite}[s]}{\mathrm{Taskzykluszeit}[s]}$$

Siehe auch Block:

ALARM

PACBM

Packe Byte in Max

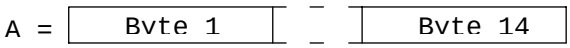


Funktion

Der PACBM-Block packt mehrere Bytes in ein Signal vom Datentyp Max. Die Anzahl der Eingänge ist variabel zwischen 1 und 14.

Ausgangsformat

Vorsicht: Das Ausgangssignal vom Typ MAX ist nur für den Teil gültig, der einen entsprechenden Eingang hat, d.h., wenn z.B. als Eingänge nur 4 Bytes verwendet werden, sind die Positionen 5...14 beim MAX-Ausgangssignal nicht definiert.



Ein-/Ausgänge

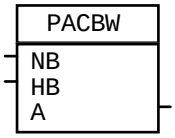
E Eingänge (Byte)
A Ausgang (Max)

Siehe auch Block:

TEILMB, PACM, TEILM

PACBW

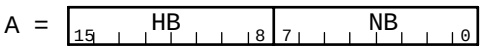
Packe Byte in Word



Funktion

Der PACB-Block packt ein niederwertiges und ein höherwertiges Byte in ein Word.

Ausgangsformat



Ein-/Ausgänge

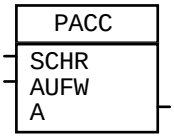
NB niederwertiges Byte
HB höherwertiges Byte
A Ausgang (Word)

Siehe auch Block:

TEILWB

PACC

Packe Control



Funktion

Der PACC-Block packt ein Richtungsbit und eine Schrittweite in ein Signal vom Datentyp Control. Ist die Schrittweite grösser als 7, ist der Ausgang unbestimmt (die Schrittweite wird mit 3 Bit abgebildet).

Ein-/Ausgänge

SCHR Schrittweite (Byte)
- 0 = stop
- 1 = 100% auf/ab dimmen
- 2 = 50% auf/ab dimmen
- 3 = 25% auf/ab dimmen
- 4 = 13% auf/ab dimmen
- 5 = 6.3% auf/ab dimmen
- 6 = 3.1% auf/ab dimmen
- 7 = 1,6% auf/ab dimmen

AUFW Aufwärts (Richtung) (Bit)
- 0 = abwärts dimmen
- 1 = aufwärts dimmen

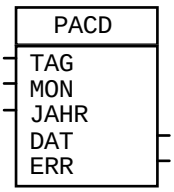
A Ausgang (Control)

Siehe auch Block:

TEILC

PACD

Packe Date



Funktion

Der PACD-Block packt drei Signale (Tag, Monat und Jahr) in ein Signal vom Datentyp Date. Hat Tag, Monat oder Jahr einen ungültigen Wertebereich, ist der Ausgang Null und das Fehlerbit ist gesetzt.

Ein-/Ausgänge

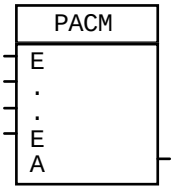
TAG	Tag (Byte). Wertebereich: 1 .. 31
MON	Monat (Byte). Wertebereich: 1 .. 12
JAHR	Jahr (Byte). Wertebereich: 0 .. 99
DAT	Datum (Date)
ERR	Fehler (Bit)

Siehe auch Block:

TEILD

PACM

Packe Word inMax

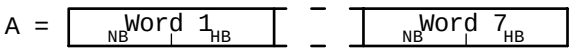


Funktion

Der PACM-Block packt mehrere Words in ein Signal vom Datentyp Max. Die Anzahl der Eingänge ist variabel zwischen 1 und 7.

Ausgangsformat

Vorsicht: Das niederwertige Byte ist jeweils vor dem höherwertigen Byte im Datentyp Max abgelegt, d.h., das niederwertige Byte von Eingang 1 ist das erste Byte am Ausgang.



Ein-/Ausgänge

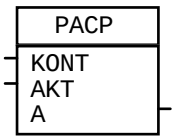
E Eingänge (Word)
A Ausgang (Max)

Siehe auch Block:

TEILM, PACBM, TEILMB

PACP

Packe Pcontrol



Funktion

Der PACP-Block packt ein Kontrollbit und ein Aktionsbit in ein Signal vom Datentyp Pcontrol.

Wahrheitstabelle

KONT	AKT	Funktion
0	0	Kontrolle ausgeschaltet, Aktionsbit nicht aktiv
0	1	Kontrolle ausgeschaltet, Aktionsbit nicht aktiv
1	0	Kontrolle eingeschaltet, Aktion aus
1	1	Kontrolle eingeschaltet, Aktion ein

Ein-/Ausgänge

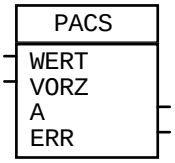
KONT Kontrollbit, Bit 1 auf dem Bus (Bit)
AKT Aktionsbit, Bit 0 auf dem Bus (Bit)
A Ausgang (Pcontrol)

Siehe auch Block:

TEILP
(*Vorsicht:* Beim TEILP-Block ist die Reihenfolge der KONT- und AKT-Ausgänge umgekehrt.)

PACS

Packe Sint



Funktion

Der PACS-Block packt einen Wert vom Datentyp Word und ein Vorzeichenbit in ein Signal vom Datentyp Sint. Eine 0 bedeutet positives Vorzeichen, eine 1 bedeutet negatives Vorzeichen.

Ist WERT zu gross für den Datentyp Sint, ist der Ausgangswert unbestimmt und das Überlaufbit ist gesetzt. Der Wertebereich für Sint reicht von –32768 bis +32767.

Ein-/Ausgänge

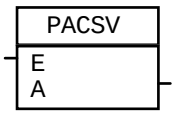
WERT	positiver Wert (Word)
VORZ	Vorzeichen (Bit) 0 = positiv 1 = negativ
A	Ausgang (Sint)
ERR	Überlauf (Bit)

Siehe auch Block:

TEILS

PACSV

Packe Sint in Value



Funktion

Der PACSV-Block packt ein Signal vom Datentyp Sint in ein Signal vom Datentyp Value.

Ein-/Ausgänge

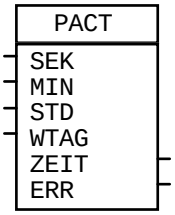
E Eingang (Sint)
A Ausgang (Value)

Siehe auch Block:

PACVS

PACT

Packe Time



Funktion

Der PACT-Block packt vier Signale (Sekunden, Minuten, Stunden und Wochentag) in ein Signal vom Datentyp Time. Haben Sekunden, Minuten, Stunden oder der Wochentag einen ungültigen Wertebereich, ist der Ausgang Null und das Fehlerbit wird gesetzt.

Ein-/Ausgänge

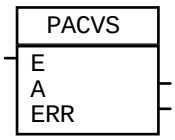
SEK Sekunden (Byte). Wertebereich: 0 .. 59
MIN Minuten (Byte). Wertebereich: 0 .. 59
STD Stunden (Byte). Wertebereich: 0 .. 23
WTAG Wochentag (Byte)
0 = kein Tag (neutral)
1 = Montag
7 = Sonntag
ZEIT Zeit (Time)
ERR Fehler (Bit)

Siehe auch Block:

TEILT

PACVS

Packe Value in Sint



Funktion

Der PACVS-Block packt ein Singal vom Datentyp Value in ein Signal vom Datentyp Sint. Es wird immer abgerundet.

Ist das Eingangssignal zu gross für den Datentyp Sint, ist der Ausgangswert unbestimmt und das Überlaufbit ist gesetzt. Der Wertebereich für Sint reicht von –32768 bis +32767.

Ein-/Ausgänge

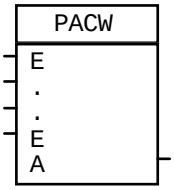
E Eingang (Value). Wertebereich: –32768.0 .. +32767.9
A Ausgang (Sint)
ERR Überlauf (Bit)

Siehe auch Block:

PACSV

PACW

Packe Word



Funktion

Der PACW-Block packt mehrere Bits in ein Word. Die Anzahl der Eingänge ist variabel zwischen 1 und 16. Eingang Nummer 1 ist das niederwertigste Bit. Werden weniger als 16 Eingänge verwendet, sind die höherwertigen Bits 0.

Ausgangsformat

A = | E16 | E15 | E14 | E13 | E12 | E11 | E10 | E9 | E8 | E7 | E6 | E5 | E4 | E3 | E2 | E1 |
15 8 7 0

Ein-/Ausgänge

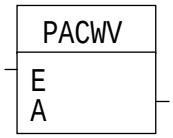
E Eingänge (Bit)
A Ausgang (Word)

Siehe auch Block:

TEILW

PACWV

Packe Word in Value



Funktion

Der PACWV-Block packt ein Signal vom Datentyp Word in ein Signal vom Datentyp Value.

Ein-/Ausgänge

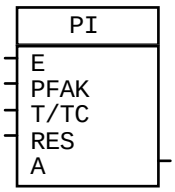
E Eingang (Word)
A Ausgang (Value)

Siehe auch Block:

TEILVW

PI

PI-Glied



Funktion

Dies ist ein PI-Glied für Regelaufgaben. Es besteht aus einem Proportionalteil und einem Integratorteil. Mit Reset wird der Integratorteil auf Null gesetzt.

Damit eine vernünftige Rechengenauigkeit erreicht wird, soll die Taskzykluszeit mindestens 10 mal kleiner sein als die Periodendauer der kritischen Grenzfrequenz des Regelsystems.

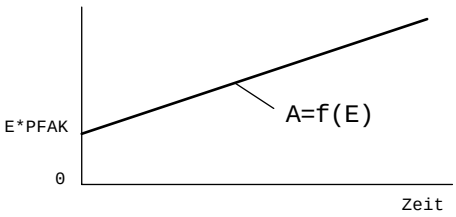
Der Ausgangswert ist auf den Wertebereich des Datentypen Value begrenzt. Ist T/TC negativ oder Null, verändert sich der Integratorwert nicht. Ist der Proportionalfaktor negativ, so wird er auf Null gesetzt.

Die Auflösung des Ausgangssignals ist beschränkt. Ist die theoretische Änderung des Ausgangssignals pro Taskzyklus kleiner als 0.000’313, ändert sich der Ausgangswert nicht. Der Wert am T/TC-Eingang darf also nicht zu hoch gewählt werden.

Die Werte am PFAK- und am T/TC-Eingang dürfen zu jedem beliebigen Zeitpunkt geändert werden.

Nach einem Reset am Funktionsmodul startet der Integrationswert bei 0.

Sprungantwort



Ein-/Ausgänge

E	Eingangssignal (Value)
P_{FAK}	Proportionalfaktor (Value)
T/TC	Zeitkonstante (Value)
RES	Reset (Bit)
Y	Integrationswert (Value)
A	Ausgangssignal (Value)

Formeln

Blockfunktion:

$Y = Y_{alt} + E/(T/TC)$

Y_{alt} bedeutet: Integrationswert aus dem letzten Taskzyklus.

$A = Y + E*P_{FAK}$

Der Eingangswert für T/TC kann folgendermassen berechnet werden:

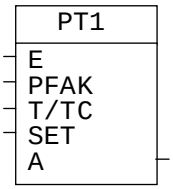
$$T / TC = \frac{\text{Zeitkonstante}[s]}{\text{Taskzykluszeit}[s]}$$

Siehe auch Block:

DT1, INTE, PT1

PT1

PT1-Glied



Funktion

Dies ist ein PT1-Glied für Regelaufgaben. Es basiert auf dem Näherungsalgorithmus von Gauss. Mit SET wird der Ausgang auf den Eingangswert gesetzt.

Damit eine vernünftige Rechengenauigkeit erreicht wird, soll die Taskzykluszeit mindestens 10 mal kleiner sein als die Periodendauer der kritischen Grenzfrequenz des Regelsystems.

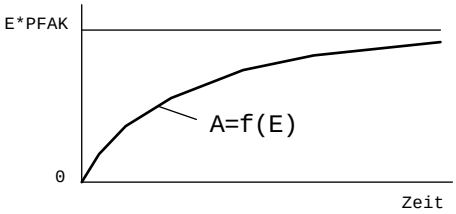
Der Ausgangswert ist auf den Wertebereich des Datentypen Value begrenzt. Ist T/TC negativ, oder ist PFAK negativ, verändert sich der Ausgangswert nicht.

Die Auflösung des Ausgangssignals ist beschränkt. Ist die theoretische Änderung des Ausgangssignals pro Taskzyklus kleiner als 0.000’313, ändert sich der Ausgangswert nicht. Der Wert am T/TC-Eingang darf also nicht zu hoch gewählt werden.

Die Werte am PFAK- und am T/TC-Eingang dürfen zu jedem beliebigen Zeitpunkt geändert werden.

Nach einem Reset am Funktionsmodul startet die Ausgangskurve bei 0.

Sprungantwort



Ein-/Ausgänge

E	Eingangssignal (Value)
PFAK	Proportionalfaktor (Value)
T/TC	Zeitkonstante (Value)
SET	Set (Bit)
A	Ausgangssignal (Value)

Formeln

Blockfunktion:

$A = (PFAK*E + A_{alt}*(T/TC)) / (1+(T/TC))$

A_{alt} bedeutet: Ausgangswert aus dem letzten Taskzyklus.

Der Eingangswert für T/TC kann folgendermassen berechnet werden:

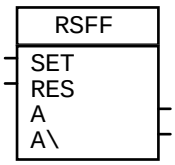
$$T / TC = \frac{\text{Zeitkonstante}[s]}{\text{Taskzykluszeit}[s]}$$

Siehe auch Block:

DT1, INTE, PI

RSFF

RS-Flip-Flop



Funktion

Der RSFF-Block ist ein Set-Reset-Flip-Flop. Ist der Set-Eingang 1, wird der A-Ausgang auf 1 gesetzt. Ist der Reset-Eingang 1, wird der A-Ausgang auf 0 gesetzt. Der Reset-Eingang hat die höhere Priorität als der Set-Eingang.

Nach einem Reset am Funktionsmodul ist der A-Ausgang 0.

Wahrheitstabelle

SET	RES	A	A\
0	0	A	A\
0	1	0	1
1	0	1	0
1	1	0	1

Ein-/Ausgänge

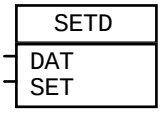
SET Set-Eingang (Bit)
RES Reset-Eingang (Bit)
A nichtinvertierter Ausgang (Bit)
A\ invertierter Ausgang (Bit)

Siehe auch Block:

TFF

SETD

Setze Datum



Funktion

Der SETD-Block setzt das Systemdatum des Funktionsmoduls auf das gegebene Datum, wenn der SET-Eingang 1 ist.

Das neue Datum ist erst im nächsten Taskzyklus gesetzt.

Der SETD-Block prüft die Gültigkeit des Datums nicht. Wird ein ungültiges Datum gesetzt, ist das Systemdatum unbestimmt.

Ein-/Ausgänge

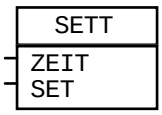
DAT Datum (Date)
SET Setzen (Bit)

Siehe auch Block:

SETT, GIBD, PACD, TEILD

SETT

Setze Zeit



Funktion

Der SETT-Block setzt die Systemzeit des Funktionsmoduls auf die gegebene Zeit, wenn der SET-Eingang 1 ist.

Die neue Systemzeit ist erst im nächsten Taskzyklus gesetzt.

Der SETT-Block prüft die Gültigkeit der Zeit nicht. Wird eine ungültige Zeit gesetzt, ist die Systemzeit unbestimmt.

Der Wochentag kann unabhängig vom Systemdatum gesetzt werden. Es besteht also kein Zusammenhang zwischen dem Wochentag und dem Datum. Wird der Wochentag auf Noday gesetzt, verändert sich der Wochentag in der Systemuhr nicht.

Ein-/Ausgänge

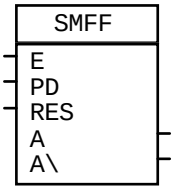
ZEIT Zeit (Time)
SET Setzen (Bit)

Siehe auch Block:

SETD, GIBT, PACT, TEILT

SMFF

Statisches, Monostabiles Flip-Flop



Funktion

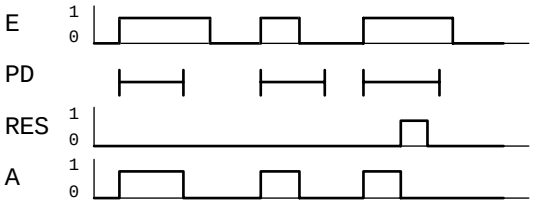
Der SMFF-Block ist ein monostabiles Flip-Flop. Er verkürzt einen Inpuls am Eingang auf eine einstellbare Pulsweite.

Der RES-Eingang setzt das Ausgangssignal jederzeit auf Null.

Die Dauer des Pulses wird am PD-Eingang eingestellt. Während der A-Ausgang 1 ist, hat eine Änderung des PD-Einganges keinen Einfluss auf die Pulsdauer. Ist PD 0, bleibt auch der Ausgang 0.

Ist der E-Eingang auf den Wert 1 initialisiert, wird dies nach einem Reset des Funktionsmoduls als steigende Flanke interpretiert und der Puls am Ausgang wird gestartet.

Zeitdiagramm



Ein-/Ausgänge

E Eingangssignal (Bit)
PD Pulsdauer (Word)
RES Reset (Bit)
A nichtinvertierter Ausgang (Bit)
A\ invertierter Ausgang (Bit)

Formeln

Der Eingangswert für PD kann folgendermassen berechnet werden:

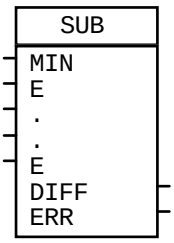
$$PD = \frac{\text{Pulsdauer}[s]}{\text{Taskzykluszeit}[s]}$$

Siehe auch Block:

DMFF

SUB

Subtrahierer



Funktion

Der SUB-Block subtrahiert ein oder mehrere Subtrahende von einem Minuend. Die Anzahl der Subtrahenden ist variabel zwischen 1 und 16.

Geht das Resultat oder ein Zwischenresultat über den Bereich des Datentypen hinaus, ist der Ausgang undefiniert und das Überlaufbit wird gesetzt.

Datentypen

Sint, Value, Word

Ein-/Ausgänge

MIN Minuend
E Subtrahenden
DIFF Differenz
ERR Überlauf (Bit)

Formeln

Blockfunktion:

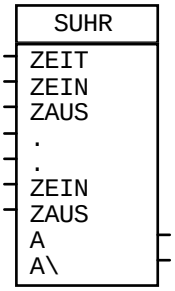
DIFF = MIN – E1 – E2 – ... – En

Siehe auch Block:

ADD, MUL, DIV

SUHR

Schaltuhr



Funktion

Die Schaltuhr schaltet den Ausgang bei programmierbaren Zeiten ein und aus.

Am ZEIT-Eingang muss die aktuelle Zeit vorhanden sein. Es können bis zu acht Einschaltbereiche angegeben werden. Ist die aktuelle Zeit innerhalb eines Einschaltbereiches, ist der A-Ausgang 1.

Wie die Einschaltbereiche defieniert werden, zeigt untenstehendes Beispiel.

Enthält die aktuelle Zeit keinen Tag (noday), werden die Einschaltbereiche, welche einen Wochentag enthalten, nicht ausgewertet.

Die Werte an den ZEIN- und ZAUS-Eingängen können zu jedem beliebigen Zeitpunkt geändert werden.

Beispiel

ZEIN, ZAUS	Bereich	Beispiel
ZEIN < ZAUS ZEIN und ZAUS haben noday	jeden Tag von ZEIN bis ZAUS eingeschaltet	ZEIN = no, 11:50:00 ZAUS = no, 13:10:00 Mittags eingeschaltet
ZEIN > ZAUS ZEIN und ZAUS haben noday	jede Nacht von ZEIN bis ZAUS eingeschaltet	ZEIN = no, 18:30:00 ZAUS = no, 05:00:00 in der Nacht eingeschaltet
ZEIN und ZAUS haben einen Wochentag	einmal pro Woche, von ZEIN bis ZAUS eingeschaltet	ZEIN = Fri, 23:50:00 ZAUS = Mon, 02:00:00 am Wochendende eingeschaltet

Ein-/Ausgänge

ZEIT	Zeit-Eingang (Time)
ZEIN	um diese Zeit wird eingeschaltet (Time)
ZAUS	um diese Zeit wird ausgeschaltet (Time)
A	nichtinvertierter Ausgang (Bit)
A\	invertierter Ausgang (Bit)

TEILC

Teile Control

Funktion

Der TEILC-Block teilt ein Signal vom Datentyp Control in eine Schrittweite und ein Richtungsbit.

Ein-/Ausgänge

E Eingang (Control)

SCHR Schrittweite (Byte)
- 0 = stop
- 1 = 100% auf/ab dimmen
- 2 = 50% auf/ab dimmen
- 3 = 25% auf/ab dimmen
- 4 = 13% auf/ab dimmen
- 5 = 6.3% auf/ab dimmen
- 6 = 3.1% auf/ab dimmen
- 7 = 1,6% auf/ab dimmen

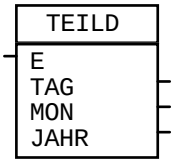
AUFW Aufwärts (Richtung) (Bit)
- 0 = abwärts dimmen
- 1 = aufwärts dimmen

Siehe auch Block:

PACC

TEILD

Teile Date



Funktion

Der TEILD-Block teilt ein Signal vom Datentyp Date in Tag, Monat und Jahr.

Ein-/Ausgänge

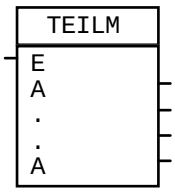
E	Eingang (Date)
TAG	Tag (Byte)
MON	Monat (Byte)
JAHR	Jahr (Byte). Wertebereich: 0 .. 99

Siehe auch Block:

PACD

TEILM

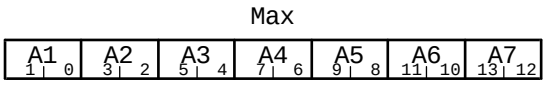
Teile Max in Word



Funktion

Der TEILM-Block teilt ein Signal vom Datentyp Max in Words. Die Anzahl Ausgänge ist variabel zwischen 1 und 7.

Aufteilung in Words
Vorsicht: Das erste Byte vom Eingang wird im niederwertigen Byte von Ausgang 1 abgelegt.



Ein-/Ausgänge

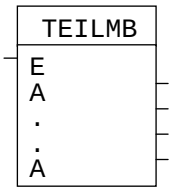
E Eingang (Max)
A Ausgänge (Word)

Siehe auch Block:

PACM, TEILMB, PACBM

TEILMB

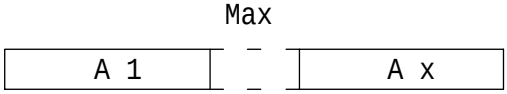
Teile Max in Byte



Funktion

Der TEILMB-Block teilt ein Signal vom Datentyp Max in Bytes. Die Anzahl Ausgänge ist variabel zwischen 1 und 14.

Aufteilung in Bytes



Ein-/Ausgänge

E Eingang (Max)
A Ausgänge (Byte)

Siehe auch Block:

PACMB, TEILM, PACM

TEILP

Teile Pcontrol

Funktion

Der TEILP-Block teilt ein Signal vom Datentyp Pcontrol in ein Aktionsbit und ein Kontrollbit.

Wahrheitstabelle

KONT	AKT	Funktion
0	0	Kontrolle ausgeschaltet, Aktionsbit nicht aktiv
0	1	Kontrolle ausgeschaltet, Aktionsbit nicht aktiv
1	0	Kontrolle eingeschaltet, Aktion aus
1	1	Kontrolle eingeschaltet, Aktion ein

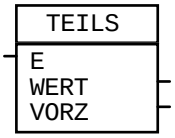
Ein-/Ausgänge

E Eingang (Pcontrol)
AKT Aktionsbit, Bit 0 vom Eingang (Bit)
KONT Kontrollbit, Bit 1 vom Eingang (Bit)

Siehe auch Block:
PACP
(*Vorsicht:* Beim PACP-Block ist die Reihenfolge der KONT- und AKT-Ausgänge umgekehrt.)

TEILS

Teile Sint



Funktion

Der TEILS-Block teilt ein Signal vom Datentyp Sint in einen Absolutwert vom Datentyp Word und in ein Vorzeichen.

Ein-/Ausgänge

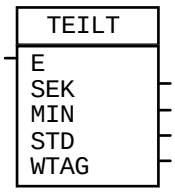
E	Eingang (Sint)
WERT	Absolutwert (Word)
VORZ	Vorzeichen (Bit) 0 = positiv 1 = negativ

Siehe auch Block:

PACS

TEILT

Teile Time



Funktion

Der TEILT-Block teilt ein Signal vom Datentyp Time in Sekunden, Minuten, Stunden und Wochentag.

Ein-/Ausgänge

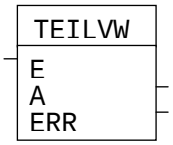
E	Eingang (Time)
SEK	Sekunden (Byte)
MIN	Minuten (Byte)
STD	Stunden (Byte)
WTAG	Wochentag (Byte)
	0 = kein Tag (neutral)
	1 = Montag
	7 = Sonntag

Siehe auch Block:

PACT

TEILVW

Teile Value in Word



Funktion

Der TEILVW-Block wandelt ein Signal vom Datentyp Value in ein Signal vom Datentyp Word. Sollte der Ausgang ausserhalb des Wertbereichs zu liegen kommen, wird das Fehlerbit ERR gesetzt und der Ausgang wird Null.

Ein-/Ausgänge

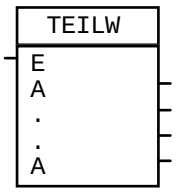
E	Eingang (Value)
A	Ausgang (Word)
ERR	Fehler (Bit)

Siehe auch Block:

PACWV

TEILW

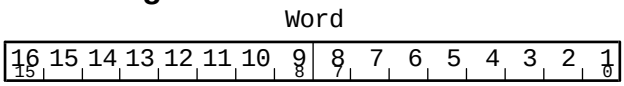
Teile Word



Funktion

Der TEILW-Block teilt ein Signal vom Datentyp Word in mehrere Bits. Die Anzahl der Ausgänge ist variabel zwischen 1 und 16. Das erste Bit ist das niederwertigste Bit.

Aufteilung in Bits



Ein-/Ausgänge

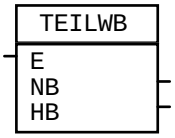
E Eingang (Word)
A Ausgänge (Bit)

Siehe auch Block:

PACW

TEILWB

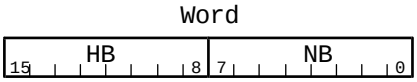
Teile Word in Byte



Funktion

Der TEILWB-Block teilt ein Signal vom Datentyp Word in ein niederwertiges und ein höherwertiges Byte.

Aufteilung in Bytes



Ein-/Ausgänge

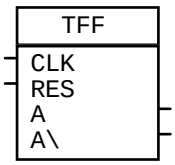
E	Eingang (Word)
NB	niederwertiges Byte
HB	höherwertiges Byte

Siehe auch Block:

PACBW

TFF

Toggle-Flip-Flop (Umschalter)



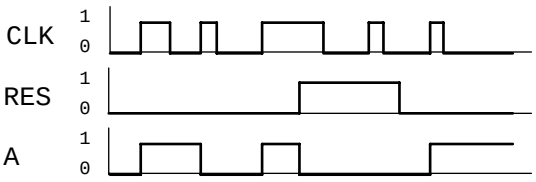
Funktion

Der TFF-Block ist ein Toggle-Flip-Flop. Bei einer steigenden Flanke am CLK-Eingang wechseln die Ausgänge ihren Wert von 0 auf 1 oder von 1 auf 0. Reset setzt den A-Ausgang immer auf 0.

Nach einem Reset am Funktionsmodul ist der A-Ausgang 0.

Ist der CLK-Eingang auf den Wert 1 initialisiert, wird dies nach einem Reset des Funktionsmoduls als steigende Flanke interpretiert und der A-Ausgang wird 1.

Zeitdiagramm



Ein-/Ausgänge

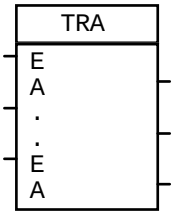
CLK	flankengesteuerter Eingang (Bit)
RES	Reset (Bit)
A	nichtinvertierter Ausgang (Bit)
A\	invertierter Ausgang (Bit)

Siehe auch Block:

RSFF

TRA

Transfer



Funktion

Der TRA-Block transferiert (kopiert) den Wert des Eingangssignals an den Ausgang. Es können bis zu 8 Signale kopiert werden.

Datentypen

Alle Datentypen

Ein-/Ausgänge

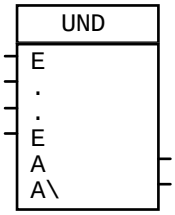
E Eingänge
A Ausgänge

Tips

- Der TRA-Block wird verwendet um ein Signal auf mehrere Signale zu kopieren. Zum Beispiel wenn ein Blockausgang intern verwendet wird (lokales Signal) und gleichzeitig auf ein Ausgangssignal geführt werden muss.

UND

Logisches UND



Funktion

Der UND-Block macht eine logische UND-Verknüpfung mit mehreren Eingängen. Sind alle Eingänge 1, ist der A-Ausgang 1. Die Anzahl der Eingänge ist variabel zwischen 2 und 16.

Wahrheitstabelle

E	E	A	A\
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

Ein-/Ausgänge

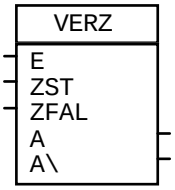
E Eingänge (Bit)
A nichtinvertierter Ausgang (Bit)
A\ invertierter Ausgang (Bit)

Siehe auch Block:

ODER

VERZ

Verzögerer



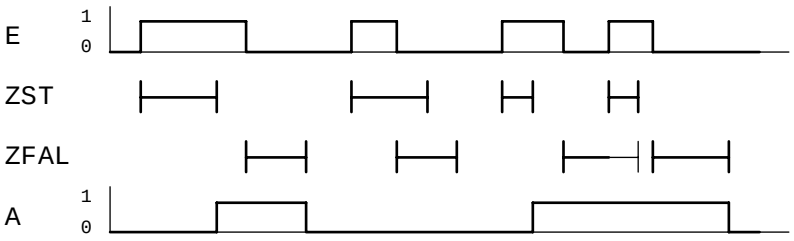
Funktion

Der VERZ-Block verzögert die steigende und die fallende Flanke vom Eingangssignal (Ein- und Ausschaltverzögerung). Die Verzögerungszeiten für das Ansteigen und Abfallen sind einzeln einstellbar. Beide Zeiten können auch auf 0 gesetzt werden.

Werden die Werte am ZST- oder ZFAL-Eingang während der Verzögerung des Ausgangssignales verändert, hat dies keinen Einfluss auf die laufende Verzögerung.

Ist der E-Eingang auf den Wert 1 initialisiert, wird dies nach einem Reset des Funktionsmodules als steigende Flanke interpretiert und der A-Ausgang wird nach der eingestellten Zeit auf 1 gesetzt.

Zeitdiagramm



Ein-/Ausgänge

E	Eingangssignal (Bit)
ZST	Verzögerungszeit der steigenden Flanke (Word)
ZFAL	Verzögerungszeit der fallenden Flanke (Word)
A	nichtinvertierter Ausgang (Bit)
A\	invertierter Ausgang (Bit)

Formeln

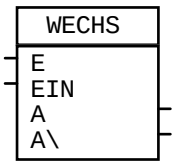
Der Eingangswert für ZST und ZFAL kann folgendermassen berechnet werden:

$$ZST = \frac{\text{Verzögerungszeit}[s]}{\text{Taskzykluszeit}[s]}$$

$$ZFAL = \frac{\text{Verzögerungszeit}[s]}{\text{Taskzykluszeit}[s]}$$

WECHS

Wechsel eines Signals

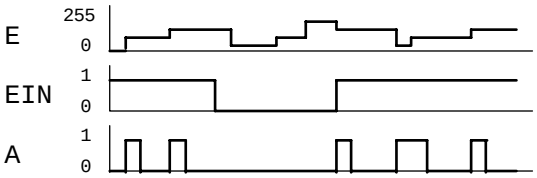


Funktion

Der WECHS-Block stellt Änderungen am Eingang fest und setzt in diesem Fall den Ausgang einen Taskzyklus lang auf 1 (kurzer Puls). Mit dem EIN-Eingang kann die Funktion ein- und ausgeschaltet werden.

Ist der E-Eingang auf einen anderen Wert als 0 initialisiert, wird dies nach einem Reset des Funktionsmoduls als Änderung interpretiert und am Ausgang wird ein Puls erzeugt.

Zeitdiagramm



Datentypen

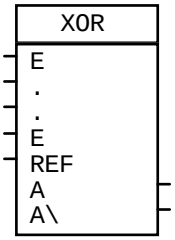
Alle Datentypen

Ein-/Ausgänge

E	Eingangssignal
EIN	Ein-/Ausschalten (Bit)
A	nichtinvertierter Ausgang (Bit)
A\	invertierter Ausgang (Bit)

XOR

Logisches XOR



Funktion

Der XOR-Block addiert alle Bits an den Eingängen die 1 sind. Ist die Summe dieser Bits nicht 0 und nicht grösser als der Wert am REF-Eingang, ist der A-Ausgang 1. Die Anzahl der Eingänge kann von 2 bis 16 eingestellt werden.

Beispiel

Beispiel mit drei E-Eingängen

E	E	E	REF	A
0	0	0	2	0
1	0	0	2	1
0	1	0	2	1
0	1	1	2	1
1	1	1	2	0
1	0	0	1	1
1	1	0	1	0

Ein-/Ausgänge

E Eingänge (Bit)
REF Referenzwert (Word)
A nichtinvertierter Ausgang (Bit)
A\ invertierter Ausgang (Bit)

Tips

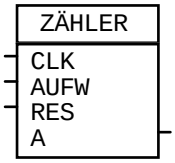
- Ein einfaches XOR-Tor mit zwei Eingängen kann auch mit dem GLEICH-Block realisiert werden (siehe GLEICH-Block).

Siehe auch Block:

GLEICH

ZÄHLER

Zähler



Funktion

Der ZÄHLER-Block zählt die steigenden Flanken am CLK-Eingang. Mit dem AUFW-Eingang kann die Zählrichtung (aufwärts/abwärts) gewählt werden. Reset setzt den Ausgangswert auf 0.

Der Zähler zählt im Kreis, d.h. beim Erreichen des Maximalwertes 32767 wird bei -32768 fortgefahren. In Abwärtszählrichtung umgekehrt.

Nach einem Reset am Funktionsmodul ist der Ausgangswert 0.

Ist der CLK-Eingang auf den Wert 1 initialisiert, wird dies nach einem Reset des Funktionsmoduls als steigende Flanke interpretiert und der Ausgang wird um 1 erhöht.

Wahrheitstabelle

CLK	AUFW	RES	A
↑	1	0	A+1
↑	0	0	A–1
X	X	1	0

↑ steigende Flanke
X kann den Wert 0 oder 1 haben

Datentypen

Der Ausgang kann vom Typ Sint oder Word sein.

Ein-/Ausgänge

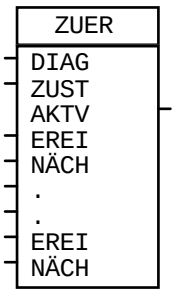
CLK flankengesteuerter Eingang (Bit)
AUFW Zählrichtung (Bit)
0 = abwärts
1 = aufwärts
RES Reset
A Ausgang (Sint)

Siehe auch Block:

INTE

ZUER

Zustands-Ereignis-Block



Funktion

Mit dem ZUER-Block können Zustands-Ereignisdiagramme programmiert werden. Jeder ZUER-Block, sowie ein ZUINIT-Block und ZUSET-Blöcke, werden über ein gemeinsames Signal am DIAG-Eingang miteinander verbunden.

Ein ZUER-Block entspricht einem bestimmten Zustand im Diagramm. Die Zustandsnummer wird am ZUST-Eingang angegeben. Ist der jeweilige Zustand aktiv, ist der AKTV-Ausgang 1.

In jedem Zustand können Ereignisse auftreten. Diese können am ZUER-Block an den EREI-Eingang angeschlossen werden. Ist nun der betreffende Zustand aktiv und ein Ereignis ist 1, wird in den neuen Zustand umgeschalten, welcher am NÄCH-Eingang angegeben ist. Treffen gleichzeitig mehrere Ereignisse ein, so wird das erste (das oberste am ZUER-Block) ausgewertet. Der neue Zustand ist immer erst im nachfolgenden Taskzyklus aktiv. D.h. der Zustandswechsel erfolgt nicht sofort.

Die Anzahl der EREI- und NÄCH-Eingänge ist variabel zwischen 1 und 8. Reicht die Anzahl dieser Eingänge nicht aus, können mehrere ZUER-Blöcke mit der gleichen Zustandsnummer eingesetzt werden.

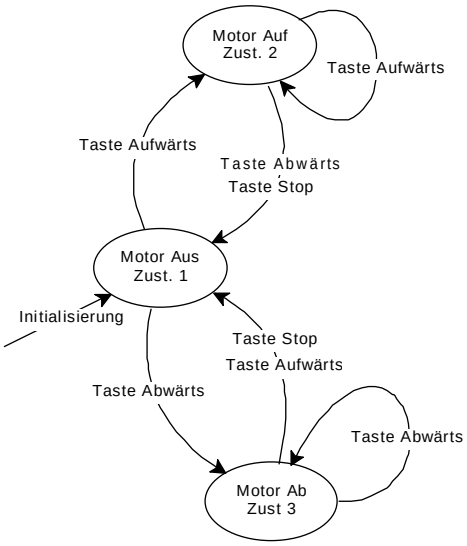
Ein Zustands-Ereignisdiagramm kann bis zu 65’536 verschiedene Zustände haben. Die Anzahl ZUER-Blöcke in einem Diagramm ist durch die Kapazität des Funktionsmoduls beschränkt.

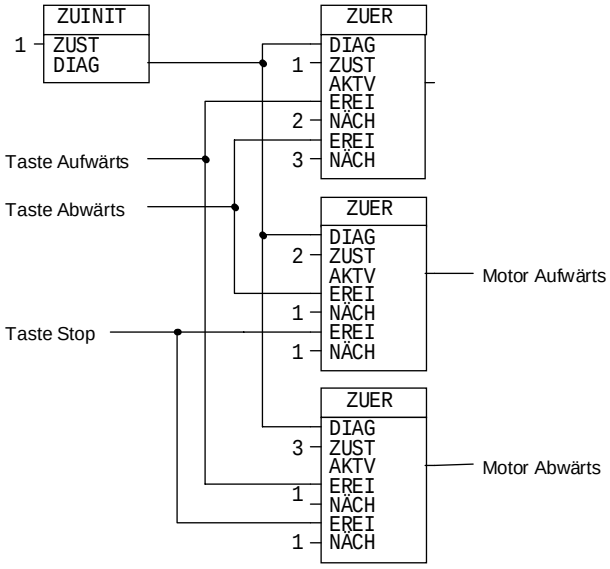
Wichtig

Alle ZUER- und ZUSET-Blöcke vom gleichen Zustands-Ereignisdiagramm müssen sich im gleichen Task befinden.

Beispiel

Das Zustands-Ereignisdiagramm und die Realisierung mit Funktionsblöcken:





Ein-/Ausgänge

DIAG Diagrammsignal (Max)
ZUST Zustandsnummer (Word)
AKTV Aktiv (Bit)
EREI Ereignis-Eingänge (Bit)
NÄCH nächster Zustand nach Ereignis (Word)

Siehe auch Block:

ZUINIT, ZUSET

ZUINIT

Zustand initialisieren

Funktion
Der ZUINIT-Block initialisiert ein Zustands-Ereignisdiagramm. Der Start-Zustand kann am ZUST-Eingang angegeben werden. Jedes Zustands-Ereignisdiagramm muss mit einem ZUINIT-Block verbunden sein.

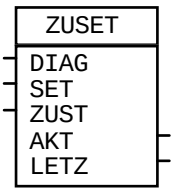
Wichtig
Dieser Block muss sich im Inittask befinden.

Ein-/Ausgänge
ZUST Initialisierungs-Zustand (Word)
DIAG Diagrammsignal (Max)

Siehe auch Block:
ZUER, ZUSET

ZUSET

Zustand setzen



Funktion

Der ZUSET-Block kann ein Zustands-Ereignisdiagramm zu einem beliebigen Zeitpunkt auf einen bestimmten Zustand setzten. Dazu muss der SET-Eingang auf 1 gesetzt und am ZUST-Eingang der neue Zustand angelegt werden. Der neue Zustand ist erst im nächsten Taskzyklus aktiv.

Am AKT-Ausgang liegt immer die Nummer des aktiven Zustandes, und am LETZ-Ausgang liegt immer die Zustandsnummer vor dem letzten Wechsel an.

Wichtig

Alle ZUER- und ZUSET-Blöcke vom gleichen Zustands-Ereignisdiagramm müssen sich im gleichen Task befinden.

Ein-/Ausgänge

DIAG Diagrammsignal (Max)
SET Setzen eines Zustandes (Bit)
ZUST zu setzender Zustand (Word)
AKT aktueller Zustand (Word)
LETZ letzter Zustand (Word)

Siehe auch Block:

ZUINIT, ZUER

Technische Dokumentation

FMTool
und
Funktionsmodul

Anhang B

Beschreibung der Kompiler-Fehlermeldungen

Hinweise

Die in diesem Unterlagen enthaltenen Angaben, Daten, Werte usw. können ohne vorherige Ankündigung geändert werden. Ebenso sind die Abbildungen unverbindlich.

Gebrüder Merten GmbH & Co. KG
Elektrotechnik • Elektronik
Fritz-Kotz-Str. 8
D-51674 Wiehl

Telefon 0 22 61 / 702-01
Telefax 0 22 61 / 702-284

Kompiler-Fehlermeldungen

Dieser Anhang beschreibt die Fehlermeldungen, die der Kompiler beim Kompilieren anzeigt. Die Fehler sind in drei Kategorien aufgeteilt:

- Fatale Fehler
- Fehler
- Warnungen

Für jeden Fehler sind die Ursache und mögliche Massnahmen zur Behebung der Fehler beschrieben.

Anstelle der in Grossbuchstaben geschriebenen Wörter (DATEI, NUMMER, SEGMENT usw.) sind in den Fehlermeldungen die Dateinamen, Nummern oder Segmentnamen usw. angegeben.

Diese Beschreibung der Kompiler-Fehlermeldungen gehört zu FMTool Vers. 1.50

Fatale Fehler

Fatale Fehler deuten auf Fehler hin, die mit dem System, dem Laufwerk usw. zusammenhängen. Tritt ein solcher Fehler auf, wird der Kompiliervorgang sofort abgebrochen.

Fehlermeldung	Ursache	Fehlerbehebung
Datei DATEI nicht gefunden!	Die Datei kann nicht mehr gefunden werden.	Backup des Projektes wieder auf PC laden (Restore).
Unable to create DATEI	Das Laufwerk ist voll oder schreibgeschützt, oder eine andere Applikation greift auf die Datei zu.	Unnötige Dateien löschen, Schreibschutz entfernen oder Zugriff auf Datei freigeben.
Unable to open DATEI	Die Datei kann nicht mehr gefunden oder darauf zugegriffen werden.	Zugriff auf Datei freigeben oder Backup des Projektes wieder auf PC laden (Restore).
Syntax in Datei DATEI!	Die Datei ist zerstört und kann nicht gelesen werden.	Backup des Projektes wieder auf PC laden (Restore).
Datei DATEI nicht gefunden! Define Task(s) first!	Die Taskbeschreibungsdatei kann nicht gefunden werden. Die Tasks wurden noch nicht definiert.	Tasks im Taskbuilder definieren.
Kompiliervorgang durch Benutzer abgebrochen!	Der "Abort"-Knopf im Kompiler-Dialog wurde gedrückt und der Kompiliervorgang wurde abgebrochen.	
Zu wenig Speicher für Rückdokumentation!	Der Arbeitsspeicher reicht nicht aus. Die Rückdokumentationsdaten können nicht erstellt werden.	Andere Windows-Applikationen schliessen.
Rückdokumentation ist zu lang! (Projekt zu gross!)	Die Rückdokumentationsdaten sind zu gross und können nicht ins Funktionsmodul geladen werden.	Das Projekt verkleinern oder optimieren. Ungebrauchte Signale löschen.

Fatale Fehler, die hier nicht aufgeführt sind, entstehen bei Fehlern im Kompiler selbst. Sollten solche Fehlermeldungen auftreten, benachrichtigen Sie bitte Ihren Vertreiber.

Fehler

'Fehler' zeigen Fehler auf, die das Projekt betreffen. Diese Fehler müssen behoben werden, bevor das Projekt ins Funktionsmodul geladen werden kann.

Fehlermeldung	Ursache	Fehlerbehebung
Eingang nicht angeschlossen an Block NUMMER in Segment SEGMENT, Gruppe GRUPPE	Ein Eingang des Blockes ist nicht mit einem Signal verbunden.	Eingang mit Signal oder Konstante verbinden.
Internes Signal SIGNAL in Segment SEGMENT, Gruppe GRUPPE ist an keinem Ausgang angeschlossen!	Das Signal ist mit keinem Blockausgang verbunden.	Einen Blockausgang mit dem Signal verbinden.
Kontrollsignal SIGNAL in Segment SEGMENT Gruppe GRUPPE ist in einem zu langsamen Task!	Dieses Kontrollsignal ist in einem langsamen Task verwendet. Kontrollsignale können so nicht zuverlässig ausgewertet werden.	Kontrollsignale nur im schnellsten Task verwenden.
Signal SIGNAL in Segment SEGMENT Gruppe GRUPPE ist nicht erlaubt im Initialisierungstask!	Dieses Eingangssignal wird im Initialisierungstask verwendet. Im Initialisierungstask können jedoch keine Bussignale empfangen werden.	Signal in einem anderen Task verwenden.
Signal SIGNAL in Segment SEGMENT Gruppe GRUPPE ist nicht erlaubt im Powerfail-Task!	Dieses Eingangssignal wird im Powerfail-Task verwendet. Im Powerfail-Task können jedoch keine Bussignale empfangen werden.	Signal in einem anderen Task verwenden.
Signal SIGNAL in Segment SEGMENT Gruppe GRUPPE ist verbunden mit einem Ausgang: Dies ist nicht erlaubt in einem Initialisierungstask!	Dieses Signal kommt von einem anderen Task in den Initialisierungstask. Im Initialisierungstask können jedoch keine anderen Signale verwendet werden.	Signal durch Konstante ersetzen.
Signal SIGNAL in Segment SEGMENT Gruppe GRUPPE ist an einem Ausgang angeschlossen: Dies ist nicht erlaubt in einem Powerfail-Task!	Dieses Signal kommt von einem anderen Task in den Powerfailtask. Im Powerfail-Task können jedoch keine anderen Signale verwendet werden.	Signal durch Konstante ersetzen.

Signal SIGNAL in Segment SEGMENT Gruppe GRUPPE im Powerfail-Task ist an einen Eingang eines anderen Tasks verbunden!	Dieses Signal geht vom Powerfail-Task in einen anderen Task. Nach dem Powerfail-Task werden jedoch keine anderen Tasks mehr gestartet.	Signal durch Konstante ersetzen.
Task TASK beinhaltet kein Segment!	Diesem Task ist kein Segment zugeordnet.	Ein Segment zuordnen oder diesen Task löschen.
Zweifelhafte Verbindung in Segment SEGMENT Gruppe GRUPPE!	Ein lokales Signal ist nur an einem Ende mit einem Block verbunden.	Signal löschen.
Zu viele Eingangssignale definiert!	Es wurden zu viele Eingangssignale (Signale vom Bus) definiert; mehr als im Funktionsmodul verarbeitet werden können.	Eingangssignale löschen. Das Projekt ist zu gross, es muss verkleinert werden.
Zu viele Eingangs- und Ausgangssignale definiert!	Es wurden zu viele externe Signale (Eingangs- und Ausgangssignale) definiert; mehr als im Funktionsmodul verarbeitet werden können.	Externe Signale löschen. Das Projekt ist zu gross, es muss verkleinert werden.
Projekt zu komplex! Nicht genügend Platz für Code im FM!	Das Projekt ist zu gross. Der erzeugte Code hat im Funktionsmodul keinen Platz.	Das Projekt verkleinern oder optimieren. Ungebrauchte Signale löschen.
Nicht genügend RAM für Signale im Funktionsmodul!	Das Projekt ist zu gross. Die Signale können im Funktionsmodul nicht gespeichert werden.	Das Projekt verkleinern. Anzahl der Signale verkleinern.

Warnungen

Warnungen machen auf Zustände aufmerksam, die vielleicht unabsichtlich entstanden sind, im Normalfall aber überhaupt nicht stören.

Fehlermeldung	Ursache	Fehlerbehebung
Ausgang nicht angeschlossen an Block NUMMER in Segment SEGMENT, Gruppe GRUPPE	Ein Ausgang des Blocks ist nicht mit einem Signal verbunden.	Eventuell mit Signal verbinden.
Internes Signal SIGNAL in Segment SEGMENT, Gruppe GRUPPE hat keine Verbindung zu einem Eingang!	Dieses interne Signal wird nirgends verwendet.	Eventuell Signal mit einem Blockeingang verbinden.

Technische Dokumentation

FMTool
und
Funktionsmodul

Anhang C

Beschreibung der Fehlermeldungen des Funktionsmoduls

Hinweise

Die in diesem Unterlagen enthaltenen Angaben, Daten, Werte usw. können ohne vorherige Ankündigung geändert werden. Ebenso sind die Abbildungen unverbindlich.

Gebrüder Merten GmbH & Co. KG
Elektrotechnik • Elektronik
Fritz-Kotz-Str. 8
D-51674 Wiehl

Telefon 0 22 61 / 702-01
Telefax 0 22 61 / 702-284

Fehlermeldungen des Funktionsmoduls

Während dem Betrieb des Funktionsmoduls auftretende Fehler und Störungen verursachen eine Fehlermeldung. Tritt ein Fehler auf, leuchtet die unterste der drei LED's am Funktionsmodul. Jeder Fehler wird im Funktionsmodul in einer Liste gespeichert. Diese Liste kann mit dem FMLoader ausgelesen werden (siehe Kapitel 9.11 'Störungsabfrage (Fehlermeldungen anzeigen)' im Handbuch).

Den verschiedenen Fehlern ist eine Identifikations-Nr.vergeben. Die folgende Liste gibt Auskunft über die Art des Fehlers und was dagegen unternommen werden kann.

Ab FMTool Version 1.50 werden aufgetretene Fehler im Klartext angezeigt, dergestalt, dass zu jedem Fehler (mit der zugehörenden Identifikations-Nr.) Angaben zur Fehlerursache und zu den Korrekturmöglichkeiten gemacht werden. Damit können aufgetretene Fehler vor Ort am PC und ohne weitere Dokumentation beurteilt werden.

Die Fehlermeldungen können in eine Datei abgelegt werden. Wird eine bestehende Datei ausgewählt, werden die neuen Meldungen angehängt. Die Datei ist ein Textfile im Tabellenformat mit einem Tabulator als Separator.

Zusätzlich zur Fehler-Information wird das Abrufdatum (Systemdatum des PC) und die physikalische Adresse des Funktionsmoduls ins File geschrieben. Somit können die Fehlermeldungen verschiedener Funktionsmodule auf dem Bus in dieselbe Datei abgelegt werden.

Diese Beschreibung der Fehlermeldungen des Funktionsmoduls gehört zum Funktionsmodul-Betriebssystem OS Version 2.10

Buffer-Overflow

ID Beschreibung

0 **Überlauf des Gruppentelegramm-Empfangspuffers**
Der Empfangspuffer für Gruppentelegramme ist überlaufen. Das bedeutet, dass zu viele Eingangstelegramme an einen langsamen Task gesendet wurden.

Massnahme: Es muss dafür gesorgt werden dass weniger Telegramme auf dem Bus vorkommen, oder die Zykluszeit des schnellsten Tasks muss kürzer gewählt werden.

1 **Überlauf des Gruppentelegramm-Sendepuffers**
Der Ausgangspuffer für Gruppentelegramme ist überlaufen. Dies passiert, wenn eine Applikation zu viele Telegramme erzeugt oder die adressierte Gruppe nicht existiert und jedes Telegramm deshalb mehrfach wiederholt wird.

Massnahme: Die Applikation muss so umprogrammiert werden, dass weniger Telegramme gesendet werden.

2 **Überlauf des Servicetelegramm-Empfangspuffers**
Der Empfangspuffer für Service-Telegramme ist überlaufen.

3 **Überlauf des Servicetelegramm-Sendepuffers**
Der Ausgangspuffer für Service-Telegramme ist überlaufen. Eventuell ist der adressierte Busteilnehmer nicht (mehr) vorhanden.

ID = Identifikations-Nr.

Kommunikationsprobleme

ID Beschreibung

10 **Verlorenes Telegramm**
Ein Telegramm ist verloren gegangen, weil ein Task innerhalb seiner Zykluszeit mehr als zwei Telegramme auf denselben Eingang empfangen hat.

Parameter 1: Gruppenadresse des verlorenen Telegrammes (dezimal)

Massnahme: Task schneller laufen lassen oder Telegramme auf dieselbe Gruppenadresse verlangsamen.

11 **Falsche Telegrammlänge**
Das auf eine entsprechende Gruppenadresse empfangene Telegramm wies eine falsche Länge (Typ) auf und wurde verworfen.

Parameter 1: Gruppenadresse des verlorenen Telegramms (dezimal)
Parameter 2: Falsche Länge des Telegramms

Massnahme: Der Busteilnehmer, der dieses Telegramm gesendet hat, muss umparmetriert werden.

12 **Zu viele Telegramme empfangen**
Die innerhalb einer Taskzykluszeit des Porthandlers erhaltenen Telegramme (entspricht Zykluszeit des langsamsten Tasks, wenn dieser schneller ist als 1s) auf verschiedene Gruppenadressen, konnten nicht alle abgearbeitet werden.

Parameter 1: Gruppenadresse des verlorenen Telegramms (dezimal)

Massnahme: Task schneller laufen lassen oder Anzahl Telegramme auf diese Eingans-Gruppenadressen des Funktionsmoduls vermindern.

13 **Verlorenes Telegramm**
Innerhalb der Zykluszeit ihres Tasks haben zu viele Gruppenadressen mehr als ein Eingangstelegramm erhalten (das demzufolge zwischengespeichert werden müsste). Das Telegramm wurde verworfen.

Parameter 1: Gruppenadresse des verlorenen Telegramms (dezimal)

Massnahme: Task schneller laufen lassen oder Telegramme auf diese Gruppenadresse verlangsamen

20 **Sendefehler**
Die maximale Anzahl Sendeversuche wurde erreicht, ohne dass das Telegramm abgesetzt werden konnte. Ein anderer Busteilnehmer hat jede Wiederholung nicht angenommen (NAK oder BUSY). Das zu sendende Telegramm wurde deshalb verworfen.
Dies deutet auf eine schlechte Busverbindung oder einen Fehlerhaften Busteilnehmer hin.

21 **BCU-Timeout**
Auf der PEI-Schnittstelle zwischen Busankopplung und Funktionsmodul ist ein Timeout aufgetreten.

22 **Physikalisch adressierte Verbindung abgebrochen**
Eine offene Verbindung zwischen zwei physikalisch adressierten Teilnehmern wurde für länger als 6s unterbrochen resp. blieb ohne Datenverkehr.
Dieser Fehler kann bei Verbindungsproblemen zwischen Funktionsmodul und FMLoader auftreten.

Massnahme: Eventuell wurde der FMLoader von einer anderen Windows-Applikation zu lange unterbrochen. Störende Applikation schliessen.

23 **Physikalisch adressiertes Telegramm ohne Bestätigung**
Ein physikalisch adressiertes Telegramm wurde länger als 3s nicht bestätigt.

30 **Fehler im Flash-EPROM**
Bei der Programmierung des Flash-EPROM’s ist ein Fehler aufgetreten.
Dies deutet auf einen Hardware-Fehler im Funktionsmodul hin.

Ungültige Meldungen

ID Beschreibung

40 **Ungültige Link-Layer Meldung**
Auf dem Bus ist eine unbekannte Link-Layer Meldung versandt worden. Dies deutet auf das fehlerhafte Funktionieren des Busteilnehmers hin, der diese Meldung gesandt hat.

50 **Ungültige Transport-Layer Meldung**
Auf dem Bus ist eine unbekannte Transport-Layer Meldung im verbindungsorientierten Mode versandt worden. Dies deutet auf das fehlerhafte Funktionieren des Busteilnehmers hin, der diese Meldung gesandt hat.

51 **Ungültige Transport-Layer Meldung, verbindungslos**
Es wurde eine unbekannte, physikalisch adressierte Transport-Layer Meldung empfangen, ohne dass eine Verbindung offen war. Dies deutet auf das fehlerhafte Funktionieren des Busteilnehmers hin, der diese Meldung gesandt hat.

60 **Ungültige Meldung eines Gruppentelegrammes**
Auf dem Bus ist eine unbekannte Meldung auf eine Gruppenadresse, die einem Eingangssignal entspricht, versandt worden. Dies deutet auf das fehlerhafte Funktionieren des Busteilnehmers hin, der diese Meldung gesandt hat.

100 **Ungültige Broadcast-Meldung**
Auf dem Bus ist eine unbekannte Broadcast-Meldung versandt worden. Dies deutet auf das fehlerhafte Funktionieren des Busteilnehmers hin, der diese Meldung gesandt hat.

101 **Ungültige GetValue-Meldung**
Auf dem Bus wurde eine unbekannte GetValue-Meldung (Speicher und Adresse auslesen der Busteilnehmer usw.) versandt. Dies deutet auf das fehlerhafte Funktionieren des Busteilnehmers hin, der diese Meldung gesandt hat.

102 **Ungültige UserMsg-Meldung**
Auf dem Bus wurde eine unbekannte UserMsg-Meldung versandt. Dies deutet auf das nicht korrekte Funktionieren des Busteilnehmers hin, der diese Meldung gesandt hat.

Applikationsfehler

ID Beschreibung

200 **Falsche OS-Version oder falsche FBL-Version**
Dies kann mit 'Version anzeigen!' im FMLoader überprüft werden. Damit die Applikation läuft, müssen folgende Punkte erfüllt sein:
Die OS-Hauptversion im Funktionsmodul (OS = Operating System) muss mit der verlangten OS-Hauptversion der Applikation (erforderliche OS-Version) übereinstimmen (die ersten beiden Ziffern, z.B. 02XX). Die Subversion (die letzten beiden Ziffern, z.B. XX06) der verlangten OS-Version darf nicht grösser sein als die im Funktionsmodul enthaltene Subversion.
Dasselbe gilt für die FBL-Versionen: Die FBL-Hauptversion im Funktionsmodul (FBL = Funktionsblock-Bibliothek) muss mit der verlangten FBL-Hauptversion der Applikation (erforderliche FBL-Version) übereinstimmen (die ersten beiden Ziffern, z.B. 01XX). Die Subversion (die letzten beiden Ziffern, z.B. XX04) der verlangten FBL-Version darf nicht grösser sein als die im Funktionsmodul enthaltene Subversion.

Massnahme: Andere OS- oder FBL-Version ins Funktionsmodul laden oder die Applikaton mit einer anderen FMTool-Version kompilieren (ändert die verlangte OS-Version) oder die FBL-Version der Applikation wechseln.

201 **Zu viele Tasks**
Die Applikation enthält zu viele Tasks und kann deshalb nicht gestartet werden.

Massnahme: Die Anzahl Tasks dieser Applikation muss mit dem FMTool vermindert werden.

202 **Zu viele Eingangs- oder Ausgangssignale**
Die Applikation enthält zu viele Eingangs- oder Ausgangsadressen.

Massnahme: Das Projekt muss mit dem FMTool redimensioniert oder auf zwei Funktionsmodule aufgeteilt werden.

203 **Fehler beim kreieren eines Tasks**
Es sind Probleme beim kreieren eines Tasks aufgetreten.

Massnahme: Die Anzahl Tasks muss mit dem FMTool vermindert werden.

Systemfehler

ID Beschreibung

99 **Funktionsmodul-Überlastung**
Ein Task brauchte für seine Abarbeitung länger, als seine Zykluszeit dauert.

Massnahme: Das Projekt muss verkleinert werden oder die die Taskzykluszeit muss vergrössert werden.

1000 **Ungültiger Entscheidungswert**
In einer CASE-Verarbeitung wurde ein ungültiger Eingangswert entdeckt. Dies deutet auf ein Sytemfehler hin. Bitte benachrichtigen Sie Ihren Vertreiber.

9999 **Überlauf des Exceptionpuffers**
Der Fehler-Puffer ist voll. Die weiteren Fehlermeldungen konnten nicht mehr gespeichert werden.

Massnahme: Die Fehlermeldungen auswerten und geeignete Lösungen ausarbeiten, damit diese Fehler nicht mehr auftreten.
Durch das Auslesen derFehlermeldungen wird der Puffer wieder geleert.

Technische Dokumentation

FMTool
und
Funktionsmodul

Anhang D

Release-Notes zu FMTool Version 1.50, Funktionsmodul Betriebssystem OS 2.10 und Funktionsblockbibliothek FBL Version 1.51

Hinweise

Die in diesem Unterlagen enthaltenen Angaben, Daten, Werte usw. können ohne vorherige Ankündigung geändert werden. Ebenso sind die Abbildungen unverbindlich.

Gebrüder Merten GmbH & Co. KG
Elektrotechnik • Elektronik
Fritz-Kotz-Str. 8
D-51674 Wiehl

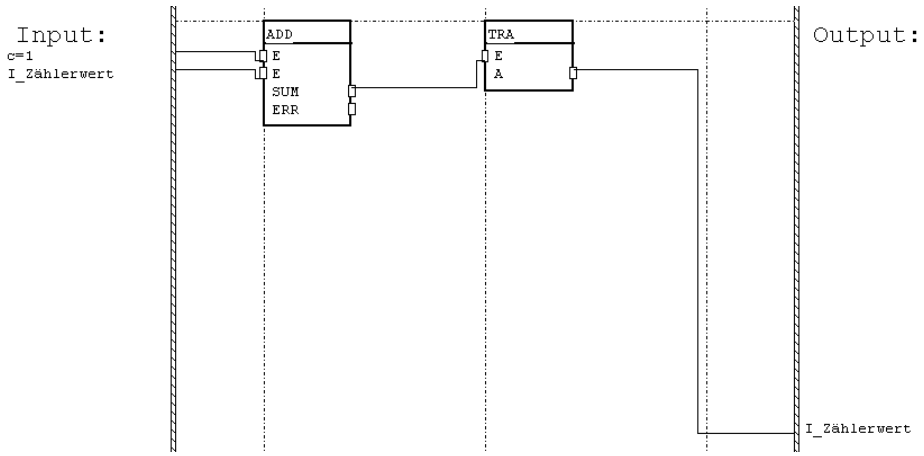
Telefon 0 22 61 / 702-01
Telefax 0 22 61 / 702-284

Release-Notes

Rückführungen von Signalen

Bei der Funktionsblockbibliothek (FBL) Version 1.51 gibt es Blöcke, die bei der Berechnung den Ausgang als Zwischenspeicher verwenden. Dies kann zu unerwarteten Ergebnissen kommen, falls die Ausgangssignale auf die Eingänge zurückgeführt werden.

Korrektur:
Zwischen den entsprechenden Funktionsblock und das Ausgangssignal ist ein TRA-Block zu schalten.



Liste der betroffenen Blöcke
Die folgende Liste enthält alle Funktionsblöcke, die Probleme bei rückgeführten Signalen machen. Es ist jeweils angegeben, welches die kritischen Ausgänge sind und welche Datentypen betroffen sind.

Funktionsblock	Datentyp	Ausgang
DIV	Value	QUOT
DT1		A
DMFF		A
INTE	alle	A
MUL	alle	A
PI		A
PT1		A
RSFF		A
SMFF		A
TFF		A

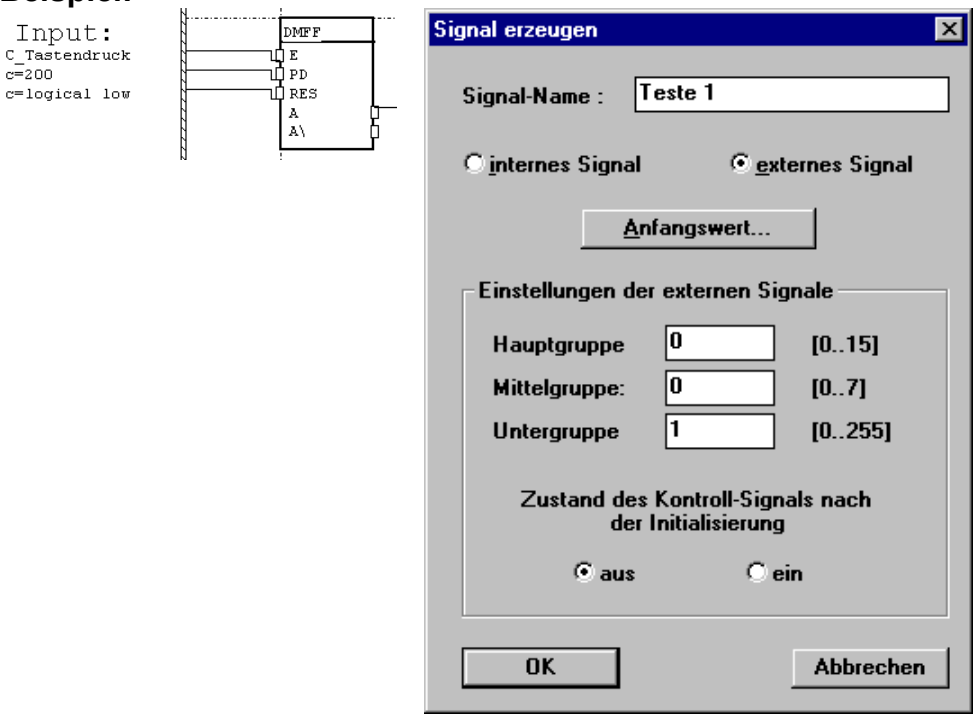
Initialisierung der Eingangs-Kontrollsignale

Die Eingangs-Kontrollsignale dürfen nicht initialisiert werden, d.h., die Einstellung ‘Zustand des Kontroll-Signals nach der Initialisierung’ muss immer auf ‘aus’ sein.

Ist die Einstellung auf ‘ein’ gesetzt, empfängt das Funktionsmodul dieses Signal nicht. Es generiert eine Fehlermeldung (ID 12; Zu viele Telegramme empfangen) falls ein Telegramm mit dieser Gruppenadresse auf den Bus gesendet wird.

Dies gilt für die Funktionsmodul-Betriebssystem Versionen OS 2.10 und alle älteren Versionen (tiefere Versionsnummern).

Beispiel:



Zusätzliche Informationen zu FBL 1.04 und älter

Benutzung der Ausgangs-Kontrollsignale

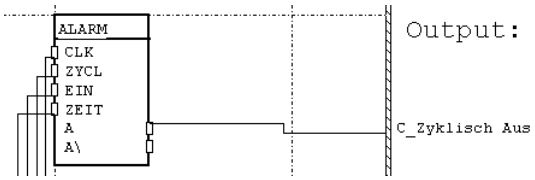
Wird die Funktionsblockbibliothek (FBL) mit der Version 1.04 oder älter verwendet, dürfen nicht alle Funktionsblockausgänge an ein Ausgangs-Kontrollsignal angeschlossen werden. Die folgende Liste enthält alle Funktionsblöcke mit den betroffenen Ausgängen, bei welchen Funktionsstörungen auftreten können. Bei allen nicht erwähnten Ausgängen treten keine Probleme auf.

Block	Ausgang	Was geschieht wenn der Ausgang an ein Kontrollsignal angeschlossen wird
AAS	A	Ausgänge die nicht durch den SEL-Eingang angewählt sind, werden durch das Anschliessen des Kontrollsignals automatisch immer auf 0 gesetzt. Der ausgewählte Ausgang hat immer den Wert des Einganges.
AASZ	A	Der Ausgang der nicht durch den SEL-Eingang angewählt ist, wird durch das Anschliessen des Kontrollsignals automatisch immer auf 0 gesetzt. Der ausgewählte Ausgang hat immer den Wert des Einganges.
ALARM	A	Wenn A mit einem Kontrollsignal verbunden ist, funktioniert das zyklische wiederholen des Pulses am Ausgang nicht (Eingang ZYKL auf 1).
DMFF	A	Wenn A mit einem Kontrollsignal verbunden ist, ist der Ausgang nur einen Taskzyklus lang auf 1. Der Ausgang A\ ist dann auch nur einen Taskzyklus lang auf 0. Ist der Eingang PD auf 1 gesetzt, treten keine Funktionsstörungen auf.
OSZI	A	Wenn A mit einem Kontrollsignal verbunden ist, ist der Ausgang nur einen Taskzyklus lang auf 1. Die Zykluszeit ändert sich aber nicht. Der Ausgang A\ ist dann auch nur einen Taskzyklus lang auf 0. Ist der Eingang PW auf 1 gesetzt, treten keine Funktionsstörungen auf.
RSFF	A	Wenn A mit einem Kontrollsignal verbunden ist, ist der Ausgang nur einen Taskzyklus lang auf 1. Der Ausgang A\ ist dann auch nur einen Taskzyklus lang auf 0.
SMFF	A	Wenn A mit einem Kontrollsignal verbunden ist, ist der Ausgang nur einen Taskzyklus lang auf 1. Der Ausgang A\ ist dann auch nur einen Taskzyklus lang auf 0. Ist der Eingang PD auf 1 gesetzt, treten keine Funktionsstörungen auf.
TFF	A	Wenn A mit einem Kontrollsignal verbunden ist, ist der Ausgang nur einen Taskzyklus lang auf 1. Der Ausgang A\ ist dann auch nur einen Taskzyklus lang auf 0.

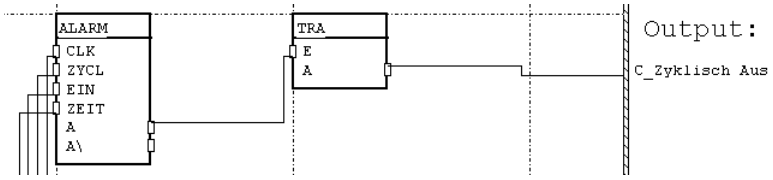
Dieses Problem kann umgangen werden, wenn ein TRA-Block zwischen dem Block-Ausgang und dem Kontrollsignal eingefügt wird.

Beispiel:

Falsch:



Korrektur:



Rückführungen von Signalen

Bei FBL 1.04 und älter gilt das Kapitel ‘Rückführung von Signalen’ dieses Anhangs zusätzlich für die Funktionsblöcke ADD, SUB und VERZ.