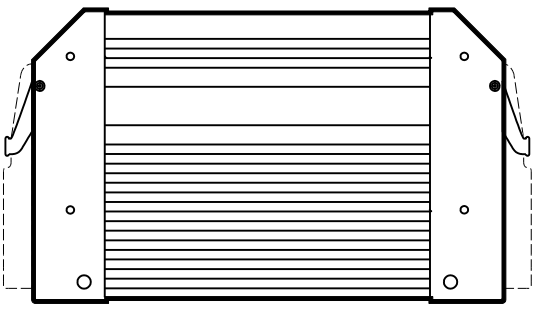
Original-Programmierhandbuch
ExtendedController

CR0232

Laufzeitsystem V01.00.03
CODESYS® V2.3

Deutsch

7390673 / 08 01 / 2018



Inhaltsverzeichnis

1 Vorbemerkung

Inhalt

202

1.1 Copyright

6088

Alle auf unseren Seiten verwendeten Produktnamen, -Bilder, Unternehmen oder sonstige Marken sind Eigentum der jeweiligen Rechteinhaber:

- AS-i ist Eigentum der AS-International Association, (→ www.as-interface.net)
- CAN ist Eigentum der CiA (CAN in Automation e.V.), Deutschland (→ www.can-cia.org)
- CODESYS™ ist Eigentum der 3S – Smart Software Solutions GmbH, Deutschland (→ www.codesys.com)
- DeviceNet™ ist Eigentum der ODVA™ (Open DeviceNet Vendor Association), USA (→ www.odva.org)
- EtherNet/IP® ist Eigentum der →ODVA™
- EtherCAT® ist eine eingetragene Marke und patentierte Technologie, lizenziert durch die Beckhoff Automation GmbH, Deutschland
- IO-Link® (→ www.io-link.com) ist Eigentum der →PROFIBUS Nutzerorganisation e.V., Deutschland
- ISOBUS ist Eigentum der AEF – Agricultural Industry Electronics Foundation e.V., Deutschland (→ www.aef-online.org)
- Microsoft® ist Eigentum der Microsoft Corporation, USA (→ www.microsoft.com)
- PROFIBUS® ist Eigentum der PROFIBUS Nutzerorganisation e.V., Deutschland (→ www.profibus.com)
- PROFINET® ist Eigentum der →PROFIBUS Nutzerorganisation e.V., Deutschland
- Windows® ist Eigentum der →Microsoft Corporation, USA

1.2 Übersicht: Anwender-Dokumentation für

22853

Die Dokumentation für das Gerät besteht aus folgenden Modulen:
(Downloads von der Homepage → **ifm weltweit • ifm worldwide • ifm à l'échelle internationale** (→ S. 264))

Dokument	Inhalt / Beschreibung
Datenblatt	Technische Daten in Tabellenform
Montageanleitung (gehört zum Lieferumfang des Geräts)	• Anleitung für Montage, elektrische Installation und Inbetriebnahme • Technische Daten
Programmierhandbuch	• Funktionen des Setup-Menüs des Gerät • Erstellen eines CODESYS-Projekts mit diesem Gerät • Zielsystem einstellen mit CODESYS • Geräteinterne SPS mit CODESYS programmieren • Beschreibung der gerätespezifischen CODESYS-Funktionsbibliotheken
Systemhandbuch "Know-How ecomatmobile"	Hintergrundwissen zu folgenden Themen (Beispiele): • Übersicht Templates und Demo-Programme • CAN, CANopen • Ausgänge steuern • Visualisierungen • Übersicht Dateien und Bibliotheken

1.3 Was bedeuten die Symbole und Formatierungen?

203

Folgende Symbole oder Piktogramme verdeutlichen Ihnen unsere Hinweise in unseren Anleitungen:

⚠ WARNUNG

Tod oder schwere irreversible Verletzungen sind möglich.

⚠ VORSICHT

Leichte reversible Verletzungen sind möglich.

ACHTUNG

Sachschaden ist zu erwarten oder möglich.

	Wichtiger Hinweis Fehlfunktionen oder Störungen sind bei Nichtbeachtung möglich
	Information Ergänzender Hinweis
► ...	Handlungsaufforderung
> ...	Reaktion, Ergebnis
→ ...	"siehe"
abc	Querverweis
123 0x123 0b010	Dezimalzahl Hexadezimalzahl Binärzahl
[...]	Bezeichnung von Tasten, Schaltflächen oder Anzeigen

1.4 Wie ist diese Dokumentation aufgebaut?

204
1508

Diese Dokumentation ist eine Kombination aus verschiedenen Anleitungstypen. Sie ist eine Lernanleitung für den Einsteiger, aber gleichzeitig auch eine Nachschlageanleitung für den versierten Anwender. Dieses Dokument richtet sich an die Programmierer der Anwendungen.

Und so finden Sie sich zurecht:

- Um gezielt zu einem bestimmten Thema zu gelangen, benutzen Sie bitte das Inhaltsverzeichnis.
- Mit dem Stichwortregister "Index" gelangen Sie ebenfalls schnell zu einem gesuchten Begriff.
- Am Anfang eines Kapitels geben wir Ihnen eine kurze Übersicht über dessen Inhalt.
- Abkürzungen und Fachbegriffe → Anhang.

Bei Fehlfunktionen oder Unklarheiten setzen Sie sich bitte mit dem Hersteller in Verbindung:
Kontakt → **ifm weltweit • ifm worldwide • ifm à l'échelle internationale** (→ S. 264)

Wir wollen immer besser werden! Jeder eigenständige Abschnitt enthält in der rechten oberen Ecke eine Identifikationsnummer. Wenn Sie uns über Unstimmigkeiten unterrichten wollen, dann nennen Sie uns bitte diese Nummer zusammen mit Titel und Sprache dieser Dokumentation. Vielen Dank für Ihre Unterstützung!

Im Übrigen behalten wir uns Änderungen vor, so dass sich Abweichungen vom Inhalt der vorliegenden Dokumentation ergeben können. Die aktuelle Version finden Sie auf der ifm-Homepage:
→ **ifm weltweit • ifm worldwide • ifm à l'échelle internationale** (→ S. 264)

1.5 Historie der Anleitung (CR0232)

9188

Was hat sich wann in dieser Anleitung geändert? Ein Überblick:

Datum	Thema	Änderung
2010-09-13	Konfigurationen Q16_MODE_E...Q31_MODE_E	Voreinstellwert korrigiert
2010-11-10	Abschlusswiderstände	Korrektur in Topic 1244
2011-02-14	TIMER_READ_US (FB)	Umrechnung max. Zählwert korrigiert
2011-04-05	Speicherbausteine FRAMREAD, FRAMWRITE, FLASHREAD, FLASHWRITE	zulässige Werte der Parameter SRC, LEN, DST
2011-04-13	CANopen Übersicht	neu: CANopen-Tabellen im Anhang
2011-12-13	INPUT_ANALOG	Parameter MODE
2012-10-04	diverse	Korrekturen
2013-06-24	diverse	neue Dokumentenstruktur
2014-04-28	diverse FBs	Beschreibung FB-Eingang CHANNEL präzisiert
2014-06-24	FB PID2	Grafik korrigiert
2014-06-30	Name der Dokumentation	"Systemhandbuch" umbenannt zu "Programmierhandbuch"
2014-07-04	Geräte-Ausgang ERROR (Klemme 13)	Ausgang ist nicht vorhanden. Hinweise darauf entfernt.
2014-07-31	FB PHASE	Beschreibung Parameter der Ausgänge C, ET korrigiert
2014-07-31	FB OUTPUT_CURRENT_CONTROL	Wenn Sollwert=0 mA >> Regelung auf 0 "innerhalb von 100 ms" anstatt "sofort"
2014-08-26	Beschreibung Eingänge, Ausgänge	highside / lowside ersetzt durch plusschaltend / minusschaltend

Datum	Thema	Änderung
2014-11-12	Kapitel "Ausgänge (Technologie)"	Abschnitt "Diagnose der binären Ausgänge" ergänzt oder korrigiert
2015-01-13	Dokumentationsstruktur Fehlercodes, Systemmerker	• Fehlermerker: nur noch im Anhang, Kapitel **Systemmerker** • CAN / CANopen Fehler und Fehlerbehandlung: nur noch im Systemhandbuch "Know-How" • Fehlercodes, EMCY-Codes: nun im Anhang, Kapitel **Fehler-Tabellen**
2015-03-10	Verfügbarer Speicher	Darstellung verbessert
2015-05-26	FB J1939_x_GLOBAL_REQUEST	Beschreibung präzisiert
2015-06-10	diverse FBs	Beschreibung FB-Eingang CHANNEL korrigiert
2015-07-27	FB GET_IDENTITY	ergänzt um Ausgang SERIALNUMBER
2015-07-27	FB GET_IDENTITY_EIOS	neu
2015-09-22	FB GET_IDENTITY_EIOS	Korrektur
2015-09-22	englisches Handbuch	defekte Grafiken erneuert
2015-10-22	Systemmerker-Bit SERIAL_MODE	Debugging des Anwenderprogramms über USB nicht möglich
2016-04-27	FBs für schnelle Eingänge	Hinweis im Fall von höheren Frequenzen ergänzt
2017-01-13	Software-Handbuch für CODESYS 2.3	Hinweis auf Download von ifm-Homepage entfernt
2017-06-02	FRAM, MEMCPY, MEMSET: Angabe für "vom Anwender frei verfügbarer remanenter Speicher"	aus Handbuch entfernt, weil Wert und Startadresse von Hardware und Software des Geräts abhängen
2017-06-02	Schnelle Eingänge	Innenwiderstand der Signalquelle muss wesentlich geringer sein als der Eingangswiderstand des verwendeten Eingangs
2017-06-02	Adressen / Variablen der E/As	Beschreibung der Eingangs- und Ausgangsbytes entfernt, weil ungültig
2017-12-18	Liste der ifm-Niederlassungen	aktualisiert

2 Sicherheitshinweise

Inhalt

213

2.1 Beachten!

214
11212

Mit den in dieser Anleitung gegebenen Informationen, Hinweisen und Beispielen werden keine Eigenschaften zugesichert. Die abgebildeten Zeichnungen, Darstellungen und Beispiele enthalten weder Systemverantwortung noch anwendungsspezifische Besonderheiten.

- ► Die Sicherheit der Maschine/Anlage muss auf jeden Fall eigenverantwortlich durch den Hersteller der Maschine/Anlage gewährleistet werden.
- ► Beachten Sie die nationalen Vorschriften des Landes, in welchem die Maschine/Anlage in Verkehr gebracht werden soll!

⚠ WARNUNG

Bei Nichtbeachten der Hinweise in dieser Anleitung sind Sach- oder Körperschäden möglich!
Die **ifm electronic gmbh** übernimmt hierfür keine Haftung.

- ► Die handelnde Person muss vor allen Arbeiten an und mit diesem Gerät die Sicherheitshinweise und die betreffenden Kapitel dieser Anleitung gelesen und verstanden haben.
- ► Die handelnde Person muss zu Arbeiten an der Maschine/Anlage autorisiert sein.
- ► Die handelnde Person muss für die auszuführende Arbeit über die erforderliche Ausbildung und Qualifikation verfügen.
- ► Beachten Sie die Technischen Daten der betroffenen Geräte!
  Das aktuelle Datenblatt finden Sie auf der **ifm**-Homepage.
- ► Beachten Sie die Montage- und Anschlussbedingungen sowie die bestimmungsgemäße Verwendung der betroffenen Geräte!
  → mitgelieferte Montageanleitung oder auf der **ifm**-Homepage
- ► Beachten Sie die Korrekturen und Hinweise in den "Release-Notes" zur vorhandenen Hardware, Software und Dokumentation auf der **ifm**-Homepage

Homepage → **ifm weltweit • ifm worldwide • ifm à l'échelle internationale** (→ S. 264)

5020

ACHTUNG

Der Treiberbaustein der seriellen Schnittstelle kann beschädigt werden!

Beim Trennen oder Verbinden der seriellen Schnittstelle unter Spannung kann es zu undefinierten Zuständen kommen, die zu einer Schädigung des Treiberbausteins führen.

- ► Die serielle Schnittstelle nur im spannungslosen Zustand trennen oder verbinden!

2.2 Welche Vorkenntnisse sind notwendig?

215

Das Dokument richtet sich an Personen, die über Kenntnisse der Steuerungstechnik und SPS-Programmierkenntnisse mit IEC 61131-3 verfügen.

Zum Programmieren der SPS sollten die Personen zusätzlich mit der Software CODESYS vertraut sein.

Das Dokument richtet sich an Fachkräfte. Dabei handelt es sich um Personen, die aufgrund ihrer einschlägigen Ausbildung und ihrer Erfahrung befähigt sind, Risiken zu erkennen und mögliche Gefährdungen zu vermeiden, die der Betrieb oder die Instandhaltung eines Produkts verursachen kann. Das Dokument enthält Angaben zum korrekten Umgang mit dem Produkt.

Lesen Sie dieses Dokument vor dem Einsatz, damit Sie mit Einsatzbedingungen, Installation und Betrieb vertraut werden. Bewahren Sie das Dokument während der gesamten Einsatzdauer des Gerätes auf.

Befolgen Sie die Sicherheitshinweise.

2.3 Anlaufverhalten der Steuerung

6827
15233
11575

⚠ WARNUNG

Gefahr durch unbeabsichtigtes und gefährliches Anlaufen von Maschinen- oder Anlagenteilen!

- ► Der Programmierer muss bei der Programmerstellung verhindern, dass nach Auftreten eines Fehlers (z.B. NOT-HALT) und der anschließenden Fehlerbeseitigung unbeabsichtigt Maschinen- oder Anlagenteile gefährlich anlaufen können!
  ⇨ Wiederanlaufsperre realisieren!
- ► Dazu im Fehlerfall die in Frage kommenden Ausgänge im Programm logisch abschalten!

Ein Wiederanlauf kann z.B. verursacht werden durch:
- Spannungswiederkehr nach Spannungsausfall
- Reset nach Watchdog-Ansprechen wegen zu langer Zykluszeit
- Fehlerbeseitigung nach NOT-HALT

So erreichen Sie sicheres Verhalten der Steuerung:

- ► Spannungsversorgung im Anwendungsprogramm überwachen.
- ► Im Fehlerfall alle relevanten Ausgänge im Anwendungsprogramm ausschalten.
- ► Aktuatoren, die zu gefahrbringenden Bewegungen führen können, zusätzlich im Anwendungsprogramm überwachen (Feedback).
- ► Relaiskontakte, die zu gefahrbringenden Bewegungen führen können, zusätzlich im Anwendungsprogramm überwachen (Feedback).
- ► Bei Bedarf im Anwendungsprojekt sicherstellen, dass verschweißte Relaiskontakte keine gefahrbringenden Bewegungen auslösen oder fortführen können.

2.4 Hinweise: Seriennummer

20780

- ► In der Fertigung des Anwenders einen Netzwerkplan mit allen Steuerungen in der Maschine erstellen. In den Netzwerkplan die Seriennummer jeder verbauten Steuerung eintragen.
- ► Vor dem Download einer Software-Komponente diese Seriennummer auslesen und mit Hilfe des Netzwerkplans prüfen, dass man auf die richtige Steuerung zugreift.

2.5 Hinweise: TEST-Eingänge

20781

- ► Die TEST-Eingänge aller Steuerungen in der Maschine einzeln verdrahten und eindeutig markieren, so dass eine Zuordnung zu den Steuerungen eindeutig hergestellt werden kann.
- ► Bei einem Maintenance-Zugriff darf immer nur der TEST-Eingang der Steuerung aktiviert werden, auf die zugegriffen werden soll.

3 Systembeschreibung

Inhalt

975

3.1 Angaben zum Gerät

6269

Diese Anleitung beschreibt aus der Gerätefamilie für den mobilen Einsatz, **ecomat*mobile*** der **ifm electronic gmbh**:

- ExtendedController: CR0232

3.2 Hardware-Beschreibung

Inhalt

14081

3.2.1 Hardwareaufbau

Inhalt

15332

Startvoraussetzung

19658

Das Gerät startet erst, wenn am Versorgungsanschluss VBBs (unter anderem Versorgung der Relais auf der Standardseite) und an Klemme 15 eine ausreichende Spannung anliegt.
Klemme 15 ist in Fahrzeugen die vom Zündschloss geschaltete Plusleitung.

- zulässiger Versorgungsspannungsbereich = 8...32 V
- Einschaltbedingung: VBBs > 10 V

Relais

19663

Der ExtendedController verfügt über 4 interne Ausgangsrelais:
- Standard-Seite: 2 Relais trennen jeweils 8 Ausgänge von der Klemmenspannung VBBx (x=o|r),
- Extended-Seite: 2 Relais trennen jeweils 16 Ausgänge von der Klemmenspannung VBBx (x=1|2|3|4).

Das Trennen erfolgt mit Ausschalten der Relais.

Die Relais werden nur unter folgender Voraussetzung aktiviert:
- das globale Bit ERROR = FALSE

UND
- das Bit RELAIS_VBBx = TRUE

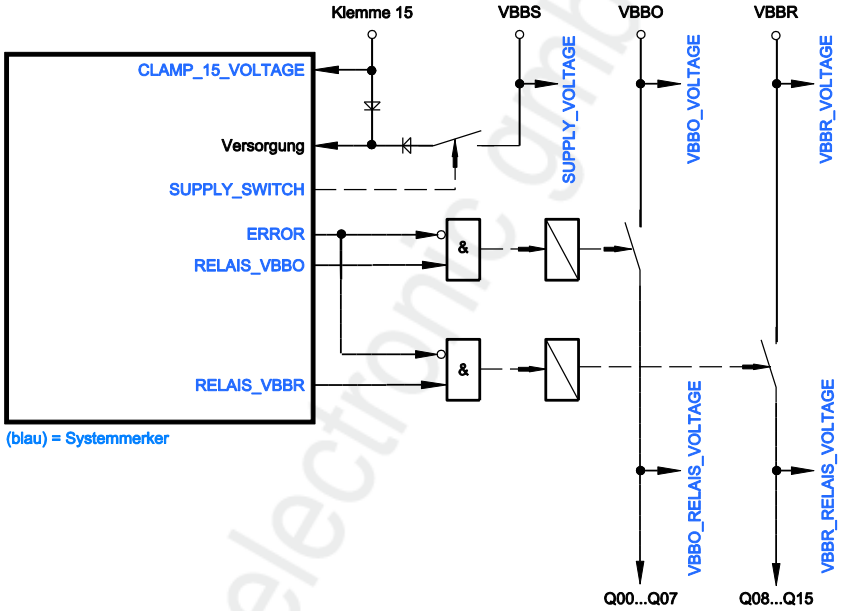
Im aktivierten Zustand legen die Relaiskontakte die Ausgänge an die Klemmenspannung VBBx.

! Zugehörige Ausgänge erst ≥ 45 ms nach Einschalten der Relais aktivieren!

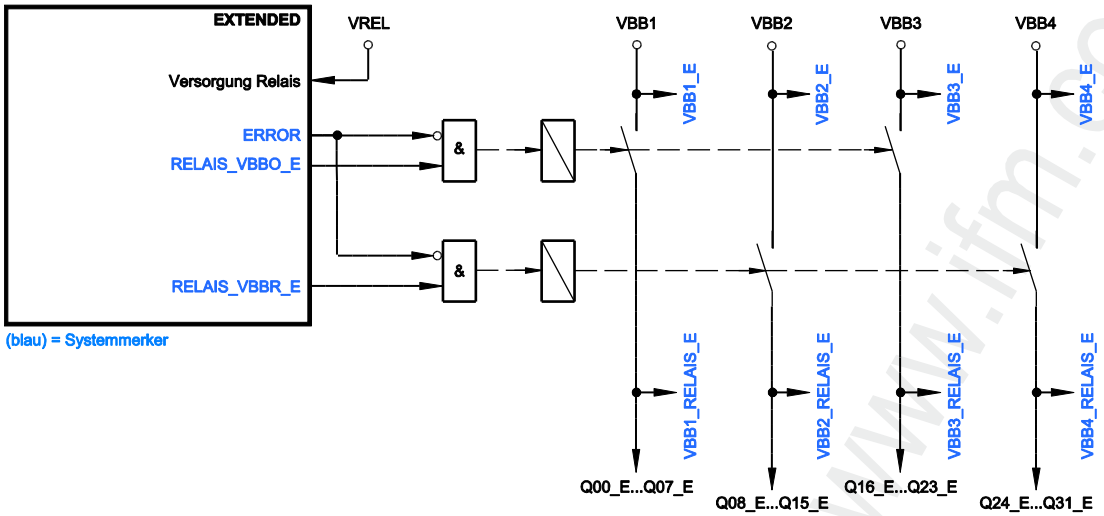
Prinzipschaltung

19664

Aus den nachfolgenden Prinzipschaltbildern kann die Abhängigkeit der Relais von den anliegenden Signalen und den logischen Zuständen der Systemmerker entnommen werden.



Grafik: Prinzipaufbau der Versorgung und der Relais (Standard-Seite)



Grafik: Prinzipaufbau der Versorgung und der Relais (Extended-Seite)

Verfügbarer Speicher

13736

FLASH-Speicher

8136

FLASH-Speicher (nichtflüchtiger, langsamer Speicher) insgesamt im Gerät vorhanden	2 176 kByte

Davon sind folgende Speicherbereiche reserviert für ...

maximale Größe für das Anwendungsprogramm	1 280 kByte
Daten außerhalb des Anwendungsprogramms Anwender kann Daten speichern, z.B. Files, Bitmaps, Fonts	128 kByte
Daten außerhalb des Anwendungsprogramms Daten mit **FLASHREAD** (→ S. 195) lesen oder mit **FLASHWRITE** (→ S. 196) schreiben (bei Files: abzüglich 128 Byte für Header)	64 kByte

Der verbleibende Speicher ist reserviert für system-interne Zwecke.

SRAM

8360

SRAM (flüchtiger, schneller Speicher) insgesamt im Gerät vorhanden SRAM steht hier allgemein für alle Arten von flüchtigen, schnellen Speichern.	2 216 kByte

Davon sind folgende Speicherbereiche reserviert für ...

vom Anwendungsprogramm reservierte Daten	192 kByte

Der verbleibende Speicher ist reserviert für system-interne Zwecke.

FRAM

19547

FRAM (nichtflüchtiger, schneller Speicher) insgesamt im Gerät vorhanden FRAM steht hier allgemein für alle Arten von nichtflüchtigen, schnellen Speichern.	128 kByte

Davon sind folgende Speicherbereiche reserviert für ...

im Anwendungsprogramm als VAR_RETAIN deklarierte Variablen	4 kByte
als remanent definierte Merker (ab %MB0...) ▶ Ende des Speicherbereichs im FB **MEMORY_RETAIN_PARAM** (→ S. 192) angeben!	4 kByte

Der verbleibende Speicher ist reserviert für system-interne Zwecke.

3.2.2 Funktionsweise der verzögerten Abschaltung

993

Werden die Controller von der Versorgungsspannung getrennt, werden im Normalfall sofort alle Ausgänge abgeschaltet, keine Eingangssignale mehr eingelesen und die Abarbeitung der Steuerungssoftware (Laufzeitsystem und Anwendungsprogramm) abgebrochen. Dieses geschieht unabhängig davon, in welchem Programmschritt sich der Controller befindet.

Wenn dieses Verhalten nicht gewünscht ist, muss der Controller programmgesteuert abgeschaltet werden. Das ermöglicht nach Abschalten der Zündung zum Beispiel das Sichern von Speicherständen.

Die ClassicController können durch eine entsprechende Beschaltung der Versorgungsspannungs-Eingänge und die Auswertung der zugehörigen Systemmerker, programmgesteuert abgeschaltet werden. Das Prinzipschaltbild im Kapitel **Hardwareaufbau** (→ S. 13) zeigt schematisch die Zusammenhänge der einzelnen Strompfade.

Klemme VBB15 mit Zündschalter verbinden

2418

Über die Klemme VBB15 wird die interne Steuerungselektronik initialisiert, wenn an Klemme VBBs Versorgungsspannung anliegt.

Diese Klemmen VBB15 und VBBs werden intern überwacht. Die anliegende Klemmenspannung VBB15 kann über den Systemmerker CLAMP_15_VOLTAGE überwacht werden. Die anliegende Klemmenspannung VBBs kann über den Systemmerker SUPPLY_VOLTAGE überwacht werden.

Selbsthaltung

2419

Einschalten der Steuerung:

- Der Zündschalter legt Spannung an VBB15 (Klemme 15*).
- Der Systemmerker CLAMP_15_VOLTAGE erkennt die angelegte Spannung und aktiviert den Systemmerker SUPPLY_SWITCH.
- SUPPLY_SWITCH aktiviert die Verbindung zum Potential VBBs.

> Somit ist der Zündschalter überbrückt, die Selbsthaltung der Steuerspannung ist hergestellt.

Ausschalten der Steuerung über Klemme 15:

- Der Systemmerker CLAMP_15_VOLTAGE erkennt das Abschalten der Versorgungsspannung an Klemme VBB15.

► Im Anwendungsprogramm den Systemmerker SUPPLY_SWITCH zurücksetzen.

> Die Selbsthaltung über VBBs ist aufgehoben und der Controller wird vollständig abgeschaltet.

*) Klemme 15 ist in Fahrzeugen die vom Zündschloss geschaltete Plusleitung.

3.2.3 Relais: wichtige Hinweise!

12976

Zuordnung Relais – Potentiale: → Datenblatt
Max. Summenstrom je Relaiskontakt (= je Ausgangsgruppe): → Datenblatt

ACHTUNG

Gefahr der Zerstörung der Relaiskontakte!
"Klebende" Relaiskontakte können auch im Notfall nicht mehr die Ausgänge von der Versorgung trennen!

Falls VBBS (VBBrel) und Klemme 15 gleichzeitig von der Versorgung getrennt werden, jedoch die Potentiale VBBx an der Versorgung angeschlossen bleiben, dann können die Relais schon abfallen, bevor die Ausgänge vom System deaktiviert werden.

In diesem Fall trennen die Relais **unter Last** die Ausgänge von der Versorgung. Dies schränkt die Lebensdauer der Relais deutlich ein.

► Bei dauerhaftem Anschluss von VBBx an Versorgung:
  • auch VBBS (VBBrel) dauerhaft anschließen und
  • die Ausgänge programmgesteuert mit Hilfe von Klemme 15 abschalten.

3.2.4 Überwachungskonzept

Inhalt

991

Die Steuerung überwacht die Versorgungsspannungen und die System-Fehlermerker.
Je nach Zustand ...
- die Steuerung schaltet die internen Relais ab
  > die Ausgänge werden stromlos, behalten aber ihren logischen Zustand
  > das Programm läuft weiter

oder:
- die Steuerung schaltet vollständig ab
  > das Programm stoppt
  > die Ausgänge werden stromlos und gehen auf logisch "0"
  > die Status-LED erlischt

Überwachung der Versorgungsspannungen

6752

Im Fehlerfall unterscheiden wir 2 Szenarien:

Klemmenspannung VBBx fällt unter den Grenzwert von 5,25 V

15752

> Die Steuerung erkennt Unterspannung. Die von der Klemmenspannung VBBx versorgten Ausgänge werden deaktiviert.

> Erholt sich die Klemmenspannung und befindet sich wieder im regulären Bereich (> 10 V), werden die Ausgänge wieder aktiviert.

13975

⚠ WARNUNG

Gefährlicher Wiederanlauf möglich!
Gefahr von Personenschaden! Gefahr von Sachschaden an der Maschine/Anlage!

Wird ein Ausgang im Fehlerfall hardwaremäßig abgeschaltet, ändert sich der durch das Anwendungsprogramm erzeugte logische Zustand dadurch nicht.

► Abhilfe:
- Die Ausgänge zunächst im Anwendungsprogramm logisch zurücksetzen!
- Fehler beseitigen!
- Ausgänge situationsabhängig wieder setzen.

Versorgungsspannung VBBs fällt unter den Grenzwert von 10 V

20638

> Die Steuerung läuft weiter, bis die Spannung so weit gefallen ist, dass die daraus erzeugten internen Spannungen einbrechen.

Unterhalb von 10 V werden keine Retain-Daten gespeichert. → Merker RETAIN_WARNING

> Brechen die internen Spannungen ein, geht der Controller in den Reset.
Die Ausführung von Laufzeitsystem und Anwendungsprogramm wird abgebrochen.
Dies geschieht unabhängig davon, in welchem Programmschritt sich die Steuerung befindet.

> Ein Wiederanlauf der Steuerung erfolgt erst, wenn die Versorgungsspannungen wieder oberhalb des Grenzwerts sind.

Überwachungs- und Sicherungsmechanismen

2421

⚠ WARNUNG

Gefahr durch unbeabsichtigtes Abschalten aller Ausgänge!

Falls Überwachungsroutinen einen Systemfehler feststellen:
> das Gerät schaltet die Energie für alle Ausgänge aus.

Während des Programmablaufes stehen die Ausgangsrelais unter voller Software-Kontrolle des Anwenders. So kann z.B. ein paralleler Kontakt der Sicherheitskette als Eingangssignal ausgewertet und das Ausgangsrelais entsprechend abgeschaltet werden. Zur weiteren Sicherheit müssen die entsprechenden nationalen Vorschriften beachtet werden.

Tritt während des Programmablaufs ein Fehler auf, können durch das Systemmerker-Bit ERROR die Relais spannungsfrei geschaltet werden, um kritische Anlagenteile abzutrennen.

Manuelles Setzen von einem Merker-Bit ERROR_VBB... hat KEINE Auswirkungen auf die Relais!

11575

⚠ WARNUNG

Gefahr durch unbeabsichtigtes und gefährliches Anlaufen von Maschinen- oder Anlagenteilen!

- Der Programmierer muss bei der Programmerstellung verhindern, dass nach Auftreten eines Fehlers (z.B. NOT-HALT) und der anschließenden Fehlerbeseitigung unbeabsichtigt Maschinen- oder Anlagenteile gefährlich anlaufen können!
  ⇨ Wiederanlaufsperre realisieren!
- Dazu im Fehlerfall die in Frage kommenden Ausgänge im Programm logisch abschalten!

Bei Auftreten eines Watchdog-Fehlers ...
> die Programmabarbeitung wird automatisch unterbrochen
> die Ausgänge werden stromlos und gehen auf logisch "0"
> der Controller wird zurückgesetzt
> der Controller startet anschließend neu, wie nach einem Power-On.

Referenzspannungsausgang

2250
13934

Der Referenzspannungsausgang dient der Versorgung von Sensoren mit einer stabilen Spannung, die nicht den Schwankungen der Versorgungsspannung unterworfen ist.

13402

ACHTUNG

Referenzspannungsausgang kann beschädigt werden!

► Von außen KEINE Spannung anlegen!

Über die binären Systemvariablen REFERENCE_VOLTAGE_5 oder REFERENCE_VOLTAGE_10 wird die Spannung am Referenzspannungsausgang [V_{REF} OUT] eingestellt:

REFERENCE_VOLTAGE_10	REFERENCE_VOLTAGE_5	Referenzspannung [V_{REF} OUT]
FALSE	FALSE	0 V
FALSE	TRUE	5 V
TRUE	FALSE	10 V
TRUE	TRUE	0 V

► Wenn Referenzspannungsausgang = 10 V gewählt:
die Steuerung mit mindestens 13 V versorgen!

► Überwachen der Spannung am Referenzspannungsausgang mit Systemvariable REF_VOLTAGE.

> Wenn Systemvariable ERROR = TRUE:
der Referenzspannungsausgang wird deaktiviert (Ausgabe = 0 V).

3.2.5 Eingänge (Technologie)

Inhalt

14090

Analogeingänge

2426

Die Analogeingänge können über das Anwendungsprogramm konfiguriert werden. Der Messbereich kann zwischen folgenden Bereichen umgeschaltet werden:

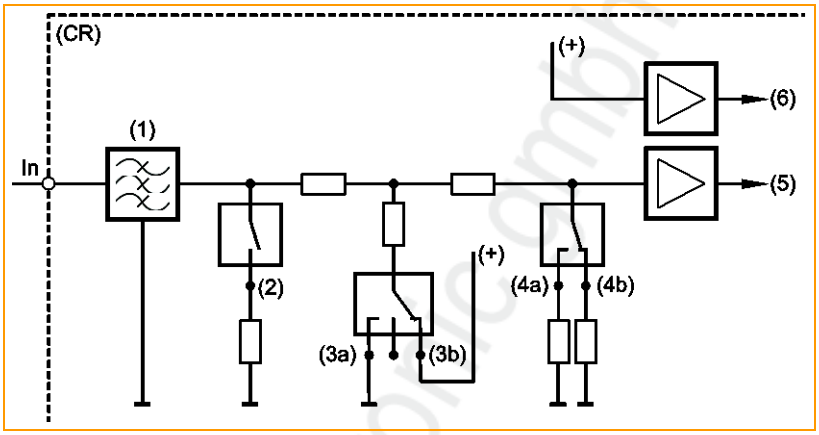
- Stromeingang 0...20 mA
- Spannungseingang 0...10 V
- Spannungseingang 0...32 V

Die Spannungsmessung kann auch ratiometrisch erfolgen (0...1000 ‰, über FBs einstellbar). Das bedeutet, ohne zusätzliche Referenzspannung können Potentiometer oder Joysticks ausgewertet werden. Ein Schwanken der Versorgungsspannung hat auf diesen Messwert keinen Einfluss.

Alternativ kann ein Analog-Kanal auch binär ausgewertet werden.

> Bei ratiometrischer Messung müssen die angeschlossenen Sensoren mit VBBs des Geräts versorgt werden. Dadurch werden Fehlmessungen durch Spannungsverschiebungen vermieden.

8971



In = Anschluss Multifunktionseingang n
(CR) = Gerät
(1) = Eingangsfilter
(2) = analoge Strommessung
(3a) = Binäreingang plus-schaltend
(3b) = Binäreingang minus-schaltend
(4a) = analoge Spannungsmessung 0...10 V
(4b) = analoge Spannungsmessung 0...32 V
(5) = Spannung
(6) = Referenzspannung

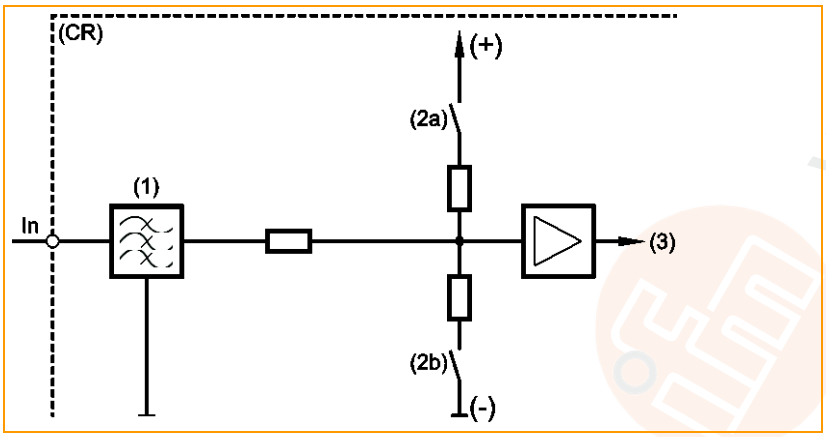
Grafik: Prinzipschaltung Multifunktionseingang

Binäreingänge

1015
7345

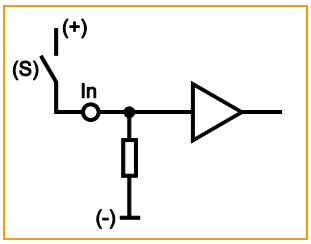
Der Binäreingang kann in folgenden Modi betrieben werden:
- binärer Eingang plus-schaltend (BL) für positives Gebersignal
- binärer Eingang, minus-schaltend (BH) für negatives Gebersignal

Je nach Gerät können auch die Binäreingänge unterschiedlich konfiguriert werden. Neben den Schutzmechanismen gegen Störungen werden die Binäreingänge intern über eine Analogstufe ausgewertet. Das ermöglicht die Diagnose der Eingangssignale. Im Anwendungsprogramm steht das Schaltsignal aber direkt als Bit-Information zur Verfügung.



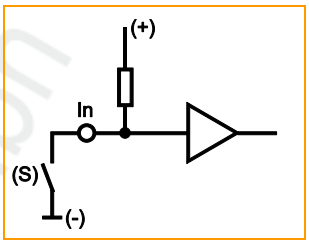
In = Anschluss Binäreingang n
(CR) = Gerät
(1) = Input-Filter
(2a) = Eingang minus-schaltend
(2b) = Eingang plus-schaltend
(3) = Spannung

Grafik: Prinzipschaltung Binäreingang minus-schaltend / plus-schaltend für negative und positive Gebersignale



In = Anschluss Binäreingang n
(S) = Sensor

Prinzipschaltung Binäreingang plus-schaltend (BL)
für positives Sensorsignal:
Eingang = offen ⇨ Signal = Low (GND)



In = Anschluss Binäreingang n
(S) = Sensor

Prinzipschaltung Binäreingang minus-schaltend (BH)
für negatives Sensorsignal:
Eingang = offen ⇨ Signal = High (Supply)

Bei einem Teil dieser Eingänge (→ Datenblatt) kann das Potential gewählt werden, gegen das geschaltet wird.

Eingangsgruppe I00...I15

20390

Bei diesen Eingängen handelt es sich um eine Gruppe von Multifunktionskanälen.

Jeder einzelne dieser Eingänge ist wahlweise wie folgt konfigurierbar:
- analoger Eingang 0...20 mA
- analoger Eingang 0...10 V
- analoger Eingang 0...32 V
- Spannungsmessung ratiometrisch 0...1000 ‰
- binärer Eingang, minus-schaltend (BH) für negatives Gebersignal
- binärer Eingang plus-schaltend (BL) für positives Gebersignal
- schneller Eingang für z.B. Inkrementalgeber und Frequenz- oder Periodendauermessung

→ Kapitel **Mögliche Betriebsarten Ein-/Ausgänge** (→ S. 234)

Diagnosefähige Sensoren nach NAMUR können ausgewertet werden.

Alle Eingänge zeigen das gleiche Verhalten bei Funktion und Diagnose.

ⓘ Detaillierte Beschreibung → Kapitel **Adressbelegung Ein-/Ausgänge**

Im Anwendungsprogramm können die Systemvariablen ANALOG00...ANALOGxx zur kundenspezifischen Diagnose der Eingänge dienen.

Werden die Analogeingänge auf Strommessung konfiguriert, wird bei Überschreiten des Endwertes (21,7 mA) in den sicheren Spannungsmessbereich (0...32 V DC) geschaltet und das jeweilige Fehlerbit im Merkerbyte ERROR_CURRENT_Ix gesetzt.
Einmal je Sekunde prüft das Gerät, ob der Stromwert wieder unter dem Grenzwert liegt. Ist der Wert wieder im gültigen Bereich, schaltet der Eingang selbsttätig auf den Strommessbereich zurück.

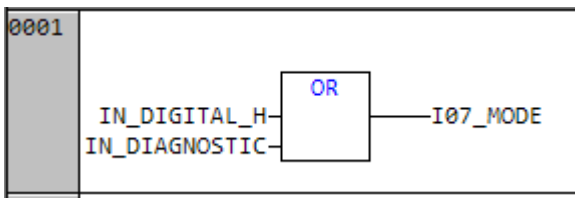
► Die Konfiguration jedes einzelnen Eingangs erfolgt über das Anwendungsprogramm:
- FB **INPUT_ANALOG** (→ S. 128) > Eingang MODE
- Konfigurationsbyte Ixx_MODE
- schnelle Eingänge mit folgenden FBs:

FAST_COUNT (→ S. 136)	Zählerbaustein für schnelle Eingangsimpulse
FREQUENCY (→ S. 138)	misst die Frequenz des am gewählten Kanal ankommenden Signals
FREQUENCY_PERIOD (→ S. 140)	misst die Frequenz und die Periodendauer (Zykluszeit) in [µs] am angegebenen Kanal
INC_ENCODER (→ S. 142)	Vorwärts-/Rückwärts-Zählerfunktion zur Auswertung von Drehgebern
PERIOD (→ S. 144)	misst am angegebenen Kanal die Frequenz und die Periodendauer (Zykluszeit) in [µs]
PHASE (→ S. 148)	liest ein Kanalpaar mit schnellen Eingängen ein und vergleicht die Phasenlage der Signale

15380

Beispiel mit Konfigurationsbyte Ixx_MODE

Die Zuweisung setzt den gewählten Eingang auf die Betriebsart IN_DIGITAL_H mit Diagnose:



13956

> Das Diagnose-Ergebnis zeigen z.B. folgende Systemmerker:

Systemmerker (Symbolname)	Typ	Beschreibung
ERROR_BREAK_Ix (x=0...n; Wert abhängig vom Gerät, → Datenblatt)	DWORD	Eingangs-Doppelwort x: Leiterbruch-Fehler oder (Widerstandseingang): Schluss nach Versorgung [Bit 0 für Eingang 0] ... [Bit z für Eingang z] dieser Gruppe Bit = TRUE: Fehler Bit = FALSE: kein Fehler
ERROR_SHORT_Ix (x=0...n; Wert abhängig vom Gerät, → Datenblatt)	DWORD	Eingangs-Doppelwort x: Kurzschluss-Fehler nur wenn Eingang = IN_DIGITAL_H [Bit 0 für Eingang 0] ... [Bit z für Eingang z] dieser Gruppe Bit = TRUE: Fehler Bit = FALSE: kein Fehler

Eingangsgruppe I00_E...I15_E

6887

Prinzipiell gelten die gleichen Aussagen wie für die Eingangsgruppe I00...I15.
Abweichungen:
- Die symbolischen Adressen der Eingänge lauten Inn_*E*.
- Die symbolischen Adressen der Konfigurationsvariablen lauten Inn_MODE_*E*.
- Die symbolischen Adressen der Filter lauten Inn_FILTER_*E*
- Die symbolischen Adressen der digitalen Filter lauten Inn_DFILTER_*E*
- Die symbolischen Adressen der anderen Merker enden ebenfalls auf '_*E*'.

3.2.6 Ausgänge (Technologie)

Inhalt

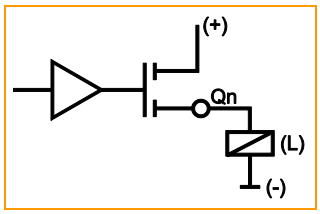
14093

Binärausgänge

14094

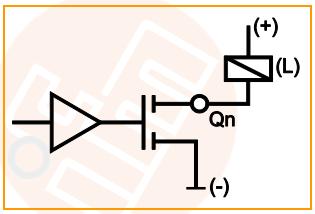
Bei den Geräte-Ausgängen sind folgende Betriebsarten möglich (→ Datenblatt):
- binärer Ausgang, plus-schaltend (BH) mit/ohne Diagnosefunktion
- binärer Ausgang, minus-schaltend (BL) ohne Diagnosefunktion

15450



Qn = Anschluss Ausgang n
(L) = Last

Prinzipschaltung Ausgang plus-schaltend (BH)
für positives Ausgangssignal



Qn = Anschluss Ausgang n
(L) = Last

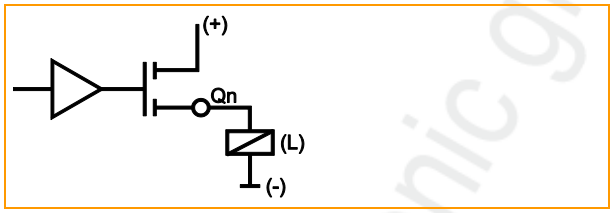
Prinzipschaltung Ausgang minus-schaltend (BL)
für negatives Ausgangssignal

PWM-Ausgänge

14095

Bei den Geräte-Ausgängen sind folgende Betriebsarten möglich (→ Datenblatt):
- PWM-Ausgang, plus-schaltend (BH) ohne Diagnosefunktion

15451



Qn = Anschluss Ausgang n
(L) = Last

Prinzipschaltung Ausgang plus-schaltend (BH)
für positives Ausgangssignal

Ausgangsgruppe Q00...Q15

2244

Bei diesen Ausgängen handelt es sich um eine Gruppe von Multifunktionskanälen.

Jeder einzelne dieser Ausgänge ist wahlweise wie folgt konfigurierbar:
- binärer Ausgang, plus-schaltend (BH), teilweise auch minus-schaltend (BL)
- analoger Ausgang, stromgeregelt (PWMi)
- analoger Ausgang mit Pulsweitenmodulation (PWM), teilweise als H-Brücke

→ Kapitel **Mögliche Betriebsarten Ein-/Ausgänge** (→ S. 234)

Werden die Ausgänge nicht als PWM-Ausgänge genutzt, wird die Diagnose über die integrierten Strommesskanäle realisiert, die auch für die stromgeregelten Ausgangsfunktionen genutzt werden.

► Die Konfiguration jedes einzelnen Ausgangs erfolgt über das Anwendungsprogramm:
Lastströme anzeigen → FB **OUTPUT_CURRENT** (→ S. 154)
PWM-Ausgang: → FB **PWM1000** (→ S. 158)
H-Brücke steuern → FB **OUTPUT_BRIDGE** (→ S. 151)

► Strommessbereich konfigurieren für die Ausgänge Q00…Q03 und Q08…Q11 (wahlweise 2 A oder 4 A):
→ FB SET_OUTPUT_MODE > Eingang CURRENT_RANGE

Bei Einsatz der H-Brücke wird die Stromregelung nicht unterstützt.

Die Ausgänge werden in 2 Gruppen im Fehlerfall (z.B. Kurzschluss) über Relaiskontakte abgeschaltet.

13975

⚠ WARNUNG

Gefährlicher Wiederanlauf möglich!
Gefahr von Personenschaden! Gefahr von Sachschaden an der Maschine/Anlage!

Wird ein Ausgang im Fehlerfall hardwaremäßig abgeschaltet, ändert sich der durch das Anwendungsprogramm erzeugte logische Zustand dadurch nicht.

► Abhilfe:
- Die Ausgänge zunächst im Anwendungsprogramm logisch zurücksetzen!
- Fehler beseitigen!
- Ausgänge situationsabhängig wieder setzen.

Die Ausgänge im PWM-Modus unterstützen keine Diagnosefunktionen.

Bei der Nutzung als Binärausgang erfolgt die Konfiguration jedes Ausgangs mit den Systemvariablen Qxx_MODE. Soll die Diagnose genutzt werden, muss sie zusätzlich aktiviert werden.

Leiterbruch und Kurzschluss des Ausgangssignals werden (gebündelt je Ausgangsgruppe) getrennt über die Systemvariablen ERROR_BREAK_Qx oder ERROR_SHORT_Qx angezeigt. Die einzelnen Ausgangs-Fehlerbits können im Anwendungsprogramm bei Bedarf ausmaskiert werden.

HINWEIS

Um die internen Messwiderstände zu schützen, sollte OUT_OVERLOAD_PROTECTION immer aktiv sein (voreingestellt). Je nach gewähltem Strommessbereich besteht Schutz ab 2,25 A oder ab 4,5 A. Die Funktion wird **nicht** im reinen PWM-Modus unterstützt und kann bei Bedarf abgeschaltet werden.

Zu den Grenzwerten unbedingt das Datenblatt beachten!

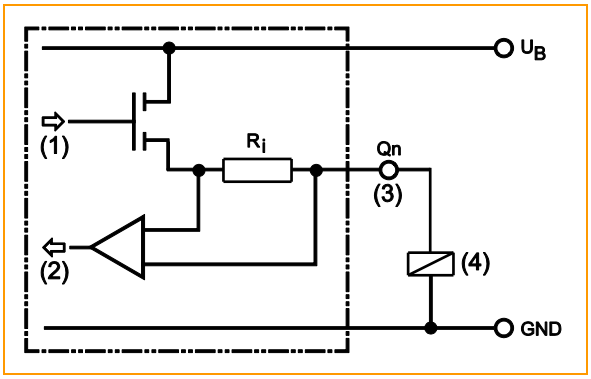
Abhängig von der Umgebungstemperatur kann ab einem bestimmten Kurzschlussstrom ein Kurzschluss eventuell nicht mehr zuverlässig erkannt werden, da die Ausgangstreiber sich zum Schutz vor Zerstörung selbsttätig zeitweise deaktivieren.

Die Leiterbruch- und die Kurzschlusserkennung sind aktiv, wenn...
- der Ausgang ist als "binär plusschaltend" (BH) konfiguriert UND
- der Ausgang ist EINgeschaltet.

Diagnose: binäre Ausgänge (via Strommessung)

19398
19396

Die Diagnose dieser Ausgänge erfolgt über eine interne Strommessung im Ausgang:



Grafik: Prinzipschaltung
(1) Ausgangskanal
(2) Rücklesekanal für Diagnose
(3) Anschluss Ausgang
(4) Last

Diagnose: Überlast (via Strommessung)

19437
15249

Überlast kann nur an einem Ausgang mit Strommessung erkannt werden.
Überlast ist definiert als ...
"nominaler Maximalstrom laut Datenblatt + 12,5 %".

Diagnose: Leiterbruch (via Strommessung)

19400

Die Diagnose erfolgt über den Rücklese-Kanal des Ausgangs.

Voraussetzung zur Diagnose:	Ausgang = TRUE
Diagnose = Leiterbruch:	über den Widerstand R_i fließt kein Strom (keine Spannung fällt ab). Ohne den Leiterbruch fließt durch den Längswiderstand R_i der Laststrom und erzeugt damit einen Spannungsabfall, der über den Rücklesekanal ausgewertet wird.

Diagnose: Kurzschluss (via Strommessung)

19401

Die Diagnose erfolgt über den Rücklese-Kanal des Ausgangs.

Voraussetzung zur Diagnose:	Ausgang = TRUE
Diagnose = Kurzschluss gegen GND:	über den Längswiderstand R_i fällt die Versorgungsspannung ab

Ausgangsgruppe Q00_E...Q15_E

6884

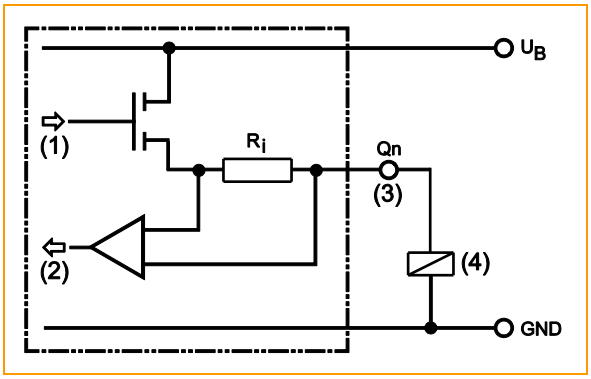
Prinzipiell gelten die gleichen Aussagen wie für die erste Ausgangsgruppe.
Abweichungen:
- Die symbolischen Adressen der Ausgänge lauten Qnn_*E*.
- Die symbolischen Adressen der Konfigurationsvariablen lauten Qnn_MODE_*E*.
- Die symbolischen Adressen der anderen Merker enden ebenfalls auf '_*E*'.

Zu den Grenzwerten unbedingt das Datenblatt beachten!

Diagnose: binäre Ausgänge (via Strommessung)

19398
19396

Die Diagnose dieser Ausgänge erfolgt über eine interne Strommessung im Ausgang:



Grafik: Prinzipschaltung
(1) Ausgangskanal
(2) Rücklesekanal für Diagnose
(3) Anschluss Ausgang
(4) Last

Diagnose: Überlast (via Strommessung)

19437
15249

Überlast kann nur an einem Ausgang mit Strommessung erkannt werden.
Überlast ist definiert als ...
"nominaler Maximalstrom laut Datenblatt + 12,5 %".

Diagnose: Leiterbruch (via Strommessung)

19400

Die Diagnose erfolgt über den Rücklese-Kanal des Ausgangs.

Voraussetzung zur Diagnose:	Ausgang = TRUE
Diagnose = Leiterbruch:	über den Widerstand R_i fließt kein Strom (keine Spannung fällt ab). Ohne den Leiterbruch fließt durch den Längswiderstand R_i der Laststrom und erzeugt damit einen Spannungsabfall, der über den Rücklesekanal ausgewertet wird.

Diagnose: Kurzschluss (via Strommessung)

19401

Die Diagnose erfolgt über den Rücklese-Kanal des Ausgangs.

Voraussetzung zur Diagnose:	Ausgang = TRUE
Diagnose = Kurzschluss gegen GND:	über den Längswiderstand R_i fällt die Versorgungsspannung ab

Ausgangsgruppe Q16_E...Q31_E

10955

Prinzipiell gelten die gleichen Aussagen wie für die erste Ausgangsgruppe.
Abweichungen:

- Die symbolischen Adressen der Ausgänge lauten Qnn_*E*.
- Die symbolischen Adressen der Konfigurationsvariablen lauten Qnn_MODE_*E*.
- Die symbolischen Adressen der anderen Merker enden ebenfalls auf '_*E*'.
- Die Ausgänge sind maximal mit 2 A belastbar (fest eingestellt).
- Die Ausgänge sind auf binär plus-schaltend fest eingestellt.
- Es gibt keine Systemvariable Qnn_FILTER_*E*.
- Die Ausgänge sind nicht diagnosefähig.

Zu den Grenzwerten unbedingt das Datenblatt beachten!

3.2.7 Hinweise zur Anschlussbelegung

1426

Die Anschlussbelegungen (→ Montageanleitungen der Geräte, Kapitel "Anschlussbelegung") beschreiben die Standard-Gerätekonfigurationen. Die Anschlussbelegung dient der Zuordnung der Ein- und Ausgangskanäle zu den IEC-Adressen und den Geräteanschlussklemmen.

Die einzelnen Kürzel haben folgende Bedeutung:

A	Analoger Eingang
BH	Binärer highside-Eingang: minus-schaltend für negatives Sensorsignal Binärer highside-Ausgang: plus-schaltend für positives Ausgangssignal
BL	Binärer lowside-Eingang: plus-schaltend für positives Sensorsignal Binärer lowside-Ausgang: minus-schaltend für negatives Ausgangssignal
CYL	Eingang Periodendauermessung
ENC	Eingang Drehgebersignale
FRQ	Frequenzeingang
H-Bridge	Ausgang mit H-Brücken-Funktion
PWM	**P**uls**w**eiten-**m**oduliertes Signal
PWMi	PWM-Ausgang mit Strommessung
IH	Impuls-/Zählereingang, highside, minus-schaltend für negatives Sensorsignal
IL	Impuls-/Zählereingang, lowside, plus-schaltend für positives Sensorsignal
R	Rücklesekanal für einen Ausgang

Zuordnung der Ein-/Ausgangskanäle: → Katalog, Montageanleitung oder Datenblatt

3.2.8 Sicherheitshinweise zu Reed-Relais

7348

Beim Einsatz von nichtelektronischen Schaltern Folgendes beachten:

6915

Kontakte von Reed-Relais können (reversibel) verkleben, wenn sie ohne Vorwiderstand an den Geräte-Eingängen angeschlossen werden.

- **Abhilfe:** Vorwiderstand zum Reed-Relais installieren:
  Vorwiderstand = max. Eingangsspannung / zulässiger Strom im Reed-Relais
  Beispiel: 32 V / 500 mA = 64 Ohm
- Der Vorwiderstand darf 5 % des Eingangswiderstands RE des Geräte-Eingangs (→ Datenblatt) nicht überschreiten. Sonst wird das Signal nicht als TRUE erkannt.
  Beispiel:
  RE = 3 000 Ohm
  ⇨ max. Vorwiderstand = 150 Ohm

3.2.9 Rückspeisung bei extern beschalteten Ausgängen

2422

In manchen Anwendungen werden Aktuatoren nicht nur von Ausgängen der SPS gesteuert, sondern zusätzlich von externen Schaltern. In solchen Fällen müssen die extern beschalteten Ausgänge mit Sperrdioden geschützt werden (→ Grafik unten).

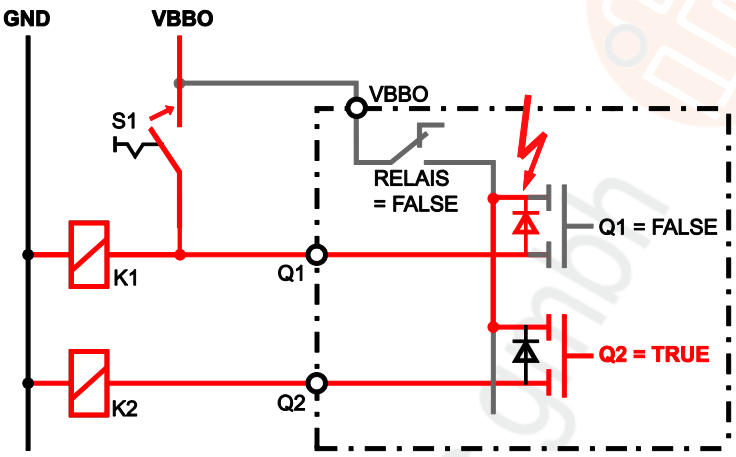
ACHTUNG

Zerstörung von Ausgängen bei unzulässiger Rückspeisung!

Werden Aktoren von extern angesteuert, darf die Potentialschiene derselben Ausgangsgruppe nicht potentialfrei werden (z.B. bei RELAIS = FALSE).

Andernfalls findet über die integrierte Schutzdiode im Ausgangstreiber des extern beschalteten Ausgangs eine Rückspeisung der Klemmenspannung VBB auf die Potentialschiene der Ausgangsgruppe statt. Dadurch steuert ein gesetzter anderer Ausgang derselben Gruppe seine an ihm angeschlossene Last an. Durch den Laststrom wird der rückspeisende Ausgang zerstört.

► Extern beschaltete Ausgänge mit Sperrdioden schützen!

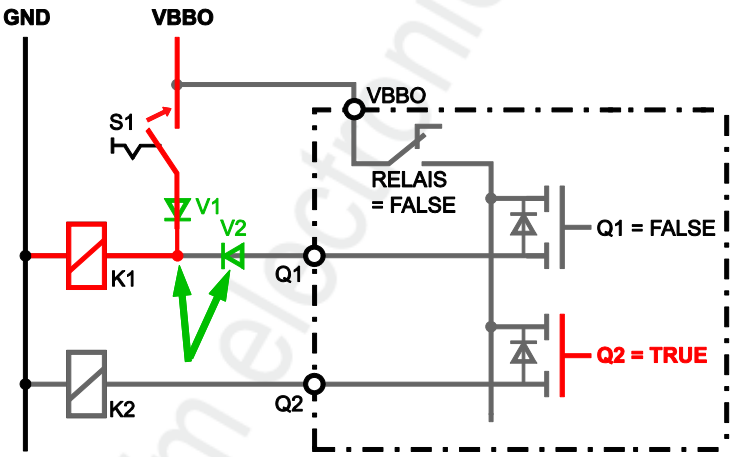


Beispiel:

Merker RELAIS schaltet die Versorgung VBBo der Ausgangsgruppe aus.

Ohne Sperrdioden speist der externe Schalter S1 die Versorgung VBBo über die interne Schutzdiode (rot) von Ausgang Q1 auf die interne Potentialschiene der Ausgänge.

Wird Ausgang Q2 = TRUE (→ Grafik), dann bekommt K2 trotz RELAIS = FALSE Spannung über die Schutzdiode von Q1 (rote Linien). Wegen Überlastung brennt diese Schutzdiode durch und der Ausgang Q1 wird zerstört!



Grafik: Beispiel Beschaltung mit Sperrdioden wegen Gefahr der Rückspeisung

Abhilfe:
Sperrdioden V1 und V2 einsetzen (→ grüne Pfeile)!

Erfolg:
Wenn RELAIS = FALSE, dann bleibt K2 ausgeschaltet, auch wenn Q2 = TRUE.

HINWEIS

Abhilfe bei extern beschalteten Ausgängen

► Die extern beschalteten Ausgänge so über Dioden entkoppeln, dass keine externe Spannung an die Ausgangsklemme der Steuerung geschaltet werden kann!

3.2.10 Status-LED

20809

Die Betriebszustände werden durch die integrierte Status-LED (Voreinstellung) angezeigt.

LED-Farbe	Anzeige	Beschreibung
Aus	konstant aus	keine Betriebsspannung
		
Gelb	kurzzeitig ein	Initialisierung oder Reset Checks
		(Zeitraster = 200 ms)
Orange	blinkt 0,2 Hz	TEST=FALSE: kein Laufzeitsystem geladen
		(Zeitraster = 1 s)
Grün	blinkt 5 Hz	TEST=TRUE: kein Laufzeitsystem geladen
		(Zeitraster = 200 ms)
Grün	blinkt 2 Hz	Anwendung = RUN
		(Zeitraster = 200 ms)
Grün	konstant ein	Anwendung = STOP
		
Rot	blinkt 2 Hz	Anwendung = RUN mit Fehler
		(Zeitraster = 200 ms)
Rot	kurzzeitig ein	FATAL ERROR
		(Zeitraster = 200 ms)
Rot	konstant ein	TEST=TRUE: Anwendung = STOP und FATAL ERROR TEST=FALSE: ERROR STOP / SYSTEM STOP
		

Für die Betriebszustände STOP und RUN kann die Status-LED vom Programmiersystem geändert werden.

LED im Anwendungsprogramm steuern

13142

Bei diesem Gerät kann die Status-LED auch durch das Anwendungsprogramm gesetzt werden. Dazu dienen folgende Systemvariablen (→ Kapitel **Systemmerker** (→ S. 214)):

Systemmerker (Symbolname)	Typ	Beschreibung
LED	WORD	LED-Farbe für "LED eingeschaltet": 0x0000 = LED_GREEN (voreingestellt) 0x0001 = LED_BLUE 0x0002 = LED_RED 0x0003 = LED_WHITE 0x0004 = LED_BLACK 0x0005 = LED_MAGENTA 0x0006 = LED_CYAN 0x0007 = LED_YELLOW
LED_X	WORD	LED-Farbe für "LED ausgeschaltet": 0x0000 = LED_GREEN 0x0001 = LED_BLUE 0x0002 = LED_RED 0x0003 = LED_WHITE 0x0004 = LED_BLACK (voreingestellt) 0x0005 = LED_MAGENTA 0x0006 = LED_CYAN 0x0007 = LED_YELLOW
LED_MODE	WORD	LED-Blinkfrequenz: 0x0000 = LED_2HZ (blinkt mit 2 Hz; voreingestellt) 0x0001 = LED_1HZ (blinkt mit 1 Hz) 0x0002 = LED_05HZ (blinkt mit 0,5 Hz) 0x0003 = LED_0HZ (leuchtet dauernd mit Wert in LED)

HINWEIS

► Im Anwendungsprogramm NICHT die LED-Farbe ROT verwenden.

> Im Fehlerfall wird die LED-Farbe ROT durch das Laufzeitsystem gesetzt.
ABER: Werden die Farben und/oder Blinkmodi im Anwendungsprogramm geändert, gilt die obige Tabelle der Voreinstellung nicht mehr.

3.3 Schnittstellen-Beschreibung

Inhalt

14098

3.3.1 Serielle Schnittstelle

14099

Dieses Gerät bietet eine serielle Schnittstelle.

Grundsätzlich kann die serielle Schnittstelle mit folgenden Funktionen genutzt werden:
- Programm-Download
- Debugging
- freie Nutzung in der Anwendung

12998

HINWEIS

Voreingestellt steht die serielle Schnittstelle dem Anwender nicht zur Verfügung, da sie für den Programm-Download und das Debugging genutzt wird.

Setzt der Anwender das Systemmerkerbit SERIAL_MODE=TRUE, dann kann die Schnittstelle frei genutzt werden. Ein Debugging des Anwendungsprogramms ist dann nur noch über eine (beliebige) CAN-Schnittstelle möglich.

Anschlüsse und Daten → Datenblatt

3.3.2 USB-Schnittstelle

14100

Dieses Gerät bietet eine USB-Schnittstelle für den Programm-Download und das Debugging.

Anschlüsse und Daten → Datenblatt
USB-Treiber auf dem PC installieren → Montageanleitung / Betriebsanleitung
Einstellungen in CODESYS für [Online] > [Kommunikationsparameter...] via USB:

Gerät	Laufzeitsystem-Version	Parameter	Wert
CR0032	< V03.00.00	Baudrate	115200
CR0032	≥ V03.00.01	Baudrate	4800...57600
CR0033, CR0133	≤ V02.00.01	Baudrate	115200
CR0033, CR0133	≥ V02.00.02	Baudrate	4800...57600
CR0232, CR0233	alle	Baudrate	115200
CR0234, CR0235	alle	Baudrate	4800...57600
CR7n32	≤ V01.00.04	Baudrate	115200
CR7n32	≥ V01.00.05	Baudrate	4800...57600
CR0n3n, CR7n32	alle	Motorola byteorder	No
CR0n3n, CR7n32	alle	Flow Control	On

3.3.3 CAN-Schnittstellen

Inhalt

14101

Anschlüsse und Daten → Datenblatt

CAN: Schnittstellen und Protokolle

13820
14587

Die Geräte werden je nach Aufbau der Hardware mit mehreren CAN-Schnittstellen ausgerüstet. Grundsätzlich können alle Schnittstellen unabhängig voneinander mit folgenden Funktionen genutzt werden:
- Layer 2: CAN auf Ebene 2 (→ Kapitel **Bausteine: CAN Layer 2** (→ S. 73))
- CANopen-Master (→ Kapitel **Bausteine: CANopen-Master** (→ S. 82))
- CANopen-Slave (→ Kapitel **Bausteine: CANopen-Slave** (→ S. 92))
- CANopen-Netzwerkvariablen (via CODESYS)
- SAE J1939 (für Antriebsmanagement, → Kapitel **Bausteine: SAE J1939** (→ S. 105))
- Buslast-Erkennung
- Errorframe-Zähler
- Download-Schnittstelle
- 100 % Buslast ohne Paketverlust

11793

In diesem **ecomatmobile**-Gerät sind folgende CAN-Schnittstellen und CAN-Protokolle verfügbar:

CAN-Schnittstelle	CAN 1	CAN 2	CAN 3	CAN 4
voreingestellte Download-ID	ID 127	ID 126	ID 125	ID 124
CAN-Protokolle	CAN Layer 2	CAN Layer 2	CAN Layer 2	CAN Layer 2
	CANopen	CANopen	CANopen	CANopen
	SAE J1939	SAE J1939	SAE J1939	SAE J1939

Standard-Baudrate = 125 kBit/s

ⓘ Welche CANopen-fähige Schnittstelle mit welchem CANopen-Protokoll arbeitet, entscheidet die Reihenfolge, mit der Sie in der Steuerungskonfiguration die Unterelemente anhängen:
CODESYS > [Steuerungskonfiguration] > [CR0232 Configuration Vxx] > [Unterelement anhängen] > [CANopen Master] oder [CANopen Slave]

3.4 Software

Inhalt

14107

3.4.1 Software-Module für das Gerät

Inhalt

14110

Die Software in diesem Gerät setzt wie folgt auf der Hardware auf:

Software-Modul	Anwender kann das Modul ändern?	womit?
Anwendungsprogramm mit Bibliotheken	ja	CODESYS, MaintenanceTool
Laufzeitsystem (LZS) *)	Upgrade ja Downgrade ja	MaintenanceTool
Bootloader	nein	---
(Hardware)	nein	---

*) Die Laufzeitsystem-Versionsnummer muss der Target-Versionsnummer in der CODESYS-Zielsystemeinstellung entsprechen!
→ Kapitel **Target einrichten** (→ S. 55)

Nachfolgend beschreiben wir diese Software-Module:

Bootloader

14111

Im Auslieferungszustand enthalten **ecomat*mobile***-Controller nur den Bootloader.
Der Bootloader ist ein Startprogramm, mit dem das Laufzeitsystem und das Anwendungsprogramm auf dem Gerät nachgeladen werden können.
Der Bootloader enthält Grundroutinen...
- zur Kommunikation der Hardware-Module untereinander,
- zum Nachladen des Laufzeitsystems.

Der Bootloader ist das erste Software-Modul, das im Gerät gespeichert sein muss.

Laufzeitsystem

14112

Grundprogramm im Gerät, stellt die Verbindung her zwischen der Hardware des Gerätes und dem Anwendungsprogramm.
→ Kapitel **Software-Module für das Gerät** (→ S. 39)

Im Auslieferungszustand ist im Normalfall kein Laufzeitsystem im Controller geladen (LED blinkt grün mit 5 Hz). In diesem Betriebszustand ist nur der Bootloader aktiv. Dieser stellt die minimalen Funktionen für den Laufzeitsystem-Ladevorgang zur Verfügung, u.a. die Unterstützung der Schnittstellen (z.B. CAN).

Der Laufzeitsystem-Download muss im Normalfall nur einmalig durchgeführt werden. Das Anwendungsprogramm kann anschließend (auch mehrmals) in den Controller geladen werden, ohne das Laufzeitsystem zu beeinflussen.

Das Laufzeitsystem wird zusammen mit dieser Dokumentation auf einem separaten Datenträger zur Verfügung gestellt. Zusätzlich kann auch die aktuelle Version von der Homepage der **ifm electronic gmbh** heruntergeladen werden:
→ **ifm weltweit • ifm worldwide • ifm à l'échelle internationale** (→ S. 264)

Anwendungsprogramm

14118

Software, die speziell für die Anwendung vom Hersteller in die Maschine programmiert wird. Die Software enthält üblicherweise logische Sequenzen, Grenzwerte und Ausdrücke zum Steuern der entsprechenden Ein- und Ausgänge, Berechnungen und Entscheidungen.

8340

⚠ WARNUNG

Für die sichere Funktion der Anwendungsprogramme, die vom Anwender erstellt werden, ist dieser selbst verantwortlich. Bei Bedarf muss er zusätzlich entsprechend der nationalen Vorschriften eine Abnahme durch entsprechende Prüf- und Überwachungsorganisationen durchführen lassen.

Bibliotheken

14117

ifm electronic bietet passend für jedes Gerät eine Reihe von Bibliotheken (`*.LIB`) an, die Programmmodule für das Anwendungsprogramm enthalten. Beispiele:

Bibliothek	Verwendung
`ifm_CR0232_Vxxyyzz.LIB`	gerätespezifische Bibliothek Muss immer im Anwendungsprogramm enthalten sein!
`ifm_CR0232_CANopenxMaster_Vxxyyzz.LIB` x = 1...4 = Nummer der CAN-Schnittstelle	(optional) wenn eine CAN-Schnittstelle des Geräts als CANopen-Master betrieben werden soll
`ifm_CR0232_CANopenxSlave_Vxxyyzz.LIB` x = 1...4 = Nummer der CAN-Schnittstelle	(optional) wenn eine CAN-Schnittstelle des Geräts als CANopen-Slave betrieben werden soll
`ifm_CR0232_J1939_Vxxyyzz.LIB`	(optional) wenn eine CAN-Schnittstelle des Geräts mit einer Motorsteuerung kommunizieren soll

Details: → Kapitel **ifm-Bibliotheken für das Gerät CR0232** (→ S. 68)

3.4.2 Programmierhinweise für CODESYS-Projekte

Inhalt

7426

Hier erhalten Sie Tipps zum Programmieren des Geräts.

► Beachten Sie die Hinweise im CODESYS-Programmierhandbuch.

FB, FUN, PRG in CODESYS

8473

In CODESYS unterscheiden wir folgende Typen von Bausteinen (POUs):

FB = function block = Funktionsbaustein
- Ein FB kann mehrere Eingänge und mehrere Ausgänge haben.
- Ein FB darf in einem Projekt mehrmals aufgerufen werden.
- Für jeden Aufruf muss eine Instanz deklariert werden.
- Erlaubt: Im FB aufrufen von FB und FUN.

FUN = function = Funktion
- Eine Funktion kann mehrere Eingänge, aber nur einen Ausgang haben.
- Der Ausgang ist vom gleichen Datentyp wie die Funktion selbst.

PRG = program = Programm
- Ein PRG kann mehrere Eingänge und mehrere Ausgänge haben.
- Ein PRG darf in einem Projekt nur einmal aufgerufen werden.
- Erlaubt: im PRG aufrufen von PRG, FB und FUN.

HINWEIS

Funktionsbausteine dürfen NICHT in Funktionen aufgerufen werden!
Sonst: Bei der Ausführung stürzt das Anwendungsprogramm ab.

Alle Bausteine (POUs) dürfen NICHT rekursiv aufgerufen werden, auch nicht indirekt!

Eine IEC-Anwendung darf maximal 8000 Bausteine (POU) enthalten!

Hintergrund:

Alle Variablen von Funktionen...
- werden beim Aufruf initialisiert und
- werden nach der Rückkehr zum Aufrufer ungültig.

Funktionsbausteine haben 2 Aufrufe:
- einen Initialisierungsaufruf und
- den eigentlichen Aufruf, um irgend etwas zu tun.

Folglich heißt das für den FB-Aufruf in einer Funktion:
- jedesmal erfolgt ein zusätzlicher Initialisierungsaufruf und
- die Daten des letzten Aufrufs gehen verloren.

Berechnungen und Konvertierungen im Anwendungsprogramm

20779

HINWEIS

Falls folgende Elemente im Anwendungsprogramm erforderlich sind:
- mathematische Funktionen (z.B. ATAN),
- Berechnungen,
- Konvertierungen (z.B. REAL_TO_BYTE),

dann gilt für die Werte an den Eingängen und Ausgängen der entsprechenden Operatoren:

► Den zulässigen Wertebereich in jedem Einzelfall unbedingt einhalten!

> Ansonsten kann es zu einem FPU-Fehler in der Steuerung kommen.

Beispiele:

20777

Der maximal darstellbare Wert des Zielformats wird überschritten.
Beispiel:

REAL_TO_INT (12345678.3)

> INT ist begrenzt auf -32768...+32767 (nur ganze Zahlen)

20778

Eine vorhandene Realzahl liegt augenscheinlich im Wertebereich des Zielformats.
Tatsächlich (wegen der internen Darstellung der Realzahl) liegt die Zahl außerhalb des Zielformats.
Beispiel:

DW := REAL_TO_DWORD (4294967295.0);

> Die genaueste Darstellung für 4294967295 in REAL ist 4.294967296E9
> Der Wert ist damit um 1 höher als der maximal erlaubte Wert des Zielformats.
> DWORD ist begrenzt auf 0...4294967295.

Zykluszeit beachten!

8006

Bei den frei programmierbaren Geräten aus der Controller-Familie ecomat*mobile* stehen in einem großen Umfang Bausteine zur Verfügung, die den Einsatz der Geräte in den unterschiedlichsten Anwendungen ermöglichen.

Da diese Bausteine je nach Komplexität mehr oder weniger Systemressourcen belegen, können nicht immer alle Bausteine gleichzeitig und mehrfach eingesetzt werden.

ACHTUNG

Gefahr von zu trägem Verhalten des Geräts!
Zykluszeit darf nicht zu lang werden!

► Beim Erstellen des Anwendungsprogramms die oben aufgeführten Empfehlungen beachten und durch Austesten überprüfen.

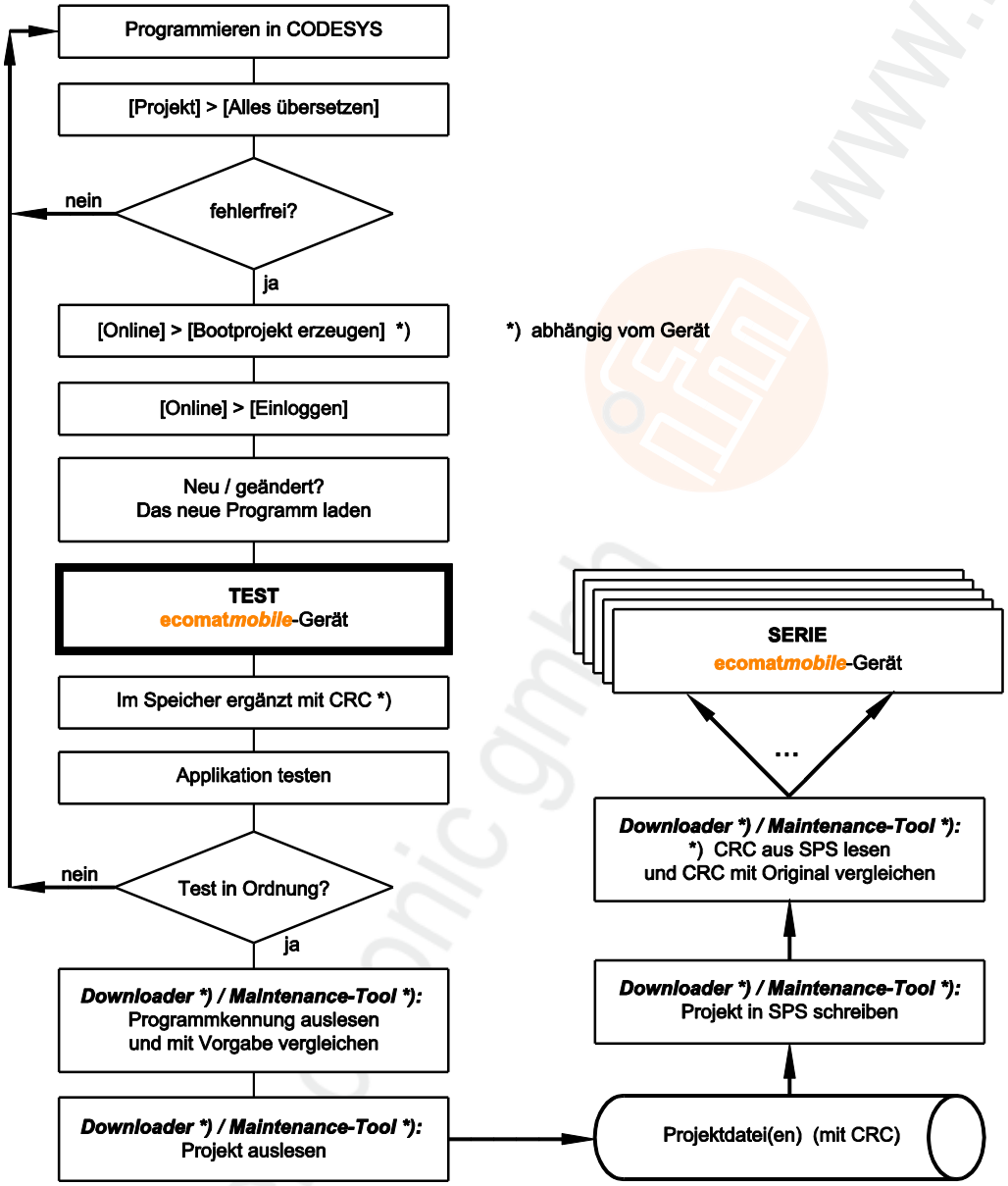
► Bei Bedarf durch Neustrukturieren der Software und des Systemaufbaus die Zykluszeit vermindern.

Anwendungsprogramm erstellen

8007

Das Anwendungsprogramm wird mit dem Programmiersystem CODESYS 2.3 erstellt und während der Programmentwicklung mehrfach zum Testen in die Steuerung geladen:
In CODESYS: [Online] > [Einloggen] > das neue Programm laden.

Für jeden derartigen Download via CODESYS 2.3 wird dazu der Quellcode neu übersetzt. Daraus resultiert, dass auch jedes Mal im Speicher der Steuerung eine neue Prüfsumme gebildet wird. Auch für Sicherheitssteuerungen ist dieses Verfahren bis zur Freigabe der Software zulässig.



Grafik: Erstellen und Verteilen der Software

Boot-Projekt speichern

7430

Speichern Sie im Gerät zusammen mit Ihrem Anwendungsprogramm immer auch das zugehörige Boot-Projekt! Nur so ist das Anwendungsprogramm auch nach einem Spannungsausfall im Gerät verfügbar.

HINWEIS

Beachten: das Boot-Projekt ist etwas größer als das eigentliche Programm.

Jedoch: das Speichern des Boot-Projekts im Gerät wird scheitern, wenn das Boot-Projekt größer wird als der vorhandene IEC-Code-Speicherbereich. Nach Power-On-Reset ist das Boot-Projekt wieder gelöscht oder ungültig.

► CODESYS-Menü [Online] > [Bootprojekt erzeugen]
Dies muss auch nach jeder Änderung erneut erfolgen!

> Nach einem Neustart startet das Gerät mit dem zuletzt gespeicherten Boot-Projekt.

> Falls noch KEIN Boot-Projekt gespeichert wurde:
- das Gerät bleibt nach dem Neustart im STOP-Betrieb
- das Anwendungsprogramm ist nicht (mehr) vorhanden
- die LED leuchtet grün.

ifm-Downloader nutzen

8008

Der ifm-Downloader dient dem einfachen Übertragen des Programmcodes vom Programmierplatz in die Steuerung. Grundsätzlich kann jedes Anwendungsprogramm mit dem ifm-Downloader auf die Steuerungen kopiert werden. Vorteil: Dazu ist kein Programmiersystem mit einer CODESYS-Lizenz erforderlich.

Hier finden Sie den aktuellen ifm-Downloader (min. V06.18.26):
Homepage → **ifm weltweit • ifm worldwide • ifm à l'échelle internationale** (→ S. 264)

ifm-Maintenance-Tool nutzen

8492

Das ifm-Maintenance-Tool dient dem einfachen Übertragen des Programmcodes vom Programmierplatz in das Gerät. Grundsätzlich kann jedes Anwendungsprogramm mit dem ifm-Maintenance-Tool auf die Geräte kopiert werden. Vorteil: Dazu ist kein Programmiersystem mit einer CODESYS-Lizenz erforderlich.

Hier finden Sie das aktuelle ifm-Maintenance-Tool:
Homepage → **ifm weltweit • ifm worldwide • ifm à l'échelle internationale** (→ S. 264)

3.4.3 Betriebszustände

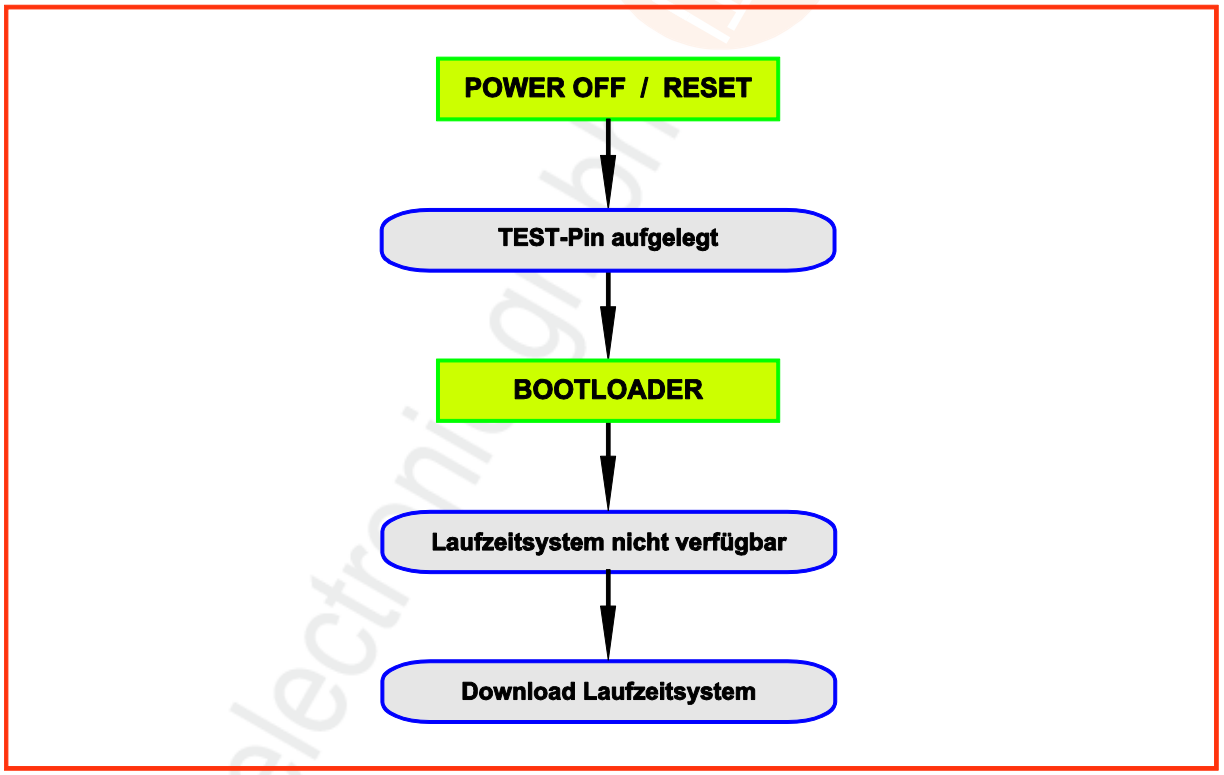
Inhalt

14120

Nach Anlegen der Versorgungsspannung kann sich das ecomatmobile-Gerät in einem von fünf möglichen Betriebszuständen befinden:
- BOOTLOADER
- INIT
- STOP
- RUN
- SYSTEM STOP (nach ERROR STOP)

Betriebszustände

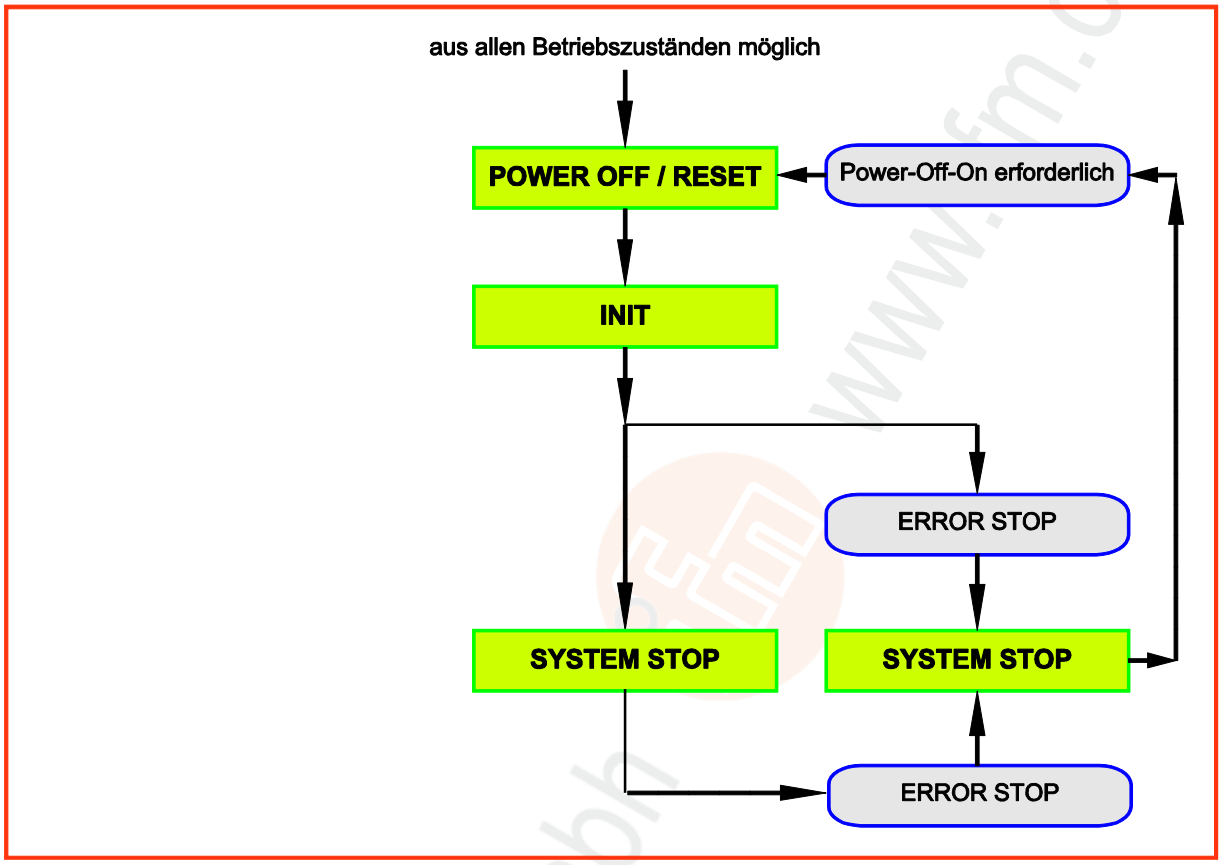
19217



Grafik: Betriebszustände (hier: Laufzeitsystem ist nicht verfügbar)

Betriebszustände: Anwendungsprogramm nicht verfügbar

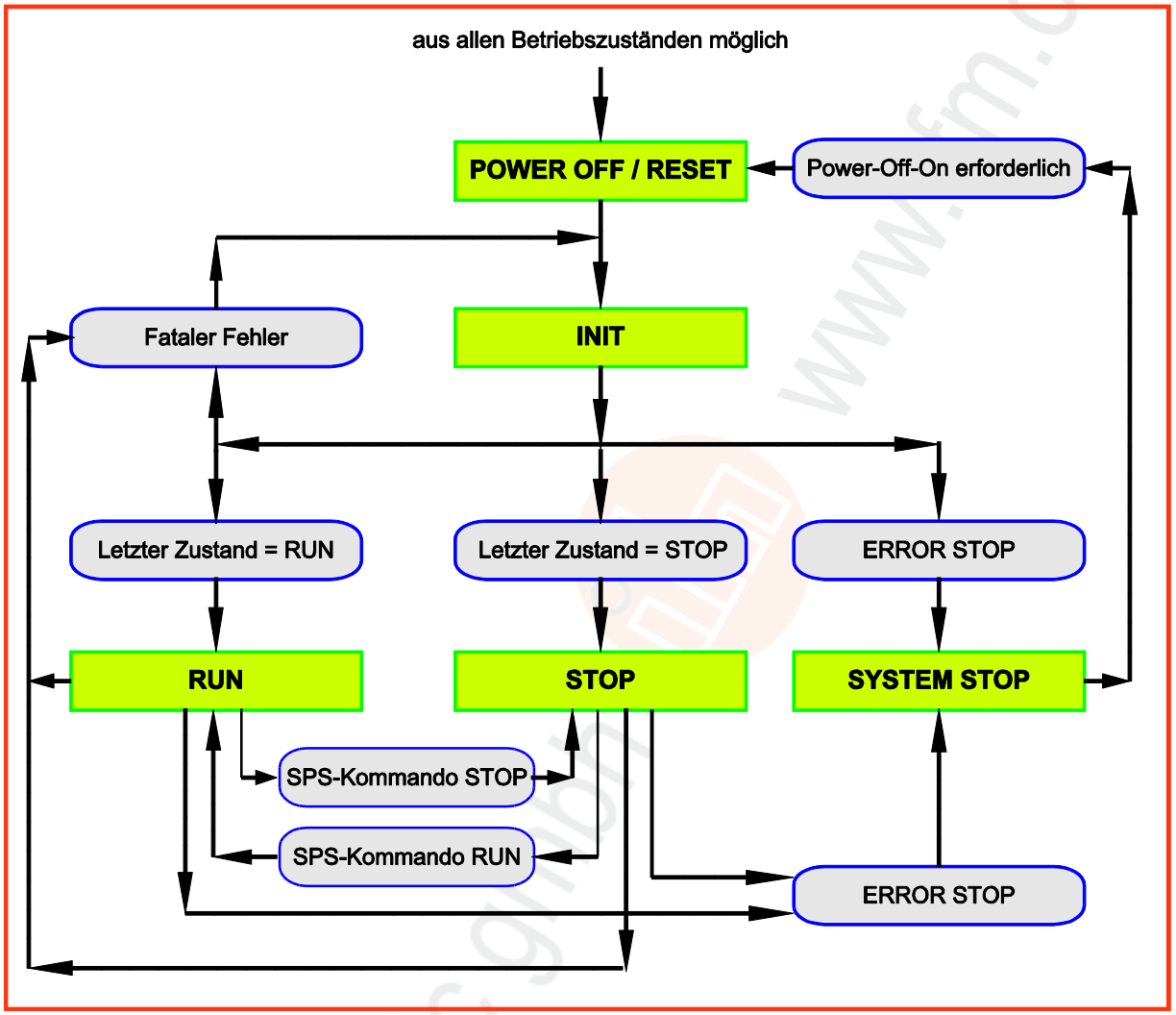
19218



Grafik: Betriebszustände (hier: Anwendungsprogramm ist nicht verfügbar)

Betriebszustände: Anwendungsprogramm verfügbar

19219



Grafik: Betriebszustände (hier: Anwendungsprogramm ist verfügbar)

Bootloader-Zustand

1080

Es wurde kein Laufzeitsystem geladen. Der ecomatmobile-Controller befindet sich im Bootloader-Zustand. Vor dem Laden des Anwendungsprogramms muss ein Laufzeitsystem-Download durchgeführt werden.

> Die LED blinkt grün (5 Hz).

INIT-Zustand (Reset)

1076

Voraussetzung: ein gültiges Laufzeitsystem ist installiert.

Dieser Zustand wird nach jedem Power-On-Reset durchlaufen:

> Das Laufzeitsystem wird initialisiert.

> Verschiedene Checks werden durchgeführt, z.B. Warten auf gültige Versorgungsspannung.

> Dieser nur temporäre Zustand wird vom RUN- oder STOP-Zustand abgelöst.

> Die LED leuchtet gelb.

Wechsel aus diesem Zustand in einen der folgenden Zustände möglich:
- RUN
- STOP

STOP-Zustand

1078

Dieser Zustand wird in folgenden Fällen erreicht:

- Aus dem RESET-Zustand, wenn:
  - kein Anwendungsprogramm ist geladen oder
  - der letzte Zustand vor dem RESET-Zustand war der STOP-Zustand
- Aus dem RUN-Zustand durch das STOP-Kommando
  - nur bei Betriebsmodus = TEST (→ Kapitel **TEST-Betrieb** (→ S. 50))

> Die LED leuchtet grün.

RUN-Zustand

1077

Dieser Zustand wird in folgenden Fällen erreicht:

- Aus dem RESET-Zustand, wenn:
  - der letzte Zustand vor dem RESET-Zustand war der RUN-Zustand
- Aus dem STOP-Zustand durch das RUN-Kommando
  - nur bei Betriebsmodus = TEST (→ Kapitel **TEST-Betrieb** (→ S. 50))

> Die LED blinkt grün (2 Hz).

SYSTEM-STOP-Zustand

19222

In diesen Zustand fällt der ecomatmobile-Controller, wenn ein nicht tolerierbarer Fehler (ERROR STOP) festgestellt wurde. Dieser Zustand kann nur durch einen Power-Off-On-Reset verlassen werden.

> Die LED leuchtet rot.

3.4.4 Betriebsmodi

1083

Unabhängig von den Betriebszuständen kann der Controller in verschiedenen Betriebsmodi betrieben werden.

TEST-Betrieb

1084

ACHTUNG

Verlust der gespeicherten Software möglich!

Im Test-Betrieb besteht kein Schutz der gespeicherten Laufzeitsystem- und Anwendungs-Software.

14892

HINWEIS

► Erst NACH dem Anschließen des OPC-Client den TEST-Anschluss mit der Versorgungsspannung verbinden!

Dieser Betriebsmodus wird durch Anlegen von Versorgungsspannung am Test-Eingang erreicht (→ Montageanleitung > Kapitel "Technische Daten" > Kapitel "Anschlussbelegung").

Jetzt kann der Controller im RUN- oder STOP-Zustand Kommandos über eine der Schnittstellen entgegennehmen und z.B. mit dem Programmiersystem kommunizieren.

Nur im TEST-Betrieb ist ein Software-Download im Controller möglich.

Über den Merker TEST kann der Zustand vom Anwendungsprogramm abgefragt werden.

Zusammenfassung Test-Eingang ist aktiv:
- Programmiermodus ist freigeben
- Software-Download ist möglich
- Zustand des Anwendungsprogramms ist abfragbar
- kein Schutz der gespeicherten Software möglich

Hinweise: TEST-Eingänge

20781

► Die TEST-Eingänge aller Steuerungen in der Maschine einzeln verdrahten und eindeutig markieren, so dass eine Zuordnung zu den Steuerungen eindeutig hergestellt werden kann.

► Bei einem Maintenance-Zugriff darf immer nur der TEST-Eingang der Steuerung aktiviert werden, auf die zugegriffen werden soll.

SERIAL_MODE

2548

Die serielle Schnittstelle steht für den Datenaustausch in der Anwendung zur Verfügung. Ein Debugging des Anwendungsprogramms ist dann nur noch über alle 4 CAN-Schnittstellen möglich.

Diese Funktion ist standardmäßig abgeschaltet (FALSE). Über den Merker SERIAL_MODE kann der Zustand über das Anwendungsprogramm oder das Programmiersystem gesteuert und abgefragt werden.

(→ Kapitel **Bausteine: serielle Schnittstelle** (→ S. 117))

DEBUG-Modus

1086

Wird der Eingang DEBUG von **SET_DEBUG** (→ S. 207) auf TRUE gesetzt, kann z.B. das Programmiersystem oder der Downloader mit dem Gerät kommunizieren und spezielle Systemkommandos ausführen (z.B. für Servicefunktionen über das GSM-Modem CANremote).

Ein Software-Download ist in dieser Betriebsart nicht möglich, da der Test-Eingang (→ Kapitel **TEST-Betrieb** (→ S. 50)) nicht mit Versorgungsspannung verbunden wird.

3.4.5 Leistungsgrenzen des Geräts

7358

Leistungsgrenzen des Geräts beachten! → Datenblatt

Verhalten des Watchdog

11786

Ein Watchdog überwacht in diesem Gerät die Programmlaufzeit der CODESYS-Anwendung.

Wird die maximale Watchdog-Zeit (ca. 100 ms) überschritten:
> das Gerät führt einen Reset durch und startet neu

Zu erkennen im Merker LAST_RESET.

CODESYS-Funktionen

2254

Folgende Grenzen sollten Sie berücksichtigen:

- Bis zu 2048 Bausteine (PB, FB...) werden unterstützt.
- Für Anwender verfügbare Merker → Kapitel **Verfügbarer Speicher** (→ S. 16).
  Beschreibung der Retain-Merker → bei den jeweiligen FBs.

4 Konfigurationen

Inhalt

1016

Die in den jeweiligen Montage- und Installationsanweisungen oder dem **Anhang** (→ S. 214) dieser Dokumentation beschriebenen Gerätekonfigurationen stehen als Standardgeräte (Lagerware) zur Verfügung. Diese decken bei den meisten Anwendungen die geforderten Spezifikationen ab.

Entsprechend den Kundenanforderungen bei Serieneinsatz ist es aber auch möglich, dass andere Gerätekonfigurationen z.B. hinsichtlich der Zusammenstellung der Ein- und Ausgänge und der Ausführung der Analogkanäle eingesetzt werden.

4.1 Laufzeitsystem einrichten

Inhalt

14091

4.1.1 Laufzeitsystem neu installieren

14092
2733

Im Auslieferungszustand ist im Normalfall kein Laufzeitsystem im Gerät geladen (LED blinkt grün mit 5 Hz). In diesem Betriebszustand ist nur der Bootloader aktiv. Dieser stellt die minimalen Funktionen für den Laufzeitsystem-Ladevorgang zur Verfügung, u.a. die Unterstützung der Schnittstellen (z.B. RS232, CAN).

Der Laufzeitsystem-Download muss im Normalfall nur einmalig durchgeführt werden. Das Anwendungsprogramm kann anschließend (auch mehrmals) in das Gerät geladen werden, ohne das Laufzeitsystem zu beeinflussen.

Das Laufzeitsystem wird zusammen mit dieser Dokumentation auf einem separaten Datenträger zur Verfügung gestellt. Zusätzlich kann auch die aktuelle Version von der Homepage der **ifm electronic gmbh** heruntergeladen werden:
→ **ifm weltweit • ifm worldwide • ifm à l'échelle internationale** (→ S. 264)

2689

HINWEIS

Es müssen immer die zum gewählten Target passenden Software-Stände zum Einsatz kommen:
- des Laufzeitsystems (`ifm_CR0232_Vxxyyzz.H86`),
- der Steuerungskonfiguration (`ifm_CR0232_Vxx.CFG`),
- der Gerätebibliothek (`ifm_CR0232_Vxxyyzz.LIB`) und
- der weiteren Dateien

V	Version
xx: 00...99	Versionsnummer
yy: 00...99	Release-Nummer
zz: 00...99	Patch-Nummer

Dabei müssen der Basisdateiname (z.B. "`CR0232`") und die Software-Versionsnummer "xx" (z.B. "`02`") überall den gleichen Wert haben! Andernfalls geht das Gerät in den STOP-Zustand.

Die Werte für "yy" (Release-Nummer) und "zz" (Patch-Nummer) müssen **nicht** übereinstimmen.

4368

Folgende Dateien müssen ebenfalls geladen sein:
- die zum Projekt erforderlichen internen Bibliotheken (in IEC 61131 erstellt),
- die Konfigurationsdateien (`*.CFG`)
- und die Target-Dateien (`*.TRG`).

Es kann vorkommen, dass das Zielsystem mit Ihrer aktuell installierten Version von CODESYS nicht oder nur teilweise programmiert werden kann. Im diesem Fall wenden Sie sich bitte an den technischen Support der **ifm electronic gmbh**.
Kontakt → **ifm weltweit • ifm worldwide • ifm à l'échelle internationale** (→ S. 264)

Das Laufzeitsystem wird mit dem eigenständigen Programm "**ifm**-Downloader" in das Gerät übertragen.
Der **ifm**-Downloader und dessen Dokumentation kann bei Bedarf von der **ifm**-Homepage heruntergeladen werden:
→ **ifm weltweit • ifm worldwide • ifm à l'échelle internationale** (→ S. 264)

Das Anwendungsprogramm wird im Normalfall über das Programmiersystem in das Gerät geladen. Es kann aber ebenfalls mit dem **ifm**-Downloader geladen werden, wenn es zuvor aus dem Gerät ausgelesen wurde (→ Upload).

4.1.2 Laufzeitsystem aktualisieren

13269

Auf dem Gerät ist bereits ein älteres Laufzeitsystem installiert. Nun möchten Sie das Laufzeitsystem auf dem Gerät aktualisieren?

14158

ACHTUNG

Gefahr von Datenverlust!

Beim Löschen oder Aktualisieren des Laufzeitsystems werden alle Daten und Programme auf dem Gerät gelöscht.

► Alle erforderlichen Daten und Programme sichern, bevor das Laufzeitsystem gelöscht oder aktualisiert wird!

Prinzipiell gelten für diesen Vorgang die gleichen Hinweise, wie zuvor im Kapitel 'Laufzeitsystem neu installieren' gegeben wurden.

4.1.3 Installation verifizieren

14407
14406

► Nach dem Laden des Laufzeitsystems in die Steuerung:
  • Prüfen, ob das Laufzeitsystem korrekt übertragen wurde!
  • Prüfen, ob sich das richtige Laufzeitsystem auf der Steuerung befindet!

► 1. Prüfung:
mit dem ifm-Downloader oder mit dem Maintenance-Tool prüfen, ob die richtige Laufzeitsystem-Version geladen wurde:
  • Name, Version und die CRC des Laufzeitsystems im Gerät auslesen!
  • Diese Daten manuell mit den Soll-Daten vergleichen!

► 2. Prüfung (optional):
Im Anwendungsprogramm prüfen, ob die richtige Laufzeitsystem-Version geladen wurde:
  • Name und die Version des Laufzeitsystems im Gerät auslesen!
  • Diese Daten mit fest vorgegebenen Werten vergleichen!
Zum Auslesen der Daten dient folgender FB:

GET_IDENTITY (→ S. 205)	liest die im Gerät gespeicherten spezifischen Kennungen: • Hardware-Name und Hardware-Version des Geräts • Name des Laufzeitsystems im Gerät • Version und Ausgabe des Laufzeitsystems im Gerät • Name der Anwendung (wurde zuvor mit **SET_IDENTITY** (→ S. 208) gespeichert) • Seriennummer des Geräts

► Wird durch die Anwendung eine falsche Laufzeitsystem-Version erkannt:
alle Sicherheitsfunktionen in den sicheren Zustand schalten!

4.2 Programmiersystem einrichten

Inhalt

3968

4.2.1 Programmiersystem manuell einrichten

Inhalt

3963

Target einrichten

2687
11379

Beim Erstellen eines neuen Projektes in CODESYS muss die dem Gerät entsprechende Target-Datei geladen werden.

► Im Dialog-Fenster [Zielsystem Einstellungen] im Menü [Konfiguration] die gewünschte Target-Datei wählen.

> Die Target-Datei stellt für das Programmiersystem die Schnittstelle zur Hardware her.

> Gleichzeitig mit Wahl des Targets werden automatisch einige wichtige Bibliotheken und die Steuerungskonfiguration geladen.

► Bei Bedarf im Fenster [Zielsystem Einstellungen] > Reiter [Netzfunktionen] > [Parameter-Manager unterstützen] und / oder [Netzvariablen unterstützen] aktivieren.

► Bei Bedarf geladene (3S-)Bibliotheken wieder entfernen oder durch weitere (ifm-)Bibliotheken ergänzen.

► Immer die passende Geräte-Bibliothek `ifm_CR0232_Vxxyyzz.LIB` manuell ergänzen!

Steuerungskonfiguration aktivieren (z.B. CR0033)

15824

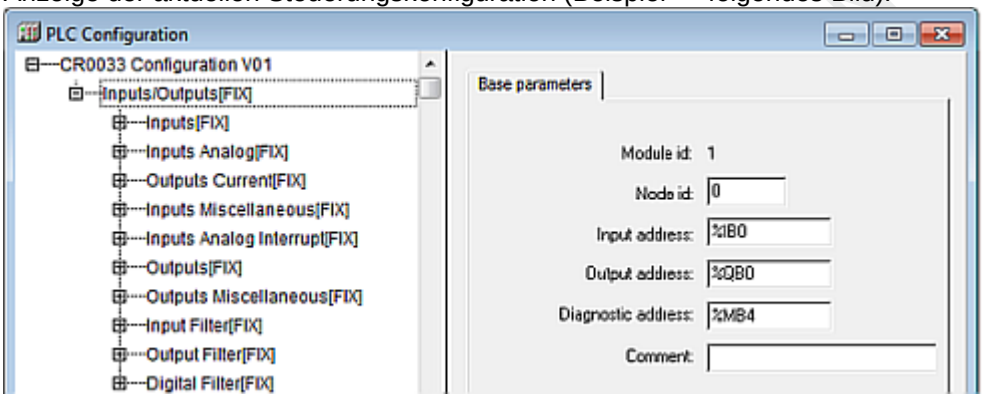
Bei der Konfiguration des Programmiersystems (→ vorheriger Abschnitt) erfolgte automatisch auch die Steuerungskonfiguration.

► Den Punkt [Steuerungskonfiguration] erreicht man über den Reiter [Ressourcen].
Mit Doppelklick auf den Punkt [Steuerungskonfiguration] öffnet sich das entsprechende Fenster.

► In CODESYS den Reiter [Ressourcen] klicken:

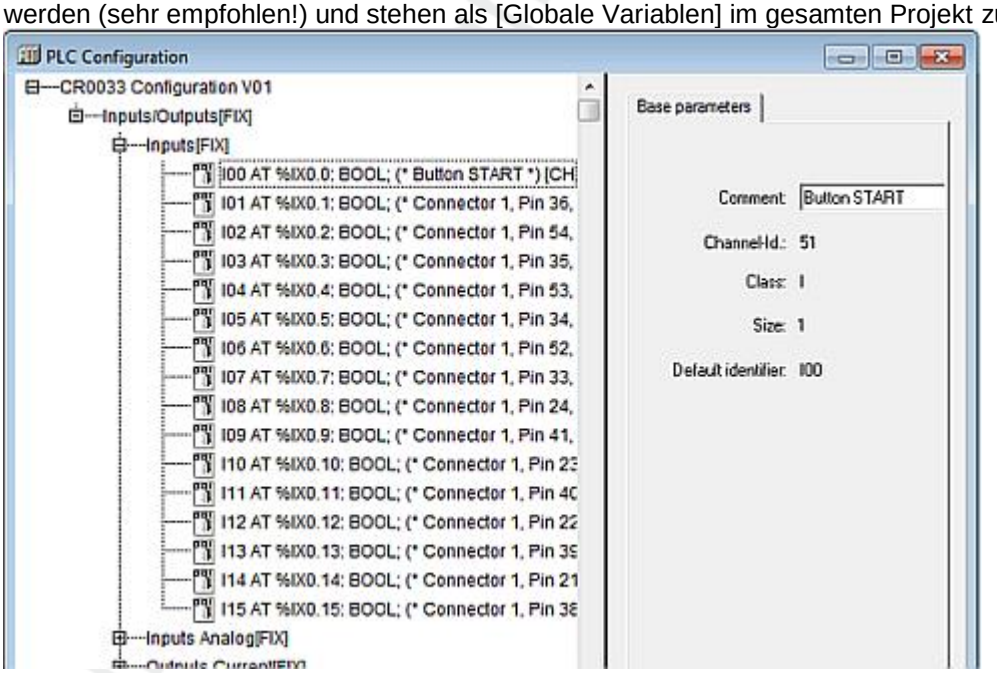
► In der linken Spalte Doppelklick auf [Steuerungskonfiguration]

> Anzeige der aktuellen Steuerungskonfiguration (Beispiel → folgendes Bild):



Durch die Konfiguration ist für den Anwender in der Programmumgebung Folgendes verfügbar:

- alle wichtigen System- und Fehlermerker
  Je nach Anwendung und Anwendungsprogramm müssen diese Merker bearbeitet und ausgewertet werden. Der Zugriff erfolgt über deren symbolischen Namen.
- die Struktur der Ein- und Ausgänge
  Diese können im Fenster [Steuerungskonfiguration] (→ Bild unten) direkt symbolisch bezeichnet werden (sehr empfohlen!) und stehen als [Globale Variablen] im gesamten Projekt zur Verfügung.



4.2.2 Programmiersystem über Templates einrichten

13745

ifm bietet vorgefertigte Templates (Programm-Vorlagen), womit Sie das Programmiersystem schnell, einfach und vollständig einrichten können.

970

Beim Installieren der **ecomat*mobile***-DVD "Software, tools and documentation" wurden auch Projekte mit Vorlagen auf Ihrem Computer im Programmverzeichnis abgelegt:
`…\ifm electronic\CoDeSys V…\Projects\Template_DVD_V…`

► Die gewünschte dort gespeicherte Vorlage in CODESYS öffnen mit:
[Datei] > [Neu aus Vorlage…]

> CODESYS legt ein neues Projekt an, dem der prinzipielle Programmaufbau entnommen werden kann. Es wird dringend empfohlen, dem gezeigten Schema zu folgen.

4.3 Funktionskonfiguration, allgemein

Inhalt

3971

4.3.1 Konfiguration der Ein- und Ausgänge (Voreinstellung)

2249

- Alle Ein-/Ausgänge sind im Auslieferungszustand im Binär-Modus (plus-schaltend!).
- Die Diagnosefunktion ist nicht aktiv.
- Der Überlastschutz ist aktiv.

4.3.2 Systemvariablen

2252
13519
15576

Alle Systemvariablen (→ Kapitel **Systemmerker** (→ S. 214)) liegen auf festen, nicht verschiebbaren Adressen.

> Zur Anzeige und Verarbeitung eines Watchdog-Fehlers oder Ursachen eines Neustarts wird die Systemvariable LAST_RESET gesetzt.

> Anzeige der gewählten E/A-Konfiguration über Mode-Bytes

4.4 Funktionskonfiguration der Ein- und Ausgänge

Inhalt

1812
1394

Bei bestimmten Ein- und Ausgängen sind zusätzliche Diagnosefunktionen aktivierbar. Damit kann das jeweilige Ein- und Ausgangssignal überwacht werden und im Fehlerfall kann das Anwendungsprogramm darauf reagieren.

Je nach Ein- und Ausgang müssen bei der Nutzung der Diagnose bestimmte Randbedingungen beachtet werden:

► Anhand des Datenblattes prüfen, für welche Ein- und Ausgänge des Geräts welche Diagnosemöglichkeit zur Verfügung steht!

- Zur Konfiguration der Ein- und Ausgänge sind in den Gerätebibliotheken (`ifm_CR0232_Vxxyyzz.LIB`) Konstanten vordefiniert (z.B. IN_DIGITAL_H). Ausführliche Angaben → Kapitel **Mögliche Betriebsarten Ein-/Ausgänge** (→ S. 234).

Nur ExtendedController:
Die Namen der Ein- und Ausgänge in der zweiten Steuerungshälfte werden durch ein angehängtes "_E" gekennzeichnet.

4.4.1 Eingänge konfigurieren

Inhalt

3973

Zulässige Betriebsarten → Kapitel **Mögliche Betriebsarten Ein-/Ausgänge** (→ S. 234)

Sicherheitshinweise zu Reed-Relais

7348

Beim Einsatz von nichtelektronischen Schaltern Folgendes beachten:

6915

Kontakte von Reed-Relais können (reversibel) verkleben, wenn sie ohne Vorwiderstand an den Geräte-Eingängen angeschlossen werden.

► **Abhilfe:** Vorwiderstand zum Reed-Relais installieren:
Vorwiderstand = max. Eingangsspannung / zulässiger Strom im Reed-Relais
Beispiel: 32 V / 500 mA = 64 Ohm

► Der Vorwiderstand darf 5 % des Eingangswiderstands RE des Geräte-Eingangs (→ Datenblatt) nicht überschreiten. Sonst wird das Signal nicht als TRUE erkannt.
Beispiel:
RE = 3 000 Ohm
⇨ max. Vorwiderstand = 150 Ohm

Schnelle Eingänge

2193

Die Geräte verfügen über schnelle Zähl-/Impulseingänge für eine Eingangsfrequenz bis 30 kHz (→ Datenblatt).

19102

Der Eingangswiderstand der schnellen Eingänge schaltet automatisch um, je nach verwendetem Modus oder Funktionsblock:

Eingangswiderstand	bei Modus / FB
3,2 kOhm	(Standard) FAST_COUNT, FREQUENCY, INC_ENCODER, PERIOD und ähnliche FBs
50,7 kOhm	Eingang mit festem Schaltpegel 32 V

23900

Der Innenwiderstand R_i der Signalquelle muss wesentlich geringer sein als der Eingangswiderstand R_{input} des verwendeten Eingangs (Prinzip Spannungsanpassung).
Ansonsten kann das Eingangssignal des schnellen Eingangs verfälscht werden (Tiefpass-Charakteristik).

14677

Werden z.B. mechanische Schalter an diesen Eingängen angeschlossen, kann es durch Kontaktprellen zu Fehlsignalen in der Steuerung kommen.

▶ Bei Bedarf diese "Fehlsignale" über die Filter Ixx_DFILTER ausfiltern.
(→ Kapitel **Systemmerker** (→ S. 214)) (nicht für alle Eingänge verfügbar)

Geeignete Funktionsbausteine sind z.B.:

FAST_COUNT (→ S. 136)	Zählerbaustein für schnelle Eingangsimpulse
FREQUENCY (→ S. 138)	misst die Frequenz des am gewählten Kanal ankommenden Signals
FREQUENCY_PERIOD (→ S. 140)	misst die Frequenz und die Periodendauer (Zykluszeit) in [µs] am angegebenen Kanal
INC_ENCODER (→ S. 142)	Vorwärts-/Rückwärts-Zählerfunktion zur Auswertung von Drehgebern
PERIOD (→ S. 144)	misst am angegebenen Kanal die Frequenz und die Periodendauer (Zykluszeit) in [µs]
PERIOD_RATIO (→ S. 146)	misst die Frequenz und die Periodendauer (Zykluszeit) in [µs] über die angegebenen Perioden am angegebenen Kanal. Zusätzlich wird das Puls-/Periodenverhältnis in [‰] angegeben.
PHASE (→ S. 148)	liest ein Kanalpaar mit schnellen Eingängen ein und vergleicht die Phasenlage der Signale

Bei Einsatz dieser Bausteine werden automatisch die dort parametrierten Ein-/Ausgänge konfiguriert. Der Programmierer der Anwendung ist hiervon entlastet.

Software-Filter der Eingänge konfigurieren

6883

Über die Systemvariablen Ixx_FILTER kann ein Software-Filter konfiguriert werden, der die gemessene Eingangsspannung an den Analogeingängen filtert. Der Filter verhält sich bei einer Sprungantwort wie ein klassischer Tiefpassfilter, wobei die Grenzfrequenz durch den in die Systemvariable eingetragenen Wert eingestellt wird. Es sind Werte von 0...8 möglich.

Tabelle: Grenzfrequenz Software-Tiefpassfilter am Analogeingang

Ixx_FILTER	Filterfrequenz [Hz]	Signalanstiegszeit	Hinweise
0	Filter deaktiviert		
1	390	1 ms	
2	145	2,5 ms	
3	68	5 ms	
4	34	10 ms	empfohlen, Voreinstellung
5	17	21 ms	
6	8	42 ms	
7	4	84 ms	
8	2	169 ms	
≥ 9	34	10 ms	→ Voreinstellung

12969

Nach dem Ändern der Filtereinstellung wird der Wert dieses Ein- oder Ausgangs nicht sofort richtig ausgegeben. Erst nach der Signalanstiegszeit (→ Tabelle) ist der Wert wieder korrekt.

Die Signalanstiegszeit ist die Zeitdauer, die ein Signal am Ausgang des Filters benötigt, um von 10 % auf 90 % des Endwerts zu kommen, wenn am Eingang ein Sprung angelegt wird. Die Signalabstiegszeit ist die Zeitdauer von 90 % bis 10 %.

Hardware-Filter konfigurieren

9154

Über die Systemvariable Ixx_DFILTER kann ein digitaler Hardware-Filter an den schnellen Zähl- und Impulseingängen konfiguriert werden. Der Wert in µs (max. 100 000) gibt an, wie lange ein binärer Pegel ohne Unterbrechung anliegen muss, bevor er übernommen wird. Voreinstellung = 0 µs.

Der Pegelwechsel des Eingangssignals wird um den im Filter eingestellten Wert verzögert.

Nur bei folgenden Funktionsbausteinen hat der Filter Auswirkungen auf die erfassten Signale:

FAST_COUNT (→ S. 136)	Zählerbaustein für schnelle Eingangsimpulse
FREQUENCY (→ S. 138)	misst die Frequenz des am gewählten Kanal ankommenden Signals
FREQUENCY_PERIOD (→ S. 140)	misst die Frequenz und die Periodendauer (Zykluszeit) in [µs] am angegebenen Kanal
INC_ENCODER (→ S. 142)	Vorwärts-/Rückwärts-Zählerfunktion zur Auswertung von Drehgebern
PERIOD (→ S. 144)	misst am angegebenen Kanal die Frequenz und die Periodendauer (Zykluszeit) in [µs]
PERIOD_RATIO (→ S. 146)	misst die Frequenz und die Periodendauer (Zykluszeit) in [µs] über die angegebenen Perioden am angegebenen Kanal. Zusätzlich wird das Puls-/Periodenverhältnis in [‰] angegeben.

Digitale Filter stehen nicht für alle schnellen Zähl- und Impulseingänge zur Verfügung.

4.4.2 Ausgänge konfigurieren

Inhalt

3976

Zulässige Betriebsarten → Kapitel **Mögliche Betriebsarten Ein-/Ausgänge** (→ S. 234)

Zulässige Konfigurationen für Q00_MODE...Q15_MODE

6903

Overload	Diagnose	--	4 A [1])	2 A	--	LS	HS	Konfig.-Wert	
7	6	5	4	3	2	1	0	[hex]	[dez]
0	0	0	0	1	0	0	1	09	9
0	0	0	1	0	0	0	1	11	17
0	1	0	0	1	0	0	1	49	73
0	1	0	1	0	0	0	1	51	81
1	0	0	0	1	0	0	1	89	137
1	0	0	1	0	0	0	1	91	145
1	1	0	0	1	0	0	1	C9	201
1	1	0	1	0	0	0	1	D1	209
0	0	0	0	0	0	1	0	02	2

= diese Konfiguration ist voreingestellt

[1]) nur möglich bei Ausgängen Q00...Q03 + Q08...Q11

Zulässige Konfigurationen für Q00_MODE_E...Q15_MODE_E

6904

Overload	Diagnose	--	4 A [1])	2 A	--	LS	HS	Konfig.-Wert	
7	6	5	4	3	2	1	0	[hex]	[dez]
0	0	0	0	1	0	0	1	09	9
0	0	0	1	0	0	0	1	11	17
0	1	0	0	1	0	0	1	49	73
0	1	0	1	0	0	0	1	51	81
1	0	0	0	1	0	0	1	89	137
1	0	0	1	0	0	0	1	91	145
1	1	0	0	1	0	0	1	C9	201
1	1	0	1	0	0	0	1	D1	209
0	0	0	0	0	0	1	0	02	2

= diese Konfiguration ist voreingestellt

[1]) nur möglich bei Ausgängen Q00_E...Q03_E + Q08_E...Q11_E

Zulässige Konfigurationen für Q16_MODE_E...Q31_MODE_E

17190

Overload	Diagnose	--	4 A [1])	2 A	--	LS	HS	Konfig.-Wert	
7	6	5	4	3	2	1	0	[hex]	[dez]
0	0	0	0	X (0)	0	0	1	01	1
0	1	0	0	X (0)	0	0	1	41	65
[1])	0	0	0	X (0)	0	0	1	81	129

= diese Konfiguration ist voreingestellt

[1]) hier nicht möglich

Software-Filter der Ausgänge konfigurieren

6882

Über die Systemvariablen Qxx_FILTER kann ein Software-Filter konfiguriert werden, der die gemessenen Stromwerte filtert.

- Der Filter verhält sich bei einer Sprungantwort wie ein klassischer Tiefpassfilter, wobei die Grenzfrequenz durch den in die Systemvariable eingetragenen Wert eingestellt wird.
- Die Einstellung des Filters bei der Strommessung hat Einfluss auf die Diagnosezeit.

Tabelle: Grenzfrequenz Software-Tiefpassfilter bei der Strommessung am Ausgang

Qxx_FILTER	Filterfrequenz [Hz]	Signalanstiegszeit	Hinweise
0	Filter deaktiviert		
1	580	0,6 ms	
2	220	1,6 ms	
3	102	3,5 ms	
4	51	7 ms	empfohlen, Voreinstellung
5	25	14 ms	
6	12	28 ms	
7	6	56 ms	
8	3	112 ms	
≥ 9	51	7 ms	→ Voreinstellung

12969

Nach dem Ändern der Filtereinstellung wird der Wert dieses Ein- oder Ausgangs nicht sofort richtig ausgegeben. Erst nach der Signalanstiegszeit (→ Tabelle) ist der Wert wieder korrekt.

Die Signalanstiegszeit ist die Zeitdauer, die ein Signal am Ausgang des Filters benötigt, um von 10 % auf 90 % des Endwerts zu kommen, wenn am Eingang ein Sprung angelegt wird. Die Signalabstiegszeit ist die Zeitdauer von 90 % bis 10 %.

Binär- und PWM-Ausgänge

2423

Bei den Geräte-Ausgängen sind folgende Betriebsarten möglich (→ Datenblatt):
- binärer Ausgang, plus-schaltend (BH) mit/ohne Diagnosefunktion
- binärer Ausgang, plus-schaltend (BH), teilweise auch minus-schaltend (BL)
- PWM-Ausgang, plus-schaltend (BH) ohne Diagnosefunktion
- PWM-Ausgangspaar H-Brücke ohne Diagnosefunktion

PWM-Ausgänge können mit und ohne Stromregelfunktion betrieben werden.
ⓘ Stromgeregelte PWM-Ausgänge werden überwiegend zur Ansteuerung von proportionalen Hydraulikfunktionen genutzt.
ⓘ Der mittlere Strom über ein PWM-Signal kann über den FB **OUTPUT_CURRENT** (→ S. 154) nur dann korrekt ermittelt werden, wenn der im eingeschaltetem Zustand fließende Strom immer innerhalb des Messbereichs liegt.

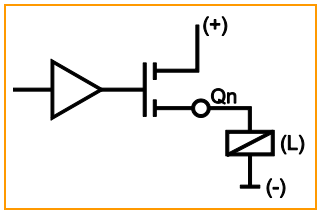
14713

⚠ WARNUNG

Sach- oder Körperschäden möglich durch Fehlfunktionen!

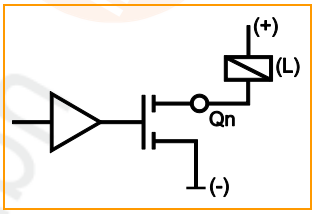
Für Ausgänge im PWM-Modus gilt:
- es gibt keine Diagnosefunktionen
- es werden keine ERROR-Merker gesetzt
- der Überlastschutz OUT_OVERLOAD_PROTECTION ist NICHT aktiv

15450



Qn = Anschluss Ausgang n
(L) = Last

Prinzipschaltung Ausgang plus-schaltend (BH)
für positives Ausgangssignal



Qn = Anschluss Ausgang n
(L) = Last

Prinzipschaltung Ausgang minus-schaltend (BL)
für negatives Ausgangssignal

13975

⚠ WARNUNG

Gefährlicher Wiederanlauf möglich!
Gefahr von Personenschaden! Gefahr von Sachschaden an der Maschine/Anlage!

Wird ein Ausgang im Fehlerfall hardwaremäßig abgeschaltet, ändert sich der durch das Anwendungsprogramm erzeugte logische Zustand dadurch nicht.

► Abhilfe:
- Die Ausgänge zunächst im Anwendungsprogramm logisch zurücksetzen!
- Fehler beseitigen!
- Ausgänge situationsabhängig wieder setzen.

14931

ⓘ HINWEIS

► Im laufenden Betrieb die Ausgänge NICHT umkonfigurieren!
Ändern von PWM-Ausgang nach Binär-Ausgang ist nicht zulässig.

> Ansonsten könnten die Ausgänge unvorhersehbar reagieren.

Verfügbarkeit von PWM

12058

Gerät	Anzahl verfügbare PWM-Ausgänge	davon stromgeregelt (PWMi)	PWM-Frequenz [Hz]
CRn032, CR0033	16	16	20...250
CRn232, CR0233	32	32	20...250

Stromregelung mit PWM (= PWMi)

13829

Über die im Controller integrierten Strommesskanäle kann eine Strommessung des Spulenstroms durchgeführt werden. Dadurch kann zum Beispiel der Strom bei einer Spulenerwärmung nachgeregelt werden. Damit bleiben die Hydraulikverhältnisse im System gleich.

Grundsätzlich sind die stromgeregelten Ausgänge gegen Kurzschluss geschützt.

4.5 Variablen

Inhalt

3130

In diesem Kapitel erfahren Sie mehr über den Umgang mit Variablen.

14486

Das Gerät unterstützt folgende Variablentypen:

Variable	Deklarationsort	Gültigkeitsbereich	Speicherverhalten
lokal	im Deklarationsteil des Bausteins	gilt nur im Baustein (POU), in dem sie konfiguriert wurde	flüchtig
lokal Retain			nicht flüchtig
global	in [Ressourcen] > [Globale Variablen] > [Globale_Variablen]	gilt in allen Bausteinen (POUs) dieses CODESYS-Projekts	flüchtig
global Retain			nicht flüchtig
Netzwerk	in [Ressourcen] > [Globale Variablen] > Deklarationsliste	Werte stehen allen CODESYS-Projekten im gesamten Netzwerk zur Verfügung, wenn die Variable in ihren Deklarationslisten enthalten ist.	flüchtig
Netzwerk Retain			nicht flüchtig

→ CODESYS-Programmierhandbuch

4.5.1 Retain-Variablen

15454

Als `RETAIN` deklarierte Variablen erzeugen remanente Daten. Retain-Variablen behalten beim Aus-/Einschalten des Geräts oder einem Online-Reset die in ihnen gespeicherten Werte.

Die Inhalte der Retain-Variablen gehen verloren, falls beim Ausschalten das Gerät im STOP-Zustand ist!

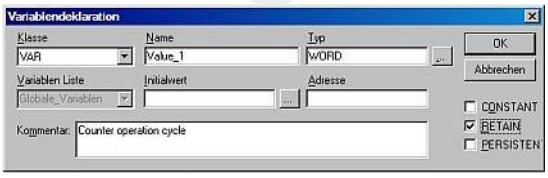
14166

Typische Einsätze für Retain-Variablen sind z.B.:
- Betriebsstunden, die zur Laufzeit der Maschine fortgeschrieben werden,
- Positionswerte von Inkrementalgebern,
- im Bildschirmgerät eingetragene Sollwerte,
- Maschinenparameter,

also alle Variablen, deren Werte beim Ausschalten des Geräts nicht verloren gehen dürfen.

Als Retain können alle Variablentypen, auch komplexe Stukturen (z.B. Timer), gekennzeichnet werden.

► Dazu in der Variablen-Deklaration das Kontrollfeld [RETAIN] aktivieren (→ Bild).



4.5.2 Netzwerkvariablen

9856

Globale Netzwerkvariablen dienen dem Datenaustausch zwischen Controllern im Netzwerk. Die Werte von globalen Netzwerkvariablen stehen allen CODESYS-Projekten im gesamten Netzwerk zur Verfügung, wenn die Variablen in deren Deklarationslisten enthalten sind.

► Dazu folgende Bibliothek(en) in das CODESYS-Projekt einbinden:
- `3S_CANopenNetVar.lib`

5 ifm-Funktionselemente

Inhalt

13586

Alle CODESYS-Funktionselemente (FBs, PRGs, FUNs) sind in Bibliotheken zusammengefasst. Nachfolgend zeigen wir Ihnen alle ifm-Bibliotheken, die Sie zusammen mit diesem Gerät nutzen können.

Anschließend finden Sie eine thematisch gegliederte Beschreibung der Funktionselemente.

5.1 ifm-Bibliotheken für das Gerät CR0232

Inhalt

14235

Legende für ..._Vxxyyzz.LIB:

V	Version
xx: 00...99	Versionsnummer
yy: 00...99	Release-Nummer
zz: 00...99	Patch-Nummer

Hier finden Sie die für dieses Gerät passenden ifm-Funktionselemente aufgelistet, nach CODESYS-Bibliotheken sortiert.

5.1.1 Bibliothek ifm_CR0232_V010003.LIB

18429

Dies ist die Geräte-Bibliothek. Diese ifm-Bibliothek enthält folgende Bausteine:

Baustein	Kurzbeschreibung
CANx (→ S. 74)	initialisiert die CAN-Schnittstelle x x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)
CANx_BAUDRATE (→ S. 75)	stellt die Übertragungsrate für den Busteilnehmer an der CAN-Schnittstelle x ein x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)
CANx_BUSLOAD (→ S. 76)	ermittelt die aktuelle Buslast an der CAN-Schnittstelle x und zählt die aufgetretenen Error-Frames x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)
CANx_DOWNLOADID (→ S. 77)	stellt den Download-Identifier für die CAN-Schnittstelle x ein x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)
CANx_ERRORHANDLER (→ S. 78)	führt ein "manuelles" Bus-Recover auf der CAN-Schnittstelle x durch x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)
CANx_RECEIVE (→ S. 79)	CAN-Schnittstelle x: konfiguriert ein Datenempfangsobjekt und liest den Empfangspuffer des Datenobjektes aus x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)
CANx_SDO_READ (→ S. 101)	CAN-Schnittstelle x: liest das SDO mit den angegebenen Indizes aus dem Knoten aus x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)
CANx_SDO_WRITE (→ S. 103)	CAN-Schnittstelle x: schreibt das SDO mit den angegebenen Indizes in den Knoten x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)
CANx_TRANSMIT (→ S. 81)	übergibt in jedem Aufruf ein CAN-Datenobjekt (Message) an die CAN-Schnittstelle x zur Übertragung x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)
CHECK_DATA (→ S. 203)	erzeugt über einen konfigurierbaren Speicherbereich eine Prüfsumme (CRC) und prüft die Daten des Speicherbereichs auf ungewollte Veränderung
DELAY (→ S. 176)	verzögert die Ausgabe des Eingangswertes um die Zeit T (Totzeit-Glied)
FAST_COUNT (→ S. 136)	Zählerbaustein für schnelle Eingangsimpulse
FAST_COUNT_E	= **FAST_COUNT** (→ S. 136) für die Extended-Seite
FLASHREAD (→ S. 195)	liest unterschiedliche Datentypen direkt aus dem Flash-Speicher in den RAM
FLASHWRITE (→ S. 196)	schreibt unterschiedliche Datentypen direkt in den Flash-Speicher
FRAMREAD (→ S. 198)	liest unterschiedliche Datentypen direkt aus dem FRAM-Speicher in den RAM FRAM steht hier allgemein für alle Arten von nichtflüchtigen, schnellen Speichern.
FRAMWRITE (→ S. 199)	schreibt unterschiedliche Datentypen direkt in den FRAM-Speicher FRAM steht hier allgemein für alle Arten von nichtflüchtigen, schnellen Speichern.
FREQUENCY (→ S. 138)	misst die Frequenz des am gewählten Kanal ankommenden Signals
FREQUENCY_E	= **FREQUENCY** (→ S. 138) für die Extended-Seite
FREQUENCY_PERIOD (→ S. 140)	misst die Frequenz und die Periodendauer (Zykluszeit) in [µs] am angegebenen Kanal
FREQUENCY_PERIOD_E	= **FREQUENCY_PERIOD** (→ S. 140) für die Extended-Seite
GET_IDENTITY (→ S. 205)	liest die im Gerät gespeicherten spezifischen Kennungen: • Hardware-Name und Hardware-Version des Geräts • Name des Laufzeitsystems im Gerät • Version und Ausgabe des Laufzeitsystems im Gerät • Name der Anwendung (wurde zuvor mit **SET_IDENTITY** (→ S. 208) gespeichert) • Seriennummer des Geräts
GET_IDENTITY_EIOS (→ S. 206)	FB liest die im Gerät für die Extended-Seite gespeicherten spezifischen Kennungen: • Name des Laufzeitsystems der Extended-Seite (EIOS) im Gerät • Version und Ausgabe des Laufzeitsystems der Extended-Seite (EIOS) im Gerät
INC_ENCODER (→ S. 142)	Vorwärts-/Rückwärts-Zählerfunktion zur Auswertung von Drehgebern
INC_ENCODER_E	= **INC_ENCODER** (→ S. 142) für die Extended-Seite

Baustein	Kurzbeschreibung
INPUT_ANALOG (→ S. 128)	Analoger Eingangskanal: Wahlweise Messung von... • Strom • Spannung
INPUT_ANALOG_E	= **INPUT_ANALOG** (→ S. 128) für die Extended-Seite
MEMCPY (→ S. 200)	schreibt und liest unterschiedliche Datentypen direkt in den Speicher
MEMORY_RETAIN_PARAM (→ S. 192)	legt das remanente Verhalten der Daten für verschiedene Ereignisse fest
MEMSET (→ S. 201)	beschreibt einen bestimmten Datenbereich
NORM (→ S. 131)	normiert einen Wert [WORD] innerhalb festgelegter Grenzen auf einen Wert mit neuen Grenzen
NORM_DINT (→ S. 133)	normiert einen Wert [DINT] innerhalb festgelegter Grenzen auf einen Wert mit neuen Grenzen
NORM_REAL (→ S. 134)	normiert einen Wert [REAL] innerhalb festgelegter Grenzen auf einen Wert mit neuen Grenzen
OUTPUT_BRIDGE (→ S. 151)	H-Brücke an einem PWM-Kanalpaar
OUTPUT_BRIDGE_E	= **OUTPUT_BRIDGE** (→ S. 151) für die Extended-Seite
OUTPUT_CURRENT (→ S. 154)	misst den Strom (Mittelung über Dither-Periode) an einem Ausgangskanal
OUTPUT_CURRENT_E	= **OUTPUT_CURRENT** (→ S. 154) für die Extended-Seite
OUTPUT_CURRENT_CONTROL (→ S. 155)	Stromregler für einen PWMi-Ausgangskanal
OUTPUT_CURRENT_CONTROL_E	= **OUTPUT_CURRENT_CONTROL** (→ S. 155) für die Extended-Seite
PERIOD (→ S. 144)	misst am angegebenen Kanal die Frequenz und die Periodendauer (Zykluszeit) in [µs]
PERIOD_E	= **PERIOD** (→ S. 144) für die Extended-Seite
PERIOD_RATIO (→ S. 146)	misst die Frequenz und die Periodendauer (Zykluszeit) in [µs] über die angegebenen Perioden am angegebenen Kanal. Zusätzlich wird das Puls-/Periodenverhältnis in [‰] angegeben.
PERIOD_RATIO_E	= **PERIOD_RATIO** (→ S. 146) für die Extended-Seite
PHASE (→ S. 148)	liest ein Kanalpaar mit schnellen Eingängen ein und vergleicht die Phasenlage der Signale
PHASE_E	= **PHASE** (→ S. 148) für die Extended-Seite
PID1 (→ S. 177)	PID-Regler
PID2 (→ S. 179)	PID-Regler
PT1 (→ S. 181)	Regelstrecke mit Verzögerung 1. Ordnung
PWM1000 (→ S. 158)	initialisiert und parametriert einen PWM-fähigen Ausgangskanal das Puls-Pausen-Verhältnis kann in 1 ‰-Schritten angegeben werden
PWM1000_E	= **PWM1000** (→ S. 158) für die Extended-Seite
SERIAL_PENDING (→ S. 118)	ermittelt die Anzahl der im seriellen Empfangspuffer gespeicherten Datenbytes
SERIAL_RX (→ S. 119)	liest mit jedem Aufruf ein empfangenes Datenbyte aus dem seriellen Empfangspuffer aus
SERIAL_SETUP (→ S. 120)	initialisiert die serielle RS232-Schnittstelle
SERIAL_TX (→ S. 121)	überträgt ein Datenbyte über die serielle RS232-Schnittstelle
SET_DEBUG (→ S. 207)	organisiert (abhängig vom TEST-Eingang) den DEBUG-Modus oder den Monitoring-Modus
SET_IDENTITY (→ S. 208)	setzt eine anwendungsspezifische Programmkennung
SET_INTERRUPT_I (→ S. 123)	bedingtes Ausführen eines Programmteils nach einer Interrupt-Anforderung über einen definierten Eingangskanal
SET_INTERRUPT_XMS (→ S. 125)	bedingtes Ausführen eines Programmteils im Intervall von x Millisekunden
SET_PASSWORD (→ S. 209)	setzt Benutzerkennung für Zugangskontrolle bei Programm- und Speicher-Upload
SOFTRESET (→ S. 183)	führt einen kompletten Neustart des Geräts aus
TEMPERATURE (→ S. 188)	liest die aktuelle Temperatur im Gerät aus
TIMER_READ (→ S. 185)	liest die aktuelle Systemzeit in [ms] aus Max-Wert = 49d 17h 2min 47s 295ms
TIMER_READ_US (→ S. 186)	liest die aktuelle Systemzeit in [µs] aus Max-Wert = 1h 11min 34s 967ms 295µs

5.1.2 Bibliothek ifm_CR0232_CANopenxMaster_Vxxyyzz.LIB

13707

x = 1...4 = Nummer der CAN-Schnittstelle
Diese Bibliothek enthält Bausteine für den Betrieb des Geräts als CANopen-Master.

Diese ifm-Bibliothek enthält folgende Bausteine:

Baustein	Kurzbeschreibung
CANx_MASTER_EMCY_HANDLER (→ S. 83)	verwaltet den geräteeigenen Fehlerstatus des CANopen-Masters an der CAN-Schnittstelle x x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)
CANx_MASTER_SEND_EMERGENCY (→ S. 84)	versendet anwendungsspezifische Fehlerstatus des CANopen-Masters an der CAN-Schnittstelle x x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)
CANx_MASTER_STATUS (→ S. 86)	Status-Anzeige an der CAN-Schnittstelle x des als CANopen-Master eingesetzten Gerätes x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

5.1.3 Bibliothek ifm_CR0232_CANopenxSlave_Vxxyyzz.LIB

13709

x = 1...4 = Nummer der CAN-Schnittstelle
Diese Bibliothek enthält Bausteine für den Betrieb des Geräts als CANopen-Slave.

Diese ifm-Bibliothek enthält folgende Bausteine:

Baustein	Kurzbeschreibung
CANx_SLAVE_EMCY_HANDLER (→ S. 93)	verwaltet den geräteeigenen Fehlerstatus des CANopen-Slaves an der CAN-Schnittstelle x: • Error Register (Index 0x1001) und • Error Field (Index 0x1003) des CANopen Objektverzeichnis x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)
CANx_SLAVE_NODEID (→ S. 94)	ermöglicht das Einstellen der Node-ID eines CANopen-Slaves an der CAN-Schnittstelle x zur Laufzeit des Anwendungsprogramms x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)
CANx_SLAVE_SEND_EMERGENCY (→ S. 95)	versendet anwendungsspezifische Fehlerstatus des CANopen-Slaves an der CAN-Schnittstelle x x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)
CANx_SLAVE_SET_PREOP (→ S. 97)	schaltet den Betriebsmodus dieses CANopen-Slaves an der CAN-Schnittstelle x von OPERATIONAL auf PRE-OPERATIONAL x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)
CANx_SLAVE_STATUS (→ S. 98)	zeigt den Status des an der CAN-Schnittstelle x als CANopen-Slave eingesetzten Gerätes x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

5.1.4 Bibliothek ifm_CR0232_J1939_Vxxyyzz.LIB

13711

Diese Bibliothek enthält Bausteine zur Motorsteuerung.

Diese ifm-Bibliothek enthält folgende Bausteine:

Baustein	Kurzbeschreibung
J1939_x (→ S. 106)	CAN-Schnittstelle x: Protokoll-Handler für das Kommunikationsprofil SAE J1939 x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)
J1939_x_GLOBAL_REQUEST (→ S. 107)	CAN-Schnittstelle x: organisiert globales Anfordern und Empfangen von Daten der J1939-Netzwerkteilnehmer x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)
J1939_x_RECEIVE (→ S. 109)	CAN-Schnittstelle x: empfängt eine einzelne Nachricht oder einen Nachrichtenblock x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)
J1939_x_RESPONSE (→ S. 111)	CAN-Schnittstelle x: organisiert die automatische Antwort auf ein Request-Telegramm x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Baustein	Kurzbeschreibung
J1939_x_SPECIFIC_REQUEST (→ S. 113)	CAN-Schnittstelle x: automatisches Anfordern einzelner Nachrichten von einem bestimmten (specific) J1939-Netzwerkteilnehmer x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)
J1939_x_TRANSMIT (→ S. 115)	CAN-Schnittstelle x: versendet einzelne Nachrichten oder Nachrichtenblocks x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

5.1.5 Bibliothek ifm_hydraulic_32bit_Vxxyyzz.LIB

13729

Diese Bibliothek enthält Bausteine für Hydraulik-Steuerungen.

Diese ifm-Bibliothek enthält folgende Bausteine:

Baustein	Kurzbeschreibung
CONTROL_OCC (→ S. 161)	OCC = **O**utput **C**urrent **C**ontrol (= stromgeregelter Ausgang) skaliert den Eingangswert [WORD] auf einen angegebenen Strombereich
JOYSTICK_0 (→ S. 163)	skaliert Signale [INT] aus einem Joystick auf fest definierte Kennlinien, normiert auf 0…1000
JOYSTICK_1 (→ S. 166)	skaliert Signale [INT] aus einem Joystick auf parametrierbare Kennlinien, normiert auf 0…1000
JOYSTICK_2 (→ S. 170)	skaliert Signale [INT] aus einem Joystick auf einen parametrierbaren Kennlinien-Verlauf; die Normierung ist frei bestimmbar
NORM_HYDRAULIC (→ S. 173)	normiert einen Wert [DINT] innerhalb festgelegter Grenzen auf einen Wert mit neuen Grenzen

5.2 ifm-Bausteine für das Gerät CR0232

Inhalt

13988
3826

Hier finden Sie die Beschreibung der für dieses Gerät passenden ifm-Funktionselemente, nach Thema sortiert.

5.2.1 Bausteine: CAN Layer 2

Inhalt

13754

Hier werden die CAN-Funktionsbausteine (Layer 2) zur Nutzung im Anwendungsprogramm beschrieben.

CANx

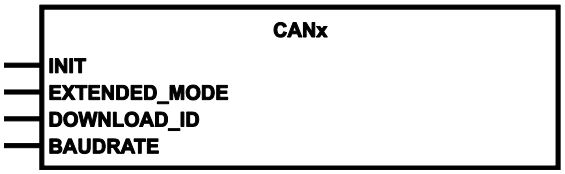
2159

x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:



Beschreibung

2162

CANx initialisiert die x. CAN-Schnittstelle.
(x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt))

Der Download-ID muss für jede Schnittstelle unterschiedlich sein.

Die Baudraten der einzelnen CANx können unterschiedlich eingestellt werden.

► Den Eingang INIT nur für einen Zyklus bei Neustart oder Restart der Schnittstelle setzen!

Eine Änderung des Download-ID und/oder der Baudrate wird erst gültig ...
- nach Spannung Aus/Ein,
- nach Soft-Reset.

Wenn der FB nicht ausgeführt wird, arbeitet die Schnittstelle mit 11-Bit-Identifier.

Parameter der Eingänge

2163

Parameter	Datentyp	Beschreibung
INIT	BOOL	TRUE (im 1. Zyklus): Baustein wird initialisiert FALSE: im weiteren Programmablauf
EXTENDED_MODE	BOOL := FALSE	TRUE: Identifier der CAN-Schnittstelle arbeitet mit 29 Bits FALSE: Identifier der CAN-Schnittstelle arbeitet mit 11 Bits
DOWNLOAD_ID	BYTE	Download-ID der CAN-Schnittstelle x x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt) zulässig = 1...127 voreingestellt = 127 - (x-1)
BAUDRATE	WORD := 125	Baudrate [kBit/s] zulässig = 20, 50, 100, 125, 250, 500, 1000

CANx_BAUDRATE

11834

x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:

Beschreibung

11839

CANx_BAUDRATE stellt die Übertragungsrate für den Busteilnehmer ein.

Mit dem FB wird für das Gerät die Übertragungsrate eingestellt. Dazu wird am Eingang BAUDRATE der entsprechende Wert in kBit/s angegeben.

Der neue Wert wird erst nach einem RESET gültig (Spannung Aus/Ein oder Soft-Reset).

Parameter der Eingänge

655

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE (im 1. Zyklus): Parameter übernehmen und aktivieren sonst: diese Funktion wird nicht ausgeführt
BAUDRATE	WORD := 125	Baudrate [kBit/s] zulässig = 20, 50, 100, 125, 250, 500, 1000

CANx_BUSLOAD

2178

x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:

Beschreibung

2180

Ermittelt die aktuelle Buslast auf dem CAN-Bus und zählt die aufgetretenen Error-Frames.

CANx_BUSLOAD ermittelt die Buslast über die Anzahl und Länge der während der Zeit PERIOD über den CAN-Bus übertragenen Telegramme, bei Berücksichtigung der aktuellen Baudrate. Der Wert BUSLOAD wird jeweils nach Ablauf der Zeit PERIOD aktualisiert.

Ist das Bit RESET dauerhaft FALSE, wird die Anzahl der Error-Frames angezeigt, die seit dem letzten RESET aufgetreten sind.

> **HINWEIS**
>
> Läuft die Kommunikation auf dem CAN-Bus über das CANopen-Protokoll, dann ist es sinnvoll, den Wert von PERIOD auf die Dauer des SYNC-Zyklus zu setzen.
>
> Die Messperiode ist dabei nicht mit dem CANopen SYNC-Zyklus synchronisiert.

Parameter der Eingänge

2181

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein ausführen FALSE: Baustein wird nicht ausgeführt > Baustein-Eingänge sind nicht aktiv > Baustein-Ausgänge sind nicht spezifiziert
INIT	BOOL	TRUE (nur 1 Zyklus lang): Konfiguration der Messdauer PERIOD FALSE: im weiteren Programmablauf
RESET	BOOL	TRUE: ERRORFRAME zurücksetzen auf "0" FALSE: Funktion wird nicht ausgeführt
PERIOD	WORD	Zeit in [ms], über welche die Buslast ermittelt wird zulässig = 20...1 000 ms

Parameter der Ausgänge

2182

Parameter	Datentyp	Beschreibung
ERRORFRAMES	WORD	Anzahl der auf dem CAN-Bus aufgetretenen Error-Frames seit dem letzten Reset
BUSLOAD	BYTE	mittlere Buslast in [%] zulässig: 0...100

CANx_DOWNLOADID

11841

= CANx Download-ID

x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:

Beschreibung

11846

CANx_DOWNLOADID stellt den Download-Identifier für die CAN-Schnittstelle x ein.

Mit dem FB kann der Kommunikations-Identifier für den Programm-Download und das Debuggen eingestellt werden. Der neue Wert wird eingetragen, wenn der Eingang ENABLE auf TRUE gesetzt wird.

Der neue Wert wird erst nach einem RESET gültig (Spannung Aus/Ein oder Soft-Reset).

Parameter der Eingänge

649

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE (im 1. Zyklus): Parameter übernehmen und aktivieren sonst: diese Funktion wird nicht ausgeführt
ID	BYTE	Download-ID der CAN-Schnittstelle x setzen x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt) zulässig = 1...127 voreingestellt = 127 - (x-1)

CANx_ERRORHANDLER

2174

x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:

Beschreibung

2329
13991

Wenn die automatische Bus-Recover-Funktion genutzt werden soll (Voreinstellung), darf CANx_ERRORHANDLER **nicht** in das Programm eingebunden und instanziert werden!

CANx_ERRORHANDLER führt ein "manuelles" Bus-Recover auf der CAN-Schnittstelle x durch.

► Nach einem erkannten CAN-Busoff den FB für einen Zyklus mit BUSOFF_RECOVER = TRUE aufrufen, damit die Steuerung wieder auf dem CAN-Bus senden und empfangen kann.

► Anschließend im Anwendungsprogramm für diese CAN-Schnittstelle das Fehlerbit CANx_BUSOFF zurücksetzen.

> Die CAN-Schnittstelle arbeitet wieder.

Parameter der Eingänge

2177

Parameter	Datentyp	Beschreibung
BUSOFF_RECOVER	BOOL	TRUE (nur 1 Zyklus lang): > Bus-off-Zustand beheben > Neustart der CAN-Schnittstelle FALSE: Funktion wird nicht ausgeführt

CANx_RECEIVE

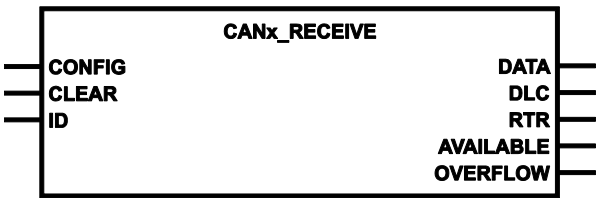
627

x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:



Beschreibung

13338

CANx_RECEIVE konfiguriert ein Datenempfangsobjekt und liest den Empfangspuffer des Datenobjektes aus.

- ► Den FB für jedes Datenobjekt in der Initialisierungsphase einmalig aufrufen, um dem CAN-Controller die Identifier der Datenobjekte bekannt zu machen.
- ► Im weiteren Programmzyklus CANx_RECEIVE zum Auslesen des jeweiligen Empfangspuffers aufrufen, bei langen Programmzyklen auch mehrfach.
- ► Je CAN-Schnittstelle sind maximal 256 Instanzen für den FB CANx_RECEIVE möglich.
- ► Im Standard Mode können alle 2048 IDs gleichzeitig genutzt werden
  Im Extended Mode können nur 256 (beliebige) IDs gleichzeitig genutzt werden.
- ► Jede ID (Standard oder Extended) kann nur einer FB-Instanz zugeordnet werden.
  Bei mehrfacher Nutzung einer ID: jeweils die letzte aufgerufene Instanz.
- ► Im FB CANx einstellen, ob CANx_RECEIVE Normal oder Extended Frames empfangen soll.
- > Wird CANx_RECEIVE für den Empfang eines Normal Frame konfiguriert, wird der Frame mit dieser ID nicht mehr an einen eventuell vorhandenen CANopen Stack weitergeleitet.
- > Wird eine ID außerhalb des zulässigen Bereichs eingestellt (abhängig von der Einstellung in CANx), wird der Funktionsbaustein nicht ausgeführt.
- ► Den Ausgang AVAILABLE auswerten, so dass neu eingegangene Datenobjekte rechtzeitig aus dem Puffer gelesen und weiterverarbeitet werden.
  Receive-Puffer: max. 16 Software-Puffer pro Identifier.
- > Jeder Aufruf des FB dekrementiert das Byte AVAILABLE um 1.
  Ist AVAILABLE = 0, sind keine Daten im Puffer.
- ► Den Ausgang OVERFLOW auswerten, um einen Überlauf des Datenpuffers zu erkennen.
  Wenn OVERFLOW = TRUE, dann ist mindestens 1 Datenobjekt verloren gegangen.

Parameter der Eingänge

2172

Parameter	Datentyp	Beschreibung
CONFIG	BOOL	TRUE (im 1. Zyklus): Datenobjekt konfigurieren FALSE: im weiteren Programmablauf
CLEAR	BOOL	TRUE: Empfangspuffer löschen FALSE: Funktion wird nicht ausgeführt
ID	DWORD	Nummer des Datenobjekt-Identifiers: Normal Frame (2^{11} IDs): 0...2 047 = 0x0000 0000...0x0000 07FF Extended Frame (2^{29} IDs): 0...536 870 911 = 0x0000 0000...0x1FFF FFFF

Parameter der Ausgänge

19810

Parameter	Datentyp	Beschreibung
DATA	ARRAY [0..7] OF BYTE	empfangene Daten (1...8 Bytes)
DLC	BYTE	Anzahl der Bytes des aus dem Empfangspuffer ausgelesenen CAN-Telegramms zulässig: 0...8
RTR	BOOL = FALSE	empfangene Nachricht war ein **R**emote **T**ransmission **R**equest (wird hier nicht unterstützt)
AVAILABLE	BYTE	Anzahl der empfangenen, aber noch nicht aus dem Empfangspuffer ausgelesenen CAN-Telegramme (vor dem Aufruf des FB). mögliche Werte = 0...16 0 = keine gültigen Daten vorhanden
OVERFLOW	BOOL	TRUE: Überlauf des Datenpuffers ⇨ Datenverlust! FALSE: Datenpuffer ist ohne Datenverlust

CANx_TRANSMIT

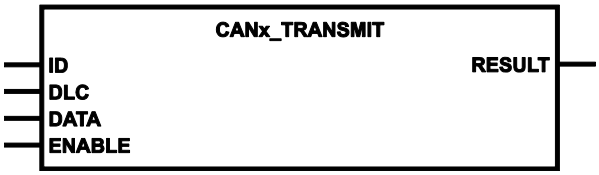
609

x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:



Beschreibung

2166

CANx_TRANSMIT übergibt in jedem Aufruf ein CAN-Datenobjekt (Message) an den CAN-Controller zur Übertragung.

► Den FB für jedes Datenobjekt im Programmzyklus aufgerufen, bei langen Programmzyklen auch mehrfach.
Transmit-Puffer: max. 16 Software- und 1 Hardware-Puffer für alle Identifier zusammen.

► Den Ausgang RESULT auswerten zur Prüfung, dass der Sendeauftrag angenommen wurde.

ⓘ Vereinfacht gilt bei 125 kBit/s, dass pro 1 ms ein Sendeauftrag ausgeführt werden kann.

Über den Eingang ENABLE kann die Ausführung des FB zeitweilig gesperrt werden (ENABLE = FALSE). Damit kann z.B. eine Busüberlastung verhindert werden.

Mehrere Datenobjekte (mit gleicher oder unterschiedlicher ID) können quasi gleichzeitig verschickt werden, wenn jedem Datenobjekt ein Merkerflag zugeordnet wird und mit diesem die Ausführung des FB über den ENABLE-Eingang gesteuert wird.

Parameter der Eingänge

19813

Parameter	Datentyp	Beschreibung
ID	DWORD	Nummer des Datenobjekt-Identifiers: Normal Frame (2^{11} IDs): 0...2 047 = 0x0000 0000...0x0000 07FF Extended Frame (2^{29} IDs): 0...536 870 911 = 0x0000 0000...0x1FFF FFFF
DLC	BYTE	Anzahl der zu übertragenden Bytes aus dem Array DATA zulässig: 0...8
DATA	ARRAY [0..7] OF BYTE	zu sendende Daten (1...8 Bytes)
ENABLE	BOOL	TRUE: Baustein ausführen FALSE: Baustein wird nicht ausgeführt > Baustein-Eingänge sind nicht aktiv > Baustein-Ausgänge sind nicht spezifiziert

Parameter der Ausgänge

2168

Parameter	Datentyp	Beschreibung
RESULT	BOOL	TRUE (nur 1 Zyklus lang): der Baustein hat den Sendeauftrag angenommen FALSE: Sendeauftrag wurde nicht angenommen

5.2.2 Bausteine: CANopen-Master

Inhalt

1870

Für den CANopen-Master stellt **ifm electronic** eine Reihe von Bausteinen zur Verfügung, die im Folgenden erklärt werden.

CANx_MASTER_EMCY_HANDLER

2006

x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_CANopenxMaster_Vxxyyzz.LIB`

Symbol in CODESYS:

Beschreibung

2009

CANx_MASTER_EMCY_HANDLER verwaltet den geräteeigenen Fehlerstatus des Masters. Der FB muss in folgenden Fällen aufgerufen werden:

- der Fehlerstatus soll ins Netzwerk übertragen werden und
- die Fehlermeldungen des Anwendungsprogramms sollen im Objektverzeichnis gespeichert werden.

Über den FB können die aktuellen Werte aus dem Error-Register (Index 0x1001/01) und Error Field (Index 0x1003/0-5) des CANopen-Objektverzeichnis ausgelesen werden.

Sollen anwendungsspezifische Fehlernachrichten im Objektverzeichnis gespeichert werden, muss CANx_MASTER_EMCY_HANDLER **nach** dem (mehrfachen) Bearbeiten von **CANx_MASTER_SEND_EMERGENCY** (→ S. 84) aufgerufen werden.

Parameter der Eingänge

2010

Parameter	Datentyp	Beschreibung
CLEAR_ERROR_FIELD	BOOL	FALSE ⇨ TRUE (Flanke): • Inhalt des ERROR_FIELD an FB-Ausgang ausgeben • Inhalt des ERROR_FIELD im Objektverzeichnis löschen sonst: diese Funktion wird nicht ausgeführt

Parameter der Ausgänge

2011

Parameter	Datentyp	Beschreibung
ERROR_REGISTER	BYTE	Zeigt den Inhalt des OBV Index 0x1001 (Error-Register)
ERROR_FIELD	ARRAY [0..5] OF WORD	Zeigt den Inhalt des OBV Index 0x1003 (Error-Field) ERROR_FIELD[0]: Anzahl der gespeicherten Fehler ERROR_FIELD[1...5]: gespeicherte Fehler, der jüngste Fehler steht im Index [1]

CANx_MASTER_SEND_EMERGENCY

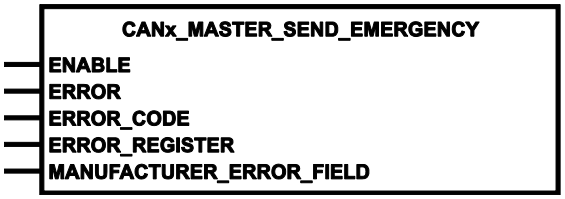
2012

x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_CANopenxMaster_Vxxyyzz.LIB`

Symbol in CODESYS:



Beschreibung

2015

CANx_MASTER_SEND_EMERGENCY versendet anwendungsspezifische Fehlerstatus. Der FB wird aufgerufen, wenn der Fehlerstatus an andere Geräte im Netzwerkverbund übertragen werden soll.

Sollen anwendungsspezifische Fehlernachrichten im Objektverzeichnis gespeichert werden, muss **CANx_MASTER_EMCY_HANDLER** (→ S. 83) **nach** dem (mehrfachen) Bearbeiten von CANx_MASTER_SEND_EMERGENCY aufgerufen werden.

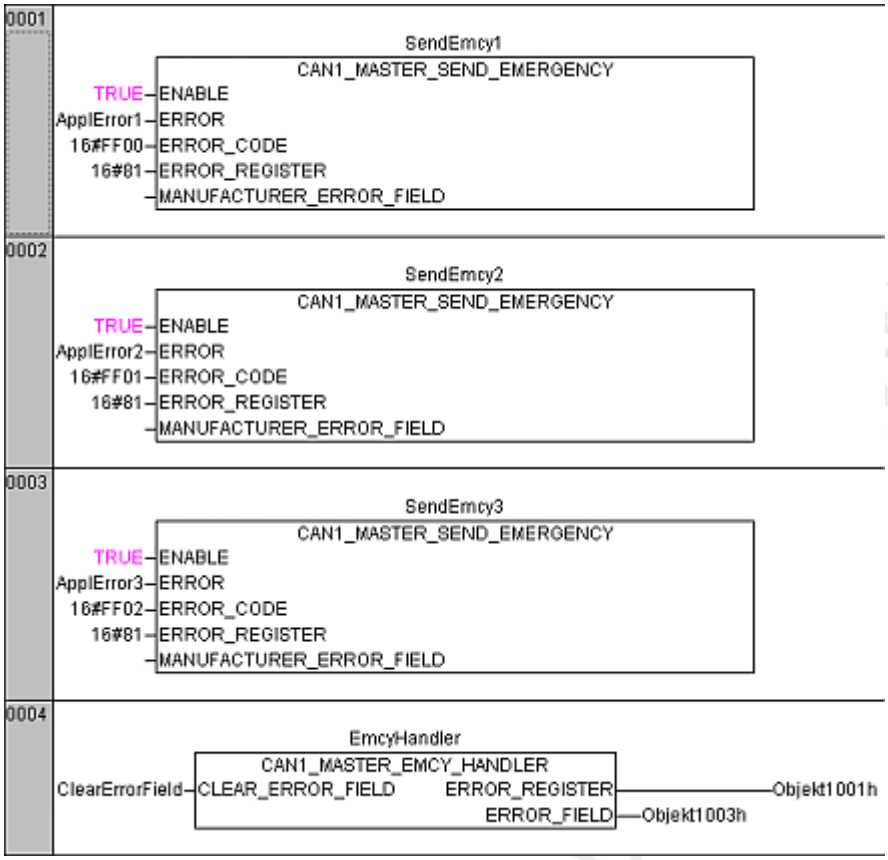
Parameter der Eingänge

2016

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein ausführen FALSE: Baustein wird nicht ausgeführt > Baustein-Eingänge sind nicht aktiv > Baustein-Ausgänge sind nicht spezifiziert
ERROR	BOOL	Über diesen Eingang wird dem FB die Information übergeben, ob der zum konfigurierten Fehlercode gehörende Fehler aktuell anliegt. FALSE ⇨ TRUE (Flanke): sendet den anstehenden Fehler-Code falls Eingang in der letzten Sekunde nicht TRUE war TRUE ⇨ FALSE (Flanke) UND Fehler steht nicht mehr an: Nach Verzögerung von ca. 1 s: > Null-Fehlermeldung wird gesendet sonst: diese Funktion wird nicht ausgeführt
ERROR_CODE	WORD	Der Error-Code gibt detailliert Auskunft über den erkannten Fehler. Die Werte sollten gemäß der CANopen-Spezifikation eingetragen werden.
ERROR_REGISTER	BYTE	ERROR_REGISTER gibt die Art des Fehlers an. Der hier angegebene Wert wird mit allen anderen aktuell aktiven Fehlernachrichten bitweise ODER-verknüpft. Der sich hierbei ergebende Wert wird ins Error-Register (Index 1001_{16}/00) geschrieben und mit der EMCY-Nachricht versendet. Die Werte sollten gemäß der CANopen-Spezifikation eingetragen werden.
MANUFACTURER_ERROR_FIELD	ARRAY [0..4] OF BYTE	Hier können bis zu 5 Bytes anwendungsspezifische Fehlerinformationen eingetragen werden. Das Format ist dabei frei wählbar.

Beispiel: CANx_MASTER_SEND_EMERGENCY

2018



In diesem Beispiel werden nacheinander 3 Fehlermeldungen generiert:

1. ApplError1, Code = 0xFF00 im Fehlerregister 0x81
2. ApplError2, Code = 0xFF01 im Fehlerregister 0x81
3. ApplError3, Code = 0xFF02 im Fehlerregister 0x81

Der FB CAN1_MASTER_EMCY_HANDLER sendet die Fehlermeldungen an das Fehler-Register "Objekt 0x1001" im Fehler-Array "Objekt 0x1003".

CANx_MASTER_STATUS

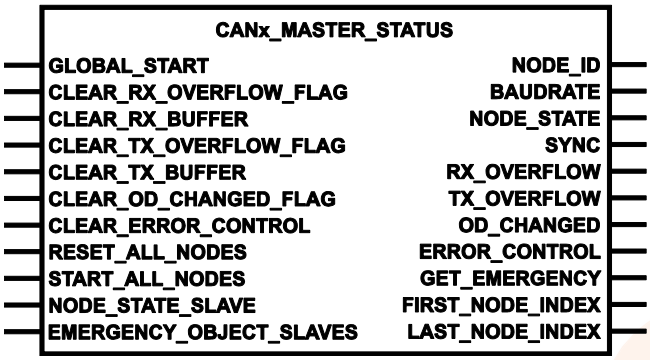
2692

x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_CANopenxMaster_Vxxyyzz.LIB`

Symbol in CODESYS:



Beschreibung

2024

Status-Anzeige des als CANopen-Master eingesetzten Gerätes

Der FB zeigt den Status des als CANopen-Master eingesetzten Gerätes an. Weitere Möglichkeiten:
- den Status des Netzwerks überwachen
- den Status der angeschlossenen Slaves überwachen
- die Slaves im Netzwerk zurücksetzen oder starten.

Der FB vereinfacht die Anwendung der CODESYS-CANopen-Master-Bibliotheken. Wir empfehlen dringend, die Auswertung des Netzwerkstatus und der Fehlermeldungen über diesen FB vorzunehmen.

Parameter der Eingänge

19861

Parameter	Datentyp	Beschreibung
GLOBAL_START	BOOL	TRUE: Alle angeschlossenen Netzwerkteilnehmer (Slaves) werden gleichzeitig bei der Netzwerkinitialisierung gestartet (⇨ Zustand OPERATIONAL). FALSE: Die angeschlossenen Netzwerkteilnehmer werden einzeln nacheinander gestartet.
CLEAR_RX_OVERFLOW_FLAG	BOOL	FALSE ⇨ TRUE (Flanke): Fehlerflag RX_OVERFLOW löschen sonst: diese Funktion wird nicht ausgeführt
CLEAR_RX_BUFFER	BOOL	FALSE ⇨ TRUE (Flanke): Daten im Empfangspuffer löschen sonst: diese Funktion wird nicht ausgeführt
CLEAR_TX_OVERFLOW_FLAG	BOOL	FALSE ⇨ TRUE (Flanke): Fehlerflag TX_OVERFLOW löschen sonst: diese Funktion wird nicht ausgeführt
CLEAR_TX_BUFFER	BOOL	FALSE ⇨ TRUE (Flanke): Daten im Sendepuffer löschen sonst: diese Funktion wird nicht ausgeführt
CLEAR_OD_CHANGED_FLAG	BOOL	FALSE ⇨ TRUE (Flanke): Flag OD_CHANGED löschen sonst: diese Funktion wird nicht ausgeführt
CLEAR_ERROR_CONTROL	BOOL	FALSE ⇨ TRUE (Flanke): Die Guard-Fehlerliste (ERROR_CONTROL) löschen sonst: diese Funktion wird nicht ausgeführt
RESET_ALL_NODES	BOOL	FALSE ⇨ TRUE (Flanke): Alle angeschlossenen Netzwerkteilnehmer (Slaves) werden per NMT-Kommando zurückgesetzt sonst: diese Funktion wird nicht ausgeführt
START_ALL_NODES	BOOL	FALSE ⇨ TRUE (Flanke): Alle angeschlossenen Netzwerkteilnehmer (Slaves) werden per NMT-Kommando gestartet sonst: diese Funktion wird nicht ausgeführt
NODE_STATE_SLAVES	DWORD	Zeigeradresse auf ein Array [0.. MAX_NODEINDEX] of CANx_NODE_STATE In das Array werden die Statusinformationen der im CANopen-Netzwerk befindlichen Slaves geschrieben. Über den Zugriff auf bestimmte Werte in den Strukturen im Array kann auch das Verhalten der Slaves gesteuert werden. MAX_NODEINDEX ist eine Konstante, die beim Übersetzen der Anwendung von CODESYS ermittelt wird. Die Adresse mit dem Operator ADR ermitteln und dem Baustein übergeben! Beispiel-Code → Kapitel **Beispiel: CANx_MASTER_STATUS** (→ S. 90)
EMERGENCY_OBJECT_SLAVES	DWORD	Zeigeradresse auf ein Array [0.. MAX_NODEINDEX] of CANx_EMERGENCY_MESSAGE Zeigt die zuletzt aufgetretenen Fehlermeldungen aller Netzwerkknoten. Die Adresse mit dem Operator ADR ermitteln und dem Baustein übergeben!

Parameter der Ausgänge

2696

Parameter	Datentyp	Beschreibung
NODE_ID	BYTE	aktuelle Knoten-ID des CANopen-Masters
BAUDRATE	WORD	aktuelle Baudrate des CANopen-Masters in [kBaud]
NODE_STATE	INT	aktueller Status des CANopen-Masters
SYNC	BOOL	SYNC-Signal des CANopen-Masters TRUE: Im letzten Zyklus wurde ein SYNC-Signal gesendet FALSE: Im letzten Zyklus wurde kein SYNC-Signal gesendet
RX_OVERFLOW	BOOL	TRUE: Fehler: Empfangspuffer-Überlauf FALSE: kein Überlauf
TX_OVERFLOW	BOOL	TRUE: Fehler: Sendepuffer-Überlauf FALSE: kein Überlauf
OD_CHANGED	BOOL	TRUE: Daten im Objektverzeichnis des CANopen-Masters wurden geändert FALSE: keine Datenänderung
ERROR_CONTROL	ARRAY [0..7] OF BYTE	Das Array enthält die Liste (max. 8) der fehlenden Netzwerkknoten (Guard- oder Heartbeat-Fehler)
GET_EMERGENCY	STRUCT CANx_EMERGENCY_MESSAGE	Am Ausgang stehen die Daten für die Struktur CANx_EMERGENCY_MESSAGE zur Verfügung. Es wird immer die zuletzt empfangene EMCY-Nachricht im CANopen-Netzwerk angezeigt. Liste aller aufgetretenen Fehler erhalten: das Array EmergencyObjectSlavesArray auswerten!
FIRST_NODE_INDEX	INT	Bereich, in dem sich die Knotennummern der an diesem CAN-Bus angeschlossenen Knoten (Slaves) befinden
LAST_NODE_INDEX	INT	

Parameter der internen Strukturen

2698

Hier sehen Sie die Strukturen der in diesem Baustein genutzten Arrays.

Die Anwendung des FB CANx_MASTER_STATUS zeigen Ihnen die Code-Fragmente am Beispiel des Controllers CR0032 → Kapitel **Beispiel: CANx_MASTER_STATUS** (→ S. 90).

Struktur von CANx_EMERGENCY_MESSAGE

13996

Die Struktur ist in den globalen Variablen der Bibliothek `ifm_CR0232_CANopenMaster_Vxxyyzz.LIB` angelegt.

Parameter	Datentyp	Beschreibung
NODE_ID	BYTE	Node-ID des Teilnehmers, von dem die EMCY-Nachricht empfangen wurde
ERROR_CODE	WORD	Error-Code mit der Information, welcher Fehler aufgetreten ist. → CANopen-Spezifikation CiA Draft Standard 301 Version 4
ERROR_REGISTER	BYTE	Wert im Error-Register (Index 0x1001/00) des sendenden Teilnehmers
MANUFACTURER_ERROR_FIELD	ARRAY [0..4] OF BYTE	herstellerspezifischer Datenbereich in der EMCY-Nachricht

Struktur von CANx_NODE_STATE

13997

Die Struktur ist in den globalen Variablen der Bibliothek `ifm_CR0232_CANopenMaster_Vxxyyzz.LIB` angelegt.

Parameter	Datentyp	Beschreibung
NODE_ID	BYTE	Node-ID des CANopen-Slaves, zu dem die Statusinformationen und Konfigurationsflags in der Struktur gehören
NODE_STATE	BYTE	aktueller Status des CANopen-Slaves aus Sicht des CANopen-Stacks des CANopen-Masters
LAST_STATE	BYTE	der letzte bekannte Status des CANopen-Slaves 0 = Bootup-Nachricht vom CANopen-Slave empfangen 4 = CANopen-Slave im Status PRE-OPERATIONAL und wird per SDO-Zugriff konfiguriert 5 = CANopen-Slave im Status OPERATIONAL 127 = CANopen-Slave im Status PRE-OPERATIONAL
RESET_NODE	BOOL	Flag zum manuellen Zurücksetzen des CANopen-Slaves (NMT-Kommando = `Reset_Node`)
START_NODE	BOOL	Flag zum manuellen Starten des CANopen-Slaves (NMT-Kommando = `start`)
PREOP_NODE	BOOL	Flag zum manuellen Versetzen des CANopen-Slaves in den Zustand PRE-OPERATIONAL (NMT-Kommando = `enter PRE-OPERATIONAL`)
SET_TIMEOUT_STATE	BOOL	Flag zum manuellen Überspringen der Initialisierung eines CANopen-Slaves, wenn Folgendes zutrifft: • Slave ist nicht im Netzwerk vorhanden • **und** Slave ist nicht als optional konfiguriert
SET_NODE_STATE	BOOL	Flag zum manuellen Einleiten der Initialisierung eines CANopen-Slaves Der Slave hatte sich beim Zugriff auf das Objekt 0x1000 als ein anderer Gerätetyp identifiert, als in der EDS-Datei angegeben ist, die in der CODESYS-Steuerungskonfiguration eingebunden wurde

Beispiel: CANx_MASTER_STATUS

2031

Slave-Informationen

2699

Damit Sie auf die Informationen der einzelnen CANopen-Knoten zugreifen können, müssen Sie ein Array der jeweiligen Struktur anlegen. Die Strukturen sind in der Bibliothek enthalten. Sie können Sie im Bibliotheksverwalter unter [Datentypen] sehen.

Die Anzahl der Array-Elemente wird bestimmt durch die Globale Variable MAX_NODEINDEX, die automatisch vom CANopen-Stack angelegt wird. Sie enthält die Anzahl der im Netzwerkkonfigurator angegebenen Slaves minus 1.

> Die Nummern der Array-Elemente entsprechen **nicht** der Node-ID. Der Identifier kann aus der jeweiligen Struktur unter NODE_ID ausgelesen werden.

Programm-Beispiel zu CAN1_MASTER_STATUS

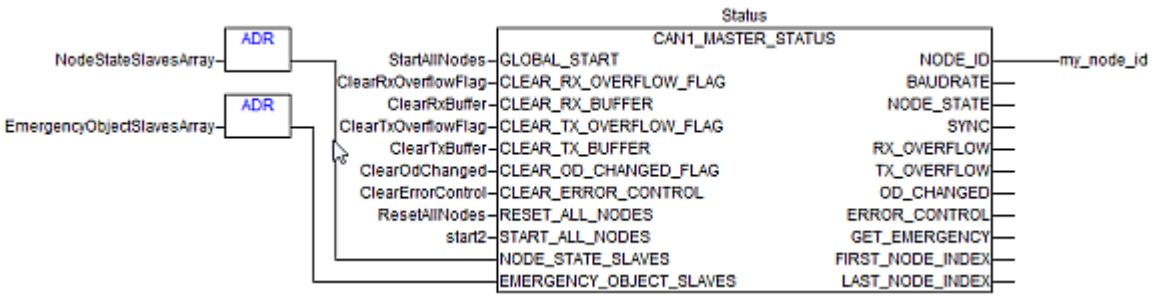
20651

Variablen-Deklaration:

```
VAR
    Status: CAN1_MASTER_STATUS;


    LedStatus: BOOL:= TRUE;
    StartAllNodes: BOOL:= TRUE;
    ClearRxOverflowFlag: BOOL;
    ClearRxBuffer: BOOL;
    ClearTxOverflowFlag: BOOL;
    ClearTxBuffer: BOOL;
    ClearOdChanged: BOOL;
    ClearErrorControl: BOOL;
    ResetAllNodes: BOOL;
    NodeStateSlavesArray: ARRAY [0..MAX_NODEINDEX] OF CAN1_NODE_STATE;
    EmergencyObjectSlavesArray: ARRAY[0..MAX_NODEINDEX] OF CAN1_EMERGENCY_MESSAGE;
    my_node_id: BYTE;
    my_baudrate: WORD;
    my_node_state: INT;
    Sync: BOOL;
    RxOverflow: BOOL;
    TxOverflow: BOOL;
    OdChanged: BOOL;
    GuardHeartbeatErrorArray: ARRAY[0..7] OF BYTE;
    GetEmergency: CAN1_EMERGENCY_MESSAGE;
    start2: BOOL;
    Ency_handler: CAN1_MASTER_EMCY_HANDLER;
    reset_emcy: BOOL;
END_VAR
```

Programm-Beispiel:



Struktur Knoten-Status

2034

```
TYPE CAN1_NODE_STATE :
STRUCT
    NODE_ID: BYTE;
    NODE_STATE: BYTE;
    LAST_STATE: BYTE;
    RESET_NODE: BOOL;
    START_NODE: BOOL;
    PREOP_NODE: BOOL;
    SET_TIMEOUT_STATE: BOOL;
    SET_NODE_STATE: BOOL;
END_STRUCT
END_TYPE
```

Struktur Emergency_Message

2035

```
TYPE CAN1_EMERGENCY_MESSAGE :
STRUCT
    NODE_ID: BYTE;
    ERROR_CODE: WORD;
    ERROR_REGISTER: BYTE;
    MANUFACTURER_ERROR_FIELD: ARRAY[0..4] OF BYTE;
END_STRUCT
END_TYPE
```

5.2.3 Bausteine: CANopen-Slave

Inhalt

1874

Für den CANopen-Slave stellt **ifm electronic** eine Reihe von Bausteinen zur Verfügung, die im Folgenden erklärt werden.

CANx_SLAVE_EMCY_HANDLER

2050

x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_CANopenxSlave_Vxxyyzz.LIB`

Symbol in CODESYS:

Beschreibung

2053

CANx_SLAVE_EMCY_HANDLER verwaltet den geräteeigenen Fehlerstatus des CANopen-Slaves:
- Error Register (Index 0x1001) und
- Error Field (Index 0x1003) des CANopen Objektverzeichnis.

► Den FB in folgenden Fällen aufrufen:
- der Fehlerstatus soll ins CAN-Netzwerk übertragen werden und
- die Fehlernachrichten des Anwendungsprogramms sollen im Objektverzeichnis gespeichert werden.

Sollen die Fehlernachrichten im Objektverzeichnis gespeichert werden?

► **Nach** dem (mehrfachen) Bearbeiten von **CANx_SLAVE_SEND_EMERGENCY** (→ S. 95) einmalig CANx_SLAVE_EMCY_HANDLER aufrufen!

Parameter der Eingänge

2054

Parameter	Datentyp	Beschreibung
CLEAR_ERROR_FIELD	BOOL	FALSE ⇨ TRUE (Flanke): • Inhalt des ERROR_FIELD an FB-Ausgang ausgeben • Inhalt des ERROR_FIELD im Objektverzeichnis löschen sonst: diese Funktion wird nicht ausgeführt

Parameter der Ausgänge

2055

Parameter	Datentyp	Beschreibung
ERROR_REGISTER	BYTE	Zeigt den Inhalt des OBV Index 0x1001 (Error-Register)
ERROR_FIELD	ARRAY [0..5] OF WORD	Zeigt den Inhalt des OBV Index 0x1003 (Error-Field) ERROR_FIELD[0]: Anzahl der gespeicherten Fehler ERROR_FIELD[1...5]: gespeicherte Fehler, der jüngste Fehler steht im Index [1]

CANx_SLAVE_NODEID

2044

= CANx Slave Node-ID

x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_CANopenxSlave_Vxxyyzz.LIB`

Symbol in CODESYS:

Beschreibung

2049

CANx_SLAVE_NODEID ermöglicht das Einstellen der Node-ID eines CANopen-Slaves zur Laufzeit des Anwendungsprogramms.

Der FB wird im Normalfall bei der Initialisierung der Steuerung einmalig, im ersten Zyklus, aufgerufen. Anschließend wird der Eingang ENABLE wieder auf FALSE gesetzt.

Parameter der Eingänge

2047

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	FALSE ⇨ TRUE (Flanke): Parameter übernehmen und aktivieren sonst: diese Funktion wird nicht ausgeführt
NODEID	BYTE	Node-ID = ID des Knotens zulässige Werte = 1...127

CANx_SLAVE_SEND_EMERGENCY

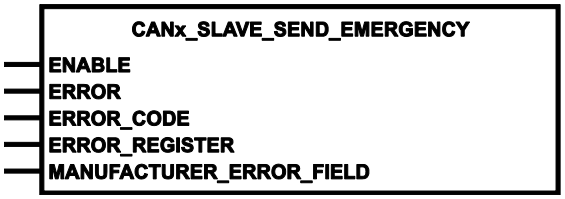
2056

x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_CANopenxSlave_Vxxyyzz.LIB`

Symbol in CODESYS:



Beschreibung

2059

CANx_SLAVE_SEND_EMERGENCY versendet anwendungsspezifische Fehlerstatus. Das sind Fehlernachrichten, die zusätzlich zu den geräteinternen Fehlernachrichten (z.B. Kurzschluss am Ausgang) gesendet werden sollen.

► Den FB aufrufen, wenn der Fehlerstatus an andere Geräte im Netzwerkverbund übertragen werden soll.

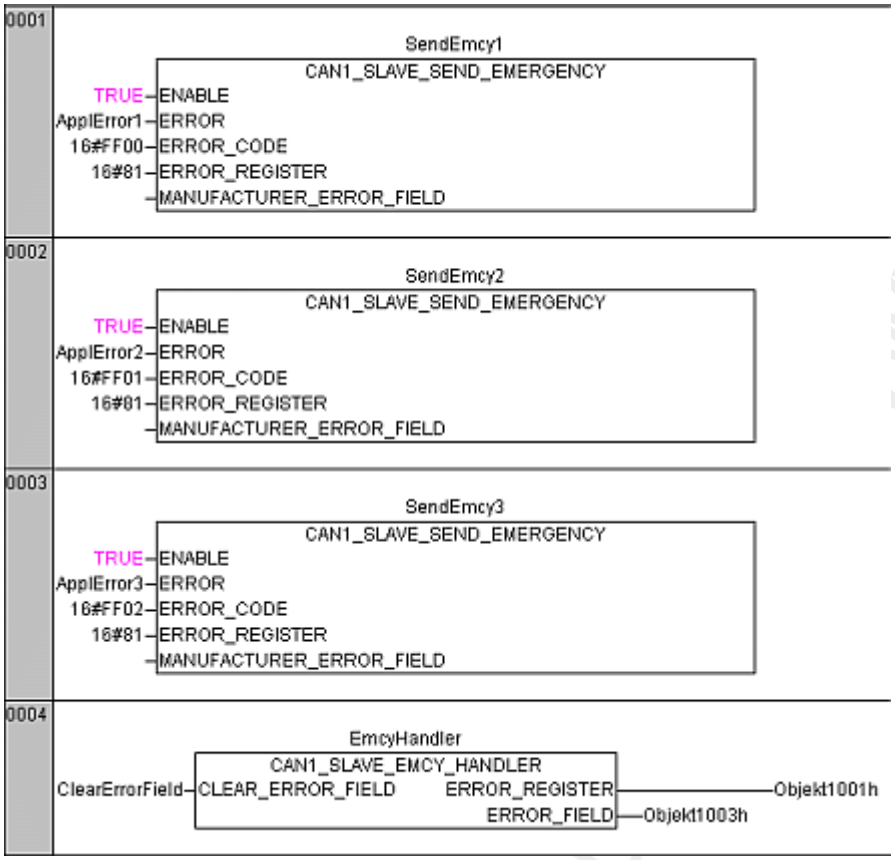
Parameter der Eingänge

2060

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein ausführen FALSE: Baustein wird nicht ausgeführt > Baustein-Eingänge sind nicht aktiv > Baustein-Ausgänge sind nicht spezifiziert
ERROR	BOOL	Über diesen Eingang wird dem FB die Information übergeben, ob der zum konfigurierten Fehlercode gehörende Fehler aktuell anliegt. FALSE ⇨ TRUE (Flanke): sendet den anstehenden Fehler-Code falls Eingang in der letzten Sekunde nicht TRUE war TRUE ⇨ FALSE (Flanke) UND Fehler steht nicht mehr an: Nach Verzögerung von ca. 1 s: > Null-Fehlermeldung wird gesendet sonst: diese Funktion wird nicht ausgeführt
ERROR_CODE	WORD	Der Error-Code gibt detailliert Auskunft über den erkannten Fehler. Die Werte sollten gemäß der CANopen-Spezifikation eingetragen werden.
ERROR_REGISTER	BYTE	ERROR_REGISTER gibt die Art des Fehlers an. Der hier angegebene Wert wird mit allen anderen aktuell aktiven Fehlernachrichten bitweise ODER-verknüpft. Der sich hierbei ergebende Wert wird ins Error-Register (Index 1001_{16}/00) geschrieben und mit der EMCY-Nachricht versendet. Die Werte sollten gemäß der CANopen-Spezifikation eingetragen werden.
MANUFACTURER_ERROR_FIELD	ARRAY [0..4] OF BYTE	Hier können bis zu 5 Bytes anwendungsspezifische Fehlerinformationen eingetragen werden. Das Format ist dabei frei wählbar.

Beispiel: CANx_SLAVE_SEND_EMERGENCY

2062



In diesem Beispiel werden nacheinander 3 Fehlermeldungen generiert:

1. ApplError1, Code = 0xFF00 im Fehlerregister 0x81
2. ApplError2, Code = 0xFF01 im Fehlerregister 0x81
3. ApplError3, Code = 0xFF02 im Fehlerregister 0x81

Der FB CAN1_SLAVE_EMCY_HANDLER sendet die Fehlermeldungen an das Fehler-Register "Objekt 0x1001" im Fehler-Array "Objekt 0x1003".

CANx_SLAVE_SET_PREOP

2700

x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_CANopenxSlave_Vxxyyzz.LIB`

Symbol in CODESYS:

Beschreibung

2703

CANx_SLAVE_SET_PREOP schaltet den Betriebsmodus dieses CANopen-Slaves von OPERATIONAL auf PRE-OPERATIONAL.

Normalerweise schaltet das Gerät im Fehlerfall wie folgt (abhängig von der Fehlerklasse):
- ein fataler Fehler führt zu Soft-Reset der Steuerung
- ein Error-Stop-Fehler führt zu einem System-Stop

Unter bestimmten Bedingungen kann es erforderlich sein, dass das Anwendungsprogramm den Betriebszustand des als Slave arbeitenden Geräts auf PRE-OPERATIONAL setzt. Dies erfolgt über den hier beschriebenen FB.

Parameter der Eingänge

2704

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	FALSE ⇨ TRUE (Flanke): Slave auf PRE-OPERATIONAL setzen sonst: diese Funktion wird nicht ausgeführt

CANx_SLAVE_STATUS

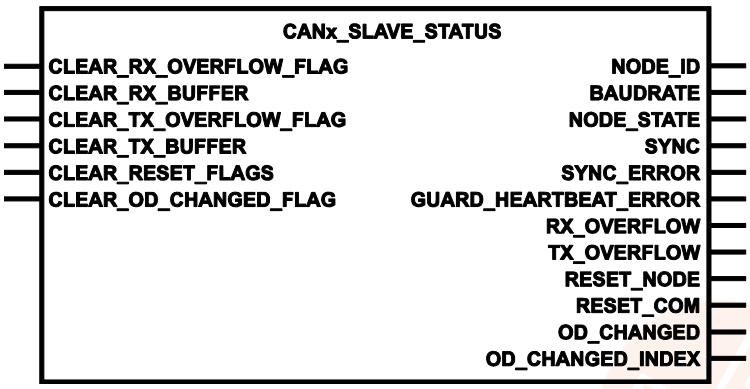
2706

x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_CANopenxSlave_Vxxyyzz.LIB`

Symbol in CODESYS:



Beschreibung

2707

CANx_SLAVE_STATUS zeigt den Status des als CANopen-Slave eingesetzten Gerätes.

Wir empfehlen dringend, die Auswertung des Netzwerkstatus über diesen FB vorzunehmen.

Parameter der Eingänge

2708

Parameter	Datentyp	Beschreibung
CLEAR_RX_OVERFLOW_FLAG	BOOL	FALSE ⇨ TRUE (Flanke): Fehlerflag RX_OVERFLOW löschen sonst: diese Funktion wird nicht ausgeführt
CLEAR_RX_BUFFER	BOOL	FALSE ⇨ TRUE (Flanke): Daten im Empfangspuffer löschen sonst: diese Funktion wird nicht ausgeführt
CLEAR_TX_OVERFLOW_FLAG	BOOL	FALSE ⇨ TRUE (Flanke): Fehlerflag TX_OVERFLOW löschen sonst: diese Funktion wird nicht ausgeführt
CLEAR_TX_BUFFER	BOOL	FALSE ⇨ TRUE (Flanke): Daten im Sendepuffer löschen sonst: diese Funktion wird nicht ausgeführt
CLEAR_RESET_FLAGS	BOOL	FALSE ⇨ TRUE (Flanke): Flag RESET_NODE löschen Flag RESET_COM löschen sonst: diese Funktion wird nicht ausgeführt
CLEAR_OD_CHANGED_FLAGS	BOOL	FALSE ⇨ TRUE (Flanke): Flag OD_CHANGED löschen Flag OD_CHANGED_INDEX löschen sonst: diese Funktion wird nicht ausgeführt

Parameter der Ausgänge

2068

Parameter	Datentyp	Beschreibung
NODE_ID	BYTE	aktuelle Knoten-ID des CANopen-Slaves
BAUDRATE	WORD	aktuelle Baudrate des CANopen-Knotens in [kBaud]
NODE_STATE	BYTE	aktueller Status des CANopen-Slaves 0 = Bootup-Nachricht versendet 4 = CANopen-Slave im Status PRE-OPERATIONAL und wird per SDO-Zugriff konfiguriert 5 = CANopen-Slave im Status OPERATIONAL 127 = CANopen-Slave im Status PRE-OPERATIONAL
SYNC	BOOL	SYNC-Signal des CANopen-Masters TRUE: Im letzten Zyklus wurde ein SYNC-Signal empfangen FALSE: Im letzten Zyklus wurde kein SYNC-Signal empfangen
SYNC_ERROR	BOOL	TRUE: Fehler: das SYNC-Signal des Masters wurde nicht oder zu spät (nach Ablauf von ComCyclePeriod) empfangen FALSE: kein SYNC-Fehler
GUARD_HEARTBEAT_ERROR	BOOL	TRUE: Fehler: das Guarding- oder Heartbeat-Signal des Masters wurde nicht oder zu spät empfangen FALSE: kein Guarding- oder Heartbeat-Fehler
RX_OVERFLOW	BOOL	TRUE: Fehler: Empfangspuffer-Überlauf FALSE: kein Überlauf
TX_OVERFLOW	BOOL	TRUE: Fehler: Sendepuffer-Überlauf FALSE: kein Überlauf
RESET_NODE	BOOL	TRUE: CANopen-Stack des Slaves vom Master zurückgesetzt FALSE: CANopen-Stack des Slaves nicht zurückgesetzt
RESET_COM	BOOL	TRUE: Kommunikations-Interface des CAN-Stack wurde vom Master zurückgesetzt FALSE: Kommunikations-Interface nicht zurückgesetzt
OD_CHANGED	BOOL	TRUE: Daten im Objektverzeichnis des CANopen-Masters wurden geändert FALSE: keine Datenänderung
OD_CHANGED_INDEX	INT	Index des zuletzt geänderten Objektverzeichnis-Eintrags

5.2.4 Bausteine: CANopen SDOs

Inhalt

2071

Hier finden Sie **ifm**-Bausteine für den Umgang von CANopen mit Service Data Objects (SDOs).

CANx_SDO_READ

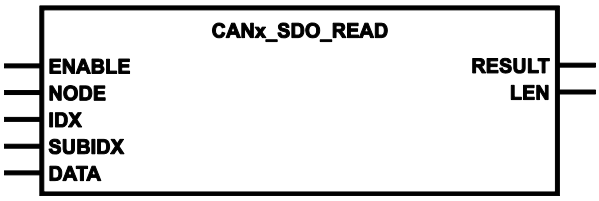
621

x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:



Beschreibung

624

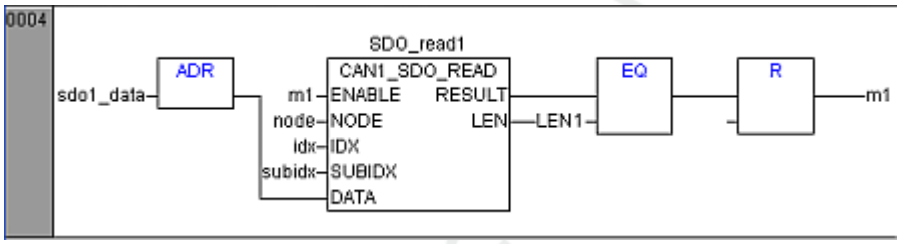
CANx_SDO_READ liest das →**SDO** (→ S. 253) mit den angegebenen Indizes aus dem Knoten aus.

Voraussetzung: Knoten muss sich im Zustand PRE-OPERATIONAL oder OPERATIONAL befinden.

Über diese Indizes können die Einträge im Objektverzeichnis gelesen werden. Dadurch ist es möglich, die Knotenparameter gezielt zu lesen.

> Gefahr von Datenverlust!
> Genügend Speicher für das angeforderte SDO bereitstellen!
> Ansonsten werden dahinter liegende Daten überschrieben

Beispiel:



Parameter der Eingänge

625

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein ausführen FALSE: Baustein wird nicht ausgeführt > Baustein-Eingänge sind nicht aktiv > Baustein-Ausgänge sind nicht spezifiziert
NODE	BYTE	CANopen-ID des Knotens zulässig = 1...127 = 0x01...0x7F
IDX	WORD	Index im Objektverzeichnis
SUBIDX	BYTE	Subindex bezogen auf den Index im Objektverzeichnis
DATA	DWORD	Adresse des Empfangsdaten-Arrays zulässige Länge = 0...255 Die Adresse mit dem Operator ADR ermitteln und dem Baustein übergeben!

Parameter der Ausgänge

626

Parameter	Datentyp	Beschreibung
RESULT	BYTE	Rückmeldung des Funktionsbausteins (mögliche Meldungen → folgende Tabelle)
LEN	WORD	Länge des Eintrags in "Anzahl der Bytes" Der Wert für LEN darf nicht größer sein als die Größe des Empfangs-Arrays. Andernfalls werden beliebige Daten in der Anwendung überschrieben.

Mögliche Ergebnisse für RESULT:

Wert dez	Wert hex	Beschreibung
0	00	FB ist inaktiv
1	01	FB-Ausführung wurde ohne Fehler beendet – Daten sind gültig
2	02	Funktionsbaustein ist aktiv (Aktion noch nicht beendet)
3	03	Fehler, keine Daten während der Überwachungszeit empfangen

CANx_SDO_WRITE

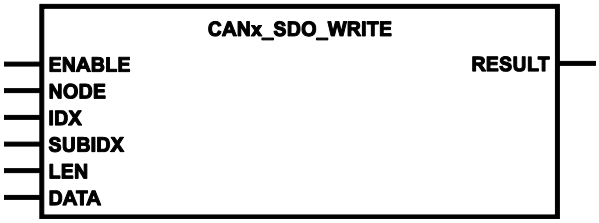
615

x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:



Beschreibung

618

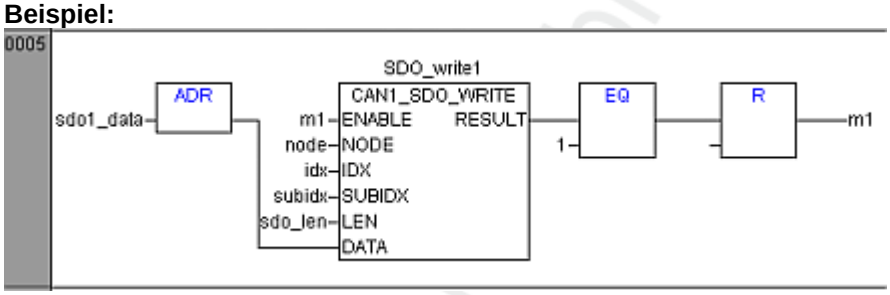
CANx_SDO_WRITE schreibt das →**SDO** (→ S. 253) mit den angegebenen Indizes in den Knoten.

Voraussetzung: Knoten muss sich im Zustand PRE-OPERATIONAL oder OPERATIONAL befinden.

Über diesen FB können die Einträge im Objektverzeichnis geschrieben werden. Dadurch ist es möglich, die Knotenparameter gezielt zu setzen.

Der Wert für LEN muss kleiner sein als die Größe des Sende-Arrays. Andernfalls werden beliebige Daten versendet.

Beispiel:



Parameter der Eingänge

619

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein ausführen FALSE: Baustein wird nicht ausgeführt > Baustein-Eingänge sind nicht aktiv > Baustein-Ausgänge sind nicht spezifiziert
NODE	BYTE	CANopen-ID des Knotens zulässig = 1...127 = 0x01...0x7F
IDX	WORD	Index im Objektverzeichnis
SUBIDX	BYTE	Subindex bezogen auf den Index im Objektverzeichnis
LEN	WORD	Länge des Eintrags in "Anzahl der Bytes" Der Wert für LEN darf nicht größer sein als die Größe des Sende-Arrays. Andernfalls werden beliebige Daten versendet.
DATA	DWORD	Adresse des Sendedaten-Arrays zulässige Länge = 0...255 Die Adresse mit dem Operator ADR ermitteln und dem Baustein übergeben!

Parameter der Ausgänge

620

Parameter	Datentyp	Beschreibung
RESULT	BYTE	Rückmeldung des Funktionsbausteins (mögliche Meldungen → folgende Tabelle)

Mögliche Ergebnisse für RESULT:

Wert dez	Wert hex	Beschreibung
0	00	FB ist inaktiv
1	01	FB-Ausführung wurde ohne Fehler beendet – Daten sind gültig
2	02	Funktionsbaustein ist aktiv (Aktion noch nicht beendet)
3	03	Fehler, Daten können nicht übertragen werden

5.2.5 Bausteine: SAE J1939

Inhalt

2273

Für SAE J1939 stellt **ifm electronic** eine Reihe von Bausteinen zur Verfügung, die im Folgenden erklärt werden.

J1939_x

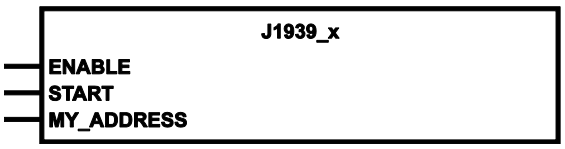
2274

x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_J1939_Vxxyyzz.LIB`

Symbol in CODESYS:



Beschreibung

2276

J1939_x dient als Protokoll-Handler für das Kommunikationsprofil SAE J1939.

Zur Abwicklung der Kommunikation muss der Protokoll-Handler in jedem Programmzyklus aufgerufen werden. Dazu wird der Eingang ENABLE auf TRUE gesetzt.

Einmal gesetzt, muss ENABLE auf TRUE bleiben!

Der Protokoll-Handler wird gestartet, wenn der Eingang START für einen Zyklus auf TRUE gesetzt wird.

Über MY_ADRESS wird dem Controller eine Geräteadresse übergeben. Sie muss sich von Adressen der anderen J1939-Busteilnehmer unterscheiden. Sie kann dann von anderen Busteilnehmern ausgelesen werden.

Parameter der Eingänge

469

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein ausführen FALSE: Baustein wird nicht ausgeführt > Baustein-Eingänge sind nicht aktiv > Baustein-Ausgänge sind nicht spezifiziert
START	BOOL	TRUE (nur 1 Zyklus lang): J1939-Protokoll an CAN-Schnittstelle x starten FALSE: im weiteren Programmablauf
MY_ADDRESS	BYTE	J1939-Adresse des Geräts

J1939_x_GLOBAL_REQUEST

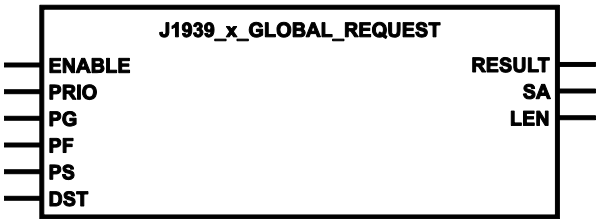
2282

x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_J1939_Vxxyyzz.LIB`

Symbol in CODESYS:



Beschreibung

2301

J1939_x_GLOBAL_REQUEST ist für das automatische Anfordern einzelner Nachrichten von allen (global) aktiven J1939-Netzwerkteilnehmern verantwortlich. Dazu werden dem FB die Parameter PG, PF, PS und die Adresse des Arrays DST übergeben, in dem die empfangenen Daten abgelegt werden.

Info

PGN = [Page] + [PF] + [PS]
PDU = [PRIO] + [PGN] + [J1939-Adresse] + [Daten]

13790

ACHTUNG

Daten können unzulässig überschrieben werden!

- ► Ein Empfangs-Array mit einer Größe von 1 785 Bytes anlegen!
  Dies ist die maximale Größe einer J1939-Nachricht.
- ► Die Anzahl empfangener Daten prüfen:
  der Wert darf nicht größer sein als das bereitgestellte Empfangs-Array!

- ► Für jede angefragte Nachricht eine eigene Instanz des FBs verwenden!
- ► Für die Zieladresse DST gilt:
  Die Adresse mit dem Operator ADR ermitteln und dem Baustein übergeben!
- ► Zusätzlich die Priorität (typisch 3, 6 oder 7) übergeben.

- ► Da das Anfordern der Daten über mehrere Steuerungszyklen abgewickelt werden kann, muss dieser Vorgang über das RESULT-Byte ausgewertet werden.
- RESULT = 2: der Baustein wartet auf Daten der Teilnehmer.
- RESULT = 1: von einem Teilnehmer wurden Daten empfangen.
  Der Ausgang LEN zeigt an, wie viele Datenbytes empfangen wurden.
  Diese neuen Daten in DST sofort speichern / auswerten!
  Der Empfang einer weiteren Nachricht überschreibt die Daten auf der Speicheradresse DST.
- RESULT = 0: innerhalb von 1,25 Sekunden hat kein Teilnehmer am Bus eine Antwort gesendet.
  Der Baustein wird wieder inaktiv.
  Erst jetzt darf ENABLE wieder auf FALSE gesetzt werden!
- ► Für das Empfangen von Daten von mehreren Teilnehmern in schneller Folge:
  den Baustein im selben SPS-Zyklus mehrmals aufrufen und direkt auswerten!

Parameter der Eingänge

463

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein ausführen FALSE: Baustein wird nicht ausgeführt > Baustein-Eingänge sind nicht aktiv > Baustein-Ausgänge sind nicht spezifiziert
PRIO	BYTE	Nachrichten-Prioritätin der PDU (Parameter Data Unit) zulässig = 0…7
PG	BYTE	Data Page Wert der definierten PGN (Parameter Group Number) zulässig = 0...1 (normalerweise = 0)
PF	BYTE	PDU format byte Wert der definierten PGN (Parameter Group Number) PDU2 (global) = 240...255
PS	BYTE	PDU specific byte Wert der definierten PGN (Parameter Group Number) GE (Group Extension) = 0...255
DST	DWORD	Startadresse im Zielspeicher Die Adresse mit dem Operator ADR ermitteln und dem Baustein übergeben!

Info

PGN = [Page] + [PF] + [PS]
PDU = [PRIO] + [PGN] + [J1939-Adresse] + [Daten]

Parameter der Ausgänge

20789

Parameter	Datentyp	Beschreibung
RESULT	BYTE	Rückmeldung des Funktionsbausteins (mögliche Meldungen → folgende Tabelle)
SA	BYTE	J1939-Adresse des antwortenden Geräts
LEN	WORD	Anzahl der empfangenen Bytes

Mögliche Ergebnisse für RESULT:

Wert dez	Wert hex	Beschreibung
0	00	FB ist inaktiv
1	01	FB-Ausführung wurde ohne Fehler beendet – Daten sind gültig
2	02	Funktionsbaustein ist aktiv (Aktion noch nicht beendet)

J1939_x_RECEIVE

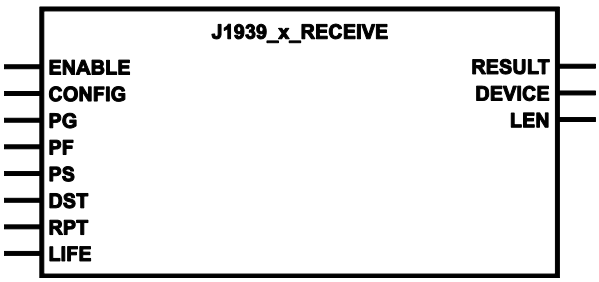
2278

x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_J1939_Vxxyyzz.LIB`

Symbol in CODESYS:



Beschreibung

2288

J1939_x_RECEIVE dient dem Empfang einer einzelnen Nachricht oder eines Nachrichtenblocks.

Dazu muss der FB über den Eingang CONFIG für einen Zyklus initialisiert werden. Bei der Initialisierung werden die Parameter PG, PF, PS, RPT, LIFE und die Speicheradresse des Datenarrays DST übergeben.

🛈 Nach dem ersten Konfigurieren können diese Parameter im laufenden Anwendungsprogramm nicht mehr verändert werden: PG, PF, PS, RPT, LIFE, DST.

13790

ACHTUNG

Daten können unzulässig überschrieben werden!

► Ein Empfangs-Array mit einer Größe von 1 785 Bytes anlegen!
Dies ist die maximale Größe einer J1939-Nachricht.

► Die Anzahl empfangener Daten prüfen:
der Wert darf nicht größer sein als das bereitgestellte Empfangs-Array!

► Für die Zieladresse DST gilt:
🛈 Die Adresse mit dem Operator ADR ermitteln und dem Baustein übergeben!

🛈 Nach dem ersten Setzen kann RPT nicht mehr verändert werden!

► Der Datenempfang muss über das RESULT-Byte ausgewertet werden. Wird RESULT = 1, können die Daten von der über DST übergebenen Speicheradresse ausgelesen und weiter verarbeitet werden.

> Der Empfang einer neuen Nachricht überschreibt die Daten auf der Speicheradresse DST.

> Die Anzahl der empfangenen Nachrichten-Bytes wird über den Ausgang LEN angezeigt.

> Wird RESULT = 3, wurden im angegebenen Zeitfenster (LIFE • RPT) keine gültigen Nachrichten empfangen.

🛈 Dieser Baustein muss auch eingesetzt werden, wenn die Nachrichten mit den FBs J1939_..._REQUEST angefordert werden.

Parameter der Eingänge

457

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein ausführen FALSE: Baustein wird nicht ausgeführt > Baustein-Eingänge sind nicht aktiv > Baustein-Ausgänge sind nicht spezifiziert
CONFIG	BOOL	TRUE (im 1. Zyklus): Datenobjekt konfigurieren FALSE: im weiteren Programmablauf
PG	BYTE	Data Page Wert der definierten PGN (Parameter Group Number) zulässig = 0...1 (normalerweise = 0)
PF	BYTE	PDU format byte Wert der definierten PGN (Parameter Group Number) PDU1 (specific) = 0...239 PDU2 (global) = 240...255
PS	BYTE	PDU specific byte Wert der definierten PGN (Parameter Group Number) Wenn PF = PDU1 ⇨ PS = DA (Destination Address) (DA = J1939-Adresse des externen Geräts) Wenn PF = PDU2 ⇨ PS = GE (Group Extension)
DST	DWORD	Startadresse im Zielspeicher Die Adresse mit dem Operator ADR ermitteln und dem Baustein übergeben!
RPT	TIME	Überwachungszeit Innerhalb dieses angegebenen Zeitfensters müssen die Telegramme zyklisch empfangen werden. > Andernfalls erfolgt eine Fehlermeldung. RPT = T#0s ⇨ keine Überwachung Nach dem ersten Setzen kann RPT nicht mehr verändert werden!
LIFE	BYTE	tolerierte Anzahl der nicht empfangenen J1939-Nachrichten

Parameter der Ausgänge

458

Parameter	Datentyp	Beschreibung
RESULT	BYTE	Rückmeldung des Funktionsbausteins (mögliche Meldungen → folgende Tabelle)
DEVICE	BYTE	J1939-Adresse des Absenders
LEN	WORD	Anzahl der empfangenen Bytes

Mögliche Ergebnisse für RESULT:

Wert dez	Wert hex	Beschreibung
0	00	FB ist inaktiv
1	01	FB-Ausführung wurde ohne Fehler beendet – Daten sind gültig
3	03	Fehler, keine Daten während der Überwachungszeit empfangen

J1939_x_RESPONSE

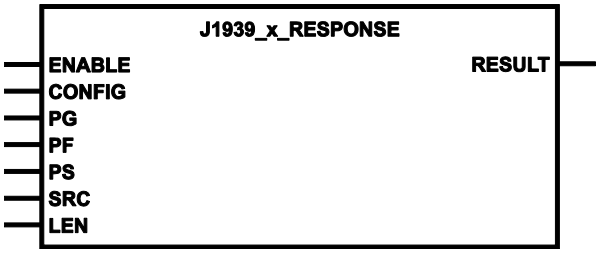
2280

x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_J1939_Vxxyyzz.LIB`

Symbol in CODESYS:



Beschreibung

2299

J1939_x_RESPONSE organisiert die automatische Antwort auf ein Request-Telegramm (Anforderungstelegramm).

Der FB ist für das automatische Versenden von Nachrichten auf "Global Requests" und "Specific Requests" verantwortlich. Dazu muss der FB über den Eingang CONFIG für einen Zyklus initialisiert werden.

Dem FB werden die Parameter PG, PF, PS, RPT und die Adresse des Datenarrays SRC übergeben.

► Für die Quelladresse SRC gilt:
  ⓘ Die Adresse mit dem Operator ADR ermitteln und dem Baustein übergeben!

► Zusätzlich die Anzahl der zu übertragenen Datenbytes übergeben.

Parameter der Eingänge

451

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein ausführen FALSE: Baustein wird nicht ausgeführt > Baustein-Eingänge sind nicht aktiv > Baustein-Ausgänge sind nicht spezifiziert
CONFIG	BOOL	TRUE (im 1. Zyklus): Datenobjekt konfigurieren FALSE: im weiteren Programmablauf
PG	BYTE	Data Page Wert der definierten PGN (Parameter Group Number) zulässig = 0...1 (normalerweise = 0)
PF	BYTE	PDU format byte Wert der definierten PGN (Parameter Group Number) PDU1 (specific) = 0...239 PDU2 (global) = 240...255
PS	BYTE	PDU specific byte Wert der definierten PGN (Parameter Group Number) Wenn PF = PDU1 ⇨ PS = DA (Destination Address) (DA = J1939-Adresse des externen Geräts) Wenn PF = PDU2 ⇨ PS = GE (Group Extension)
SRC	DWORD	Startadresse im Quellspeicher ⓘ Die Adresse mit dem Operator ADR ermitteln und dem Baustein übergeben!
LEN	WORD	Anzahl (≥ 1) der zu übertragenden Daten-Bytes

Parameter der Ausgänge

13993

Parameter	Datentyp	Beschreibung
RESULT	BYTE	Rückmeldung des Funktionsbausteins (mögliche Meldungen → folgende Tabelle)

Mögliche Ergebnisse für RESULT:

Wert dez	Wert hex	Beschreibung
0	00	FB ist inaktiv
1	01	Datenübertragung wurde ohne Fehler beendet
2	02	Funktionsbaustein ist aktiv (Aktion noch nicht beendet)
3	03	Fehler, Daten können nicht übertragen werden

J1939_x_SPECIFIC_REQUEST

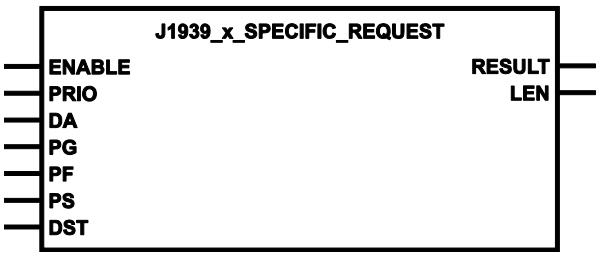
2281

x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_J1939_Vxxyyzz.LIB`

Symbol in CODESYS:



Beschreibung

2300

J1939_x_SPECIFIC_REQUEST ist für das automatische Anfordern einzelner Nachrichten von einem bestimmten (specific) J1939-Netzwerkteilnehmer verantwortlich. Dazu werden dem FB die logische Geräteadresse DA, die Parameter PG, PF, PS und die Adresse des Arrays DST übergeben, in dem die empfangenen Daten abgelegt werden.

Info

PGN = [Page] + [PF] + [PS]
PDU = [PRIO] + [PGN] + [J1939-Adresse] + [Daten]

13790

ACHTUNG

Daten können unzulässig überschrieben werden!

- ► Ein Empfangs-Array mit einer Größe von 1 785 Bytes anlegen!
  Dies ist die maximale Größe einer J1939-Nachricht.
- ► Die Anzahl empfangener Daten prüfen:
  der Wert darf nicht größer sein als das bereitgestellte Empfangs-Array!

- ► Für die Zieladresse gilt:
  Die Adresse mit dem Operator ADR ermitteln und dem Baustein übergeben!
- ► Zusätzlich die Priorität (typisch 3, 6 oder 7) übergeben.
- ► Da das Anfordern der Daten über mehrere Steuerungszyklen abgewickelt werden kann, muss dieser Vorgang über das RESULT-Byte ausgewertet werden. Wird RESULT = 1, wurden alle Daten empfangen.
- > Der Ausgang LEN zeigt an, wie viele Datenbytes empfangen wurden.
- > Wird innerhalb von 1,25 Sekunden vom angeforderten Teilnehmer keine Antwort gesendet, meldet der FB einen Fehler (⇨ RESULT = 3).

Parameter der Eingänge

445

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein ausführen FALSE: Baustein wird nicht ausgeführt > Baustein-Eingänge sind nicht aktiv > Baustein-Ausgänge sind nicht spezifiziert
PRIO	BYTE	Nachrichten-Prioritätin der PDU (Parameter Data Unit) zulässig = 0…7
DA	BYTE	J1939-Adresse des angefragten Geräts
PG	BYTE	Data Page Wert der definierten PGN (Parameter Group Number) zulässig = 0...1 (normalerweise = 0)
PF	BYTE	PDU format byte Wert der definierten PGN (Parameter Group Number) PDU1 (specific) = 0...239 PDU2 (global) = 240...255
PS	BYTE	PDU specific byte Wert der definierten PGN (Parameter Group Number) Wenn PF = PDU1 ⇨ PS = DA (Destination Address) (DA = J1939-Adresse des externen Geräts) Wenn PF = PDU2 ⇨ PS = GE (Group Extension)
DST	DWORD	Startadresse im Zielspeicher Die Adresse mit dem Operator ADR ermitteln und dem Baustein übergeben!

Info

PGN = [Page] + [PF] + [PS]
PDU = [PRIO] + [PGN] + [J1939-Adresse] + [Daten]

Parameter der Ausgänge

446

Parameter	Datentyp	Beschreibung
RESULT	BYTE	Rückmeldung des Funktionsbausteins (mögliche Meldungen → folgende Tabelle)
LEN	WORD	Anzahl der empfangenen Bytes

Mögliche Ergebnisse für RESULT:

Wert dez	Wert hex	Beschreibung
0	00	FB ist inaktiv
1	01	FB-Ausführung wurde ohne Fehler beendet – Daten sind gültig
2	02	Funktionsbaustein ist aktiv (Aktion noch nicht beendet)
3	03	Fehler

J1939_x_TRANSMIT

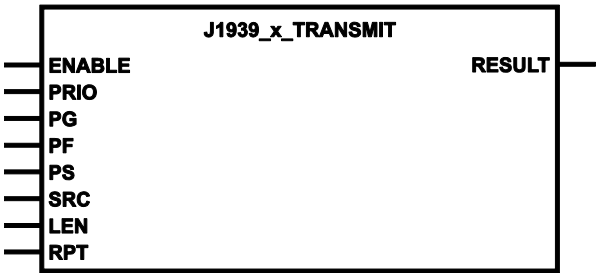
279

x = 1...n = Nummer der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_J1939_Vxxyyzz.LIB`

Symbol in CODESYS:



Beschreibung

2298

J1939_x_TRANSMIT ist für das Versenden einzelner Nachrichten oder Nachrichtenblocks verantwortlich. Dazu werden dem FB die Parameter PG, PF, PS, RPT und die Adresse des Datenarrays SRC übergeben.

Info

PGN = [Page] + [PF] + [PS]
PDU = [PRIO] + [PGN] + [J1939-Adresse] + [Daten]

- ► Für die Quelladresse SRC gilt:
  Die Adresse mit dem Operator ADR ermitteln und dem Baustein übergeben!
- ► Zusätzlich die Anzahl der zu übertragenen Datenbytes und die Priorität (typisch 3, 6 oder 7) übergeben.
- ► Da das Versenden der Daten über mehrere Steuerungszyklen abgewickelt wird, muss der Vorgang über das RESULT-Byte ausgewertet werden. Wird RESULT = 1, wurden alle Daten übertragen.

Wenn mehr als 8 Bytes gesendet werden sollen, wird ein "multi package transfer" durchgeführt.

Parameter der Eingänge

439

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein ausführen FALSE: Baustein wird nicht ausgeführt > Baustein-Eingänge sind nicht aktiv > Baustein-Ausgänge sind nicht spezifiziert
PRIO	BYTE	Nachrichten-Prioritätin der PDU (Parameter Data Unit) zulässig = 0…7
PG	BYTE	Data Page Wert der definierten PGN (Parameter Group Number) zulässig = 0...1 (normalerweise = 0)
PF	BYTE	PDU format byte Wert der definierten PGN (Parameter Group Number) PDU1 (specific) = 0...239 PDU2 (global) = 240...255
PS	BYTE	PDU specific byte Wert der definierten PGN (Parameter Group Number) Wenn PF = PDU1 ⇨ PS = DA (Destination Address) (DA = J1939-Adresse des externen Geräts) Wenn PF = PDU2 ⇨ PS = GE (Group Extension)
SRC	DWORD	Startadresse im Quellspeicher Die Adresse mit dem Operator ADR ermitteln und dem Baustein übergeben!
LEN	WORD	Anzahl der zu übertragenden Daten-Bytes zulässig = 1...1 785 = 0x0001...0x06F9
RPT	TIME	Wiederholzeit, innerhalb der die Daten-Telegramme zyklisch versendet werden sollen RPT = T#0s ⇨ nur einmalig versenden

Info

PGN = [Page] + [PF] + [PS]
PDU = [PRIO] + [PGN] + [J1939-Adresse] + [Daten]

Parameter der Ausgänge

440

Parameter	Datentyp	Beschreibung
RESULT	BYTE	Rückmeldung des Funktionsbausteins (mögliche Meldungen → folgende Tabelle)

Mögliche Ergebnisse für RESULT:

Wert dez	Wert hex	Beschreibung
0	00	FB ist inaktiv
1	01	FB-Ausführung wurde ohne Fehler beendet – Daten sind gültig
2	02	Funktionsbaustein ist aktiv (Aktion noch nicht beendet)
3	03	Fehler, Daten können nicht übertragen werden

5.2.6 Bausteine: serielle Schnittstelle

Inhalt

13011
12998

HINWEIS

Voreingestellt steht die serielle Schnittstelle dem Anwender nicht zur Verfügung, da sie für den Programm-Download und das Debugging genutzt wird.

Setzt der Anwender das Systemmerkerbit SERIAL_MODE=TRUE, dann kann die Schnittstelle frei genutzt werden. Ein Debugging des Anwendungsprogramms ist dann nur noch über eine (beliebige) CAN-Schnittstelle möglich.

Mit den folgend aufgeführten Bausteinen kann die serielle Schnittstelle im Anwendungsprogramm genutzt werden.

SERIAL_PENDING

314

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:

Beschreibung

12994

SERIAL_PENDING ermittelt die Anzahl der im seriellen Empfangspuffer gespeicherten Datenbytes.

Im Gegensatz zu **SERIAL_RX** (→ S. 119) bleibt der Inhalt des Puffers nach Aufruf dieser Funktion unverändert.

Die SERIAL-Bausteine bilden die Grundlage für die Erstellung eines anwendungsspezifischen Protokolls für die serielle Schnittstelle.
Dazu das Systemmerkerbit SERIAL_MODE=TRUE setzen!

12998

HINWEIS

Voreingestellt steht die serielle Schnittstelle dem Anwender nicht zur Verfügung, da sie für den Programm-Download und das Debugging genutzt wird.

Setzt der Anwender das Systemmerkerbit SERIAL_MODE=TRUE, dann kann die Schnittstelle frei genutzt werden. Ein Debugging des Anwendungsprogramms ist dann nur noch über eine (beliebige) CAN-Schnittstelle möglich.

Parameter der Ausgänge

12996

Parameter	Datentyp	Beschreibung
NUMBER	WORD	Anzahl der empfangenen Datenbytes (1...1 000)

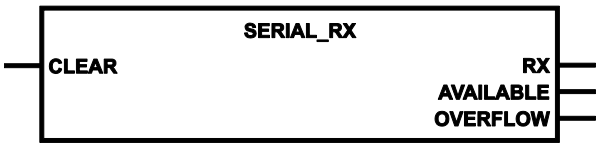
SERIAL_RX

308

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:



Beschreibung

12997

SERIAL_RX liest mit jedem Aufruf ein empfangenes Datenbyte aus dem seriellen Empfangspuffer aus.

Gehen mehr als 1 000 Datenbytes ein, läuft der Puffer über und es gehen Daten verloren. Dieses wird durch das Bit OVERFLOW angezeigt.

Wird eine 7-Bit-Datenübertragung genutzt, enthält das 8. Bit die Parität und muss gegebenenfalls vom Anwender ausgeblendet werden.

Die SERIAL-Bausteine bilden die Grundlage für die Erstellung eines anwendungsspezifischen Protokolls für die serielle Schnittstelle.
Dazu das Systemmerkerbit SERIAL_MODE=TRUE setzen!

12998

HINWEIS

Voreingestellt steht die serielle Schnittstelle dem Anwender nicht zur Verfügung, da sie für den Programm-Download und das Debugging genutzt wird.

Setzt der Anwender das Systemmerkerbit SERIAL_MODE=TRUE, dann kann die Schnittstelle frei genutzt werden. Ein Debugging des Anwendungsprogramms ist dann nur noch über eine (beliebige) CAN-Schnittstelle möglich.

Parameter der Eingänge

312

Parameter	Datentyp	Beschreibung
CLEAR	BOOL	TRUE: Empfangspuffer löschen FALSE: Funktion wird nicht ausgeführt

Parameter der Ausgänge

12931

Parameter	Datentyp	Beschreibung
RX	BYTE	empfangene Byte-Daten aus dem Empfangspuffer
AVAILABLE	WORD	Anzahl der empfangenen Bytes, die sich im Empfangspuffer befinden VOR dem Aufruf des FBs: 0 = keine Daten empfangen 1...1 000 = Anzahl von Bytes im Empfangspuffer
OVERFLOW	BOOL	TRUE: Überlauf des Datenpuffers ⇨ Datenverlust! FALSE: Datenpuffer ist ohne Datenverlust

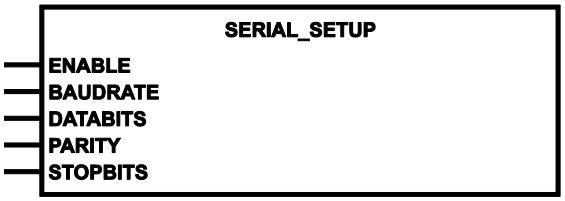
SERIAL_SETUP

302

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:



Beschreibung

13000

SERIAL_SETUP initialisiert die serielle RS232-Schnittstelle.

Der FB muss nicht zwingend ausgeführt werden, um die serielle Schnittstelle verwenden zu können. Ohne FB-Aufruf gilt der zuletzt eingestellte Wert.

Mit ENABLE=TRUE für einen Zyklus setzt der FB die serielle Schnittstelle auf die angegebenen Parameter. Die mit dem FB vorgenommenen Änderungen werden remanent gespeichert.

12998

HINWEIS

Voreingestellt steht die serielle Schnittstelle dem Anwender nicht zur Verfügung, da sie für den Programm-Download und das Debugging genutzt wird.

Setzt der Anwender das Systemmerkerbit SERIAL_MODE=TRUE, dann kann die Schnittstelle frei genutzt werden. Ein Debugging des Anwendungsprogramms ist dann nur noch über eine (beliebige) CAN-Schnittstelle möglich.

Parameter der Eingänge

13002

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE (nur 1 Zyklus lang): Schnittstelle initialisieren FALSE: im weiteren Programmablauf
BAUDRATE	DWORD	Baudrate zulässige Werte → Datenblatt Voreinstellwert → Datenblatt
DATABITS	BYTE := 8	Anzahl der Daten-Bits zulässig = 7 oder 8
PARITY	BYTE := 0	Parität zulässig: 0=keine, 1=gerade, 2=ungerade Falls DATABITS = 7 und PARITY = 0 parametriert: dann arbeitet der FB mit PARITY = 1
STOPBITS	BYTE := 1	Anzahl der Stopp-Bits zulässig = 1 oder 2

SERIAL_TX

296

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:

Beschreibung

13003

SERIAL_TX überträgt ein Datenbyte über die serielle RS232-Schnittstelle.

Der FiFo-Sendespeicher fasst 1 000 Bytes.

Mit dem Eingang ENABLE kann die Übertragung freigegeben oder gesperrt werden.

Die SERIAL-Bausteine bilden die Grundlage für die Erstellung eines anwendungsspezifischen Protokolls für die serielle Schnittstelle.
Dazu das Systemmerkerbit SERIAL_MODE=TRUE setzen!

12998

HINWEIS

Voreingestellt steht die serielle Schnittstelle dem Anwender nicht zur Verfügung, da sie für den Programm-Download und das Debugging genutzt wird.

Setzt der Anwender das Systemmerkerbit SERIAL_MODE=TRUE, dann kann die Schnittstelle frei genutzt werden. Ein Debugging des Anwendungsprogramms ist dann nur noch über eine (beliebige) CAN-Schnittstelle möglich.

Parameter der Eingänge

300

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein ausführen FALSE: Baustein wird nicht ausgeführt > Baustein-Eingänge sind nicht aktiv > Baustein-Ausgänge sind nicht spezifiziert
DATA	BYTE	zu übertragender Wert

5.2.7 Bausteine: SPS-Zyklus optimieren mit Interrupts

Inhalt

20965
8609

Hier zeigen wir Ihnen Funktionen zum Optimieren des SPS-Zyklus.

1599

Die SPS arbeitet das gespeicherte Anwendungsprogramm zyklisch in voller Länge ab. Von z.B. äußeren Ereignissen abhängige Verzweigungen im Programm (= bedingte Sprünge) lassen die Zykluszeit variieren. Für bestimmte Funktionen kann dieses Verhalten nachteilig sein.

Mit Hilfe gezielter Unterbrechungen (= Interrupts) des zyklischen Programmablaufs können zeitkritische Abläufe unabhängig vom Zyklus in festen Zeitrastern oder bei bestimmten Ereignissen aufgerufen werden.

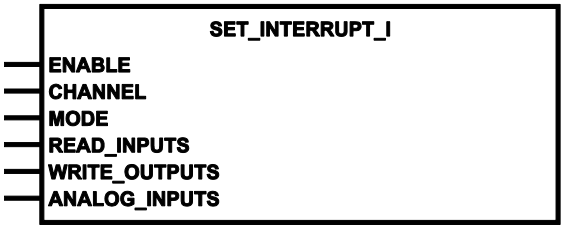
SET_INTERRUPT_I

2381

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:



Beschreibung

19361
11573

SET_INTERRUPT_I organisiert das Ausführen eines Programmteils durch eine Interrupt-Anforderung über einen Eingangskanal.

In der klassischen SPS ist die Zykluszeit das Maß der Dinge für Echtzeitbetrachtungen. Gegenüber kundenspezifischen Steuerungen ist die SPS damit im Nachteil. Auch ein "Echtzeit-Betriebssystem" ändert nichts an dieser Tatsache, wenn das gesamte Anwendungsprogramm in einem einzigen unveränderlichen Block abläuft.

Ein möglicher Lösungsansatz wäre, die Zykluszeit kurz zu halten. Dieser Weg führt oft dazu, die Anwendung auf mehrere Steuerungszyklen zu verteilen. Die Programmierung wird dadurch jedoch unübersichtlich und schwierig.

Eine andere Möglichkeit besteht darin, einen bestimmten Programmteil nur auf Anforderung durch einen Eingangsimpuls unabhängig vom Steuerungszyklus aufzurufen:

Der zeitkritische Teil des Anwendungsprogramms wird vom Anwender in einen Baustein vom Type PROGRAMM (PRG) zusammengefasst. Dieser Baustein wird zur Interrupt-Routine deklariert, indem einmalig (zur Initialisierungszeit) SET_INTERRUPT_I aufgerufen wird. Das hat zur Folge, dass dieser Programmteil immer dann ausgeführt wird, wenn eine Flanke am Eingang CHANNEL erkannt wird. Werden Ein- und Ausgänge in diesem Programmteil genutzt, werden diese ebenfalls in der Interrupt-Routine, ausgelöst durch die Eingangs-Flanke, gelesen oder beschrieben. Über die Eingänge READ_INPUTS, WRITE_OUTPUTS oder ANALOG_INPUTS kann das Lesen oder Schreiben unterbunden werden.

Innerhalb des Programmteils können also alle zeitkritischen Ereignisse bearbeitet werden, indem Eingänge oder globale Variablen verknüpft und Ausgänge beschrieben werden. So können auch Bausteine nur genau dann ausgeführt werden, wenn sie durch ein Eingangssignal angefordert werden.

HINWEIS

Damit der per Interrupt aufgerufene Programmteil nicht zusätzlich zyklisch aufgerufen wird, sollte er (mit Ausnahme des Initialisierungsaufrufes) im Zyklus übersprungen werden.

Der Eingang (CHANNEL), der zum Auslösen des Interrupt überwacht wird, kann in der Interrupt-Routine nicht initialisiert und weiter verarbeitet werden.

Die Laufzeit des Hauptzyklus plus die Summe der Laufzeiten aller per Interrupt aufgerufenen Programmteile muss stets innerhalb der max. zulässigen Zykluszeit bleiben!

Für die Datenkonsistenz zwischen Hauptprogramm und den im Interrupt laufenden Programmteilen ist der Anwender zuständig!

19866

Interrupt-Prioritäten:

- Alle per Interrupt aufgerufenen Programmteile haben die gleiche Priorität der Ausführung. Mehrere gleichzeitige Interrupts werden sequenziell in Reihenfolge ihres Auftretens abgearbeitet.
- Wird eine weitere Flanke am gleichen Eingang während der Ausführung des per Interrupt aufgerufenen Programmteils erkannt, wird dieser zur Bearbeitung eingetragen und das Programm nach Beendigung direkt wieder aufgerufen. Optional können durch Setzen des Glitch-Filters störende Mehrfachimpulse ausgefiltert werden.
- Das im Interupt laufende Programm kann durch höherpriorisierte Interrupts (z.B. CAN) unterbrochen werden.
- Belegen mehrere Interrupts den gleichen Kanal, erhält der zuletzt initialisierte FB (oder das PRG) den Kanal. Der zuvor definierte FB (oder das PRG) wird dann nicht mehr aufgerufen und liefert keine Daten mehr.

19365

HINWEIS

Die Eindeutigkeit der Ein- und Ausgänge im Zyklus wird durch die Interrupt-Routine aufgehoben. Deshalb wird nur ein Teil der Ein- und Ausgänge bedient. Wurden sie im Interrupt-Programm initialisiert, werden folgende Ein- und Ausgänge gelesen oder geschrieben:
• Eingänge: IN00...IN07
• Ausgänge: Q00...Q07

Auch globale Variablen verlieren ihre Eindeutigkeit, wenn auf sie quasi gleichzeitig im Zyklus und durch die Interrupt-Routine zugegriffen wird. Insbesondere größere Datentypen (z.B. DINT) sind von dieser Problematik betroffen.

Alle anderen Ein- und Ausgänge werden, wie üblich, einmalig im Zyklus bearbeitet.

Parameter der Eingänge

2383

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE (nur 1 Zyklus lang): Initialisierung des Bausteins FALSE: Baustein wird nicht ausgeführt
CHANNEL	BYTE	Nummer des Interrupt-Eingangs 0...7 für die Eingänge I00...I07
MODE	BYTE	Art der Flanke am Eingang CHANNEL, die den Interrupt auslöst 1 = steigende Flanke (Standard-Wert) 2 = fallende Flanke 3 = steigende und fallende Flanke > 3 = Standard-Wert
READ_INPUTS	BOOL	TRUE: die Eingänge 0..7 vor Aufruf des Programms lesen und in die Eingangsmerker I00...I07 schreiben FALSE: nur den unter CHANNEL angegebenen Kanal lesen und in den dazugehörigen Eingangsmerker Ixx schreiben
WRITE_OUTPUTS	BOOL	TRUE: die aktuellen Werte der Ausgangsmerker Q00...Q07 nach Programmablauf auf die Ausgänge schreiben FALSE: keine Ausgänge schreiben
ANALOG_INPUTS	BOOL	TRUE: die Eingänge 0..7 lesen und die ungefilterten, unkalibrierten Analogwerte in die Merker ANALOG_IRQ00...07 schreiben FALSE: die Merker ANALOG_IRQ00...07 nicht schreiben

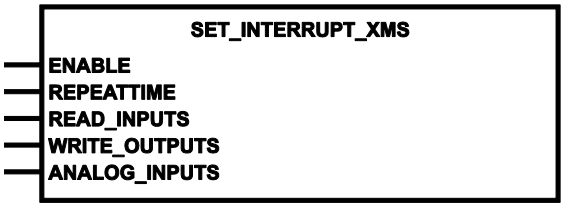
SET_INTERRUPT_XMS

272

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:



Beschreibung

275

SET_INTERRUPT_XMS organisiert das Ausführen eines Programmteils im Intervall von x ms.

In der klassischen SPS ist die Zykluszeit das Maß der Dinge für Echtzeitbetrachtungen. Gegenüber kundenspezifischen Steuerungen ist die SPS damit im Nachteil. Auch ein "Echtzeit-Betriebssystem" ändert nichts an dieser Tatsache, wenn das gesamte Anwendungsprogramm in einem einzigen unveränderlichen Block abläuft.

Ein möglicher Lösungsansatz wäre, die Zykluszeit kurz zu halten. Dieser Weg führt oft dazu, die Anwendung auf mehrere Steuerungszyklen zu verteilen. Die Programmierung wird dadurch jedoch unübersichtlich und schwierig.

Eine andere Möglichkeit besteht darin, einen bestimmten Programmteil in festen Zeitabständen (alle x ms) unabhängig vom Steuerungszyklus aufzurufen.

Der zeitkritische Teil des Anwendungsprogramms wird vom Anwender in einen Baustein vom Type PROGRAMM (PRG) zusammengefasst. Dieser Baustein wird zur Interrupt-Routine deklariert, indem einmalig (zur Initialisierungszeit) SET_INTERRUPT_XMS aufgerufen wird. Das hat zur Folge, dass dieser Programmteil immer nach Ablauf der REPEATTIME (alle x ms) abgearbeitet wird. Werden Ein- und Ausgänge in diesem Programmteil genutzt, werden diese ebenfalls im festgelegten Takt gelesen oder beschrieben. Über die Eingänge READ_INPUTS, WRITE_OUTPUTS oder ANALOG_INPUTS kann das Lesen oder Schreiben unterbunden werden.

Innerhalb des Programmteils können also alle zeitkritischen Ereignisse bearbeitet werden, indem Eingänge oder globale Variablen verknüpft und Ausgänge beschrieben werden. So können auch Zeitglieder genauer überwacht werden, als es in einem "normalen" Zyklus möglich ist.

HINWEIS

Damit der per Interrupt aufgerufene Programmteil nicht zusätzlich zyklisch aufgerufen wird, sollte er (mit Ausnahme des Initialisierungsaufrufes) im Zyklus übersprungen werden.

Es können mehrere Timer-Interrupt-Bausteine aktiv sein. Der Zeitbedarf der Interrupt-Funktionen muss so berechnet werden, dass alle aufgerufenen Bausteine ausgeführt werden können. Das gilt besonders bei Berechnungen, Gleitkomma-Arithmetik und Regler-Funktionen.

Für die Datenkonsistenz zwischen Hauptprogramm und den im Interrupt laufenden Programmteilen ist der Anwender zuständig!

Bitte beachten: Bei einer hohen CAN-Busaktivität kann die eingestellte REPEATTIME schwanken.

971

HINWEIS

Die Eindeutigkeit der Ein- und Ausgänge im Zyklus wird durch die Interrupt-Routine aufgehoben. Deshalb wird nur ein Teil der Ein- und Ausgänge bedient. Wurden sie im Interrupt-Programm initialisiert, werden folgende Ein- und Ausgänge gelesen oder geschrieben.

Eingänge, digital:

%IX0.0...%IX0.7 (Controller: CR0n3n, CR7n3n)
%IX0.12...%IX0.15, %IX1.4...%IX1.8 (übrige ClassicController, ExtendedController, SafetyController)
%IX0.0, %IX0.8 (SmartController: CR250n)
IN08...IN11 (CabinetController: CR030n)
IN0...IN3 (Platinensteuerung: CS0015)

Eingänge, analog:

%IX0.0...%IX0.7 (Controller: CR0n3n, CR7n3n)
alle Kanäle (Auswahl bitcodiert) (alle übrigen Controller)

Ausgänge, digital:

%QX0.0...%QX0.7 (ClassicController, ExtendedController, SafetyController)
%QX0.0, %QX0.8 (SmartController: CR250n)
OUT00...OUT03 CabinetController: CR030n()
OUT0...OUT7 (Platinensteuerung: CS0015)

Auch globale Variablen verlieren ihre Eindeutigkeit, wenn auf sie quasi gleichzeitig im Zyklus und durch die Interrupt-Routine zugegriffen wird. Insbesondere größere Datentypen (z.B. DINT) sind von dieser Problematik betroffen.

Alle anderen Ein- und Ausgänge werden, wie üblich, einmalig im Zyklus bearbeitet.

Parameter der Eingänge

2382

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE (nur 1 Zyklus lang): Initialisierung des Bausteins FALSE: Baustein wird nicht ausgeführt
REPEATTIME	TIME	Zeitdauer in [ms] zwischen Ende des Programms und Neustart Die Zeitdauer zwischen zwei Aufrufen ermittelt sich damit als Summe aus REPEATTIME und Laufzeit des per Interrupt aufgerufenen Programms.
READ_INPUTS	BOOL	TRUE: die Eingänge 0..7 vor Aufruf des Programms lesen und in die Eingangsmerker I00...I07 schreiben FALSE: keine Aktualisierung der Eingänge
WRITE_OUTPUTS	BOOL	TRUE: die aktuellen Werte der Ausgangsmerker Q00...Q07 nach Programmablauf auf die Ausgänge schreiben FALSE: keine Ausgänge schreiben
ANALOG_INPUTS	BOOL	TRUE: die Eingänge 0..7 lesen und die ungefilterten, unkalibrierten Analogwerte in die Merker ANALOG_IRQ00...07 schreiben FALSE: die Merker ANALOG_IRQ00...07 nicht schreiben

5.2.8 Bausteine: Eingangswerte verarbeiten

Inhalt

1602
1302

Hier zeigen wir Ihnen **ifm**-Funktionsbausteine zum Lesen und Verarbeiten der analogen oder binären Signale am Geräte-Eingang.

> **HINWEIS**
>
> Die in der Steuerungskonfiguration von CODESYS erscheinenden analogen Rohwerte kommen direkt aus dem ADC. Sie sind noch nicht korrigiert!
>
> Deshalb können in der Steuerungskonfiguration bei gleichen Geräten unterschiedliche Rohwerte erscheinen.
>
> Erst durch die **ifm**-FBs findet eine Fehlerkorrektur und Normierung statt. Die FBs liefern den korrigierten Wert.

INPUT_ANALOG

2245

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Für die Extended-Seite des ExtendedControllers endet der FB-Name mit "_E".

Symbol in CODESYS:

Beschreibung

2361
12916

INPUT_ANALOG ermöglicht die nachfolgend aufgeführten Betriebsarten an den Eingangskanälen. Details → Kapitel **Mögliche Betriebsarten Ein-/Ausgänge** (→ S. 234)

Der FB liefert den aktuellen Analogwert am gewählten Analogkanal. Die Analogwerte werden normiert ausgegeben. Gleichzeitig werden die unkalibrierten Rohwerte über die Systemmerker ANALOGxx ausgegeben.

► Für Frequenz- und Periodenmessungen sowie Zählerfunktionen: MODE=1 (= IN_DIGITAL_H) einstellen!

Die Messung und der Ausgangswert resultieren aus der über MODE angegebenen Betriebsart:

12917

MODE dez	MODE hex	Eingang Betriebsart		FB-Ausgang OUT	Einheit
0	00	deaktiviert		---	---
1	01	Binäreingang, minus-schaltend (BH)	IN_DIGITAL_H	0 / 1	---
2	02	Binäreingang, plus-schaltend (BL)	IN_DIGITAL_L	0 / 1	---
4	04	Stromeingang	IN_CURRENT	0…20 000	µA
8	08	Spannungseingang	IN_VOLTAGE_10	0…10 000	mV
16	10	Spannungseingang	IN_VOLTAGE_30	0…32 000	mV
32	20	Spannungseingang, ratiometrisch	IN_RATIO	0…1 000	‰
64	40	Diagnose	IN_DIAGNOSIS	---	---
128	80	schneller Eingang	IN_FAST	0 / 1	---

18414

Falls Eingang I15 nicht verwendet:
► Eingang I15 als Binäreingang konfigurieren!

Parameter der Eingänge

2362

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein ausführen FALSE: Baustein wird nicht ausgeführt > Baustein-Eingänge sind nicht aktiv > Baustein-Ausgänge sind nicht spezifiziert
MODE	BYTE	Betriebsart des Eingangskanals: 0 = 0x00 IN_NOMODE (Aus; Voreinstellung aktiv) 1 = 0x01 IN_DIGITAL_H voreingestellt 2 = 0x02 IN_DIGITAL_L 4 = 0x04 IN_CURRENT 0…20 000 µA 8 = 0x08 IN_VOLTAGE10 0…10 000 mV 16 = 0x10 IN_VOLTAGE30 0…32 000 mV 32 = 0x20 IN_RATIO 0...1 000 ‰ 64 = 0x40 IN_DIAGNOSTIC 128 = 0x80 IN_FAST □
CHANNEL	BYTE	Nummer des Eingangskanals 0...15 für die Eingänge I00...I15 Für den FB xxx_**E** (falls vorhanden) gilt: 0...15 für die Eingänge I00_E...I15_E

Parameter der Ausgänge

2363

Parameter	Datentyp	Beschreibung	
OUT	WORD	Ausgangswert entsprechend MODE bei ungültiger Einstellung: OUT = "0"	
ERROR	BYTE	00 = okay	
		01 = Überstrom	bei IN_CURRENT
		02 = Schluss gegen VBB	bei IN_DIGITAL_H, OUT_DIAGNOSTIC
		03 = Leiterbruch	bei IN_DIGITAL_H, OUT_DIAGNOSTIC

5.2.9 Bausteine: analoge Werte anpassen

Inhalt

1603

Wenn die Werte analoger Eingänge oder die Ergebnisse von analogen Funktionen angepasst werden müssen, helfen Ihnen die folgenden Funktionsbausteine.

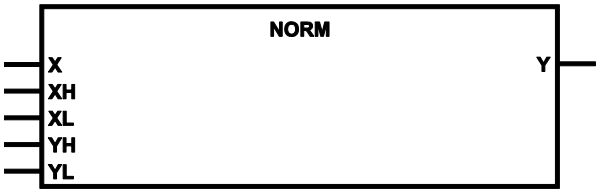
NORM

401

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:



Beschreibung

404

NORM normiert einen Wert innerhalb festgelegter Grenzen auf einen Wert mit neuen Grenzen.

Der FB normiert einen Wert vom Typ WORD, der innerhalb der Grenzen XH und XL liegt, auf einen Ausgangswert innerhalb der Grenzen YH und YL. Der FB wird z.B. bei der Erzeugung von PWM-Werten aus analogen Eingangsgrößen genutzt.

HINWEIS

- ► Der Eingangswert für X muss sich im definierten Bereich zwischen XL und XH befinden! Der FB prüft NICHT den Wert X auf Plausibilität.
- > Bedingt durch die Rundungsfehler können Abweichungen beim normierten Wert um 1 auftreten.
- > Werden die Grenzen (XH/XL oder YH/YL) invertiert angegeben, erfolgt auch die Normierung invertiert.

Parameter der Eingänge

405

Parameter	Datentyp	Beschreibung
X	WORD	Eingangswert
XH	WORD	obere Grenze des Eingangswertebereichs [Inkremente]
XL	WORD	untere Grenze des Eingangswertebereichs [Inkremente]
YH	WORD	obere Grenze des Ausgangswertebereichs
YL	WORD	untere Grenze des Ausgangswertebereichs

Parameter der Ausgänge

406

Parameter	Datentyp	Beschreibung
Y	WORD	Ausgangswert

Beispiel: NORM (1)

407

unterer Grenzwert Eingang	0	XL
oberer Grenzwert Eingang	100	XH
unterer Grenzwert Ausgang	0	YL
oberer Grenzwert Ausgang	2000	YH

dann wandelt der Funktionsbaustein das Eingangssignal z.B. wie folgt um:

von X =	50	0	100	75
	↓	↓	↓	↓
nach Y =	1000	0	2000	1500

Beispiel: NORM (2)

408

unterer Grenzwert Eingang	2000	XL
oberer Grenzwert Eingang	0	XH
unterer Grenzwert Ausgang	0	YL
oberer Grenzwert Ausgang	100	YH

dann wandelt der Funktionsbaustein das Eingangssignal z.B. wie folgt um:

von X =	1000	0	2000	1500
	↓	↓	↓	↓
nach Y =	50	100	0	25

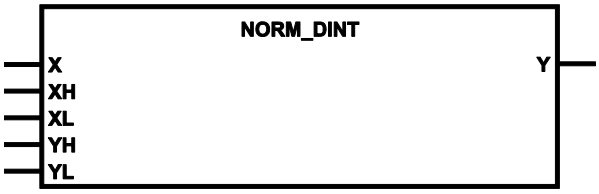
NORM_DINT

2217

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:



Beschreibung

2355

NORM_DINT normiert einen Wert innerhalb festgelegter Grenzen auf einen Wert mit neuen Grenzen.

Der FB normiert einen Wert vom Typ DINT, der innerhalb der Grenzen XH und XL liegt, auf einen Ausgangswert innerhalb der Grenzen YH und YL. Dieser FB wird z.B. bei der Erzeugung von PWM-Werten aus analogen Eingangsgrößen genutzt.

HINWEIS

- ► Der Eingangswert für X muss sich im definierten Bereich zwischen XL und XH befinden! Der FB prüft NICHT den Wert X auf Plausibilität.
- ► Das Ergebnis der Berechnung (XH-XL)•(YH-YL) muss im Wertebereich des Datentyps DINT (-2 147 483 648...2 147 483 647) bleiben!
- > Bedingt durch die Rundungsfehler können Abweichungen beim normierten Wert um 1 auftreten.
- > Werden die Grenzen (XH/XL oder YH/YL) invertiert angegeben, erfolgt auch die Normierung invertiert.

Parameter der Eingänge

2359

Parameter	Datentyp	Beschreibung
X	DINT	Eingangswert
XH	DINT	obere Grenze des Eingangswertebereichs
XL	DINT	untere Grenze des Eingangswertebereichs
YH	DINT	obere Grenze des Ausgangswertebereichs
YL	DINT	untere Grenze des Ausgangswertebereichs

Parameter der Ausgänge

2360

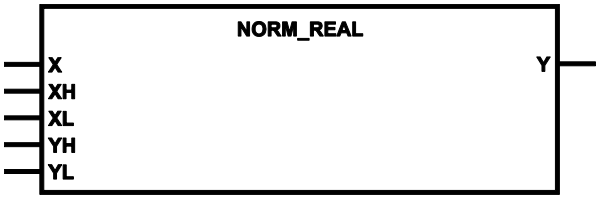
Parameter	Datentyp	Beschreibung
Y	DINT	Ausgangswert

NORM_REAL

2218

Baustein-Typ = Funktionsbaustein (FB)
Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:



Beschreibung

2358

NORM_REAL normiert einen Wert innerhalb festgelegter Grenzen auf einen Wert mit neuen Grenzen.
Der FB normiert einen Wert vom Typ REAL, der innerhalb der Grenzen XH und XL liegt, auf einen Ausgangswert innerhalb der Grenzen YH und YL. Dieser FB wird z.B. bei der Erzeugung von PWM-Werten aus analogen Eingangsgrößen genutzt.

HINWEIS

- ► Der Eingangswert für X muss sich im definierten Bereich zwischen XL und XH befinden! Der FB prüft NICHT den Wert X auf Plausibilität.
- ► Das Ergebnis der Berechnung (XH-XL)•(YH-YL) muss im Wertebereich des Datentyps REAL (-3,402823466•10^{38}...3,402823466•10^{38}) bleiben!
- > Bedingt durch die Rundungsfehler können Abweichungen beim normierten Wert um 1 auftreten.
- > Werden die Grenzen (XH/XL oder YH/YL) invertiert angegeben, erfolgt auch die Normierung invertiert.

Parameter der Eingänge

2356

Parameter	Datentyp	Beschreibung
X	REAL	Eingangswert
XH	REAL	obere Grenze des Eingangswertebereichs
XL	REAL	untere Grenze des Eingangswertebereichs
YH	REAL	obere Grenze des Ausgangswertebereichs
YL	REAL	untere Grenze des Ausgangswertebereichs

Parameter der Ausgänge

2357

Parameter	Datentyp	Beschreibung
Y	REAL	Ausgangswert

5.2.10 Bausteine: Zählerfunktionen zur Frequenz- und Periodendauermessung

Inhalt

2322

Je nach Controller werden bis zu 16*) schnelle Eingänge unterstützt, die Eingangsfrequenzen bis zu 30 kHz verarbeiten können. Neben der reinen Frequenzmessung können die Eingänge auch zur Auswertung von inkrementellen Drehgebern (Zählerfunktion) eingesetzt werden.
*) ExtendedController: bis zu 32 schnelle Eingänge

Bedingt durch die unterschiedlichen Messmethoden können Fehler bei der Frequenzermittlung auftreten.

Zur einfachen Auswertung stehen folgende Bausteine zur Verfügung:

Baustein	zulässige Werte	Erklärung
FREQUENCY	0,1...30 000 Hz	Frequenz am angegebenen Kanal messen. Messfehler verringert sich bei hohen Frequenzen
PERIOD	0,1...5 000 Hz	Frequenz und Periodendauer (Zykluszeit) am angegebenen Kanal messen
PERIOD_RATIO	0,1...5 000 Hz	Frequenz und Periodendauer (Zykluszeit) sowie Puls-Pause-Verhältnis [‰] am angegebenen Kanal messen
FREQUENCY_PERIOD	0,1...30 000 Hz	Die Funktion vereinigt die beiden Funktionen FREQUENCY und PERIOD oder PERIOD_RATIO. Automatisches Umschalten der Messmethode bei 5 kHz
PHASE	0,1...5 000 Hz	Liest ein Kanalpaar ein und vergleicht die Phasenlage der Signale
INC_ENCODER	0,1...30 000 Hz	Vorwärts-/Rückwärts-Zählerfunktion zur Auswertung von Drehgebern
FAST_COUNT	0,1...30 000 Hz	Schnelle Impulse zählen

Wichtig bei Einsatz der schnellen Eingänge als "normale" Digitaleingänge:
- ► Die erhöhte Empfindlichkeit gegen Störimpulse beachten (z.B. Kontaktprellen bei mechanischen Kontakten).
- ► Das Eingangssignal bei Bedarf entprellen! → Kapitel **Hardware-Filter konfigurieren** (→ S. 61)
- • Der Standard-Digitaleingang kann Signale bis 50 Hz auswerten.

FAST_COUNT

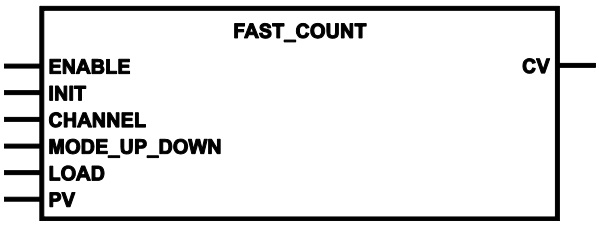
567

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Für die Extended-Seite des ExtendedControllers endet der FB-Name mit "_E".

Symbol in CODESYS:



Beschreibung

6830

FAST_COUNT arbeitet als Zählerbaustein für schnelle Eingangsimpulse.

Während ENABLE=TRUE erfasst der FB steigende Flanken an den FRQ-Eingangskanälen.
Maximale Eingangsfrequenz → Datenblatt.

Nach Rücksetzen und erneutem Setzen von ENABLE zählt der Zähler von dem Wert an weiter, der beim letzten Rücksetzen von ENABLE gültig war.

Mit Setzen von INIT (steigende Flanke) wird der Zählerwert CV=0 gesetzt.
Nach Rücksetzen des Parameters INIT zählt der Zähler von 0 an.

Am selben Eingang diesen FB **nicht** gemeinsam mit einem der folgenden FBs nutzen!
- **FREQUENCY** (→ S. 138)
- **FREQUENCY_PERIOD** (→ S. 140)
- **INC_ENCODER** (→ S. 142)
- **PERIOD** (→ S. 144)
- **PERIOD_RATIO** (→ S. 146)
- **PHASE** (→ S. 148)

14888

HINWEIS

Bei höheren Frequenzen (als den von **ifm** garantierten) können folgende Probleme auftreten:

- Die Ein- und Ausschaltzeiten der Eingänge werden zunehmend relevant.
- Die Bauteile können sich unzulässig erwärmen.

Die genannten Einflüsse sind abhängig von den im Einzelfall eingesetzten Bauteilen.
Diese möglichen Einflüsse sind nicht exakt vorhersagbar.

Parameter der Eingänge

571

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein ausführen FALSE: Baustein wird nicht ausgeführt > Zähler angehalten
INIT	BOOL	FALSE ⇨ TRUE (Flanke): Baustein wird initialisiert FALSE: im weiteren Programmablauf
CHANNEL	BYTE	Nummer des schnellen Eingangskanals 0...15 für die Eingänge I00...I15 Für den FB xxx_**E** (falls vorhanden) gilt: 0...15 für die Eingänge I00_E...I15_E
MODE_UP_DOWN	BOOL	TRUE: Zähler zählt abwärts FALSE: Zähler zählt aufwärts
LOAD	BOOL	TRUE: Startwert PV wird in CV geladen FALSE: Funktion wird nicht ausgeführt
PV	DWORD	Startwert (Preset value) für den Zähler

Parameter der Ausgänge

572

Parameter	Datentyp	Beschreibung
CV	DWORD	aktueller Zählerwert Verhalten beim Überlauf: • zählt der Zähler abwärts, bleibt er bei 0 stehen • zählt der Zähler aufwärts, gibt es einen Überlauf.

FREQUENCY

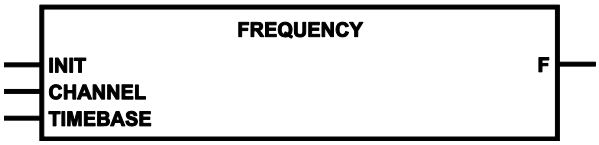
537

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Für die Extended-Seite des ExtendedControllers endet der FB-Name mit "_E".

Symbol in CODESYS:



Beschreibung

2325
20675

FREQUENCY misst die Frequenz des am gewählten Kanal (CHANNEL) ankommenden Signals. Der FB wertet dazu die positive Flanke des Signals aus.

In Abhängigkeit von der Zeitbasis (TIMEBASE) können Frequenzmessungen in einem weiten Wertebereich durchgeführt werden.
- hohe Frequenzen erfordern eine kurze Zeitbasis
- niedrige Frequenzen erfordern eine entsprechend längere Zeitbasis

Grenzwerte:

TIMEBASE	zulässige, messbare Frequenz
57 000 ms (= Maximalwert)	1 149 Hz
2 184 ms	30 000 Hz (= Maximalwert)

Je länger die Timebase für eine zu messende Frequenz ist, desto genauer wird der ermittelte Messwert.
Beispiel für Frequenz = 1 Hz:

TIMEBASE [ms]	max. Fehler [%]	Messung [Hz]
1 000	100	0...2
10 000	10	0,9...1,1

Die Frequenz wird direkt in [Hz] ausgegeben.

14888

HINWEIS

Bei höheren Frequenzen (als den von ifm garantierten) können folgende Probleme auftreten:
- Die Ein- und Ausschaltzeiten der Eingänge werden zunehmend relevant.
- Die Bauteile können sich unzulässig erwärmen.

Die genannten Einflüsse sind abhängig von den im Einzelfall eingesetzten Bauteilen.
Diese möglichen Einflüsse sind nicht exakt vorhersagbar.

7321

Bei der Frequenzmessung sicherstellen, dass der FB innerhalb des Wertes von TIMEBASE nicht mehr als 65 535 positive Flanken empfängt!
Sonst kann das interne Zählregister überlaufen und zu falschen Ergebnissen führen.

Am selben Eingang diesen FB **nicht** gemeinsam mit einem der folgenden FBs nutzen!
- **FAST_COUNT** (→ S. 136)
- **FREQUENCY_PERIOD** (→ S. 140)
- **INC_ENCODER** (→ S. 142)
- **PERIOD** (→ S. 144)
- **PERIOD_RATIO** (→ S. 146)
- **PHASE** (→ S. 148)

Parameter der Eingänge

2599

Parameter	Datentyp	Beschreibung
INIT	BOOL	TRUE (nur 1 Zyklus lang): Baustein und Schnittstelle werden initialisiert FALSE: Messung läuft oder: Messung startet, wenn zuvor INIT=TRUE war
CHANNEL	BYTE	Nummer des schnellen Eingangskanals 0...15 für die Eingänge I00...I15 Für den FB xxx_**E** (falls vorhanden) gilt: 0...15 für die Eingänge I00_E...I15_E
TIMEBASE	TIME	Zeitbasis zur Frequenzmessung (max. 57 s)

Parameter der Ausgänge

542

Parameter	Datentyp	Beschreibung
F	REAL	Frequenz des Eingangssignals in [Hz]

FREQUENCY_PERIOD

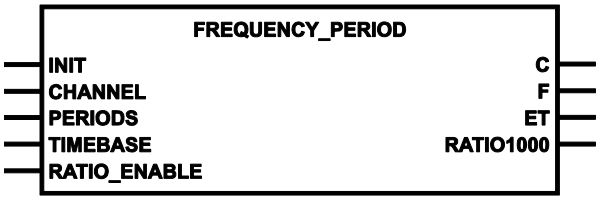
2206

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Für die Extended-Seite des ExtendedControllers endet der FB-Name mit "_E".

Symbol in CODESYS:



Beschreibung

2335
20676

FREQUENCY_PERIOD misst die Frequenz und die Periodendauer (Zykluszeit) in [µs] am angegebenen Kanal (für alle Eingänge zugelassen). Maximale Eingangsfrequenz → Datenblatt.

Der FB vereinigt PERIOD oder PERIOD_RATIO und FREQUENCY in einem gemeinsamen Funktionsbaustein. Die Umschaltung der Messmethode erfolgt automatisch bei etwa 5 kHz:
- unterhalb von 5,2 kHz verhält sich der FB wie PERIOD oder PERIOD_RATIO
- oberhalb von 5,5 kHz verhält sich der FB wie FREQUENCY.

Der FB misst die Frequenz und die Zykluszeit des am gewählten Kanal (CHANNEL) anstehenden Signals. Zur Berechnung werden alle positiven Flanken ausgewertet und der Mittelwert über die Anzahl der angegebenen Perioden (PERIODS) gebildet.

Bei einer Eingangsfrequenz > 5 kHz und aktivem FREQUENCY-Modus kann der Ratio nicht gemessen werden.

Der maximale Messbereich beträgt ca. 15 min.

14888

HINWEIS

Bei höheren Frequenzen (als den von ifm garantierten) können folgende Probleme auftreten:
- Die Ein- und Ausschaltzeiten der Eingänge werden zunehmend relevant.
- Die Bauteile können sich unzulässig erwärmen.

Die genannten Einflüsse sind abhängig von den im Einzelfall eingesetzten Bauteilen.
Diese möglichen Einflüsse sind nicht exakt vorhersagbar.

7321

Bei der Frequenzmessung sicherstellen, dass der FB innerhalb des Wertes von TIMEBASE nicht mehr als 65 535 positive Flanken empfängt!
Sonst kann das interne Zählregister überlaufen und zu falschen Ergebnissen führen.

HINWEIS

Am selben Eingang diesen FB **nicht** gemeinsam mit einem der folgenden FBs nutzen!
- **FAST_COUNT** (→ S. 136)
- **FREQUENCY** (→ S. 138)
- **INC_ENCODER** (→ S. 142)
- **PERIOD** (→ S. 144)
- **PERIOD_RATIO** (→ S. 146)
- **PHASE** (→ S. 148)

Parameter der Eingänge

2336

Parameter	Datentyp	Beschreibung
INIT	BOOL	TRUE (nur 1 Zyklus lang): Baustein und Schnittstelle werden initialisiert FALSE: Messung läuft oder: Messung startet, wenn zuvor INIT=TRUE war
CHANNEL	BYTE	Nummer des schnellen Eingangskanals 0...15 für die Eingänge I00...I15 Für den FB xxx_**E** (falls vorhanden) gilt: 0...15 für die Eingänge I00_E...I15_E
PERIODS	BYTE	Anzahl der Perioden, über die gemittelt wird (1...16) 0 : Ausgänge C und F werden nicht aktualisiert > 16 : wird auf 16 limitiert
TIMEBASE	TIME	Zeitbasis zur Frequenzmessung (max. 57 s)
RATIO_ENABLE	BOOL	TRUE: Ratio-Messung an RATIO1000 ausgeben FALSE: Ratio-Messung nicht ausgeben

Parameter der Ausgänge

2337

Parameter	Datentyp	Beschreibung
C	DWORD	Zykluszeit der erfassten Perioden in [µs] zulässig = 33...10 000 000 = 0x21...0x989680
F	REAL	Frequenz des Eingangssignals in [Hz]
ET	TIME	bei Messung der Periodendauer: (nutzbar bei sehr langsamen Signalen) RATIO_ENABLE = TRUE: Verstrichene Zeit seit dem letzten Flankenwechsel am Eingang RATIO_ENABLE = FALSE: Verstrichene Zeit seit der letzten positiven Flanke am Eingang bei anderen Messungen: ET = 0
RATIO1000	WORD	Puls-/Periode-Verhältnis in [‰] zulässig = 1...999 = 0x1...0x3E7 Voraussetzungen: • Periodendauermessung • Impulsdauer ≥ 100 µs • Frequenz < 5 kHz

INC_ENCODER

525

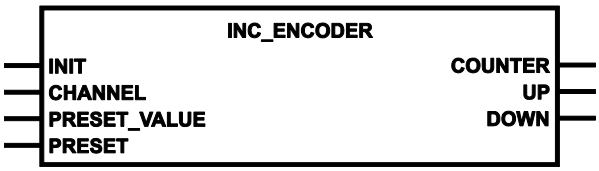
= Incremental Encoder

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Für die Extended-Seite des ExtendedControllers endet der FB-Name mit "_E".

Symbol in CODESYS:



Beschreibung

2602

INC_ENCODER bietet eine Vorwärts-/Rückwärts-Zählerfunktion zur Auswertung von Drehgebern.

Immer zwei Frequenzeingänge bilden das Eingangspaar, das über den FB ausgewertet wird.

Grenzfrequenz = 30 kHz
max. anschließbar: 4 Drehgeber (ExtendedController: max. 8 Drehgeber)

Voreinstellwert setzen:
1. Wert in PRESET_VALUE eintragen
2. PRESET für einen Zyklus auf TRUE setzen
3. PRESET wieder auf FALSE setzen

Der FB zählt die Impulse an den Eingängen, solange INIT=FALSE und PRESET=FALSE sind.
Am Ausgang COUNTER steht der aktuelle Zählerstand an.

Die Ausgänge UP und DOWN zeigen die aktuelle Zählrichtung des Zählers an. Die Ausgänge sind dann TRUE, wenn im vorangegangenen Programmzyklus der Zähler in die entsprechende Richtung gezählt hat. Bleibt der Zähler stehen, wird auch der Richtungsausgang im folgenden Programmzyklus zurückgesetzt.

Am selben Eingang diesen FB **nicht** gemeinsam mit einem der folgenden FBs nutzen!
- **FAST_COUNT** (→ S. 136)
- **FREQUENCY** (→ S. 138)
- **FREQUENCY_PERIOD** (→ S. 140)
- **PERIOD** (→ S. 144)
- **PERIOD_RATIO** (→ S. 146)
- **PHASE** (→ S. 148)

Parameter der Eingänge

529

Parameter	Datentyp	Beschreibung
INIT	BOOL	TRUE (nur 1 Zyklus lang): Baustein wird initialisiert FALSE: im weiteren Programmablauf
CHANNEL	BYTE	Nummer des Eingangskanal-Paares 0 = Kanalpaar 0 = Eingänge I00 + I01 ... 3 = Kanalpaar 3 = Eingänge I06 + I07 Für den FB xxx_**E** (falls vorhanden) gilt: 0 = Kanalpaar 0 = Eingänge I00_E + I01_E ... 3 = Kanalpaar 3 = Eingänge I06_E + I07_E
PRESET_VALUE	DINT	Zähler-Startwert
PRESET	BOOL	FALSE ⇨ TRUE (Flanke): PRESET_VALUE wird nach COUNTER geladen TRUE: Zähler ignoriert die Eingangsimpulse FALSE: Zähler zählt die Eingangsimpulse

Parameter der Ausgänge

530

Parameter	Datentyp	Beschreibung
COUNTER	DINT	aktueller Zählerstand
UP	BOOL	TRUE: Zähler zählte im letzten Zyklus aufwärts FALSE: Zähler zählte im letzten Zyklus nicht aufwärts
DOWN	BOOL	TRUE: Zähler zählte im letzten Zyklus abwärts FALSE: Zähler zählte im letzten Zyklus nicht abwärts

PERIOD

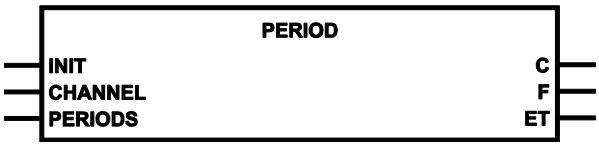
370

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Für die Extended-Seite des ExtendedControllers endet der FB-Name mit "_E".

Symbol in CODESYS:



Beschreibung

2330
20677

PERIOD misst die Frequenz und die Periodendauer (Zykluszeit) in [µs] am angegebenen Kanal (für alle Eingänge zugelassen). Maximale Eingangsfrequenz → Datenblatt.

Der FB misst die Frequenz und die Zykluszeit des am gewählten Kanal (CHANNEL) anstehenden Signals. Zur Berechnung werden alle positiven Flanken ausgewertet und der Mittelwert über die Anzahl der angegebenen Perioden (PERIODS) gebildet.

Bei niedrigen Frequenzen kommt es mit **FREQUENCY** (→ S. 138) zu Ungenauigkeiten. Um dieses zu umgehen, kann PERIOD genutzt werden. Die Zykluszeit wird direkt in [µs] ausgegeben.

Der maximale Messbereich beträgt 10 Sekunden.

14888

HINWEIS

Bei höheren Frequenzen (als den von **ifm** garantierten) können folgende Probleme auftreten:

- Die Ein- und Ausschaltzeiten der Eingänge werden zunehmend relevant.
- Die Bauteile können sich unzulässig erwärmen.

Die genannten Einflüsse sind abhängig von den im Einzelfall eingesetzten Bauteilen.
Diese möglichen Einflüsse sind nicht exakt vorhersagbar.

Am selben Eingang diesen FB **nicht** gemeinsam mit einem der folgenden FBs nutzen!

- **FAST_COUNT** (→ S. 136)
- **FREQUENCY** (→ S. 138)
- **FREQUENCY_PERIOD** (→ S. 140)
- **INC_ENCODER**
- **PERIOD_RATIO** (→ S. 146)
- **PHASE** (→ S. 148)

Parameter der Eingänge

2600

Parameter	Datentyp	Beschreibung
INIT	BOOL	FALSE ⇨ TRUE (Flanke): Baustein wird initialisiert FALSE: im weiteren Programmablauf
CHANNEL	BYTE	Nummer des schnellen Eingangskanals 0...15 für die Eingänge I00...I15 Für den FB xxx_**E** (falls vorhanden) gilt: 0...15 für die Eingänge I00_E...I15_E
PERIODS	BYTE	Anzahl der Perioden, über die gemittelt wird (1...16) 0 : Ausgänge C und F werden nicht aktualisiert > 16 : wird auf 16 limitiert

Parameter der Ausgänge

375

Parameter	Datentyp	Beschreibung
C	DWORD	Zykluszeit der erfassten Perioden in [µs] zulässig = 200...10 000 000 = 0xC8...0x989680 (= 10 Sekunden)
F	REAL	Frequenz des Eingangssignals in [Hz]
ET	TIME	Verstrichene Zeit seit der letzten positiven Flanke am Eingang (nutzbar bei sehr langsamen Signalen)

PERIOD_RATIO

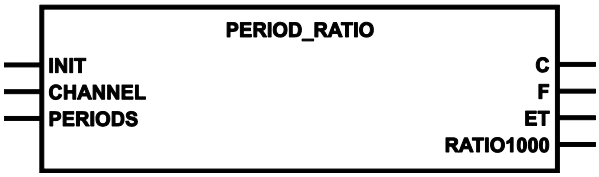
364

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Für die Extended-Seite des ExtendedControllers endet der FB-Name mit "_E".

Symbol in CODESYS:



Beschreibung

2332
20678

PERIOD_RATIO misst die Frequenz und die Periodendauer (Zykluszeit) in [µs] über die angegebenen Perioden am angegebenen Kanal (für alle Eingänge zugelassen). Zusätzlich wird das Puls-/Periode-Verhältnis in [‰] angegeben. Maximale Eingangsfrequenz → Datenblatt.

Dieser FB misst die Frequenz und die Zykluszeit des am gewählten Kanal (CHANNEL) anstehenden Signals. Zur Berechnung werden alle positiven Flanken ausgewertet und der Mittelwert über die Anzahl der angegebenen Perioden (PERIODS) gebildet. Zusätzlich wird das Puls-/Periode-Verhältnis in [‰] angegeben.
Zum Beispiel: Bei einem Signalverhältnis von 25 ms High-Pegel und 75 ms Low-Pegel, wird der Wert RATIO1000 von 250 ‰ ausgegeben.

Bei niedrigen Frequenzen kommt es mit **FREQUENCY** (→ S. 138) zu Ungenauigkeiten. Um dieses zu umgehen, kann PERIOD_RATIO genutzt werden. Die Zykluszeit wird direkt in [µs] ausgegeben.

Der maximale Messbereich beträgt 10 Sekunden.

14888

HINWEIS

Bei höheren Frequenzen (als den von **ifm** garantierten) können folgende Probleme auftreten:

- Die Ein- und Ausschaltzeiten der Eingänge werden zunehmend relevant.
- Die Bauteile können sich unzulässig erwärmen.

Die genannten Einflüsse sind abhängig von den im Einzelfall eingesetzten Bauteilen.
Diese möglichen Einflüsse sind nicht exakt vorhersagbar.

Am selben Eingang diesen FB **nicht** gemeinsam mit einem der folgenden FBs nutzen!

- **FAST_COUNT** (→ S. 136)
- **FREQUENCY** (→ S. 138)
- **FREQUENCY_PERIOD** (→ S. 140)
- **INC_ENCODER** (→ S. 142)
- **PERIOD** (→ S. 144)
- **PHASE** (→ S. 148)

Parameter der Eingänge

2601

Parameter	Datentyp	Beschreibung
INIT	BOOL	FALSE ⇨ TRUE (Flanke): Baustein wird initialisiert FALSE: im weiteren Programmablauf
CHANNEL	BYTE	Nummer des schnellen Eingangskanals 0...15 für die Eingänge I00...I15 Für den FB xxx_**E** (falls vorhanden) gilt: 0...15 für die Eingänge I00_E...I15_E
PERIODS	BYTE	Anzahl der Perioden, über die gemittelt wird (1...16) 0 : Ausgänge C und F werden nicht aktualisiert > 16 : wird auf 16 limitiert

Parameter der Ausgänge

369

Parameter	Datentyp	Beschreibung
C	DWORD	Zykluszeit der erfassten Perioden in [µs] zulässig = 200...10 000 000 = 0xC8...0x989680 (= 10 Sekunden)
F	REAL	Frequenz des Eingangssignals in [Hz]
ET	TIME	Verstrichene Zeit seit dem letzten Zustandswechsel am Eingang (nutzbar bei sehr langsamen Signalen)
RATIO1000	WORD	Puls-/Periode-Verhältnis in [‰] zulässig = 1...999 = 0x1...0x3E7 Voraussetzungen: • Periodendauermessung • Impulsdauer ≥ 100 µs • Frequenz < 5 kHz

PHASE

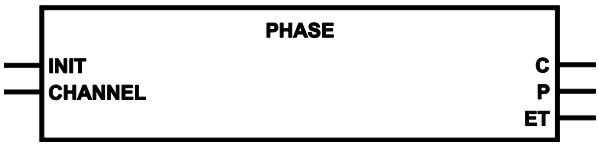
358

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Für die Extended-Seite des ExtendedControllers endet der FB-Name mit "_E".

Symbol in CODESYS:



Beschreibung

2338
20679

PHASE liest ein Kanalpaar mit schnellen Eingängen ein und vergleicht die Phasenlage der Signale. Maximale Eingangsfrequenz → Datenblatt.

Dieser FB fasst jeweils ein Kanalpaar mit schnellen Eingängen zusammen, so dass die Phasenlage zweier Signale zueinander ausgewertet werden kann. Es kann eine Periodendauer bis in den Sekundenbereich ausgewertet werden (max. 10 Sekunden).

14888

HINWEIS

Bei höheren Frequenzen (als den von **ifm** garantierten) können folgende Probleme auftreten:

- Die Ein- und Ausschaltzeiten der Eingänge werden zunehmend relevant.
- Die Bauteile können sich unzulässig erwärmen.

Die genannten Einflüsse sind abhängig von den im Einzelfall eingesetzten Bauteilen.
Diese möglichen Einflüsse sind nicht exakt vorhersagbar.

Am selben Eingang diesen FB **nicht** gemeinsam mit einem der folgenden FBs nutzen!

- **FAST_COUNT** (→ S. 136)
- **FREQUENCY** (→ S. 138)
- **FREQUENCY_PERIOD** (→ S. 140)
- **INC_ENCODER** (→ S. 142)
- **PERIOD** (→ S. 144)
- **PERIOD_RATIO** (→ S. 146)

Parameter der Eingänge

2339

Parameter	Datentyp	Beschreibung
INIT	BOOL	TRUE (nur 1 Zyklus lang): Baustein und Schnittstelle werden initialisiert FALSE: im weiteren Programmablauf
CHANNEL	BYTE	Nummer des Eingangskanal-Paares 0 = Kanalpaar 0 = Eingänge I00 + I01 ... 7 = Kanalpaar 7 = Eingänge I14 + I15 Für den FB xxx_**E** (falls vorhanden) gilt: 0 = Kanalpaar 0 = Eingänge I00_E + I01_E ... 7 = Kanalpaar 7 = Eingänge I14_E + I15_E

Parameter der Ausgänge

363

Parameter	Datentyp	Beschreibung
C	DWORD	Periodendauer des Signals am ersten Eingang des Kanalpaares in [µs]
P	INT	Winkel der Phasenverschiebung gültige Messung = 1...358 °
ET	TIME	Verstrichene Zeit seit der letzten positiven Flanke am zweiten Impulseingang des Kanalpaares

5.2.11 Bausteine: PWM-Funktionen

Inhalt

13758

Hier finden Sie ifm-Bausteine, um die Ausgänge mit Pulsweitenmodulation (PWM) betreiben zu können.

OUTPUT_BRIDGE

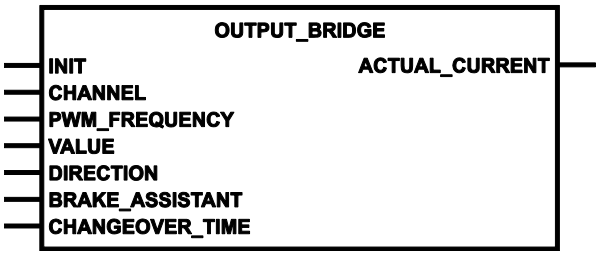
2198

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Für die Extended-Seite des ExtendedControllers endet der FB-Name mit "_E". (nicht bei CR0133)

Symbol in CODESYS:



Beschreibung

2203

OUTPUT_BRIDGE organisiert das Ansteuern der H-Brücken an den PWM-Kanälen.

Der FB dient zur einfachen Verwendung der Ausgänge als H-Brücke. Dazu werden jeweils zwei aufeinander folgende Ausgangskanäle mit minus-schaltendem Treiber zu einer Brücke zusammengefasst. Ist DIRECTION = FALSE, wird beim ersten Ausgang der plus-schaltende Treiber über ein PWM-Signal angesteuert und der minus-schaltende Treiber des zweiten Ausgangs ist durchgeschaltet.

HINWEIS

Bei Einsatz der H-Brücke wird die Stromregelung nicht unterstützt.

Ausgänge, die im PWM-Modus betrieben werden, unterstützen keine Diagnosefunktionen und es werden keine ERROR-Merker gesetzt.

Die Funktion OUT_OVERLOAD_PROTECTION ist in diesem Modus nicht aktiv!
Das Bit im Mode-Byte wird durch OUTPUT_BRIDGE zurückgesetzt.

Bei VALUE = 0 wird der Ausgang nicht komplett deaktiviert. Prinzipbedingt wird der Ausgang für die Dauer eines Timer-Ticks des PWM-Timers aktiv sein (typisch ca. 50 µs).

► FB in jedem SPS-Zyklus aufrufen!

Lage der als H-Brücke verwendbaren Ausgangskanäle → Datenblatt.

15672

HINWEIS

Soll im laufenden Betrieb am FB OUTPUT_BRIDGE der Messbereich für ACTUAL_CURRENT (auf 4 A) umgeschaltet werden?

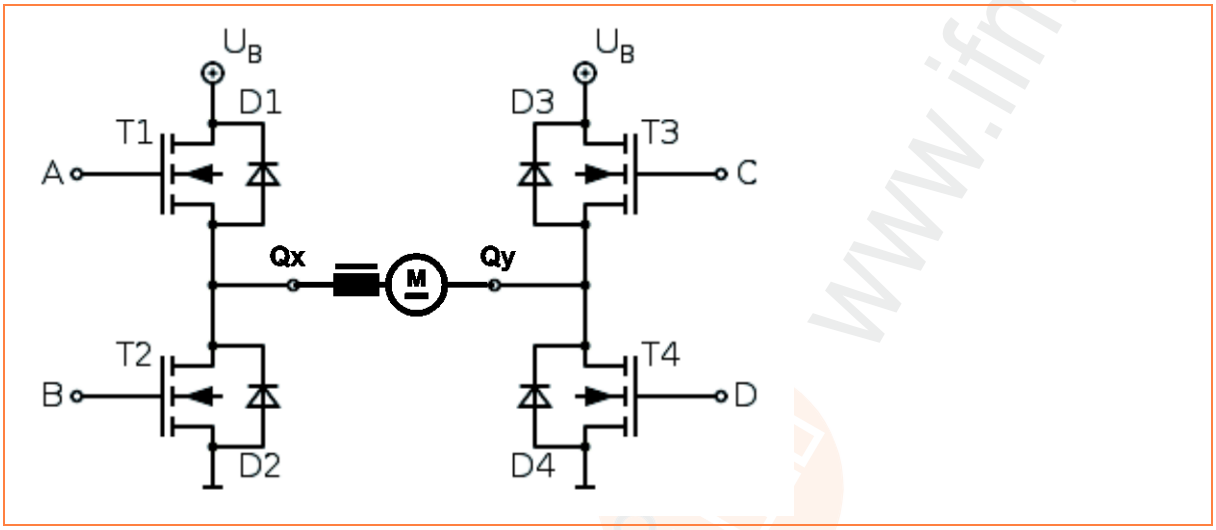
► Für beide betreffenden Ausgänge den FB SET_OUTPUT_MODE in der Init-Phase **vor** dem FB OUTPUT_BRIDGE aufrufen!
CURRENT_RANGE = 2 (für 4 A)

Prinzip der H-Brücke

9990
16411

Hier sehen Sie, wie eine H-Brücke am **ifm**-Controller via PWM-Ausgängen betrieben werden kann.

Prinzipschaltung einer H-Brücke mit PWM-Ansteuerung:



T1 und T2 bilden zusammen z.B. den Ausgang Qx.
Genauso bilden T3 und T4 z.B. den Ausgang Qy.
Dadurch werden nur zwei Anschlüsse für den DC-Motor benötigt.

Programm-Beispiel:

```
VAR
    Init1: BOOL:=TRUE;
    CycleTime:DWORD;
    MaxCycleTime:DWORD;
    ResetMax:BOOL;

    DownloadID: CAN1_DOWNLOADID;

(*****************************************************************************
CHANNEL = 1:    Motor between OUT01 (Pin17) and OUT03(Pin15)
CHANNEL = 2:    Motor between OUT09 (Pin03) and OUT11(Pin05)
*****************************************************************************)
    H_BRIDGE: OUTPUT_BRIDGE;
    PWM_value: WORD := 100;         (* current PWM value - VALUE = 0...1000 *)
    H_direction: BOOL;              (* TRUE = counter clockwise; FALSE = clockwise *)
    H_current: WORD;                (* output current in mA *)
    changeover_time: WORD := 500;   (* Space time [ms] during which the motor is not triggered
                                       (> 10 ms) in the case of a change of the rotational direction. *)
END_VAR
```

```
                    H_BRIDGE
                  OUTPUT_BRIDGE
      Init1-INIT              ACTUAL_CURRENT-H_current
          1-CHANNEL
        250-PWM_FREQUENCY
  PWM_value-VALUE
H_direction-DIRECTION
      FALSE-BRAKE_ASSISTANT
       2000-CHANGEOVER_TIME
```

Parameter der Eingänge

2204

Parameter	Datentyp	Beschreibung
INIT	BOOL	TRUE (nur 1 Zyklus lang): Baustein wird initialisiert FALSE: im weiteren Programmablauf
CHANNEL	BYTE	Name des Ausgangspaares: 1 = Brücke 1 an Q01 + Q03 2 = Brücke 2 an Q09 + Q11 Für den FB xxx_**E** (falls vorhanden) gilt: 1 = Brücke 1 an Q01_E + Q03_E 2 = Brücke 2 an Q09_E + Q11_E
PWM_FREQUENCY	WORD	PWM-Frequenz [Hz] für die Last am Ausgang > FB begrenzt den Wert auf 20...2 000 = 0x0014...0x07D0 Änderung der PWM-Frequenzen im laufenden Betrieb: nur zulässig im Bereich 40...2 000 Hz.
VALUE	WORD	PWM-Wert (Puls-Periode-Verhältnis) in [‰] zulässig = 0...1 000 = 0x0000...0x03E8 Werte > 1 000 gelten als = 1 000
DIRECTION	BOOL	Drehrichtung des Motors: TRUE: entgegen Uhrzeigersinn (ccw): Brücke 1: Stromfluss Q01(_E) ⇦ Q03(_E) Brücke 2: Stromfluss Q09(_E) ⇦ Q11(_E) FALSE: im Uhrzeigersinn (cw): Brücke 1: Stromfluss Q01(_E) ⇨ Q03(_E) Brücke 2: Stromfluss Q09(_E) ⇨ Q11(_E)
BRAKE_ASSISTANT	BOOL	TRUE: Beim Wechsel der Drehrichtung: FB schaltet beide Ausgänge gegen Masse, zwecks Bremswirkung am Motor, solange CHANGEOVER_TIME läuft. FALSE: Funktion wird nicht ausgeführt
CHANGEOVER_TIME	WORD	Pausezeit in [ms], während der bei einem Wechsel der Drehrichtung der Motor nicht angesteuert wird (≥ Zykluszeit, mindestens 10 ms) Werte < 10 ms gelten als = 10 ms

Parameter der Ausgänge

2205

Parameter	Datentyp	Beschreibung
ACTUAL_CURRENT	WORD	Ausgangsstrom in [mA]

OUTPUT_CURRENT

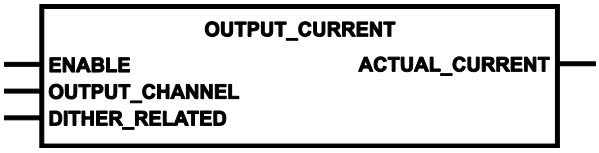
382

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Für die Extended-Seite des ExtendedControllers endet der FB-Name mit "_E".. (nicht bei CR0133)

Symbol in CODESYS:



Beschreibung

385

OUTPUT_CURRENT dient dem Messen des Stroms (optional: Mittelung über Dither-Periode) an einem Ausgangskanal.

Der FB liefert den aktuellen Ausgangsstrom, wenn die Ausgänge als PWM-Ausgänge oder als plus-schaltend benutzt werden. Die Strommessung erfolgt innerhalb des Gerätes, es werden also keine externen Messwiderstände benötigt.

Parameter der Eingänge

17894

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein ausführen FALSE: Baustein wird nicht ausgeführt > Baustein-Eingänge sind nicht aktiv > Baustein-Ausgänge sind nicht spezifiziert
OUTPUT_CHANNEL	BYTE	Nummer des stromgeregelten Ausgangskanals (0...15) 0...15 für die Ausgänge Q00...Q15 Für den FB xxx_**E** (falls vorhanden) gilt: 0...15 für die Ausgänge Q00_E...Q15_E
DITHER_RELATED	BOOL	Strom wird ermittelt als Mittelwert über... TRUE: eine Dither-Periode FALSE: eine PWM-Periode

Parameter der Ausgänge

387

Parameter	Datentyp	Beschreibung
ACTUAL_CURRENT	WORD	Ausgangsstrom in [mA]

OUTPUT_CURRENT_CONTROL

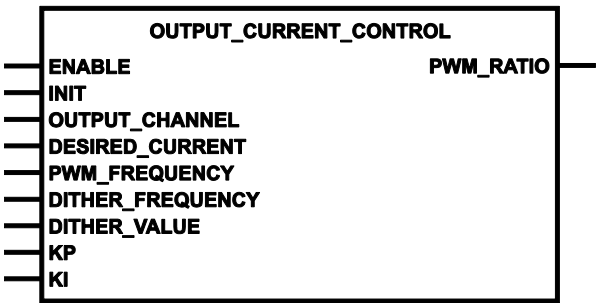
2196

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Für die Extended-Seite des ExtendedControllers endet der FB-Name mit "_E". (nicht bei CR0133)

Symbol in CODESYS:



Beschreibung

2200

OUTPUT_CURRENT_CONTROL arbeitet als Stromregler für die PWM-Ausgänge.

Die beiden Anstellparameter KI und KP repräsentieren den I- und den P-Anteil des Reglers. Zur Ermittlung der besten Einstellung des Reglers bietet sich als Startwert an, KI = 50 und KP = 50 zu setzen. Je nach gewünschtem Reglerverhalten können die Werte schrittweise vergrößert (Regler wird härter / schneller) oder verkleinert (Regler wird schwächer / langsamer) werden.

Bei Sollwert DESIRED_CURRENT=0 wird der Ausgang innerhalb von etwa 100 ms auf 0 mA heruntergeregelt , wobei die Anstellparameter ignoriert werden.

Werden Parameter während des Betriebs geändert, dann kann Folgendes geschehen:
- die Regelung kann eventuell ganz aussetzen oder
- die Regelung kann eine längere Zeit benötigen, um den Sollwert wieder einzuregeln.

▶ Daher den gemessenen Strom prüfen und gegebenenfalls die Regelung neu starten.

HINWEIS

► Bei der Definition des Parameters DITHER_VALUE darauf achten, dass das resultierende PWM-Ratio im Arbeitsbereich der Regelung zwischen 0...1000 ‰ bleibt:
  • PWM-Ratio + DITHER_VALUE < 1000 ‰ und
  • PWM-Ratio - DITHER_VALUE > 0 ‰.
  Außerhalb dieses zulässigen Bereichs wird DITHER_VALUE intern vorübergehend auf den maximal möglichen Wert reduziert, so dass der Mittelwert des PWM-Ratio dem geforderten Wert entspricht.

> Bei aktiviertem Dither werden Änderungen an PWM_FREQUENCY, DITHER_VALUE und DITHER_FREQUENCY erst nach Ende der aktuellen Dither-Periode angewendet.

► Änderungen an den Parametern im Betrieb nur bei INIT=FALSE durchführen! Die neuen Parameter werden frühestens nach Ablauf der aktuellen PWM-Periode übernommen.

► Änderung der PWM-Frequenzen im laufenden Betrieb:
  nur zulässig im Bereich 40...2 000 Hz.

> Kann der im Parameter DESIRED_CURRENT angegebene Strom nicht erreicht werden, weil das PWM-Ratioverhältnis schon bei 100 % ist, wird das durch die Systemvariable ERROR_CONTROL_Qx angezeigt.

> Bei KI = 0 findet keine Regelung statt.

> Ergibt sich bei der Regelung ein PWM_RATIO = 0, wird der Ausgang nicht komplett deaktiviert. Prinzipbedingt wird der Ausgang für die Dauer eines Timer-Ticks des PWM-Timers aktiv sein (typisch ca. 50 µs).

► Die Initialisierung des FBs (INIT=TRUE) darf nur einmalig für einen SPS Zyklus erfolgen.

► Der Aufruf dieses FB mit einem als B(L) konfigurierten Ausgang ist nicht zulässig.

► Ein als PWM-Ausgang definierter Ausgang kann anschließend nicht mehr als Binärausgang verwendet werden.

> Übersteigt der fließende Strom im eingeschalteten Zustand den Messbereich, kann keine Regelung mehr erfolgen, da der AD-Wandler am Messbereichsende ist und daher falsche Werte (den max. Wert) liefert.

Parameter der Eingänge

2201

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein ausführen FALSE: Baustein wird nicht ausgeführt > Regelung läuft weiter mit den zuletzt gültigen Parametern
INIT	BOOL	TRUE (nur 1 Zyklus lang): Baustein wird initialisiert FALSE: im weiteren Programmablauf
OUTPUT_CHANNEL	BYTE	Nummer des stromgeregelten Ausgangskanals (0...15) 0...15 für die Ausgänge Q00...Q15 Für den FB xxx_**E** (falls vorhanden) gilt: 0...15 für die Ausgänge Q00_E...Q15_E
DESIRED_CURRENT	WORD	Stromsollwert des Ausgangs in [mA] zulässig = 0...2 000 / 0...4 000 (abhängig vom Ausgang und der Konfiguration)
PWM_FREQUENCY	WORD	PWM-Frequenz [Hz] für die Last am Ausgang > FB begrenzt den Wert auf 20...2 000 = 0x0014...0x07D0 Änderung der PWM-Frequenzen im laufenden Betrieb: nur zulässig im Bereich 40...2 000 Hz.
DITHER_FREQUENCY	WORD	Dither-Frequenz in [Hz] Wertebereich = 0...FREQUENCY / 2 FREQUENCY / DITHER_FREQUENCY muss geradzahlig sein! Alle anderen Werte erhöht der FB auf den nächst passenden Wert.
DITHER_VALUE	WORD	Spitze-Spitze-Wert des Dithers in [‰] zulässig = 0...1 000 = 0x0000...0x03E8
KP	BYTE	Proportional-Anteil des Ausgangsignals
KI	BYTE	Integral-Anteil des Ausgangsignals bei KI = 0 keine Regelung

Parameter der Ausgänge

2202

Parameter	Datentyp	Beschreibung
PWM_RATIO	WORD	Zu Kontrollzwecken: Anzeige PWM-Tastverhältnis 0...999 ‰

PWM1000

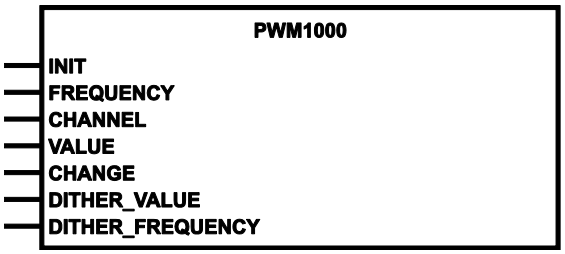
326

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Für die Extended-Seite des ExtendedControllers endet der FB-Name mit "_E". (nicht bei CR0133)

Symbol in CODESYS:



Beschreibung

2311

PWM1000 initialisiert und parametriert einen PWM-fähigen Ausgang.

Der FB ermöglicht eine einfache Anwendung der PWM-Funktion im Gerät. Für jeden Kanal kann jeweils eine eigene PWM-Frequenz, das Puls-Periode-Verhältnis und der Dither eingestellt werden.

Die PWM-Frequenz FREQUENCY kann direkt in [Hz] und das Puls-Periode-Verhältnis VALUE in 1 ‰-Schritten angegeben werden.

Bei VALUE = 0 wird der Ausgang nicht komplett deaktiviert. Prinzipbedingt wird der Ausgang für die Dauer eines Timer-Ticks des PWM-Timers aktiv sein (typisch ca. 50 µs).

- ► Bei der Definition des Parameters DITHER_VALUE darauf achten, dass das resultierende PWM-Ratio im Arbeitsbereich der Regelung zwischen 0...1000 ‰ bleibt:
  - PWM-Ratio + DITHER_VALUE < 1000 ‰ und
  - PWM-Ratio - DITHER_VALUE > 0 ‰.

  Außerhalb dieses zulässigen Bereichs wird DITHER_VALUE intern vorübergehend auf den maximal möglichen Wert reduziert, so dass der Mittelwert des PWM-Ratio dem geforderten Wert entspricht.
- ► Den FB dauerhaft aufrufen!
- ► Der Aufruf dieses FB mit einem als B(L) konfigurierten Ausgang ist nicht zulässig.

HINWEIS

Die Funktionsänderung eines als PWM-Funktion definierten Kanals im laufenden Betrieb ist nicht möglich. Die PWM-Funktion bleibt solange gesetzt, bis an der Steuerung ein Hardware-Reset durchgeführt wurde ⇨ Versorgungsspannung ausschalten und wieder einschalten.

Bei hohen PWM-Frequenzen kann es systembedingt zu Differenzen kommen zwischen eingestelltem und ausgegebenem Ratio-Verhältnis.

- ► Änderungen an den Parametern im Betrieb nur bei INIT=FALSE durchführen! Die neuen Parameter werden frühestens nach Ablauf der aktuellen PWM-Periode übernommen.
- ► Änderung der PWM-Frequenzen im laufenden Betrieb:
  nur zulässig im Bereich 40...2 000 Hz.

Änderungen während der Laufzeit:

Immer, wenn Eingang CHANGE auf TRUE gesetzt ist, übernimmt der FB den Wert ...
- FREQUENCY nach der aktuellen PWM-Periode
- VALUE nach der aktuellen PWM-Periode
- DITHER_VALUE nach der aktuellen Dither-Periode
- DITHER_FREQUENCY nach der aktuellen Dither-Periode

Parameter der Eingänge

2312

Parameter	Datentyp	Beschreibung
INIT	BOOL	TRUE (nur 1 Zyklus lang): Baustein wird initialisiert Übernahme neuer Wert von FREQUENCY FALSE: im weiteren Programmablauf
FREQUENCY	WORD	PWM-Frequenz in [Hz] > FB begrenzt den Wert auf 20...2 000 = 0x0014...0x07D0 Änderung der PWM-Frequenzen im laufenden Betrieb: nur zulässig im Bereich 40...2 000 Hz.
CHANNEL	BYTE	Nummer des PWM-Ausgangskanals 0...15 für die Ausgänge Q00...Q15 Für den FB xxx_**E** (falls vorhanden) gilt: 0...15 für die Ausgänge Q00_E...Q15_E
VALUE	WORD	PWM-Wert (Puls-Periode-Verhältnis) in [‰] zulässig = 0...1 000 = 0x0000...0x03E8 Werte > 1 000 gelten als = 1 000
CHANGE	BOOL	TRUE: Übernahme neuer Wert von ... • FREQUENCY: nach der aktuellen PWM-Periode • VALUE: nach der aktuellen PWM-Periode • DITHER_VALUE: nach der aktuellen Dither-Periode • DITHER_FREQUENCY: nach der aktuellen Dither-Periode FALSE: geänderter PWM-Wert hat keinen Einfluss auf den Ausgang
DITHER_VALUE	WORD	Spitze-Spitze-Wert des Dithers in [‰] zulässig = 0...1 000 = 0x0000...0x03E8
DITHER_FREQUENCY	WORD	Dither-Frequenz in [Hz] Wertebereich = 0...FREQUENCY / 2 FREQUENCY / DITHER_FREQUENCY muss geradzahlig sein! Alle anderen Werte erhöht der FB auf den nächst passenden Wert.

5.2.12 Bausteine: Hydraulikregelung

Inhalt

13760

Die Bibliothek `ifm_HYDRAULIC_32bit_Vxxyyzz.Lib` enthält folgende Bausteine:

CONTROL_OCC (→ S. 161)	OCC = **O**utput **C**urrent **C**ontrol (= stromgeregelter Ausgang) skaliert den Eingangswert [WORD] auf einen angegebenen Strombereich
JOYSTICK_0 (→ S. 163)	skaliert Signale [INT] aus einem Joystick auf fest definierte Kennlinien, normiert auf 0…1000
JOYSTICK_1 (→ S. 166)	skaliert Signale [INT] aus einem Joystick auf parametrierbare Kennlinien, normiert auf 0…1000
JOYSTICK_2 (→ S. 170)	skaliert Signale [INT] aus einem Joystick auf einen parametrierbaren Kennlinien-Verlauf; die Normierung ist frei bestimmbar
NORM_HYDRAULIC (→ S. 173)	normiert einen Wert [DINT] innerhalb festgelegter Grenzen auf einen Wert mit neuen Grenzen

Aus der Bibliothek `UTIL.Lib` (im CODESYS-Paket) werden folgende Bausteine benötigt:
- RAMP_INT
- CHARCURVE

Diese Bausteine werden von den FBs der Hydraulik-Bibliothek automatisch aufgerufen und parametriert.

Aus der Bibliothek `ifm_CR0232_Vxxyyzz.LIB` werden folgende Bausteine benötigt:

OUTPUT_CURRENT (→ S. 154)	misst den Strom (Mittelung über Dither-Periode) an einem Ausgangskanal
OUTPUT_CURRENT_CONTROL (→ S. 155)	Stromregler für einen PWMi-Ausgangskanal

Diese Bausteine werden von den FBs der Hydraulik-Bibliothek automatisch aufgerufen und parametriert.

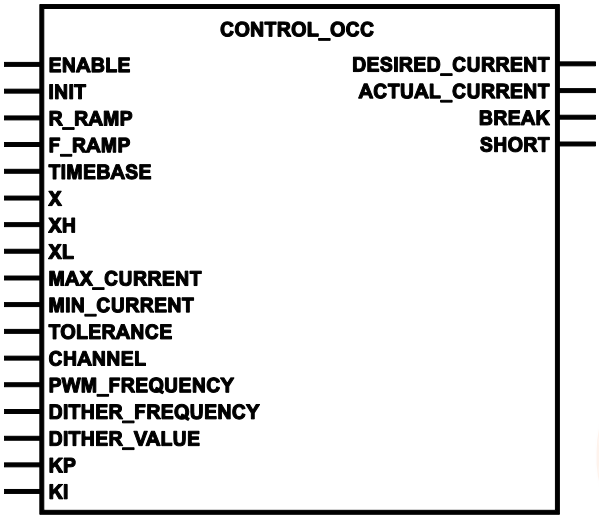
CONTROL_OCC

2735

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_HYDRAULIC_32bit_Vxxyyzz.Lib`

Symbol in CODESYS:



Beschreibung

2737

CONTROL_OCC skaliert den Eingangswert X auf einen angegebenen Strombereich.

Jede Instanz des FBs wird in jedem SPS-Zyklus einmalig aufgerufen.
Dieser FB nutzt aus der Bibliothek `ifm_CR0232_Vxxyyzz.LIB` folgende FBs:

- **OUTPUT_CURRENT_CONTROL** (→ S. 155)
- **OUTPUT_CURRENT** (→ S. 154)

Der Regler regelt in Abhängigkeit der Periodendauer des PWM Signals.

Die beiden Anstellparameter KI und KP repräsentieren den Integral- und den Proportionalanteil des Reglers. Zur Ermittlung der besten Einstellung des Reglers bietet sich als Startwert an, KI=50 und KP=50 zu setzen.

► Werte für KI und/oder KP vergrößern: ⇨ Regler wird schärfer / schneller
Werte für KI und/oder KP verkleinern: ⇨ Regler wird schwächer / langsamer

> Bei Ausgang DESIRED_CURRENT=0 wird der Ausgang **sofort** auf 0 mA geschaltet, wobei **nicht** entsprechend der eingestellten Parameter auf 0 mA heruntergeregelt wird.

Der Regler verfügt über einen schnellen Ausgleichsmechanismus bei Spannungseinbrüchen der Versorgungsspannung. In Abhängigkeit der Größe des Spannungseinbruchs wird zusätzlich zum Regelverhalten des Reglers die Ratio des PWMs dementsprechend so vergrößert, dass der Regler so schnell wie möglich den Sollwert erreicht.

ⓘ Der Eingang X von CONTROL_OCC sollte von einem Ausgang der JOYSTICK-Bausteine gespeist werden.

Parameter der Eingänge

2739

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein ausführen FALSE: Baustein wird nicht ausgeführt > Baustein-Eingänge sind nicht aktiv > Baustein-Ausgänge sind nicht spezifiziert
INIT	BOOL	FALSE ⇨ TRUE (Flanke): Baustein wird initialisiert FALSE: im weiteren Programmablauf
R_RAMP	INT	Steigende Flanke der Rampe in [Inkremente/SPS-Zyklus] 0 = keine Rampe
F_RAMP	INT	Fallende Flanke der Rampe in [Inkremente/SPS-Zyklus] 0 = keine Rampe
TIMEBASE	TIME	Referenz für steigende und fallende Flanke der Rampe: t#0s = steigende / fallende Flanke in [Inkremente/SPS-Zyklus] Schnelle Controller haben sehr kurze Zykluszeiten! sonst = steigende / fallende Flanke in [Inkremente/TIMEBASE]
X	WORD	Eingangswert
XH	WORD	obere Grenze des Eingangswertebereichs [Inkremente]
XL	WORD	untere Grenze des Eingangswertebereichs [Inkremente]
MAX_CURRENT	WORD	Max. Ventilstrom in [mA]
MIN_CURRENT	WORD	Min. Ventilstrom in [mA]
TOLERANCE	BYTE	Toleranz für min. Ventilstrom in [Inkremente] Bei Überschreiten der Toleranz erfolgt Sprung auf MIN_CURRENT
CHANNEL	BYTE	Nummer des stromgeregelten Ausgangskanals 0...15 für die Ausgänge Q00...Q15 Für den FB xxx_**E** (falls vorhanden) gilt: 0...15 für die Ausgänge Q00_E...Q15_E
PWM_FREQUENCY	WORD	PWM-Frequenz [Hz] für die Last am Ausgang
DITHER_FREQUENCY	WORD	Dither-Frequenz in [Hz] Wertebereich = 0...FREQUENCY / 2 FREQUENCY / DITHER_FREQUENCY muss geradzahlig sein! Alle anderen Werte erhöht der FB auf den nächst passenden Wert.
DITHER_VALUE	BYTE	Spitze-Spitze-Wert des Dithers in [%] zulässige Werte = 0...100 = 0x00...0x64
KP	BYTE	Proportional-Anteil des Ausgangsignals
KI	BYTE	Integral-Anteil des Ausgangsignals

Für KP, KI gilt: empfohlener Startwert = 50

Parameter der Ausgänge

602

Parameter	Datentyp	Beschreibung
DESIRED_CURRENT	WORD	Stromsollwert in [mA] für OCC (zu Kontrollzwecken)
ACTUAL_CURRENT	WORD	Ausgangsstrom in [mA]
BREAK	BOOL	Fehler: Leitung am Ausgang unterbrochen
SHORT	BOOL	Fehler: Kurzschluss in Leitung am Ausgang

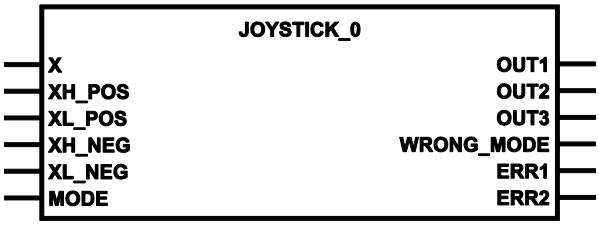
JOYSTICK_0

6250

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_hydraulic_32bit_Vxxyyzz.Lib`

Symbol in CODESYS:



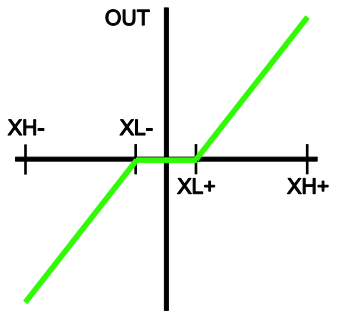
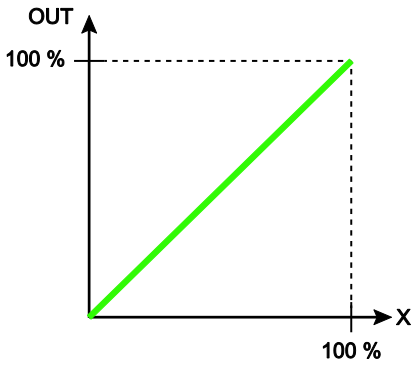
Beschreibung

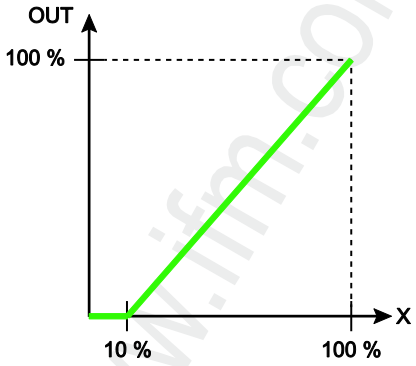
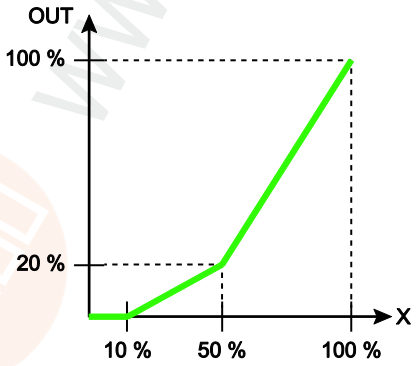
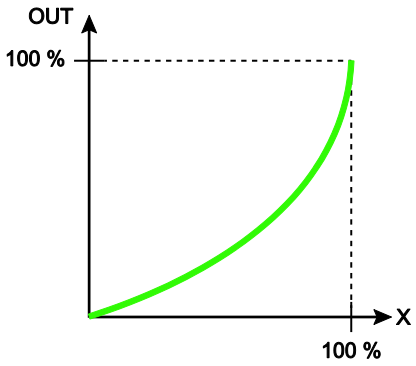
432

JOYSTICK_0 skaliert Signale aus einem Joystick auf fest definierte Kennlinien, normiert auf 0…1000.

Bei diesem FB sind die Kennlinien-Werte fest vorgegeben (→ Grafiken):

- Steigende Flanke der Rampe = 5 Inkremente/SPS-Zyklus
  Schnelle Controller haben sehr kurze Zykluszeiten!
- Fallende Flanke der Rampe = keine Rampe

Die Parameter XL_POS (XL+), XH_POS (XH+), XL_NEG (XL-) und XH_NEG (XH-) dienen dazu, die Joystickbewegung nur im erwünschten Bewegungsbereich auszuwerten. Die Werte für den positiven und den negativen Bereich dürfen sich unterscheiden. Die Werte für XL_NEG und XH_NEG sind hier negativ.	
Modus 0: Kennlinie linear für den Bereich XL bis XH	

Modus 1: Kennlinie linear mit Totbereich Werte fest eingestellt auf: Totbereich: 0…10% von 1000 Inkrementen	
Modus 2: Kennlinie 2-stufig linear mit Totbereich Werte fest eingestellt auf: Totbereich: 0…10% von 1000 Inkrementen Stufe: X = 50 % von 1000 Inkrementen Y = 20 % von 1000 Inkrementen	
Kennlinie Modus 3: Kurve ansteigend (Verlauf ist fest eingestellt)	

Parameter der Eingänge

433

Parameter	Datentyp	Beschreibung
X	INT	Eingangswert [Inkremente]
XH_POS	INT	Max. Sollwert positive Richtung [Inkremente] (auch negative Werte zulässig)
XL_POS	INT	Min. Sollwert positive Richtung [Inkremente] (auch negative Werte zulässig)
XH_NEG	INT	Max. Sollwert negative Richtung [Inkremente] (auch negative Werte zulässig)
XL_NEG	INT	Min. Sollwert negative Richtung [Inkremente] (auch negative Werte zulässig)
MODE	BYTE	Modus Auswahl Kennlinie: 0 = linear (X|OUT = 0|0 ... 1000|1000) 1 = linear mit Totbereich (X|OUT = 0|0 ... 100|0 ... 1000|1000) 2 = 2-stufig linear mit Totbereich (X|OUT = 0|0 ... 100|0 ... 500|200 ... 1000|1000) 3 = Kurve ansteigend (Verlauf ist fest eingestellt)

Parameter der Ausgänge

6252

Parameter	Datentyp	Beschreibung
OUT1	WORD	normierter Ausgangswert: 0…1000 Inkremente z.B. für Ventil links
OUT2	WORD	normierter Ausgangswert: 0…1000 Inkremente z.B. für Ventil rechts
OUT3	INT	normierter Ausgangswert: -1000…0…1000 Inkremente z.B. für Ventil an Ausgangsmodul (z.B. CR2011 oder CR2031)
WRONG_MODE	BOOL	Fehler: Ungültiger Modus
ERR1	BYTE	Fehler-Code für steigende Flanke (bezogen auf die intern verwendeten FBs CHARCURVE und RAMP_INT aus der `util.lib`) (mögliche Meldungen → folgende Tabelle)
ERR2	BYTE	Fehler-Code für fallende Flanke (bezogen auf die intern verwendeten FBs CHARCURVE und RAMP_INT aus der `util.lib`) (mögliche Meldungen → folgende Tabelle)

Mögliche Ergebnisse für ERR1 und ERR2:

Wert dez	Wert hex	Beschreibung
0	00	kein Fehler
1	01	Fehler in Zahlenreihe: Falsche Reihenfolge
2	02	Fehler: Eingangswert IN ist nicht im Wertebereich der Zahlenreihe
4	04	Fehler: Ungültige Anzahl N für Zahlenreihe

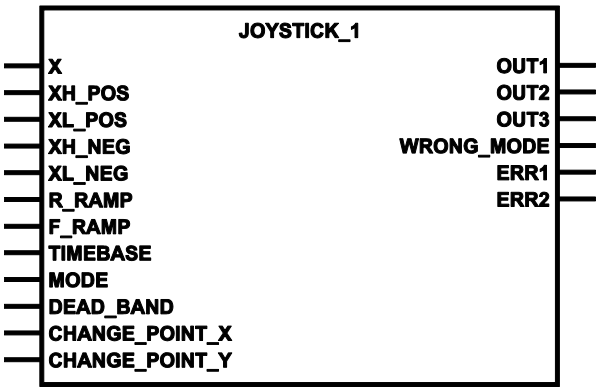
JOYSTICK_1

6255

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_hydraulic_32bit_Vxxyyzz.Lib`

Symbol in CODESYS:

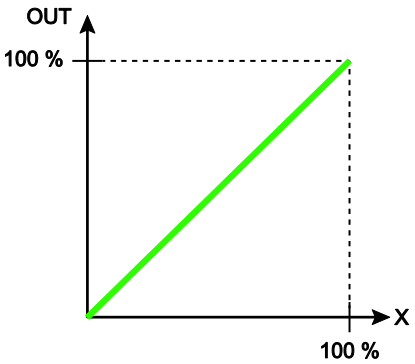
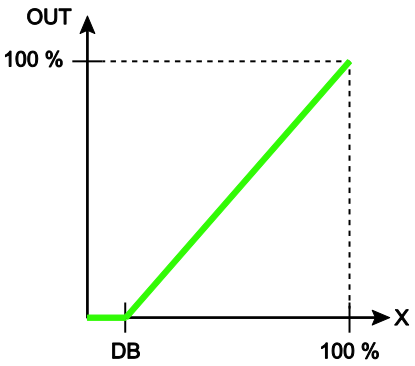


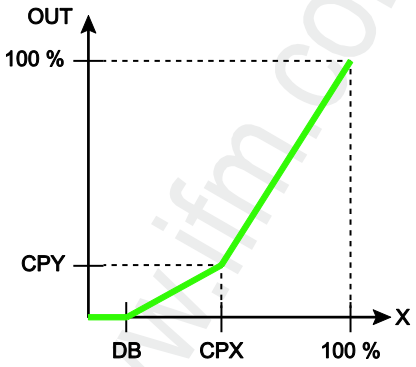
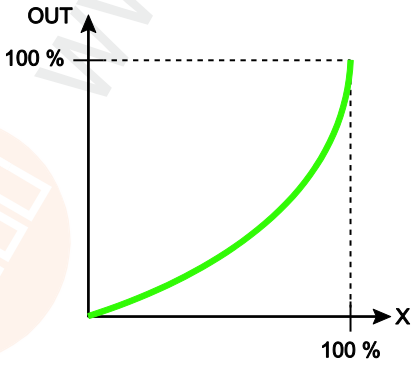
Beschreibung

425

JOYSTICK_1 skaliert Signale aus einem Joystick auf parametrierbare Kennlinien, normiert auf 0…1000.

Bei diesem FB sind die Kennlinien-Werte parametrierbar (→ Grafiken):

Modus 0: Kennlinie linear 100 % = 1000 Inkremente	
Modus 1: Kennlinie linear mit Totbereich Wert für Totbereich (DB) einstellbar in % von 1000 Inkrementen 100 % = 1000 Inkremente DB = Dead_Band	

Modus 2: Kennlinie 2-stufig linear mit Totbereich Werte parametrierbar auf: Totbereich: 0…DB in % von 1000 Inkrementen Stufe: X = CPX in % von 1000 Inkrementen Y = CPY in % von 1000 Inkrementen 100 % = 1000 Inkremente DB = Dead_Band CPX = Change_Point_X CPY = Change_Point_Y	
Kennlinie Modus 3: Kurve ansteigend (Verlauf ist fest eingestellt)	

Parameter der Eingänge

6256

Parameter	Datentyp	Beschreibung
X	INT	Eingangswert [Inkremente]
XH_POS	INT	Max. Sollwert positive Richtung [Inkremente] (auch negative Werte zulässig)
XL_POS	INT	Min. Sollwert positive Richtung [Inkremente] (auch negative Werte zulässig)
XH_NEG	INT	Max. Sollwert negative Richtung [Inkremente] (auch negative Werte zulässig)
XL_NEG	INT	Min. Sollwert negative Richtung [Inkremente] (auch negative Werte zulässig)
R_RAMP	INT	Steigende Flanke der Rampe in [Inkremente/SPS-Zyklus] 0 = keine Rampe
F_RAMP	INT	Fallende Flanke der Rampe in [Inkremente/SPS-Zyklus] 0 = keine Rampe
TIMEBASE	TIME	Referenz für steigende und fallende Flanke der Rampe: t#0s = steigende / fallende Flanke in [Inkremente/SPS-Zyklus] Schnelle Controller haben sehr kurze Zykluszeiten! sonst = steigende / fallende Flanke in [Inkremente/TIMEBASE]
MODE	BYTE	Modus Auswahl Kennlinie: 0 = linear (X|OUT = 0|0 ... 1000|1000) 1 = linear mit Totbereich (X|OUT = 0|0 ... DB|0 ... 1000|1000) 2 = 2-stufig linear mit Totbereich (X|OUT = 0|0 ... DB|0 ... CPX|CPY ... 1000|1000) 3 = Kurve ansteigend (Verlauf ist fest eingestellt)
DEAD_BAND	BYTE	Einstellbarer Totbereich in [% von 1000 Inkrementen]
CHANGE_POINT_X	BYTE	Für Modus 2: Rampenstufe, Wert für X in [% von 1000 Inkrementen]
CHANGE_POINT_Y	BYTE	Für Modus 2: Rampenstufe, Wert für Y in [% von 1000 Inkrementen]

Parameter der Ausgänge

6252

Parameter	Datentyp	Beschreibung
OUT1	WORD	normierter Ausgangswert: 0…1000 Inkremente z.B. für Ventil links
OUT2	WORD	normierter Ausgangswert: 0…1000 Inkremente z.B. für Ventil rechts
OUT3	INT	normierter Ausgangswert: -1000…0…1000 Inkremente z.B. für Ventil an Ausgangsmodul (z.B. CR2011 oder CR2031)
WRONG_MODE	BOOL	Fehler: Ungültiger Modus
ERR1	BYTE	Fehler-Code für steigende Flanke (bezogen auf die intern verwendeten FBs CHARCURVE und RAMP_INT aus der `util.lib`) (mögliche Meldungen → folgende Tabelle)
ERR2	BYTE	Fehler-Code für fallende Flanke (bezogen auf die intern verwendeten FBs CHARCURVE und RAMP_INT aus der `util.lib`) (mögliche Meldungen → folgende Tabelle)

Mögliche Ergebnisse für ERR1 und ERR2:

Wert dez	Wert hex	Beschreibung
0	00	kein Fehler
1	01	Fehler in Zahlenreihe: Falsche Reihenfolge
2	02	Fehler: Eingangswert IN ist nicht im Wertebereich der Zahlenreihe
4	04	Fehler: Ungültige Anzahl N für Zahlenreihe

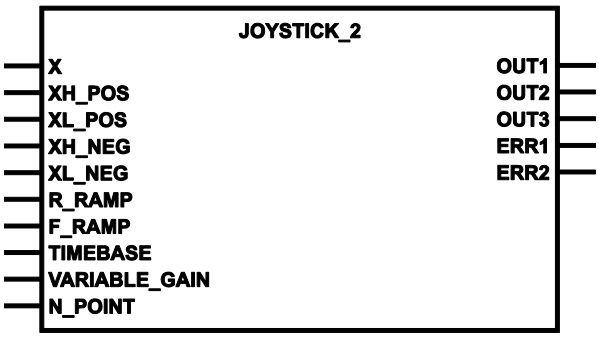
JOYSTICK_2

6258

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_hydraulic_32bit_Vxxyyzz.Lib`

Symbol in CODESYS:

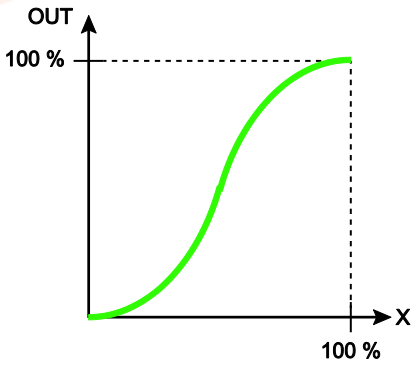


Beschreibung

418

JOYSTICK_2 skaliert Signale aus einem Joystick auf einen parametrierbaren Kennlinien-Verlauf. Die Normierung ist frei bestimmbar.

Bei diesem FB ist der Kennlinien-Verlauf frei parametrierbar (→ Grafik):

Kennlinie frei parametrierbar	

Parameter der Eingänge

6261

Parameter	Datentyp	Beschreibung
X	INT	Eingangswert [Inkremente]
XH_POS	INT	Max. Sollwert positive Richtung [Inkremente] (auch negative Werte zulässig)
XL_POS	INT	Min. Sollwert positive Richtung [Inkremente] (auch negative Werte zulässig)
XH_NEG	INT	Max. Sollwert negative Richtung [Inkremente] (auch negative Werte zulässig)
XL_NEG	INT	Min. Sollwert negative Richtung [Inkremente] (auch negative Werte zulässig)
R_RAMP	INT	Steigende Flanke der Rampe in [Inkremente/SPS-Zyklus] 0 = keine Rampe
F_RAMP	INT	Fallende Flanke der Rampe in [Inkremente/SPS-Zyklus] 0 = keine Rampe
TIMEBASE	TIME	Referenz für steigende und fallende Flanke der Rampe: t#0s = steigende / fallende Flanke in [Inkremente/SPS-Zyklus] Schnelle Controller haben sehr kurze Zykluszeiten! sonst = steigende / fallende Flanke in [Inkremente/TIMEBASE]
VARIABLE_GAIN	ARRAY [0..10] OF POINT	Wertepaare, die den Kurven-Verlauf beschreiben Es werden die ersten in N_POINT angegebenen Wertepaare verwertet. n = 2…11 **Beispiel:** 9 Wertepaare als Variable VALUES deklariert: `VALUES : ARRAY [0..10] OF POINT := (X:=0,Y:=0),(X:=200,Y:=0), (X:=300,Y:=50), (X:=400,Y:=100), (X:=700,Y:=500), (X:=1000,Y:=900), (X:=1100,Y:=950), (X:=1200,Y:=1000), (X:=1400,Y:=1050);` Zwischen den Werten dürfen auch Leerzeichen stehen.
N_POINT	BYTE	Anzahl der Punkte (Wertepaare in VARIABLE_GAIN), womit die Kurven-Charakteristik definiert ist: n = 2…11

Parameter der Ausgänge

420

Parameter	Datentyp	Beschreibung
OUT1	WORD	normierter Ausgangswert: 0…1000 Inkremente z.B. für Ventil links
OUT2	WORD	normierter Ausgangswert: 0…1000 Inkremente z.B. für Ventil rechts
OUT3	INT	normierter Ausgangswert: -1000…0…1000 Inkremente z.B. für Ventil an Ausgangsmodul (z.B. CR2011 oder CR2031)
ERR1	BYTE	Fehler-Code für steigende Flanke (bezogen auf die intern verwendeten FBs CHARCURVE und RAMP_INT aus der `util.lib`) (mögliche Meldungen → folgende Tabelle)
ERR2	BYTE	Fehler-Code für fallende Flanke (bezogen auf die intern verwendeten FBs CHARCURVE und RAMP_INT aus der `util.lib`) (mögliche Meldungen → folgende Tabelle)

Mögliche Ergebnisse für ERR1 und ERR2:

Wert dez	Wert hex	Beschreibung
0	00	kein Fehler
1	01	Fehler in Zahlenreihe: Falsche Reihenfolge
2	02	Fehler: Eingangswert IN ist nicht im Wertebereich der Zahlenreihe
4	04	Fehler: Ungültige Anzahl N für Zahlenreihe

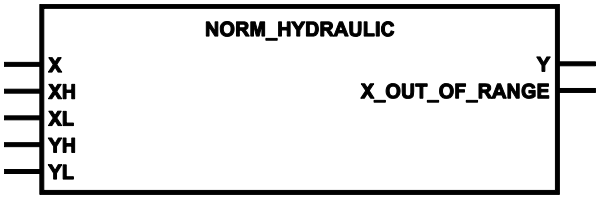
NORM_HYDRAULIC

394

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_hydraulic_32bit_Vxxyyzz.Lib`

Symbol in CODESYS:



Beschreibung

397

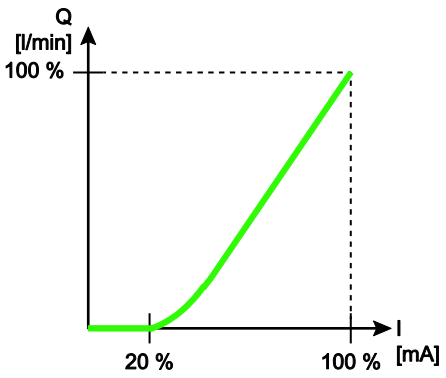
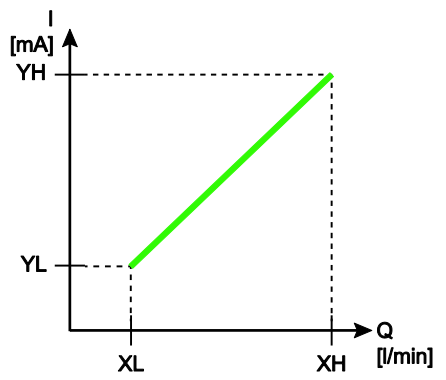
NORM_HYDRAULIC normiert Eingangswerte innerhalb festgesetzter Grenzen auf Werte mit neuen Grenzen.

Dieser FB entspricht NORM_DINT aus der CODESYS-Bibliothek `UTIL.Lib`.

Der FB normiert einen Wert vom Typ DINT, der innerhalb der Grenzen zwischen XH und XL liegt, auf einen Ausgangswert innerhalb der Grenzen zwischen YH und YL.

Bedingt durch Rundungsfehler können Abweichungen beim normierten Wert um 1 auftreten. Werden die Grenzen (XH/XL oder YH/YL) invertiert angegeben, erfolgt auch die Normierung invertiert.

Wenn X außerhalb der Grenzen XL…XH liegt, wird die Fehlermeldung X_OUT_OF_RANGE = TRUE.

Typischer Kennlinienverlauf eines Hydraulikventils: Erst bei ca. 20 % des Spulenstroms beginnt der Ölfluss. Der Ölfluss ist anfänglich nicht linear.	
Charakteristik des Funktionsbausteins	

Parameter der Eingänge

398

Parameter	Datentyp	Beschreibung
X	DINT	Eingangswert
XH	DINT	Max. Eingangswert [Inkremente]
XL	DINT	Min. Eingangswert [Inkremente]
YH	DINT	Max. Ausgangswert [Inkremente], z.B.: Ventilstrom [mA], Durchfluss [l/min]
YL	DINT	Min. Ausgangswert [Inkremente], z.B.: Ventilstrom [mA], Durchfluss [l/min]

Parameter der Ausgänge

399

Parameter	Datentyp	Beschreibung
Y	DINT	Ausgangswert
X_OUT_OF_RANGE	BOOL	Fehler: X liegt außerhalb der Grenzen von XH und XL

Beispiel: NORM_HYDRAULIC

400

Parameter	Fall 1	Fall 2	Fall 3
oberer Grenzwert Eingang XH	100	100	2000
unterer Grenzwert Eingang XL	0	0	0
oberer Grenzwert Ausgang YH	2000	0	100
unterer Grenzwert Ausgang YL	0	2000	0
nicht normierter Wert X	20	20	20
normierter Wert Y	400	1600	1

- Fall 1:
  Eingang mit relativ grober Auflösung.
  Ausgang mit hoher Auflösung.
  1 X-Inkrement ergibt 20 Y-Inkremente.
- Fall 2:
  Eingang mit relativ grober Auflösung.
  Ausgang mit hoher Auflösung.
  1 X-Inkrement ergibt 20 Y-Inkremente.
  Ausgangssignal ist gegenüber dem Eingangssignal invertiert.
- Fall 3:
  Eingang mit hoher Auflösung.
  Ausgang mit relativ grober Auflösung.
  20 X-Inkremente ergeben 1 Y-Inkrement.

5.2.13 Bausteine: Regler

Inhalt

1634

Der nachfolgende Abschnitt beschreibt im Detail die Bausteine, die zum Aufbau von Software-Reglern im ecomatmobile-Gerät bereitgestellt werden. Die Bausteine können auch als Basis für die Entwicklung von eigenen Regelungsfunktionen genutzt werden.

Einstellregel für einen Regler

1627

Für Regelstrecken, deren Zeitkonstanten nicht bekannt sind, ist das Einstellverfahren nach Ziegler und Nickols im geschlossenen Regelkreis vorteilhaft:

Einstellregel

1628

Die Regeleinrichtung wird zunächst als eine reine P-Regeleinrichtung betrieben. Dazu wird die Vorhaltezeit T_V auf 0 und die Nachstellzeit T_N auf einen sehr großen Wert (ideal auf unendlich) für eine träge Strecke eingestellt. Bei einer schnellen Regelstrecke sollte ein kleines T_N gewählt werden.

Der Proportionalbeiwert KP wird anschließend solange vergrößert, bis die Regel- und die Stellabweichung bei KP = $KP_{kritisch}$ Dauerschwingungen mit konstanter Amplitude ausführen. Es ist damit die Stabilitätsgrenze erreicht.

Anschließend muss die Periodendauer $T_{kritisch}$ der Dauerschwingung ermittelt werden.

Nur bei Bedarf einen D-Anteil hinzufügen.

T_V sollte ca. 2...10-mal kleiner sein als T_N.

KP sollte gleich groß wie KD gewählt werden.

Idealisiert ist die Regelstrecke wie folgt einzustellen:

Regeleinrichtung	KP = KD	TN	TV
P	2,0 • $KP_{kritisch}$	—	—
PI	2,2 • $KP_{kritisch}$	0,83 • $T_{kritisch}$	—
PID	1,7 • $KP_{kritisch}$	0,50 • $T_{kritisch}$	0,125 • $T_{kritisch}$

Bei diesem Einstellverfahren darauf achten, dass die Regelstrecke durch die auftretenden Schwingungen keinen Schaden nimmt. Bei empfindlichen Regelstrecken darf KP nur bis zu einem Wert erhöht werden, bei dem sicher noch keine Schwingungen auftreten.

Dämpfung von Überschwingungen

1629

Um Überschwingungen zu dämpfen, kann **PT1** (→ S. 181) (Tiefpass) eingesetzt werden. Dazu wird der Sollwert XS durch das PT1-Glied gedämpft, bevor er der Reglerfunktion zugeführt wird.

Die Einstellgröße T1 sollte ca. 4...5-mal größer sein als TN des Reglers.

DELAY

585

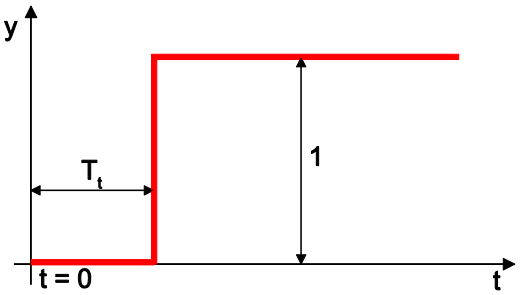
Baustein-Typ = Funktionsbaustein (FB)
Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:

Beschreibung

588

DELAY verzögert die Ausgabe des Eingangswertes um die Zeit T (Totzeit-Glied).



Grafik: Zeitlicher Verlauf von DELAY

Die Totzeit wird durch die Dauer des SPS-Zyklus beeinflusst.
Die Totzeit darf nicht länger sein als 100 • SPS-Zykluszeit (Speichergrenze!).
Wird eine größere Verzögerung eingestellt, wird die Auflösung der Werte am Ausgang des FB schlechter, wodurch kurze Werteänderungen verloren gehen können.

Damit der FB einwandfrei arbeitet: FB in jedem SPS-Zyklus aufrufen!

Parameter der Eingänge

2615

Parameter	Datentyp	Beschreibung
X	REAL	Eingangswert
T	TIME	Verzögerungszeit (Totzeit) zulässig: 0...100 • Zykluszeit

Parameter der Ausgänge

2616

Parameter	Datentyp	Beschreibung
Y	REAL	Eingangswert, verzögert um die Zeit T

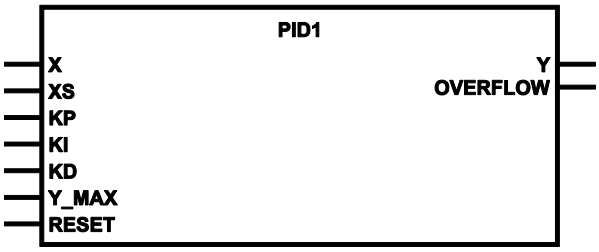
PID1

19235

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:



Beschreibung

19237

PID1 organisiert einen PID-Regler.

Die Änderung der Stellgröße eines PID-Reglers setzt sich aus einem proportionalen, integralen und differentialen Anteil zusammen.

Wenn der I-Anteil eine interne Begrenzung erreicht, weil eine Regelabweichung nicht ausgeregelt werden konnte, wird OVERFLOW = TRUE gemeldet.
OVERFLOW bleibt solange TRUE, solange die Begrenzung aktiv ist.

Parameter der Eingänge

19238

Parameter	Datentyp	Beschreibung
X	REAL	Eingangswert
XS	REAL	Sollwert
KP	REAL	Proportional-Anteil des Ausgangsignals (nur positive Werte zulässig)
KI	REAL	Integral-Anteil des Ausgangsignals (nur positive Werte zulässig)
KD	REAL	Differential-Anteil des Ausgangsignals (nur positive Werte zulässig)
Y_MAX	REAL	Maximaler Stellwert
RESET	BOOL	TRUE: Regler zurücksetzen FALSE: Funktion wird nicht ausgeführt

Parameter der Ausgänge

19241

Parameter	Datentyp	Beschreibung
Y	REAL	Ausgangswert
OVERFLOW	BOOL	TRUE: Überlauf des Datenpuffers ⇨ Datenverlust! FALSE: Datenpuffer ist ohne Datenverlust

Einstellempfehlung

19242

- ► Startwerte:
  KP = 0
  KD = 0
- ► KI dem Prozess anpassen.
- ► KP und KI anschließend schrittweise verändern.

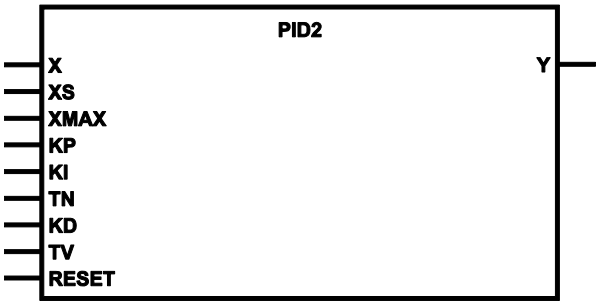
PID2

344

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:



Beschreibung

6262

PID2 organisiert einen PID-Regler.

Die Änderung der Stellgröße eines PID-Reglers setzt sich aus einem proportionalen, integralen und differentialen Anteil zusammen. Die Stellgröße ändert sich zunächst um einen von der Änderungsgeschwindigkeit der Eingangsgröße abhängigen Betrag (**D**ifferential-Anteil). Nach Ablauf der Vorhaltezeit TV geht die Stellgröße auf den dem Proportionalbereich entsprechenden Wert zurück und ändert sich dann entsprechend der Nachstellzeit TN.

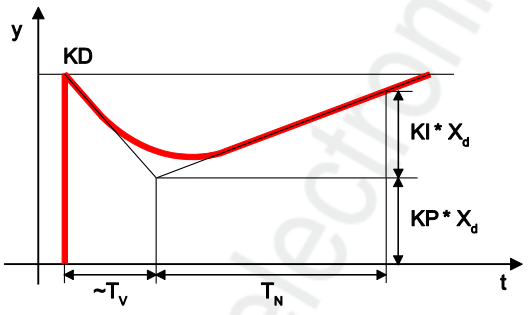
ⓘ Die Stellgröße Y ist bereits auf **PWM1000** (→ S. 158) normiert.

Regeln:

- Negative Werte bei KP, KI und KD sind nicht zulässig.
- Bei TN = 0 wird der I-Anteil nicht berechnet.
- Bei XS > XMAX wird XS auf XMAX limitiert.
- Bei X > XMAX wird Y auf 0 gesetzt.
- Wenn X > XS, dann wird die Stellgröße erhöht.
- Wenn X < XS, dann wird die Stellgröße reduziert.

Eine Führungsgröße wird intern zur Stellgröße hinzuaddiert:
Y = Y + 65 536 - (XS / XMAX • 65 536).

Die Stellgröße Y hat folgenden zeitlichen Verlauf.



Grafik: Typische Sprungantwort eines PID-Reglers

Parameter der Eingänge

12963

Parameter	Datentyp	Beschreibung
X	WORD	Eingangswert
XS	WORD	Sollwert
XMAX	WORD	Maximaler Istwert zur Festlegung des Istwert-Wertebereichs
KP	REAL	Proportional-Anteil des Ausgangsignals (nur positive Werte zulässig)
KI	REAL	Integral-Anteil des Ausgangsignals (nur positive Werte zulässig)
TN	TIME	Nachstellzeit (Integral-Anteil)
KD	REAL	Differential-Anteil des Ausgangsignals (nur positive Werte zulässig)
TV	TIME	Vorhaltezeit (Differential-Anteil)
RESET	BOOL	TRUE: Regler zurücksetzen FALSE: Funktion wird nicht ausgeführt

Parameter der Ausgänge

349

Parameter	Datentyp	Beschreibung
Y	WORD	Stellgröße (0...1000 ‰)

Einstellempfehlung

350

- ► TN gemäß des Zeitverhaltens der Strecke wählen
  (schnelle Strecke = kleines TN, träge Strecke = großes TN)
- ► KP langsam, schrittweise erhöhen bis zu einem Wert, bei dem sicher noch kein Schwingen auftritt.
- ► TN bei Bedarf nachjustieren
- ► Nur bei Bedarf D-Anteil hinzufügen:
  TV ca. 2...10-mal kleiner als TN wählen.
  KD etwa gleich groß wie KP wählen.

Beachten Sie, dass die maximale Regelabweichung + 127 beträgt. Für ein gutes Regelverhalten sollte dieser Bereich einerseits nicht überschritten, andererseits aber möglichst ausgenutzt werden.

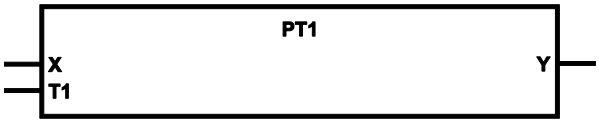
PT1

338

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:



Beschreibung

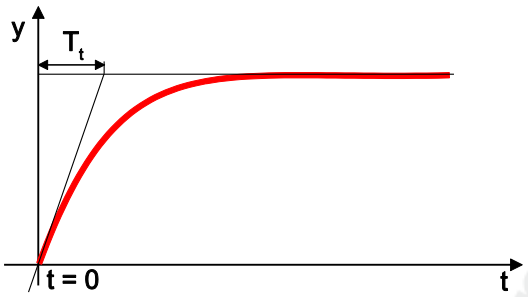
341

PT1 organisiert eine Regelstrecke mit Verzögerung 1. Ordnung.

Bei der Funktion handelt es sich um eine proportionale Regelstrecke mit Verzögerung. Sie wird z.B. zur Bildung von Rampen bei Einsatz der PWM-Funktionen genutzt.

> Der Ausgang des FB kann instabil werden, wenn T1 kleiner ist als die SPS-Zykluszeit.

Die Ausgangsvariable Y des Tiefpassfilters hat folgenden zeitlichen Verlauf (Einheitssprungfunktion):



Grafik: Zeitlicher Verlauf bei PT1

Parameter der Eingänge

2618

Parameter	Datentyp	Beschreibung
X	DINT	Eingangswert
T1	TIME	Verzögerungszeit (Zeitkonstante)

Parameter der Ausgänge

2619

Parameter	Datentyp	Beschreibung
Y	DINT	Ausgangswert

5.2.14 Bausteine: Software-Reset

Inhalt

1594

Hiermit kann die Steuerung per Kommando im Anwendungsprogramm neu gestartet werden.

SOFTRESET

260

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:

Beschreibung

263

SOFTRESET führt einen kompletten Neustart des Geräts aus.

Die Funktion kann z.B. in Verbindung mit CANopen genutzt werden, wenn ein Node-Reset ausgeführt werden soll. Der FB SOFTRESET führt einen sofortigen Neustart der Steuerung durch. Der aktuelle Zyklus wird nicht beendet.

Vor dem Neustart erfolgt das Speichern der Retain- Variablen.
Der Neustart wird im Fehlerspeicher protokolliert.

Bei einer laufenden Kommunikation: die lange Reset-Phase beachten, da andernfalls Guarding-Fehler gemeldet werden.

Parameter der Eingänge

264

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein ausführen FALSE: Baustein wird nicht ausgeführt > Baustein-Eingänge sind nicht aktiv > Baustein-Ausgänge sind nicht spezifiziert

5.2.15 Bausteine: Zeit messen / setzen

Inhalt

1601

Mit folgenden Bausteinen der **ifm electronic** können Sie...
- Zeiten messen und im Anwendungsprogramm auswerten,
- bei Bedarf Zeitwerte ändern.

TIMER_READ

236

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:

Beschreibung

239

TIMER_READ liest die aktuelle Systemzeit aus.

Mit Anlegen der Versorgungsspannung bildet das Gerät einen Zeittakt, der in einem Register aufwärts gezählt wird. Dieses Register kann mittels des Funktionsaufrufes ausgelesen und z.B. zur Zeitmessung genutzt werden.

Der System-Timer läuft maximal bis 0xFFFF FFFF (entspricht 49d 17h 2min 47s 295ms) und startet anschließend wieder mit 0.

Parameter der Ausgänge

241

Parameter	Datentyp	Beschreibung
T	TIME	Aktuelle Systemzeit [ms]

TIMER_READ_US

657

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:

Beschreibung

660

TIMER_READ_US liest die aktuelle Systemzeit in [µs] aus.

Mit Anlegen der Versorgungsspannung bildet das Gerät einen Zeittakt, der in einem Register aufwärts gezählt wird. Dieses Register kann mittels des FB-Aufrufes ausgelesen werden und z.B. zur Zeitmessung genutzt werden.

> **Info**
>
> Der System-Timer läuft maximal bis zum Zählerwert 1h 11min 34s 967ms 295µs und startet anschließend wieder mit 0.

Parameter der Ausgänge

662

Parameter	Datentyp	Beschreibung
TIME_US	DWORD	Aktuelle Systemzeit [µs]

5.2.16 Bausteine: Gerätetemperatur auslesen

Inhalt

2364

Mit folgendem Baustein zeigt Ihnen das Gerät die Innentemperatur.

TEMPERATURE

2216

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:

Beschreibung

2365

TEMPERATURE liest die aktuelle Temperatur im Gerät aus.

Der FB kann zyklisch aufgerufen werden und zeigt am Ausgang die aktuelle Gerätetemperatur an (-40...125 °C).

Parameter der Eingänge

2366

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein ausführen FALSE: Baustein wird nicht ausgeführt > Baustein-Eingänge sind nicht aktiv > Baustein-Ausgänge sind nicht spezifiziert

Parameter der Ausgänge

2367

Parameter	Datentyp	Beschreibung
TEMPERATURE	INT	Aktuelle Geräteinnentemperatur [°C]

5.2.17 Bausteine: Daten im Speicher sichern, lesen und wandeln

Inhalt

13795

Speicherarten zur Datensicherung

13805

Das Gerät bietet folgende Speicher:

Flash-Speicher

13803

Eigenschaften:
- nichtflüchtiger Speicher
- relativ langsames und nur blockweises Schreiben
- vor dem erneuten Schreiben muss Speicherinhalt gelöscht werden
- schnelles Lesen
- begrenzte Schreib-/Lesehäufigkeit
- nur zum Speichern großer Datenmengen sinnvoll einsetzbar
- Daten sichern mit FLASHWRITE
- Daten lesen mit FLASHREAD

FRAM-Speicher

13802

FRAM steht hier allgemein für alle Arten von nichtflüchtigen, schnellen Speichern.

Eigenschaften:
- schnelles Schreiben und Lesen
- unbegrenzte Schreib-/Lesehäufigkeit
- beliebige Speicherbereiche wählbar
- Daten sichern mit FRAMWRITE
- Daten lesen mit FRAMREAD

Dateisystem

2690

Das Dateisystem koordiniert, wo im Speicher welche Informationen liegen. Die Größe des Dateisystems beträgt 128 kByte.

Die Dateinamen des Dateisystems sind begrenzt:
max. Länge für Controller: CR0n3n, CR7n3n: 15 Zeichen
max. Länge für alle anderen Geräte: 11 Zeichen

Verhalten des Dateisystems im Controller: CR0n3n, CR7n3n**:**

- Der Controller versucht immer, die Datei zu schreiben, auch wenn der gleiche Dateiname bereits existiert. Gegebenenfalls wird die Datei mehrfach gespeichert.Genutzt wird nur die aktuelle Datei. Über den Download (s.u.) wird diese Mehrfach-Ablage vermieden.
- Einzelne Dateien können nicht überschrieben oder gelöscht werden.
- Das Dateisystem wird bei jedem Download (Boot-Projekt-Download oder RAM-Download) komplett gelöscht. Anschließend kann z.B. eine Symboldatei oder eine Projektdatei (Funktionen in CODESYS) geschrieben werden.
- Das Dateisystem wird ebenfalls bei einem [Reset (Ursprung)] (CODESYS-Funktion im Menü [Online]) gelöscht.

Automatische Datensicherung

Inhalt

14168
2347

Die ecomat*mobile*-Geräte bieten die Möglichkeit, Daten (BOOL, BYTE, WORD, DWORD) remanent (= spannungsausfallsicher) im Speicher zu sichern. Voraussetzung ist, dass die Daten als RETAIN-Variablen angelegt wurden (→ CODESYS).

Man unterscheidet zwischen Variablen, die als RETAIN deklariert wurden, und Variablen im Merkerbereich, der als Block mit **MEMORY_RETAIN_PARAM** (→ S. 192) als remanent konfiguriert werden kann.
Details → Kapitel **Variablen** (→ S. 66)

Der Vorteil des automatischen Speicherns ist, dass auch bei einem plötzlichen Spannungsabfall oder einer Unterbrechung der Versorgungsspannung die aktuellen Werte der Daten erhalten bleiben (z.B. Zählerstände).

Wenn Versorgungsspannung < 8 V, werden keine Retain-Daten mehr gesichert!
In diesem Fall wird Merker RETAIN_WARNING = TRUE.

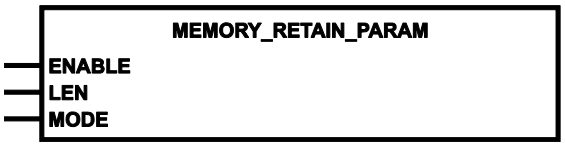
MEMORY_RETAIN_PARAM

2372

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:



Beschreibung

2374

MEMORY_RETAIN_PARAM legt das remanente Verhalten der Daten für verschiedene Ereignisse fest. In CODESYS als VAR_RETAIN deklarierte Variablen haben von vornherein ein remanentes Verhalten.

Remanente Daten behalten (wie die als VAR_RETAIN deklarierte Variablen) ihren Wert nach einem unkontrolliertem Beenden wie auch nach normalem Aus- und Einschalten der Steuerung. Bei erneutem Start arbeitet das Programm mit den gespeicherten Werten weiter.

Für (mit MODE) wählbare Gruppen von Ereignissen legt dieser FB fest, wie viele (LEN) Datenbytes (ab Merkerbyte %MB0) Retain-Verhalten haben sollen, auch wenn sie nicht ausdrücklich als VAR_RETAIN deklariert wurden.

Ereignis	MODE = 0	MODE = 1	MODE = 2	MODE = 3
Power OFF ⇨ ON	Daten werden neu initialisiert	Daten sind remanent	Daten sind remanent	Daten sind remanent
Reset warm	Daten werden neu initialisiert	Daten sind remanent	Daten sind remanent	Daten sind remanent
Reset kalt	Daten werden neu initialisiert	Daten werden neu initialisiert	Daten sind remanent	Daten sind remanent
Reset Ursprung	Daten werden neu initialisiert	Daten werden neu initialisiert	Daten sind remanent	Daten sind remanent
Anwendungsprogramm laden	Daten werden neu initialisiert	Daten werden neu initialisiert	Daten sind remanent	Daten sind remanent
Laufzeitsystem laden	Daten werden neu initialisiert	Daten werden neu initialisiert	Daten werden neu initialisiert	Daten sind remanent

Bei MODE = 0 habe nur solche Daten Retain-Verhalten wie bei MODE=1, die ausdrücklich als VAR_RETAIN deklariert wurden.

Wird der FB nie aufgerufen, verhalten sich die Merkerbytes nach MODE = 0. Die Merkerbytes, die oberhalb des konfigurierten Bereichs liegen, verhalten sich ebenfalls nach MODE = 0.

Eine einmal getätigte Konfiguration bleibt auf dem Gerät erhalten, auch wenn die Anwendung oder das Laufzeitsystem neu geladen werden.

Parameter der Eingänge

2375

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein ausführen FALSE: Baustein wird nicht ausgeführt > Baustein-Eingänge sind nicht aktiv > Baustein-Ausgänge sind nicht spezifiziert
LEN	WORD	Anzahl der Datenbytes ab Merkeradresse %MB0, die remanentes Verhalten haben sollen zulässig = 0...4 096 = 0x0...0x1000 LEN > 4 096 wird automatisch zu LEN = 4 096 korrigiert
MODE	BYTE	Ereignisse, bei denen diese Variablen Retain-Verhalten haben sollen (0...3; → Tabelle oben) Bei MODE > 3 bleibt die zuletzt gültige Einstellung erhalten

Manuelle Datensicherung

Inhalt

13801

Neben der Möglichkeit, die Daten automatisch zu sichern, können über FB-Aufrufe Anwenderdaten manuell in integrierte Speicher gesichert und von dort wieder gelesen werden.

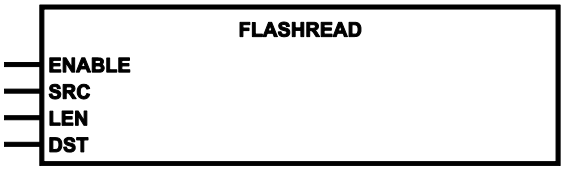
ⓘ Der Programmierer kann sich anhand der Speicheraufteilung (→ Kapitel **Verfügbarer Speicher** (→ S. 16)) darüber informieren, welcher Speicherbereich frei zur Verfügung steht.

FLASHREAD

561

Baustein-Typ = Funktionsbaustein (FB)
Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:



Beschreibung

564

FLASHREAD ermöglicht das Lesen unterschiedlicher Datentypen direkt aus dem Flash-Speicher in den RAM.

> Der FB liest den Inhalt ab der Adresse von SRC aus dem Flash-Speicher. Dabei werden genau so viele Bytes übertragen, wie diese unter LEN angegeben sind.

> Das Lesen erfolgt komplett in dem Zyklus, in dem der FB aufgerufen wird.

► Darauf achten, dass der Zielspeicherbereich im RAM groß genug ist.

► Für die Zieladresse DST gilt:
(!) Die Adresse mit dem Operator ADR ermitteln und dem Baustein übergeben!

Parameter der Eingänge

2318

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein ausführen FALSE: Baustein wird nicht ausgeführt > Baustein-Eingänge sind nicht aktiv > Baustein-Ausgänge sind nicht spezifiziert
SRC	DWORD	Relative Quell-Anfangsadresse im Speicher zulässig = 0...65 535 = 0y0000 0000...0000 FFFF (!) Falls Startadresse außerhalb des zulässigen Bereichs: > kein Datentransfer
LEN	DWORD	Anzahl der Datenbytes (max. 65 536 = 0x0001 0000) (!) Würde durch die angegebene Anzahl an Bytes der Flash-Speicherbereich überschritten werden, werden die Daten nur bis zum Ende des Flash-Speicherbereichs übertragen.
DST	DWORD	Anfangsadresse der Zielvariablen (!) Die Adresse mit dem Operator ADR ermitteln und dem Baustein übergeben!

FLASHWRITE

555

Baustein-Typ = Funktionsbaustein (FB)
Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:

Beschreibung

19245

► Für den Einsatz des FBs den TEST-Eingang aktivieren! Ansonsten tritt ein Watchdog-Fehler auf.

Test-Eingang ist aktiv:
• Programmiermodus ist freigeben
• Software-Download ist möglich
• Zustand des Anwendungsprogramms ist abfragbar
• kein Schutz der gespeicherten Software möglich

558

⚠ WARNUNG

Gefahr durch unkontrollierten Prozessablauf!

Der Zustand der Ein-/Ausgänge wird während der Ausführung von FLASHWRITE "eingefroren".

► Diesen Funktionsbaustein nicht bei laufender Maschine ausführen!

FLASHWRITE ermöglicht das Schreiben unterschiedlicher Datentypen direkt in den Flash-Speicher.

Mit diesem FB sollen während der Inbetriebnahme große Datenmengen gesichert werden, auf die im Prozess nur lesend zugegriffen wird.

Der Flash-Speicher ist in 256 Byte große Pages organisiert.

► Wurde eine Page schon einmal (auch nur teilweise) beschrieben, muss der komplette Flash-Speicherbereich vor einem erneuten Schreibzugriff auf diese Page gelöscht werden. Dies geschieht durch einen Schreibzugriff auf die Adresse 0.

► Niemals mehrfach in eine Page schreiben! Erst immer alles löschen!
Sonst entstehen Traps oder Watchdog-Fehler.

► (!) Den Flash-Speicherbereich nicht öfter als 100mal löschen, da ansonsten die Datenkonsistenz in anderen Flash-Speicherbereichen nicht mehr gewährleistet werden kann.

► In jedem SPS-Zyklus darf FLASHWRITE nur einmalig gestartet werden!

► Für die Quell-Startadresse SRC gilt:
(!) Die Adresse mit dem Operator ADR ermitteln und dem Baustein übergeben!

> Der FB schreibt den Inhalt der Adresse SRC in den Flash-Speicher. Dabei werden genau so viele Bytes übertragen, wie diese unter LEN angegeben sind.

(!) Falls Ziel-Startadresse DST außerhalb des zulässigen Bereichs: kein Datentransfer!

Parameter der Eingänge

2603

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein ausführen FALSE: Baustein wird nicht ausgeführt > Baustein-Eingänge sind nicht aktiv > Baustein-Ausgänge sind nicht spezifiziert
DST	DWORD	Relative Ziel-Anfangsadresse im Speicher zulässig = 0...65 535 = 0x0...0x0000 FFFF
LEN	DWORD	Anzahl der Datenbytes (max. 65 536 = 0x0001 0000) Würde durch die angegebene Anzahl an Bytes der Flash-Speicherbereich überschritten werden, werden die Daten nur bis zum Ende des Flash-Speicherbereichs übertragen.
SRC	DWORD	Anfangsadresse der Quellvariablen Die Adresse mit dem Operator ADR ermitteln und dem Baustein übergeben!

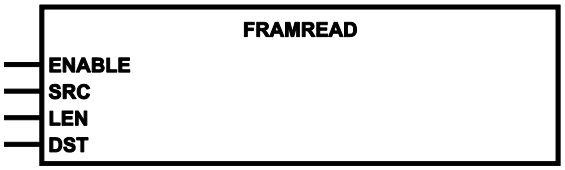
FRAMREAD

549

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:



Beschreibung

552

FRAMREAD ermöglicht das schnelle Lesen unterschiedlicher Datentypen direkt aus dem Anwender-Retain-Speicher (FRAM[1]).

Der FB liest den Inhalt ab der Adresse von SRC aus dem FRAM-Speicher. Dabei werden genau so viele Bytes übertragen, wie diese unter LEN angegeben sind.
Würde durch die angegebene Anzahl an Bytes der FRAM-Speicherbereich überschritten werden, werden nur die Daten bis zum Ende des FRAM-Speicherbereichs gelesen.

► Für die Zieladresse DST gilt:
(!) Die Adresse mit dem Operator ADR ermitteln und dem Baustein übergeben!

[1]) FRAM steht hier allgemein für alle Arten von nichtflüchtigen, schnellen Speichern.

Parameter der Eingänge

2606

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein ausführen FALSE: Baustein wird nicht ausgeführt > Baustein-Eingänge sind nicht aktiv > Baustein-Ausgänge sind nicht spezifiziert
SRC	DWORD	Relative Quell-Anfangsadresse im Speicher zulässig = 0... 16 383 = 0x0000 0000...0x0000 3FFF
LEN	DWORD	Anzahl der Datenbytes zulässig = 0...16 384 = 0x0000 0000...0x0000 4000
DST	DWORD	Anfangsadresse der Zielvariablen (!) Die Adresse mit dem Operator ADR ermitteln und dem Baustein übergeben!

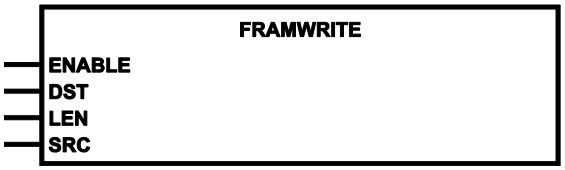
FRAMWRITE

543

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:



Beschreibung

546

FRAMWRITE ermöglicht das schnelle Schreiben unterschiedlicher Datentypen direkt in den Anwender-Retain-Speicher (FRAM[1]).

Der FB schreibt den Inhalt ab der Adresse SRC in den spannungsausfallsicheren FRAM-Speicher. Dabei werden genau so viele Bytes übertragen, wie diese über LEN angegeben sind.
Würde durch die angegebene Anzahl an Bytes der FRAM-Speicherbereich überschritten werden, werden nur die Daten bis zum Ende des FRAM-Speicherbereichs geschrieben.

► Für die Quelladresse SRC gilt:
(!) Die Adresse mit dem Operator ADR ermitteln und dem Baustein übergeben!

(!) Falls Zieladresse DST außerhalb des zulässigen Bereichs: kein Datentransfer!

[1]) FRAM steht hier allgemein für alle Arten von nichtflüchtigen, schnellen Speichern.

Parameter der Eingänge

2605

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein ausführen FALSE: Baustein wird nicht ausgeführt > Baustein-Eingänge sind nicht aktiv > Baustein-Ausgänge sind nicht spezifiziert
DST	DWORD	Relative Zieladresse im Speicher zulässig = 0...16 383 = 0x0000 0000...0x0000 3FFF
LEN	DWORD	Anzahl der Datenbytes zulässig = 0...16 384 = 0x0000 0000...0x0000 4000
SRC	DWORD	Anfangsadresse der Quellvariablen (!) Die Adresse mit dem Operator ADR ermitteln und dem Baustein übergeben!

MEMCPY

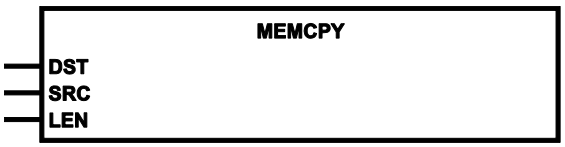
409

= Memory Copy

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:



Beschreibung

15944
412

MEMCPY ermöglicht das Schreiben und Lesen unterschiedlicher Datentypen direkt in den Speicher.

Der FB schreibt den Inhalt ab der Adresse von SRC an die Adresse DST.

► Für die Adressen SRC und DST gilt:
[!] Die Adresse mit dem Operator ADR ermitteln und dem Baustein übergeben!

> Dabei werden genau so viele Bytes übertragen, wie diese unter LEN angegeben wurden. Dadurch ist es auch möglich, genau ein Byte einer Word-Variablen zu übertragen.

> Befindet sich der Speicherbereich, in den die Daten kopiert werden sollen, nicht komplett in einem zulässigen Speicherbereich, werden die Daten nicht kopiert und es wird ein Parameterfehler gemeldet.

DST Speicherbereich	Gerät	Speichergröße
Anwendungsdaten	(alle)	192 kBytes

Tabellen "Verfügbarer Speicher" → Kapitel **Verfügbarer Speicher** (→ S. 16)

Parameter der Eingänge

413

Parameter	Datentyp	Beschreibung
DST	DWORD	Startadresse im Zielspeicher [!] Die Adresse mit dem Operator ADR ermitteln und dem Baustein übergeben!
SRC	DWORD	Startadresse im Quellspeicher [!] Die Adresse mit dem Operator ADR ermitteln und dem Baustein übergeben!
LEN	WORD	Anzahl (≥ 1) der zu übertragenden Daten-Bytes

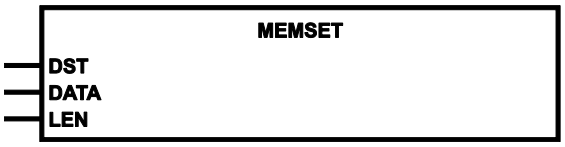
MEMSET

2348

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:



Beschreibung

2350

MEMSET ermöglicht das Beschreiben eines bestimmten Datenbereiches.

Der FB beschreibt den Speicher ab der Adresse DST mit der Anzahl von LEN Bytes mit dem Inhalt von DATA.

► Für die Ziel-Adresse DST gilt:
[!] Die Adresse mit dem Operator ADR ermitteln und dem Baustein übergeben!

> Befindet sich der Speicherbereich, in den die Daten kopiert werden sollen, nicht komplett in einem zulässigen Speicherbereich, werden die Daten nicht kopiert und es wird ein Parameterfehler gemeldet.

DST Speicherbereich	Gerät	Speichergröße
Anwendungsdaten	(alle)	192 kBytes

Parameter der Eingänge

2351

Parameter	Datentyp	Beschreibung
DST	DWORD	Startadresse im Zielspeicher [!] Die Adresse mit dem Operator ADR ermitteln und dem Baustein übergeben!
DATA	BYTE	zu schreibender Wert
LEN	WORD	Anzahl der mit DATA zu beschreibenden Datenbytes

5.2.18 Bausteine: Datenzugriff und Datenprüfung

Inhalt

1598

Die Bausteine in diesem Kapitel steuern den Datenzugriff und ermöglichen ein Prüfen der Daten.

CHECK_DATA

603

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:

Beschreibung

606

CHECK_DATA erzeugt über einen konfigurierbaren Speicherbereich eine Prüfsumme (CRC) und prüft die Daten des Speicherbereichs auf ungewollte Veränderung.

- ► Für jeden zu überwachenden Speicherbereich eine eigene Instanz des FB erzeugen.
- ► Die Adresse mit dem Operator ADR ermitteln und dem Baustein übergeben!
- ► Zusätzlich die Anzahl der Datenbytes LENGTH (Länge ab der STARTADR) angeben.

Ungewollte Änderung: Fehler!
Wenn Eingang UPDATE = FALSE und Daten im Speicher sich ungewollt verändern, wird RESULT = FALSE. Das Ergebnis kann dann für weitere Aktionen (z.B. Abschalten der Ausgänge) genutzt werden.

Gewollte Änderung:
Nur wenn der Eingang UPDATE auf TRUE gesetzt ist, sind Datenänderungen im Speicher (z.B. vom Anwendungsprogramm oder ecomatmobile-Gerät) zulässig. Der Wert der Prüfsumme wird dann neu berechnet. Der Ausgang RESULT ist wieder permanent TRUE.

Parameter der Eingänge

2612

Parameter	Datentyp	Beschreibung
STARTADR	DWORD	Startadresse des überwachten Datenspeichers (WORD-Adresse ab %MW0) Die Adresse mit dem Operator ADR ermitteln und dem Baustein übergeben!
LENGTH	DWORD	Länge des überwachten Datenspeichers in [Byte]
UPDATE	BOOL	TRUE: Daten wurden geändert > FB berechnet eine neue Prüfsumme FALSE: Daten wurden nicht geändert > FB prüft den Speicherbereich

Parameter der Ausgänge

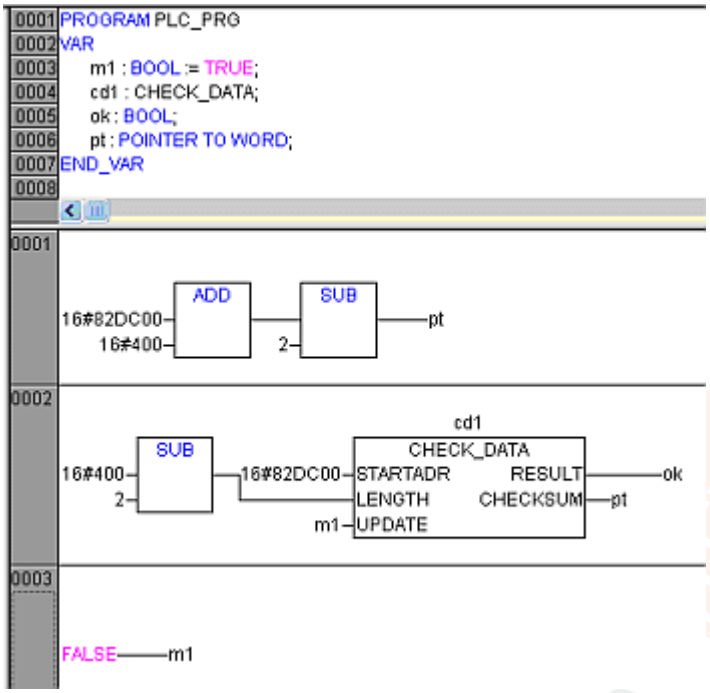
2613

Parameter	Datentyp	Beschreibung
RESULT	BOOL	TRUE: CRC-Prüfsumme in Ordnung: Daten sind gewollt verändert oder nicht verändert FALSE: CRC-Prüfsumme fehlerhaft: Daten wurden ungewollt verändert
CHECKSUM	DWORD	aktuelle CRC-Prüfsumme

Beispiel: CHECK_DATA

4168

Im folgenden Beispiel ermittelt das Programm die Prüfsumme und legt sie über den Pointer pt im RAM ab:



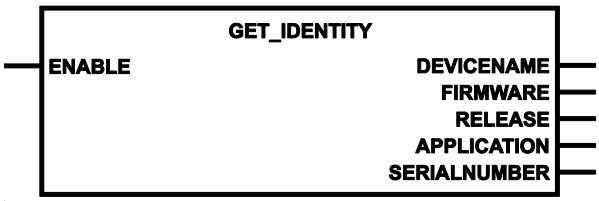
GET_IDENTITY

19287

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`
Neuer Ausgang SERIALNUMBER ist enthalten in:
- CR0032 ab LZS V02.01.06
- CR0033 ab LZS V01.00.09
- CR0133 ab LZS V01.00.09
- CR0232 ab LZS V01.00.03
- CR0233 ab LZS V01.00.09

Symbol in CODESYS:



Beschreibung

19288

GET_IDENTITY liest die im Gerät gespeicherten spezifischen Kennungen:
- Hardware-Name und Hardware-Version des Geräts
- Name des Laufzeitsystems im Gerät
- Version und Ausgabe des Laufzeitsystems im Gerät
- Name der Anwendung (wurde zuvor mit **SET_IDENTITY** (→ S. 208) gespeichert)
- Seriennummer des Geräts

Parameter der Eingänge

2609

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein ausführen FALSE: Baustein wird nicht ausgeführt > Baustein-Eingänge sind nicht aktiv > Baustein-Ausgänge sind nicht spezifiziert

Parameter der Ausgänge

19289

Parameter	Datentyp	Beschreibung
DEVICENAME	STRING(31)	Hardware-Name und Hardware-Version des Geräts als Zeichenkette von max. 31 Zeichen z.B.: "CR0403 01.00.00"
FIRMWARE	STRING(31)	Name des Laufzeitsystems im Gerät als Zeichenkette von max. 31 Zeichen z.B.: "CR0403"
RELEASE	STRING(31)	Version und Ausgabe des Laufzeitsystems im Gerät als Zeichenkette von max. 31 Zeichen z.B.: "V01.00.00 120215"
APPLICATION	STRING(79)	Name der Anwendung als String von max. 79 Zeichen z.B.: "Crane1704"
SERIALNUMBER	STRING(31)	Seriennummer des Geräts als Zeichenkette von max. 31 Zeichen z.B.: "12345678"

GET_IDENTITY_EIOS

19247

EIOS = Extended IO System = Laufzeitsystem der Extended-Seite

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`
Baustein ist enthalten in:
- CR0232 ab LZS V01.00.03
- CR0233 ab LZS V01.00.09
- CR0234
- CR0235

Symbol in CODESYS:

Beschreibung

19249

GET_IDENTITY_EIOS liest die im Gerät für die Extended-Seite gespeicherten spezifischen Kennungen:
- Name des Ein-/Ausgangssystems der Extended-Seite (EIOS) im Gerät
- Version und Ausgabe des Ein-/Ausgangssystems der Extended-Seite (EIOS) im Gerät

Parameter der Eingänge

19250

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein ausführen FALSE: Baustein wird nicht ausgeführt > Baustein-Eingänge sind nicht aktiv > Baustein-Ausgänge sind nicht spezifiziert

Parameter der Ausgänge

19251

Parameter	Datentyp	Beschreibung
FIRMWARE	STRING(31)	Name des Ein-/Ausgangssystems der Extended-Seite im Gerät als Zeichenkette von max. 31 Zeichen
RELEASE	STRING(31)	Version und Ausgabe des Ein-Ausgangssystems der Extended-Seite im Gerät als Zeichenkette von max. 31 Zeichen

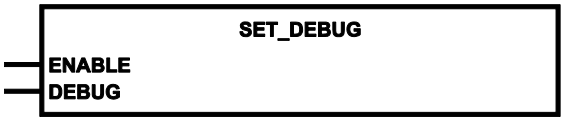
SET_DEBUG

290

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:



Beschreibung

293

SET_DEBUG organisiert den DEBUG-Modus ohne aktiven Test-Eingang (→ Kapitel **TEST-Betrieb** (→ S. 50)).

Wird der Eingang DEBUG auf TRUE gesetzt, kann z.B. das Programmiersystem oder der Downloader mit dem Gerät kommunizieren und einige, spezielle Systemkommandos ausführen (z.B. für Servicefunktionen über das GSM-Modem CANremote).

Ein Software-Download ist in dieser Betriebsart nicht möglich, da der Test-Eingang nicht mit Versorgungsspannung verbunden wird. Nur lesender Zugriff ist möglich.

Parameter der Eingänge

294

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein ausführen FALSE: Baustein wird nicht ausgeführt > Baustein-Eingänge sind nicht aktiv > Baustein-Ausgänge sind nicht spezifiziert
DEBUG	BOOL	TRUE: Debugging über die Schnittstellen möglich FALSE: Debugging über die Schnittstellen nicht möglich

SET_IDENTITY

11927

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:

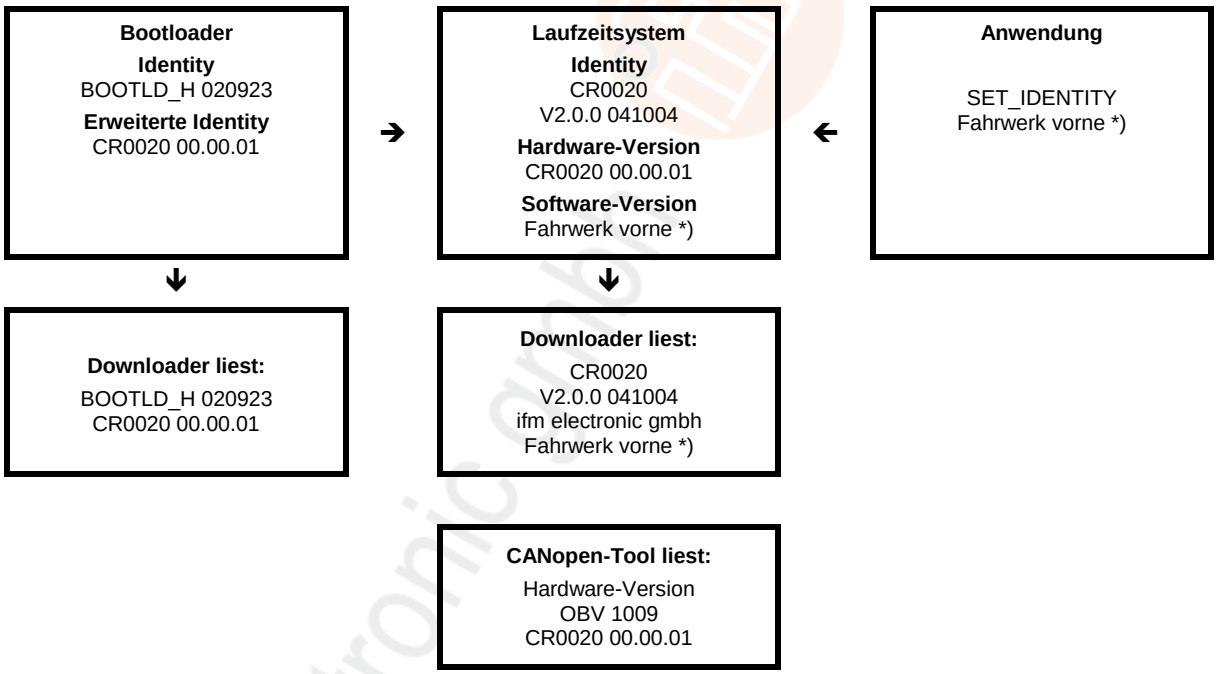
Beschreibung

287

SET_IDENTITY setzt eine anwendungsspezifische Programmkennung.

Mit dem FB kann durch das Anwendungsprogramm eine Programmkennung erzeugt werden. Diese Kennung kann zur Identifizierung des geladenen Programms über das Software-Tool DOWNLOADER.EXE als Software-Version ausgelesen werden.

Die nachfolgende Grafik zeigt die Zusammenhänge der unterschiedlichen Kennungen, wie sie mit den unterschiedlichen Software-Tools angezeigt werden. (Beispiel: ClassicController CR0020):



*) 🛈 'Fahrwerk vorne' steht hier stellvertretend für einen kundenspezifischen Text.

Parameter der Eingänge

11928

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein ausführen FALSE: Baustein wird nicht ausgeführt > Baustein-Eingänge sind nicht aktiv > Baustein-Ausgänge sind nicht spezifiziert
ID	STRING(79)	beliebiger Text mit einer maximalen Länge von 79 Zeichen

SET_PASSWORD

266

Baustein-Typ = Funktionsbaustein (FB)

Baustein ist enthalten in Bibliothek `ifm_CR0232_Vxxyyzz.LIB`

Symbol in CODESYS:

Beschreibung

269

SET_PASSWORD setzt Benutzerkennung für Programm- und Speicher-Upload mit dem DOWNLOADER.

Ist die Benutzerkennung aktiv, kann durch das Software-Tool DOWNLOADER das Anwendungsprogramm oder der Datenspeicher nur ausgelesen werden, wenn das richtige Password eingegeben wurde.

Wird an den Eingang PASSWORD ein Leer-String (Default-Zustand) übergeben, ist ein Upload des Anwendungsprogramms oder des Datenspeichers jederzeit möglich.

Ein neues Passwort wird nur nach dem Löschen des bisherigen Passwortes übernommen.

> Beim Laden eines neuen Anwendungsprogramms als Boot-Projekt wird die Kennung wieder zurückgesetzt.

Parameter der Eingänge

2353

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	FALSE ⇨ TRUE (Flanke): Baustein initialisieren (nur 1 Zyklus) > Baustein-Eingänge lesen TRUE: Baustein ausführen FALSE: Baustein wird nicht ausgeführt > Baustein-Eingänge sind nicht aktiv > Baustein-Ausgänge sind nicht spezifiziert
PASSWORD	STRING(16)	Benutzerkennung Wenn PASSWORD = "", dann ist Zugriff ohne Passworteingabe möglich.

6 Diagnose und Fehlerbehandlung

Inhalt

19598

Das Laufzeitsystem (LZS) überprüft das Gerät durch interne Fehler-Checks:
• in der Startphase (Reset-Phase)
• während der Ausführung des Anwendungsprogramms
→ Kapitel **Betriebszustände** (→ S. 46)

So wird eine möglichst hohe Betriebssicherheit gewährleistet.

6.1 Diagnose

19601

Bei der Diagnose wird der "Gesundheitszustand" des Gerätes geprüft. Es soll festgestellt werden, ob und gegebenenfalls welche →Fehler im Gerät vorhanden sind.

Je nach Gerät können auch die Ein- und Ausgänge auf einwandfreie Funktion überwacht werden:
- Drahtbruch,
- Kurzschluss,
- Wert außerhalb des Sollbereichs.

Zur Diagnose können Konfigurations-Dateien herangezogen werden, die während des "normalen" Betriebs des Gerätes erzeugt wurden.
Der korrekte Start der Systemkomponenten wird während der Initialisierungs- und Startphase überwacht.
Zur weiteren Diagnose können auch Selbsttests durchgeführt werden.

6.2 Fehler

19602

Ein Fehler ist die Unfähigkeit einer Einheit, eine geforderte Funktion auszuführen.
Kein Fehler ist diese Unfähigkeit während vorbeugender Wartung oder anderer geplanter Handlungen oder aufgrund des Fehlers externer Mittel.
Ein Fehler ist oft das Resultat eines Ausfalls der Einheit selbst, kann aber ohne vorherigen Ausfall bestehen.
In der ISO 13849-1 ist mit "Fehler" der "zufällige Fehler" gemeint.

6.3 Reaktion im Fehlerfall

19603
12217

Bei erkannten Fehlern kann im Anwendungsprogramm zusätzlich der Systemmerker ERROR gesetzt werden. Im Fehlerfall reagiert die Steuerung dann wie folgt:

> die Betriebs-LED leuchtet rot,
> die Ausgangsrelais schalten ab,
> die darüber gesicherten Ausgänge sind spannungsfrei,
> die logischen Signalzustände der Ausgänge ändern sich dadurch NICHT.

HINWEIS

Bei Abschalten der Ausgänge durch die Relais bleiben die logischen Signalzustände unverändert.

► Der Programmierer muss das ERROR-Bit auswerten und so im Fehlerfall die Ausgänge auch logisch zurücksetzen.

Vollständige Aufstellung der gerätespezifischen Fehler-Codes und Diagnosemeldungen
→ Kapitel **Systemmerker** (→ S. 214)

6.4 Relais: wichtige Hinweise!

14034

ACHTUNG

Vorzeitiger Verschleiß der Relaiskontakte möglich.

► Im Normalfall die Relais nur lastfrei schalten!
Dazu via Anwendungsprogramm alle relevanten Ausgänge auf FALSE setzen!

6.5 Reaktion auf System-Fehler

14033
4320

Für die sichere Verarbeitung der Daten im Anwendungsprogramm ist allein dessen Programmierer verantwortlich.

► Die spezifischen Fehlermerker und / oder Fehler-Codes im Anwendungsprogramm verarbeiten!
Über den Fehlermerker/Fehler-Code erhält man eine Fehlerbeschreibung.
Dieser Fehlermerker/Fehler-Code kann bei Bedarf weiter verarbeitet werden.

Nach der Analyse und Beseitigung der Fehler-Ursache:

► Grundsätzlich alle Fehlermerker durch das Anwendungsprogramm zurücksetzen.
Ohne ausdrückliches Rücksetzen der Fehlermerker bleiben die Merker gesetzt mit entsprechender Auswirkung im Anwendungsprogramm.

6.6 CAN / CANopen: Fehler und Fehlerbehandlung

19604

→ Systemhandbuch "Know-How ecomat*mobile*"
→ Kapitel **CAN / CANopen: Fehler und Fehlerbehandlung**

7 Anhang

Inhalt

1664

Hier stellen wir Ihnen – ergänzend zu den Angaben in den Datenblättern – zusammenfassende Tabellen zur Verfügung.

7.1 Systemmerker

Inhalt

12167

Die zu den Systemmerkern gehörenden Merkeradressen können sich bei einer Erweiterung der Steuerungskonfiguration ändern.

► Für die Programmierung nur die Symbolnamen der Systemmerker nutzen!

→ Systemhandbuch "Know-How ecomat*mobile*"
→ Kapitel **Fehler-Codes und Diagnoseinformationen**

7.1.1 Systemmerker: CAN

12820

Systemmerker (Symbolname)	Typ	Beschreibung	
CANx_BAUDRATE	WORD	CAN-Schnittstelle x: eingestellte Baudrate in [kBaud]	
CANx_BUSOFF	BOOL	CAN-Schnittstelle x: Fehler "CAN-Bus off" ℹ Zurücksetzen des Fehler-Codes setzt auch den Merker zurück	
CANx_DOWNLOADID	BYTE	CAN-Schnittstelle x: eingestellter Download-Identifier	
CANx_ERRORCOUNTER_RX	BYTE	CAN-Schnittstelle x: Fehlerzähler Empfang ℹ Reset des Merkers ist via Schreibzugriff möglich	
CANx_ERRORCOUNTER_TX	BYTE	CAN-Schnittstelle x: Fehlerzähler Versand ℹ Reset des Merkers ist via Schreibzugriff möglich	
CANx_LASTERROR	BYTE	CAN-Schnittstelle x: Fehlernummer der letzten CAN-Übertragung:	
		0 = kein Fehler	Initial-Wert
		1 = Stuff Error	mehr als 5 gleiche Bits in Reihe auf dem Bus
		2 = Form Error	empfangenes Telegramm hatte falsches Format
		3 = Ack Error	gesendetes Telegramm wurde nicht bestätigt
		4 = Bit1 Error	außerhalb des Arbitrierungsbereichs wurde ein rezessives Bit gesendet, aber ein dominates Bit auf dem Bus gelesen
		5 = Bit0 Error	es wurde versucht, ein dominantes Bit zu senden, aber es wurde ein rezessiver Pegel gelesen ODER: während Bus-off Recovery wurde eine Sequenz von 11 rezessiven Bits gelesen
		6 = CRC Error	die Prüfsumme der empfangenen Nachricht war falsch
CANx_WARNING	BOOL	CAN-Schnittstelle x: Warnschwelle erreicht (≥ 96) ℹ Reset des Merkers ist via Schreibzugriff möglich	

CANx steht für x = 1...4 = Nummer der CAN-Schnittstelle

7.1.2 Systemmerker: SAE-J1939

12815

Systemmerker (Symbolname)	Typ	Beschreibung
J1939_RECEIVE_OVERWRITE	BOOL	Einstellung gilt nur für J1939 Daten, die nicht über ein J1939-Transportprotokoll übertragen wurden. TRUE: Alte Daten werden durch die neuen Daten überschrieben, wenn die alten Daten noch nicht aus der Funktionsbaustein-Instanz ausgelesen wurden FALSE: Neue Daten werden verworfen, solange die alten Daten noch nicht aus der Funktionsbaustein-Instanz ausgelesen wurden ⓘ Neue Daten können eintreffen, bevor die alten ausgelesen wurden, wenn der IEC-Zyklus länger ist als die Aktualisierungsfrequenz der J1939-Daten
J1939_TASK	BOOL	Mit J1939_TASK wird die Zeitanforderung beim Versenden von J1939-Telegrammen eingehalten. Sollen J1939-Telegramme mit einer Wiederholzeit ≤ 50 ms versendet werden, setzt das Laufzeitsystem automatisch J1939_TASK=TRUE. Für Anwendungen, bei denen die Zeitanforderungen ≥ SPS-Zykluszeit sind: ▶ Systemlast reduzieren mit J1939_TASK=FALSE! TRUE: J1939-Task ist aktiv (= Initialwert) Der Task wird alle 2 ms aufgerufen Der J1939-Stack sendet seine Telegramme im benötigten Zeitraster FALSE: J1939-Task ist nicht aktiv

7.1.3 Systemmerker: Fehlermerker (Standard-Seite)

12821

Systemmerker (Symbolname)	Typ	Beschreibung
ERROR	BOOL	TRUE = Sammelfehlermeldung setzen, Relais ausschalten
ERROR_BREAK_Ix (x=0...n; Wert abhängig vom Gerät, → Datenblatt)	WORD	Eingangs-Wort x: Leiterbruch-Fehler [Bit 0 für Eingang 0] ... [Bit z für Eingang z] dieser Gruppe Bit = TRUE: Fehler Bit = FALSE: kein Fehler
ERROR_BREAK_Qx (x=0...n; Wert abhängig vom Gerät, → Datenblatt)	WORD	Ausgangs-Wort x: Leiterbruch-Fehler [Bit 0 für Ausgang 0] ... [Bit z für Ausgang z] dieser Gruppe Bit = TRUE: Fehler Bit = FALSE: kein Fehler
ERROR_CONTROL_Qx (x=0...n; Wert abhängig vom Gerät, → Datenblatt)	WORD	Ausgangs-Wort x: Fehler Stromregelung Endwert kann nicht erreicht werden [Bit 0 für Ausgang 0] ... [Bit z für Ausgang z] dieser Gruppe Bit = TRUE: Fehler Bit = FALSE: kein Fehler
ERROR_CURRENT_Ix (x=0...n; Wert abhängig vom Gerät, → Datenblatt)	WORD	Eingangs-Wort x: Überstrom-Fehler nur wenn Ixx_MODE = IN_CURRENT [Bit 0 für Eingang 0] ... [Bit z für Eingang z] dieser Gruppe Bit = TRUE: Fehler Bit = FALSE: kein Fehler
ERROR_IO	BOOL	Sammelfehlermeldung Ein-/Ausgangsfehler TRUE: Fehler FALSE: kein Fehler
ERROR_POWER	BOOL	Überspannungs-Fehler für VBBs / Klemme 15: TRUE: Wert außerhalb des zulässigen Bereichs oder: Differenz (VBB15 - VBBs) > 1 V > allgemeiner Fehler > Anwendung STOP > Ausgänge = inaktiv > keine Kommunikation > Meldung "Überspannung Klemme 15" FALSE: Wert in Ordnung
ERROR_SHORT_Ix (x=0...n; Wert abhängig vom Gerät, → Datenblatt)	WORD	Eingangs-Wort x: Kurzschluss-Fehler [Bit 0 für Eingang 0] ... [Bit z für Eingang z] dieser Gruppe Bit = TRUE: Fehler Bit = FALSE: kein Fehler
ERROR_SHORT_Qx (x=0...n; Wert abhängig vom Gerät, → Datenblatt)	WORD	Ausgangs-Wort x: Kurzschluss-Fehler [Bit 0 für Ausgang 0] ... [Bit z für Ausgang z] dieser Gruppe Bit = TRUE: Fehler Bit = FALSE: kein Fehler
ERROR_TEMPERATURE	BOOL	Temperatur-Fehler TRUE: Wert außerhalb des zulässigen Bereichs > allgemeiner Fehler FALSE: Wert in Ordnung
ERROR_VBBx	BOOL	Versorgungsspannungs-Fehler an VBBx (x = o | r): TRUE: Wert außerhalb des zulässigen Bereichs > allgemeiner Fehler FALSE: Wert in Ordnung
LAST_RESET	BYTE	Grund für den letzten Reset: 00 = Reset der Anwendung 01 = PowerOn-Reset 02 = Watchdog-Reset 03 = Soft-Reset 04 = Grund nicht feststellbar

7.1.4 Systemmerker: Fehlermerker (Extended-Seite)

12823

Systemmerker (Symbolname)	Typ	Beschreibung
BOARD_LINK_ERROR	BOOL	Die Verbindung zur Extended-Seite ist... TRUE: unterbrochen die Extended-Seite ist offline Nach Unterbrechen der Verbindung ist keine automatische Neuverbindung möglich. ▶ Gerät neu starten! FALSE: in Ordnung
BOARD_LINK_WARNING	BOOL	Die Verbindung zur Extended-Seite ist... TRUE: gestört, aber noch funktionsfähig FALSE: in Ordnung
ERROR_BREAK_Ix_E (x=0...n; Wert abhängig vom Gerät, → Datenblatt)	WORD	Extended-Eingangs-Wort x: Leiterbruch-Fehler [Bit 0 für Eingang 0] ... [Bit z für Eingang z] dieser Gruppe Bit = TRUE: Fehler Bit = FALSE: kein Fehler
ERROR_BREAK_Q0_E	DWORD	1. extended-Ausgangs-Doppelwort: Leiterbruch-Fehler [Bit 0 für Ausgang 0] ... [Bit z für Ausgang z] dieser Gruppe Bit = TRUE: Fehler Bit = FALSE: kein Fehler
ERROR_CONTROL_Qx_E (x=0...n; Wert abhängig vom Gerät, → Datenblatt)	WORD	Extended-Ausgangs-Wort x: Fehler Stromregelung Endwert kann nicht erreicht werden [Bit 0 für Ausgang 0] ... [Bit z für Ausgang z] dieser Gruppe Bit = TRUE: Fehler Bit = FALSE: kein Fehler
ERROR_CURRENT_Ix_E (x=0...n; Wert abhängig vom Gerät, → Datenblatt)	WORD	Extended-Eingangs-Wort x: Überstromfehler nur wenn Ixx_MODE_E = IN_CURRENT [Bit 0 für Eingang 0] ... [Bit z für Eingang z] dieser Gruppe Bit = TRUE: Fehler Bit = FALSE: kein Fehler
ERROR_IO_E	BOOL	Sammelfehlermeldung Ein-/Ausgangsfehler Extended-Seite TRUE: Fehler FALSE: kein Fehler
ERROR_POWER_E	BOOL	Spannungs-Fehler Extended-Seite: TRUE: Wert außerhalb des zulässigen Bereichs FALSE: Wert in Ordnung
ERROR_SHORT_Ix_E (x=0...n; Wert abhängig vom Gerät, → Datenblatt)	WORD	Extended-Eingangs-Wort x: Kurzschlussfehler [Bit 0 für Eingang 0] ... [Bit z für Eingang z] dieser Gruppe Bit = TRUE: Fehler Bit = FALSE: kein Fehler
ERROR_SHORT_Qx_E (x=0...n; Wert abhängig vom Gerät, → Datenblatt)	WORD	Extended-Ausgangs-Wort x: Kurzschluss-Fehler [Bit 0 für Ausgang 0] ... [Bit z für Ausgang z] dieser Gruppe Bit = TRUE: Fehler Bit = FALSE: kein Fehler
ERROR_VBBx_E	BOOL	Versorgungsspannungs-Fehler an Extended-VBBx x = 1 | 2 | 3 | 4 TRUE: Wert außerhalb des zulässigen Bereichs FALSE: Wert in Ordnung
ERROR_VBBREL_E	BOOL	Versorgungsspannungs-Fehler an Relaisversorgung: TRUE: Wert außerhalb des zulässigen Bereichs FALSE: Wert in Ordnung

7.1.5 Systemmerker: Status-LED (Standard-Seite)

12817

Systemmerker (Symbolname)	Typ	Beschreibung
LED	WORD	LED-Farbe für "LED eingeschaltet": 0x0000 = LED_GREEN (voreingestellt) 0x0001 = LED_BLUE 0x0002 = LED_RED 0x0003 = LED_WHITE 0x0004 = LED_BLACK 0x0005 = LED_MAGENTA 0x0006 = LED_CYAN 0x0007 = LED_YELLOW
LED_X	WORD	LED-Farbe für "LED ausgeschaltet": 0x0000 = LED_GREEN 0x0001 = LED_BLUE 0x0002 = LED_RED 0x0003 = LED_WHITE 0x0004 = LED_BLACK (voreingestellt) 0x0005 = LED_MAGENTA 0x0006 = LED_CYAN 0x0007 = LED_YELLOW
LED_MODE	WORD	LED-Blinkfrequenz: 0x0000 = LED_2HZ (blinkt mit 2 Hz; voreingestellt) 0x0001 = LED_1HZ (blinkt mit 1 Hz) 0x0002 = LED_05HZ (blinkt mit 0,5 Hz) 0x0003 = LED_0HZ (leuchtet dauernd mit Wert in LED)

7.1.6 Systemmerker: Status-LED (Extended-Seite)

12824

Systemmerker (Symbolname)	Typ	Beschreibung
LED_E	WORD	LED-Farbe für "LED eingeschaltet": 0x0000 = LED_GREEN (voreingestellt) 0x0001 = LED_BLUE 0x0002 = LED_RED 0x0003 = LED_WHITE 0x0004 = LED_BLACK 0x0005 = LED_MAGENTA 0x0006 = LED_CYAN 0x0007 = LED_YELLOW
LED_X_E	WORD	LED-Farbe für "LED ausgeschaltet": 0x0000 = LED_GREEN 0x0001 = LED_BLUE 0x0002 = LED_RED 0x0003 = LED_WHITE 0x0004 = LED_BLACK (voreingestellt) 0x0005 = LED_MAGENTA 0x0006 = LED_CYAN 0x0007 = LED_YELLOW
LED_MODE_E	WORD	LED-Blinkfrequenz: 0x0000 = LED_2HZ (blinkt mit 2 Hz; voreingestellt) 0x0001 = LED_1HZ (blinkt mit 1 Hz) 0x0002 = LED_05HZ (blinkt mit 0,5 Hz) 0x0003 = LED_0HZ (leuchtet dauernd mit Wert in LED_E)

7.1.7 Systemmerker: Spannungen (Standard-Seite)

12822

Systemmerker (Symbolname)	Typ	Beschreibung
CLAMP_15_VOLTAGE	WORD	Spannung an Klemme 15 in [mV]
REF_VOLTAGE	WORD	Spannung am Referenzspannungsausgang in [mV]
REFERENCE_VOLTAGE_5	BOOL	Referenzspannungsausgang mit 5 V aktiviert
REFERENCE_VOLTAGE_10	BOOL	Referenzspannungsausgang mit 10 V aktiviert
RELAIS_VBBy y = o | r	BOOL	TRUE: Relais für VBBy aktiviert Ausgangsgruppe x wird mit Spannung versorgt (x = 1 | 2) FALSE: Relais für VBBy ausgeschaltet Ausgangsgruppe x ist spannungslos
SERIAL_MODE	BOOL	serielle Schnittstelle (RS232) für die Verwendung in der Anwendung aktivieren TRUE: RS232-Schnittstelle kann in der Anwendung verwendet werden, jedoch nicht mehr zum Programmieren, Debuggen oder Monitoren des Geräts. FALSE: RS232-Schnittstelle kann in der Anwendung nicht verwendet werden. Programmieren, Debuggen oder Monitoren des Geräts ist möglich.
SUPPLY_SWITCH	BOOL	Bit zum Abschalten der Versorgungs-Selbsthaltung VBBs. Das Rücksetzen des Merkers wird vom Laufzeitsystem nur akzeptiert, wenn die Spannung an Klemme 15 < 4 V ist, ansonsten wird der Merker wieder aktiviert. Die Trennung von VBBs erfolgt vor dem Beginn des nächsten SPS-Zyklus. Abhängig vom Ladezustand der internen Kondensatoren kann es noch eine gewisse Zeit dauern, bis das Gerät abschaltet. TRUE: Versorgung des Geräts über VBBs ist aktiv FALSE: Versorgung des Geräts über VBBs wird deaktiviert
SUPPLY_VOLTAGE	WORD	Versorgungsspannung an VBBs in [mV]
TEST	BOOL	TRUE: Test-Eingang ist aktiv: • Programmiermodus ist freigeben • Software-Download ist möglich • Zustand des Anwendungsprogramms ist abfragbar • kein Schutz der gespeicherten Software möglich FALSE: laufender Betrieb der Anwendung
VBBx_RELAIS_VOLTAGE x = o | r	WORD	Versorgungsspannung an VBBx nach Relaiskontakt in [mV]
VBBx_VOLTAGE x = o | r	WORD	Versorgungsspannung an VBBx in [mV]

7.1.8 Systemmerker: Spannungen (Extended-Seite)

12203

Systemmerker (Symbolname)	Typ	Beschreibung
RELAIS_VBBy_E y = o | r	BOOL	TRUE: Relais für VBBy aktiviert VBBo → VBB1 + VBB3 VBBr → VBB2 + VBB4 Ausgangsgruppe x wird mit Spannung versorgt (x = 1 | 2 | 3 | 4) FALSE: Relais für VBBy ausgeschaltet Ausgangsgruppe x ist spannungslos
VBBx_RELAIS_VOLTAGE_E x = 1 | 2 | 3 | 4	WORD	Versorgungsspannung an VBBx_E nach Relaiskontakt in [mV]
VBB_RELAIS_VOLTAGE_E	WORD	Versorgungsspannung für Relaisversorgung in [mV]

7.1.9 Systemmerker: 16 Eingänge und 16 Ausgänge

13119

Systemmerker (Symbolname)	Typ	Beschreibung
ANALOGxx xx = 00...15	WORD	Analog-Eingang xx: gefilterter A/D-Wandler-Rohwert (12 Bit) ohne Kalibrierung und Normierung
ANALOG_IRQxx xx = 00...07	WORD	Analogeingang Kanal xx: ungefilterter A/D-Wandler-Rohwert (12 Bit) ohne Kalibrierung, ohne Normierung Verwendung im FB **SET_INTERRUPT_I** (→ S. 123) oder **SET_INTERRUPT_XMS** (→ S. 125)
CURRENTxx xx = 00...15	WORD	PWM-Ausgang xx: gefilterte A/D-Wandler-Rohwerte (12 Bit) der Strommessung ohne Kalibrierung und Normierung
Ixx xx = 00...15	BOOL	Status am Binäreingang xx Voraussetzung: Eingang ist als Binäreingang konfiguriert (MODE = IN_DIGITAL_H oder IN_DIGITAL_L) TRUE: Spannung am Binäreingang > 70 % von VBBS FALSE: Spannung am Binäreingang < 30 % von VBBS oder: nicht als Binäreingang konfiguriert oder: falsch konfiguriert
Ixx_DFILTER xx = 00...11	DWORD	Impulseingang xx: Impulsdauer in [µs], die als Glitch ignoriert werden soll. Die Erfassung des Eingangssignals verzögert sich um die eingestellte Zeit. zugelassen = 0...100 000 µs voreingestellt = 0 µs = kein Filter
Ixx_FILTER xx = 00...15	BYTE:=4	Binär- und Analogeingang xx: Grenzfrequenz (oder Signalanstiegszeit) des Software-Tiefpass-Filters erster Ordnung 0 = 0x00 = kein Filter 1 = 0x01 = 390 Hz (1 ms) 2 = 0x02 = 145 Hz (2,5 ms) 3 = 0x03 = 68 Hz (5 ms) 4 = 0x04 = 34 Hz (10 ms) (voreingestellt) 5 = 0x05 = 17 Hz (21 ms) 6 = 0x06 = 8 Hz (42 ms) 7 = 0x07 = 4 Hz (84 ms) 8 = 0x08 = 2 Hz (169 ms) größer = → voreingestellter Wert
Ixx_MODE xx = 00...15	BYTE	Betriebsart des Eingangs Ixx → Kapitel **Mögliche Betriebsarten Ein-/Ausgänge** (→ S. 234)
Qxx xx = 00...15	BOOL	Status am Binärausgang xx: Voraussetzung: Ausgang ist als Binärausgang konfiguriert TRUE: Ausgang aktiviert FALSE: Ausgang deaktiviert (= Initialwert) oder: nicht als Binärausgang konfiguriert

Systemmerker (Symbolname)	Typ	Beschreibung
Qxx_FILTER xx = 00...15	BYTE	Ausgang xx: Grenzfrequenz des Software-Tiefpass-Filters erster Ordnung für die Strommessung nur wenn Qxx_MODE = OUT_DIGITAL_H nicht bei PWM-Betrieb 0 = 0x00 = kein Filter 1 = 0x01 = 580 Hz (0,6 ms) 2 = 0x02 = 220 Hz (1,6 ms) 3 = 0x03 = 102 Hz (3,5 ms) 4 = 0x04 = 51 Hz (7 ms) (voreingestellt) 5 = 0x05 = 25 Hz (14 ms) 6 = 0x06 = 12 Hz (28 ms) 7 = 0x07 = 6 Hz (56 ms) 8 = 0x08 = 3 Hz (112 ms) größer = → voreingestellter Wert
Qxx_MODE xx = 00...15	BYTE	Betriebsart des Ausgangs Qxx → Kapitel **Mögliche Betriebsarten Ein-/Ausgänge** (→ S. 234)

7.1.10 Systemmerker: 16 Eingänge und 32 Ausgänge (Extended-Seite)

13120

Systemmerker (Symbolname)	Typ	Beschreibung
ANALOGxx_E xx = 00...15	WORD	Extended-Analog-Eingang xx: gefilterter A/D-Wandler-Rohwert (12 Bit) ohne Kalibrierung und Normierung
CURRENTxx_E xx = 00...15	WORD	Extended-PWM-Ausgang xx: gefilterte A/D-Wandler-Rohwerte (12 Bit) der Strommessung ohne Kalibrierung und Normierung
Ixx_E xx = 00...15	BOOL	Status am Extended-Binäreingang xx_E Voraussetzung: Eingang ist als Binäreingang konfiguriert (MODE = IN_DIGITAL_H oder IN_DIGITAL_L) TRUE: Spannung am Binäreingang > 70 % von VBBS FALSE: Spannung am Binäreingang < 30 % von VBBS oder: nicht als Binäreingang konfiguriert oder: falsch konfiguriert
Ixx_DFILTER_E xx = 00...11	DWORD	Extended-Impulseingang xx: Impulsdauer in [µs], die als Glitch ignoriert werden soll. Die Erfassung des Eingangssignals verzögert sich um die eingestellte Zeit. zugelassen = 0...100 000 µs voreingestellt = 0 µs = kein Filter
Ixx_FILTER_E xx = 00...15	BYTE:=4	Extended-Binär- und Analogeingang xx_E: Grenzfrequenz (oder Signalanstiegszeit) des Software-Tiefpass-Filters erster Ordnung 0 = 0x00 = kein Filter 1 = 0x01 = 390 Hz (1 ms) 2 = 0x02 = 145 Hz (2,5 ms) 3 = 0x03 = 68 Hz (5 ms) 4 = 0x04 = 34 Hz (10 ms) (voreingestellt) 5 = 0x05 = 17 Hz (21 ms) 6 = 0x06 = 8 Hz (42 ms) 7 = 0x07 = 4 Hz (84 ms) 8 = 0x08 = 2 Hz (169 ms) größer = → voreingestellter Wert
Ixx_MODE_E xx = 00...15	BYTE	Betriebsart des Extended-Eingangs Ixx → Kapitel **Mögliche Betriebsarten Ein-/Ausgänge** (→ S. 234)
Qxx_E xx = 00...31	BOOL	Status am Extended-Binärausgang xx_E: Voraussetzung: Ausgang ist als Binärausgang konfiguriert TRUE: Ausgang aktiviert FALSE: Ausgang deaktiviert (= Initialwert) oder: nicht als Binärausgang konfiguriert

Systemmerker (Symbolname)	Typ	Beschreibung
Qxx_FILTER_E xx = 00...15	BYTE	Extended-Ausgang xx: Grenzfrequenz des Software-Tiefpass-Filters erster Ordnung für die Strommessung nur wenn Qxx_MODE_E = OUT_DIGITAL_H nicht bei PWM-Betrieb 0 = 0x00 = kein Filter 1 = 0x01 = 580 Hz (0,6 ms) 2 = 0x02 = 220 Hz (1,6 ms) 3 = 0x03 = 102 Hz (3,5 ms) 4 = 0x04 = 51 Hz (7 ms) (voreingestellt) 5 = 0x05 = 25 Hz (14 ms) 6 = 0x06 = 12 Hz (28 ms) 7 = 0x07 = 6 Hz (56 ms) 8 = 0x08 = 3 Hz (112 ms) größer = → voreingestellter Wert
Qxx_MODE_E xx = 00...31	BYTE	Betriebsart des Extended-Ausgangs Qxx_E → Kapitel **Mögliche Betriebsarten Ein-/Ausgänge** (→ S. 234)

7.2 Adressbelegung und E/A-Betriebsarten

Inhalt

1656

→ auch Datenblatt

7.2.1 Adressen / Variablen der E/As

Inhalt

2376

Eingänge: Adressen und Variablen (Standard-Seite) (16 Eingänge)

13352

Abkürzungen → Kapitel **Hinweise zur Anschlussbelegung** (→ S. 33)
Betriebsarten der Ein- und Ausgänge → Kapitel **Mögliche Betriebsarten Ein-/Ausgänge** (→ S. 234)

IEC-Adresse	E/A-Variable	Bemerkung
%IX0.0	I00	Binäreingang Kanal 0
%IX0.1	I01	Binäreingang Kanal 1
%IX0.2	I02	Binäreingang Kanal 2
%IX0.3	I03	Binäreingang Kanal 3
%IX0.4	I04	Binäreingang Kanal 4
%IX0.5	I05	Binäreingang Kanal 5
%IX0.6	I06	Binäreingang Kanal 6
%IX0.7	I07	Binäreingang Kanal 7
%IX0.8	I08	Binäreingang Kanal 8
%IX0.9	I09	Binäreingang Kanal 9
%IX0.10	I10	Binäreingang Kanal 10
%IX0.11	I11	Binäreingang Kanal 11
%IX0.12	I12	Binäreingang Kanal 12
%IX0.13	I13	Binäreingang Kanal 13
%IX0.14	I14	Binäreingang Kanal 14
%IX0.15	I15	Binäreingang Kanal 15
%IW2	ANALOG00	Analogeingang Kanal 0
%IW3	ANALOG01	Analogeingang Kanal 1
%IW4	ANALOG02	Analogeingang Kanal 2
%IW5	ANALOG03	Analogeingang Kanal 3
%IW6	ANALOG04	Analogeingang Kanal 4
%IW7	ANALOG05	Analogeingang Kanal 5
%IW8	ANALOG06	Analogeingang Kanal 6
%IW9	ANALOG07	Analogeingang Kanal 7
%IW10	ANALOG08	Analogeingang Kanal 8
%IW11	ANALOG09	Analogeingang Kanal 9
%IW12	ANALOG10	Analogeingang Kanal 10
%IW13	ANALOG11	Analogeingang Kanal 11
%IW14	ANALOG12	Analogeingang Kanal 12
%IW15	ANALOG13	Analogeingang Kanal 13
%IW16	ANALOG14	Analogeingang Kanal 14
%IW17	ANALOG15	Analogeingang Kanal 15
%IW18	CURRENT00	Ausgangsstrom (Rohwert) an Q00
%IW19	CURRENT01	Ausgangsstrom (Rohwert) an Q01
%IW20	CURRENT02	Ausgangsstrom (Rohwert) an Q02
%IW21	CURRENT03	Ausgangsstrom (Rohwert) an Q03
%IW22	CURRENT04	Ausgangsstrom (Rohwert) an Q04

IEC-Adresse	E/A-Variable	Bemerkung
%IW23	CURRENT05	Ausgangsstrom (Rohwert) an Q05
%IW24	CURRENT06	Ausgangsstrom (Rohwert) an Q06
%IW25	CURRENT07	Ausgangsstrom (Rohwert) an Q07
%IW26	CURRENT08	Ausgangsstrom (Rohwert) an Q08
%IW27	CURRENT09	Ausgangsstrom (Rohwert) an Q09
%IW28	CURRENT10	Ausgangsstrom (Rohwert) an Q10
%IW29	CURRENT11	Ausgangsstrom (Rohwert) an Q11
%IW30	CURRENT12	Ausgangsstrom (Rohwert) an Q12
%IW31	CURRENT13	Ausgangsstrom (Rohwert) an Q13
%IW32	CURRENT14	Ausgangsstrom (Rohwert) an Q14
%IW33	CURRENT15	Ausgangsstrom (Rohwert) an Q15
%IW34	SUPPLY_VOLTAGE	Versorgungsspannung an VBBs in [mV]
%IW35	CLAMP_15_VOLTAGE	Spannung Klemme 15
%IW36	VBBO_VOLTAGE	Versorgungsspannung an VBBo in [mV]
%IW37	VBBR_VOLTAGE	Versorgungsspannung an VBBr in [mV]
%IW38	VBBO_RELAIS_VOLTAGE	Versorgungsspannung VBBo nach Relaiskontakt in [mV]
%IW39	VBBR_RELAIS_VOLTAGE	Versorgungsspannung VBBr nach Relaiskontakt in [mV]
%IW40	REF_VOLTAGE	Spannung am Referenzausgang Pin 51
%IW41	ANALOG_IRQ00	Interrupt zu Analogeingang Kanal 0
%IW42	ANALOG_IRQ01	Interrupt zu Analogeingang Kanal 1
%IW43	ANALOG_IRQ02	Interrupt zu Analogeingang Kanal 2
%IW44	ANALOG_IRQ03	Interrupt zu Analogeingang Kanal 3
%IW45	ANALOG_IRQ04	Interrupt zu Analogeingang Kanal 4
%IW46	ANALOG_IRQ05	Interrupt zu Analogeingang Kanal 5
%IW47	ANALOG_IRQ06	Interrupt zu Analogeingang Kanal 6
%IW48	ANALOG_IRQ07	Interrupt zu Analogeingang Kanal 7
%MB7960	ERROR_CURRENT_I0	Fehler DWORD Überstrom
%MB7964	ERROR_SHORT_I0	Fehler DWORD Kurzschluss
%MB7968	ERROR_BREAK_I0	Fehler DWORD Leiterbruch

Eingänge: Adressen und Variablen (Extended-Seite) (16 Eingänge)

12082

Abkürzungen → Kapitel **Hinweise zur Anschlussbelegung** (→ S. 33)
Betriebsarten der Ein- und Ausgänge → Kapitel **Mögliche Betriebsarten Ein-/Ausgänge** (→ S. 234)

IEC-Adresse	E/A-Variable	Bemerkung
%IX128.0	I00_E	Binäreingang Kanal 0
%IX128.1	I01_E	Binäreingang Kanal 1
%IX128.2	I02_E	Binäreingang Kanal 2
%IX128.3	I03_E	Binäreingang Kanal 3
%IX128.4	I04_E	Binäreingang Kanal 4
%IX128.5	I05_E	Binäreingang Kanal 5
%IX128.6	I06_E	Binäreingang Kanal 6
%IX128.7	I07_E	Binäreingang Kanal 7
%IX128.8	I08_E	Binäreingang Kanal 8
%IX128.9	I09_E	Binäreingang Kanal 9
%IX128.10	I10_E	Binäreingang Kanal 10
%IX128.11	I11_E	Binäreingang Kanal 11
%IX128.12	I12_E	Binäreingang Kanal 12
%IX128.13	I13_E	Binäreingang Kanal 13
%IX128.14	I14_E	Binäreingang Kanal 14
%IX128.15	I15_E	Binäreingang Kanal 15
%IW130	ANALOG00_E	Analogeingang Kanal 0
%IW131	ANALOG01_E	Analogeingang Kanal 1
%IW132	ANALOG02_E	Analogeingang Kanal 2
%IW133	ANALOG03_E	Analogeingang Kanal 3
%IW134	ANALOG04_E	Analogeingang Kanal 4
%IW135	ANALOG05_E	Analogeingang Kanal 5
%IW136	ANALOG06_E	Analogeingang Kanal 6
%IW137	ANALOG07_E	Analogeingang Kanal 7
%IW138	ANALOG08_E	Analogeingang Kanal 8
%IW139	ANALOG09_E	Analogeingang Kanal 9
%IW140	ANALOG10_E	Analogeingang Kanal 10
%IW141	ANALOG11_E	Analogeingang Kanal 11
%IW142	ANALOG12_E	Analogeingang Kanal 12
%IW143	ANALOG13_E	Analogeingang Kanal 13
%IW144	ANALOG14_E	Analogeingang Kanal 14
%IW145	ANALOG15_E	Analogeingang Kanal 15
%IW146	CURRENT00_E	Ausgangsstrom (Rohwert) an Q00_E
%IW147	CURRENT01_E	Ausgangsstrom (Rohwert) an Q01_E
%IW148	CURRENT02_E	Ausgangsstrom (Rohwert) an Q02_E
%IW149	CURRENT03_E	Ausgangsstrom (Rohwert) an Q03_E
%IW150	CURRENT04_E	Ausgangsstrom (Rohwert) an Q04_E

IEC-Adresse	E/A-Variable	Bemerkung
%IW151	CURRENT05_E	Ausgangsstrom (Rohwert) an Q05_E
%IW152	CURRENT06_E	Ausgangsstrom (Rohwert) an Q06_E
%IW153	CURRENT07_E	Ausgangsstrom (Rohwert) an Q07_E
%IW154	CURRENT08_E	Ausgangsstrom (Rohwert) an Q08_E
%IW155	CURRENT09_E	Ausgangsstrom (Rohwert) an Q09_E
%IW156	CURRENT10_E	Ausgangsstrom (Rohwert) an Q10_E
%IW157	CURRENT11_E	Ausgangsstrom (Rohwert) an Q11_E
%IW158	CURRENT12_E	Ausgangsstrom (Rohwert) an Q12_E
%IW159	CURRENT13_E	Ausgangsstrom (Rohwert) an Q13_E
%IW160	CURRENT14_E	Ausgangsstrom (Rohwert) an Q14_E
%IW161	CURRENT15_E	Ausgangsstrom (Rohwert) an Q15_E
%IW162	VBB1_E	Versorgungsspannung an VBB1 in [mV]
%IW163	VBB2_E	Versorgungsspannung an VBB2 in [mV]
%IW164	VBB3_E	Versorgungsspannung an VBB3 in [mV]
%IW165	VBB4_E	Versorgungsspannung an VBB4 in [mV]
%IW166	VBB1_RELAIS_VOLTAGE	Versorgungsspannung VBB1 nach Relaiskontakt in [mV]
%IW167	VBB2_RELAIS_VOLTAGE	Versorgungsspannung VBB2 nach Relaiskontakt in [mV]
%IW168	VBB3_RELAIS_VOLTAGE	Versorgungsspannung VBB3 nach Relaiskontakt in [mV]
%IW169	VBB4_RELAIS_VOLTAGE	Versorgungsspannung VBB4 nach Relaiskontakt in [mV]
%IW170	VBB_RELAIS_VOLTAGE	Versorgungsspannung an VBBrel in [mV]
%MB8048	ERROR_CURRENT_I0_E	Fehler DWORD Überstrom
%MB8052	ERROR_SHORT_I0_E	Fehler DWORD Kurzschluss
%MB8056	ERROR_BREAK_I0_E	Fehler DWORD Leiterbruch

Ausgänge: Adressen und Variablen (Standard-Seite) (16 Ausgänge)

13354

Abkürzungen → Kapitel **Hinweise zur Anschlussbelegung** (→ S. 33)
Betriebsarten der Ein- und Ausgänge → Kapitel **Mögliche Betriebsarten Ein-/Ausgänge** (→ S. 234)

IEC-Adresse	E/A-Variable	Bemerkung
%QX0.0	Q00	Binärausgang / PWM-Ausgang Kanal 0
%QX0.1	Q01	Binärausgang / PWM-Ausgang Kanal 1
%QX0.2	Q02	Binärausgang / PWM-Ausgang Kanal 2
%QX0.3	Q03	Binärausgang / PWM-Ausgang Kanal 3
%QX0.4	Q04	Binärausgang / PWM-Ausgang Kanal 4
%QX0.5	Q05	Binärausgang / PWM-Ausgang Kanal 5
%QX0.6	Q06	Binärausgang / PWM-Ausgang Kanal 6
%QX0.7	Q07	Binärausgang / PWM-Ausgang Kanal 7
%QX0.8	Q08	Binärausgang / PWM-Ausgang Kanal 8
%QX0.9	Q09	Binärausgang / PWM-Ausgang Kanal 9
%QX0.10	Q10	Binärausgang / PWM-Ausgang Kanal 10
%QX0.11	Q11	Binärausgang / PWM-Ausgang Kanal 11
%QX0.12	Q12	Binärausgang / PWM-Ausgang Kanal 12
%QX0.13	Q13	Binärausgang / PWM-Ausgang Kanal 13
%QX0.14	Q14	Binärausgang / PWM-Ausgang Kanal 14
%QX0.15	Q15	Binärausgang / PWM-Ausgang Kanal 15
%QB2	REFERENCE_VOLTAGE_5	Aktivieren des Referenzspannungsausgangs mit 5 V
%QB3	REFERENCE_VOLTAGE_10	Aktivieren des Referenzspannungsausgangs mit 10 V
%QB68	I00_FILTER	Filterbyte für %IX0.0 / %IW2
%QB69	I01_FILTER	Filterbyte für %IX0.1 / %IW3
%QB70	I02_FILTER	Filterbyte für %IX0.2 / %IW4
%QB71	I03_FILTER	Filterbyte für %IX0.3 / %IW5
%QB72	I04_FILTER	Filterbyte für %IX0.4 / %IW6
%QB73	I05_FILTER	Filterbyte für %IX0.5 / %IW7
%QB74	I06_FILTER	Filterbyte für %IX0.6 / %IW8
%QB75	I07_FILTER	Filterbyte für %IX0.7 / %IW9
%QB76	I08_FILTER	Filterbyte für %IX0.8 / %IW2
%QB77	I09_FILTER	Filterbyte für %IX0.9 / %IW3
%QB78	I10_FILTER	Filterbyte für %IX0.10 / %IW4
%QB79	I11_FILTER	Filterbyte für %IX0.11 / %IW5
%QB80	I12_FILTER	Filterbyte für %IX0.12 / %IW6
%QB81	I13_FILTER	Filterbyte für %IX0.13 / %IW7
%QB82	I14_FILTER	Filterbyte für %IX0.14 / %IW8
%QB83	I15_FILTER	Filterbyte für %IX0.15 / %IW9
%QB84	Q00_FILTER	Filter-Byte für %QX0.0
%QB85	Q01_FILTER	Filter-Byte für %QX0.1
%QB86	Q02_FILTER	Filter-Byte für %QX0.2

IEC-Adresse	E/A-Variable	Bemerkung
%QB87	Q03_FILTER	Filter-Byte für %QX0.3
%QB88	Q04_FILTER	Filter-Byte für %QX0.4
%QB89	Q05_FILTER	Filter-Byte für %QX0.5
%QB90	Q06_FILTER	Filter-Byte für %QX0.6
%QB91	Q07_FILTER	Filter-Byte für %QX0.7
%QB92	Q08_FILTER	Filter-Byte für %QX0.8
%QB93	Q09_FILTER	Filter-Byte für %QX0.9
%QB94	Q10_FILTER	Filter-Byte für %QX0.10
%QB95	Q11_FILTER	Filter-Byte für %QX0.11
%QB96	Q12_FILTER	Filter-Byte für %QX0.12
%QB97	Q13_FILTER	Filter-Byte für %QX0.13
%QB98	Q14_FILTER	Filter-Byte für %QX0.14
%QB99	Q15_FILTER	Filter-Byte für %QX0.15
%QD25	I00_DFILTER	Filterwert Zähl-/Impulseingang 0
%QD26	I01_DFILTER	Filterwert Zähl-/Impulseingang 1
%QD27	I02_DFILTER	Filterwert Zähl-/Impulseingang 2
%QD28	I03_DFILTER	Filterwert Zähl-/Impulseingang 3
%QD29	I04_DFILTER	Filterwert Zähl-/Impulseingang 4
%QD30	I05_DFILTER	Filterwert Zähl-/Impulseingang 5
%QD31	I06_DFILTER	Filterwert Zähl-/Impulseingang 6
%QD32	I07_DFILTER	Filterwert Zähl-/Impulseingang 7
%QD33	I08_DFILTER	Filterwert Zähl-/Impulseingang 8
%QD34	I09_DFILTER	Filterwert Zähl-/Impulseingang 9
%QD35	I10_DFILTER	Filterwert Zähl-/Impulseingang 10
%QD36	I11_DFILTER	Filterwert Zähl-/Impulseingang 11
%MB7948	ERROR_SHORT_Q0	Fehler DWORD Kurzschluss
%MB7952	ERROR_BREAK_Q0	Fehler DWORD Leiterbruch
%MB7956	ERROR_CONTROL_Q0	Fehler DWORD Stromregelung

Ausgänge: Adressen und Variablen (Extended-Seite) (32 Ausgänge)

12084

Abkürzungen → Kapitel **Hinweise zur Anschlussbelegung** (→ S. 33)
Betriebsarten der Ein- und Ausgänge → Kapitel **Mögliche Betriebsarten Ein-/Ausgänge** (→ S. 234)

IEC-Adresse	E/A-Variable	Bemerkung
%QX128.0	Q00_E	Binärausgang / PWM-Ausgang Kanal 0
%QX128.1	Q01_E	Binärausgang / PWM-Ausgang Kanal 1
%QX128.2	Q02_E	Binärausgang / PWM-Ausgang Kanal 2
%QX128.3	Q03_E	Binärausgang / PWM-Ausgang Kanal 3
%QX128.4	Q04_E	Binärausgang / PWM-Ausgang Kanal 4
%QX128.5	Q05_E	Binärausgang / PWM-Ausgang Kanal 5
%QX128.6	Q06_E	Binärausgang / PWM-Ausgang Kanal 6
%QX128.7	Q07_E	Binärausgang / PWM-Ausgang Kanal 7
%QX128.8	Q08_E	Binärausgang / PWM-Ausgang Kanal 8
%QX128.9	Q09_E	Binärausgang / PWM-Ausgang Kanal 9
%QX128.10	Q10_E	Binärausgang / PWM-Ausgang Kanal 10
%QX128.11	Q11_E	Binärausgang / PWM-Ausgang Kanal 11
%QX128.12	Q12_E	Binärausgang / PWM-Ausgang Kanal 12
%QX128.13	Q13_E	Binärausgang / PWM-Ausgang Kanal 13
%QX128.14	Q14_E	Binärausgang / PWM-Ausgang Kanal 14
%QX128.15	Q15_E	Binärausgang / PWM-Ausgang Kanal 15
%QX128.16	Q16_E	Binärausgang / PWM-Ausgang Kanal 16
%QX128.17	Q17_E	Binärausgang / PWM-Ausgang Kanal 17
%QX128.18	Q18_E	Binärausgang / PWM-Ausgang Kanal 18
%QX128.19	Q19_E	Binärausgang / PWM-Ausgang Kanal 19
%QX128.20	Q20_E	Binärausgang / PWM-Ausgang Kanal 20
%QX128.21	Q21_E	Binärausgang / PWM-Ausgang Kanal 21
%QX128.22	Q22_E	Binärausgang / PWM-Ausgang Kanal 22
%QX128.23	Q23_E	Binärausgang / PWM-Ausgang Kanal 23
%QX128.24	Q24_E	Binärausgang / PWM-Ausgang Kanal 24
%QX128.25	Q25_E	Binärausgang / PWM-Ausgang Kanal 25
%QX128.26	Q26_E	Binärausgang / PWM-Ausgang Kanal 26
%QX128.27	Q27_E	Binärausgang / PWM-Ausgang Kanal 27
%QX128.28	Q28_E	Binärausgang / PWM-Ausgang Kanal 28
%QX128.29	Q29_E	Binärausgang / PWM-Ausgang Kanal 29
%QX128.30	Q30_E	Binärausgang / PWM-Ausgang Kanal 30
%QX128.31	Q31_E	Binärausgang / PWM-Ausgang Kanal 31
%QB356	I00_FILTER_E	Filterbyte für %IX128.0 / %IW130
%QB357	I01_FILTER_E	Filterbyte für %IX128.1 / %IW131
%QB358	I02_FILTER_E	Filterbyte für %IX128.2 / %IW132
%QB359	I03_FILTER_E	Filterbyte für %IX128.3 / %IW133
%QB360	I04_FILTER_E	Filterbyte für %IX128.4 / %IW134
%QB361	I05_FILTER_E	Filterbyte für %IX128.5 / %IW135

IEC-Adresse	E/A-Variable	Bemerkung
%QB362	I06_FILTER_E	Filterbyte für %IX128.6 / %IW136
%QB363	I07_FILTER_E	Filterbyte für %IX128.7 / %IW137
%QB364	I08_FILTER_E	Filterbyte für %IX128.8 / %IW138
%QB365	I09_FILTER_E	Filterbyte für %IX128.9 / %IW139
%QB366	I10_FILTER_E	Filterbyte für %IX128.10 / %IW140
%QB367	I11_FILTER_E	Filterbyte für %IX128.11 / %IW141
%QB368	I12_FILTER_E	Filterbyte für %IX128.12 / %IW142
%QB369	I13_FILTER_E	Filterbyte für %IX128.13 / %IW143
%QB370	I14_FILTER_E	Filterbyte für %IX128.14 / %IW144
%QB371	I15_FILTER_E	Filterbyte für %IX128.15 / %IW145
%QB372	Q00_FILTER_E	Filter-Byte für %QX128.0
%QB373	Q01_FILTER_E	Filter-Byte für %QX128.1
%QB374	Q02_FILTER_E	Filter-Byte für %QX128.2
%QB375	Q03_FILTER_E	Filter-Byte für %QX128.3
%QB376	Q04_FILTER_E	Filter-Byte für %QX128.4
%QB377	Q05_FILTER_E	Filter-Byte für %QX128.5
%QB378	Q06_FILTER_E	Filter-Byte für %QX128.6
%QB379	Q07_FILTER_E	Filter-Byte für %QX128.7
%QB380	Q08_FILTER_E	Filter-Byte für %QX128.8
%QB381	Q09_FILTER_E	Filter-Byte für %QX128.9
%QB382	Q10_FILTER_E	Filter-Byte für %QX128.10
%QB383	Q11_FILTER_E	Filter-Byte für %QX128.11
%QB384	Q12_FILTER_E	Filter-Byte für %QX128.12
%QB385	Q13_FILTER_E	Filter-Byte für %QX128.13
%QB386	Q14_FILTER_E	Filter-Byte für %QX128.14
%QB387	Q15_FILTER_E	Filter-Byte für %QX128.15
%QD97	I00_DFILTER_E	Filterwert Zähl-/Impulseingang 0
%QD98	I01_DFILTER_E	Filterwert Zähl-/Impulseingang 1
%QD99	I02_DFILTER_E	Filterwert Zähl-/Impulseingang 2
%QD100	I03_DFILTER_E	Filterwert Zähl-/Impulseingang 3
%QD101	I04_DFILTER_E	Filterwert Zähl-/Impulseingang 4
%QD102	I05_DFILTER_E	Filterwert Zähl-/Impulseingang 5
%QD103	I06_DFILTER_E	Filterwert Zähl-/Impulseingang 6
%QD104	I07_DFILTER_E	Filterwert Zähl-/Impulseingang 7
%QD105	I08_DFILTER_E	Filterwert Zähl-/Impulseingang 8
%QD106	I09_DFILTER_E	Filterwert Zähl-/Impulseingang 9
%QD107	I10_DFILTER_E	Filterwert Zähl-/Impulseingang 10
%QD108	I11_DFILTER_E	Filterwert Zähl-/Impulseingang 11
%MB8036	ERROR_SHORT_Q0_E	Fehler DWORD Kurzschluss
%MB8040	ERROR_BREAK_Q0_E	Fehler DWORD Leiterbruch
%MB8044	ERROR_CONTROL_Q0_E	Fehler DWORD Stromregelung

7.2.2 Mögliche Betriebsarten Ein-/Ausgänge

Inhalt

2386

Eingänge: Betriebsarten (Standard-Seite) (16 Eingänge)

15548

= diese Konfiguration ist voreingestellt

Eingänge	mögliche Betriebsart		einstellen mit FB	FB-Eingang	Wert	
					dez	hex
I00…I15	IN_NOMODE	Aus	INPUT_ANALOG SET_INPUT_MODE	MODE	0	00
	IN_DIGITAL_H	plus	INPUT_ANALOG SET_INPUT_MODE	MODE	1	01
	IN_DIGITAL_L	minus	INPUT_ANALOG SET_INPUT_MODE	MODE	2	02
	IN_CURRENT	0…20 000 µA	INPUT_ANALOG SET_INPUT_MODE	MODE	4	04
	IN_VOLTAGE10	0…10 000 mV	INPUT_ANALOG SET_INPUT_MODE	MODE	8	08
	IN_VOLTAGE30	0…32 000 mV	INPUT_ANALOG SET_INPUT_MODE	MODE	16	10
	IN_RATIO	0…1 000 ‰	INPUT_ANALOG SET_INPUT_MODE	MODE	32	20
	Diagnose	bei IN_DIGITAL_H	SET_INPUT_MODE	DIAGNOSTICS	TRUE	
	Frequenzmessung Periodendauermessung Phasenmessung	0…30 000 Hz	FREQUENCY FREQUENCY_PERIOD PHASE			
	Periodendauermessung	0,1...5 000 Hz	PERIOD			
	Periodendauer- und Ratiomessung	0,1...5 000 Hz	PERIOD_RATIO			
	Zähler	0…30 000 Hz	FAST_COUNT			
I00...I07	Drehgeber erfassen	0…30 000 Hz 0...5 000 Hz	INC_ENCODER INC_ENCODER_HR			

Betriebsarten mit folgendem Funktionsbaustein einstellen:

FAST_COUNT (→ S. 136)	Zählerbaustein für schnelle Eingangsimpulse
FREQUENCY (→ S. 138)	misst die Frequenz des am gewählten Kanal ankommenden Signals
FREQUENCY_PERIOD (→ S. 140)	misst die Frequenz und die Periodendauer (Zykluszeit) in [µs] am angegebenen Kanal
INC_ENCODER (→ S. 142)	Vorwärts-/Rückwärts-Zählerfunktion zur Auswertung von Drehgebern
INC_ENCODER_HR	Vorwärts-/Rückwärts-Zählerfunktion zur hochauflösenden Auswertung von Drehgebern
INPUT_ANALOG (→ S. 128)	Analoger Eingangskanal: Wahlweise Messung von... • Strom • Spannung
PERIOD (→ S. 144)	misst am angegebenen Kanal die Frequenz und die Periodendauer (Zykluszeit) in [µs]
PERIOD_RATIO (→ S. 146)	misst die Frequenz und die Periodendauer (Zykluszeit) in [µs] über die angegebenen Perioden am angegebenen Kanal. Zusätzlich wird das Puls-/Periodenverhältnis in [‰] angegeben.
PHASE (→ S. 148)	liest ein Kanalpaar mit schnellen Eingängen ein und vergleicht die Phasenlage der Signale
SET_INPUT_MODE	weist einem Eingangskanal eine Betriebsart zu

Eingänge: Betriebsarten (Extended-Seite) (16 Eingänge)

19370

= diese Konfiguration ist voreingestellt

Eingänge	mögliche Betriebsart		einstellen mit FB	FB-Eingang	Wert	
					dez	hex
I00_E...I15_E	IN_NOMODE	Aus	INPUT_ANALOG_E SET_INPUT_MODE_E	MODE	0	00
	IN_DIGITAL_H	plus	INPUT_ANALOG_E SET_INPUT_MODE_E	MODE	1	01
	IN_DIGITAL_L	minus	INPUT_ANALOG_E SET_INPUT_MODE_E	MODE	2	02
	IN_CURRENT	0…20 000 µA	INPUT_ANALOG_E SET_INPUT_MODE_E	MODE	4	04
	IN_VOLTAGE10	0…10 000 mV	INPUT_ANALOG_E SET_INPUT_MODE_E	MODE	8	08
	IN_VOLTAGE30	0…32 000 mV	INPUT_ANALOG_E SET_INPUT_MODE_E	MODE	16	10
	IN_RATIO	0…1 000 ‰	INPUT_ANALOG_E SET_INPUT_MODE_E	MODE	32	20
	Diagnose	bei IN_DIGITAL_H	SET_INPUT_MODE_E	DIAGNOSTICS	TRUE	
	Frequenzmessung Periodendauermessung Phasenmessung	0…30 000 Hz	FREQUENCY_E FREQUENCY_PERIOD_E PHASE_E			
	Periodendauermessung	0,1...5 000 Hz	PERIOD_E			
	Periodendauer- und Ratiomessung	0,1...5 000 Hz	PERIOD_RATIO_E			
	Zähler	0…30 000 Hz	FAST_COUNT_E			
I00_E...I07_E	Drehgeber erfassen	0…30 000 Hz 0...5 000 Hz	INC_ENCODER_E INC_ENCODER_HR_E			

Betriebsarten mit folgendem Funktionsbaustein einstellen:

FAST_COUNT_E	= **FAST_COUNT** (→ S. 136) für die Extended-Seite
FREQUENCY_E	= **FREQUENCY** (→ S. 138) für die Extended-Seite
FREQUENCY_PERIOD_E	= **FREQUENCY_PERIOD** (→ S. 140) für die Extended-Seite
INC_ENCODER_E	= **INC_ENCODER** (→ S. 142) für die Extended-Seite
INC_ENCODER_HR_E	= **INC_ENCODER_HR** für die Extended-Seite
INPUT_ANALOG_E	= **INPUT_ANALOG** (→ S. 128) für die Extended-Seite
PERIOD_E	= **PERIOD** (→ S. 144) für die Extended-Seite
PERIOD_RATIO_E	= **PERIOD_RATIO** (→ S. 146) für die Extended-Seite
PHASE_E	= **PHASE** (→ S. 148) für die Extended-Seite
SET_INPUT_MODE_E	= **SET_INPUT_MODE** für die Extended-Seite

Ausgänge: Betriebsarten (Standard-Seite) (16 Ausgänge)

15523

= diese Konfiguration ist voreingestellt

Ausgänge	mögliche Betriebsart		einstellen mit FB	FB-Eingang	Wert	
					dez	hex
Q00…Q15	OUT_DIGITAL_H	plus	SET_OUTPUT_MODE	MODE	1	0001
	OUT_DIGITAL_L	minus	SET_OUTPUT_MODE	MODE	2	0002
	Diagnose	bei OUT_DIGITAL_H via Strommessung	SET_OUTPUT_MODE	DIAGNOSTICS	TRUE	
	Überlastschutz	bei OUT_DIGITAL_H mit Strommessung	SET_OUTPUT_MODE	PROTECTION	TRUE	
	Strommessbereich	keine Strommessung	SET_OUTPUT_MODE	CURRENT_RANGE	0	00
		2 A / 3 A	SET_OUTPUT_MODE	CURRENT_RANGE	1	01
		4 A	SET_OUTPUT_MODE	CURRENT_RANGE	2	02

Details → Kapitel **Ausgänge Q00...Q15: zulässige Betriebsarten (→ S. 238)**

Betriebsarten mit folgendem Funktionsbaustein einstellen:

OUTPUT_BRIDGE (→ S. 151)	H-Brücke an einem PWM-Kanalpaar
OUTPUT_CURRENT_CONTROL (→ S. 155)	Stromregler für einen PWMi-Ausgangskanal
PWM1000 (→ S. 158)	initialisiert und parametriert einen PWM-fähigen Ausgangskanal das Puls-Pausen-Verhältnis kann in 1 ‰-Schritten angegeben werden
SET_OUTPUT_MODE	setzt die Betriebsart des gewählten Ausgangskanals

Ausgänge Q00...Q15: zulässige Betriebsarten

19296

Betriebsart		Q00	Q01	Q02	Q03	Q04	Q05	Q06	Q07
OUT_DIGITAL_H	plus	X	X	X	X	X	X	X	X
OUT_DIGITAL_L	minus	--	X	--	X	--	--	--	--
Diagnose	bei OUT_DIGITAL_H via Strommessung	X	X	X	X	X	X	X	X
Überlastschutz	bei OUT_DIGITAL_H mit Strommessung	X	X	X	X	X	X	X	X
Strommessbereich	2 A	X	X	X	X	X	X	X	X
	4 A	X	X	X	X	--	--	--	--
PWM		X	X	X	X	X	X	X	X
PWMi		X	X	X	X	X	X	X	X
H-Brücke		--	X	--	X	--	--	--	--

Betriebsart		Q08	Q09	Q10	Q11	Q12	Q13	Q14	Q15
OUT_DIGITAL_H	plus	X	X	X	X	X	X	X	X
OUT_DIGITAL_L	minus	--	X	--	X	--	--	--	--
Diagnose	bei OUT_DIGITAL_H via Strommessung	X	X	X	X	X	X	X	X
Überlastschutz	bei OUT_DIGITAL_H mit Strommessung	X	X	X	X	X	X	X	X
Strommessbereich	2 A	X	X	X	X	X	X	X	X
	4 A	X	X	X	X	--	--	--	--
PWM		X	X	X	X	X	X	X	X
PWMi		X	X	X	X	X	X	X	X
H-Brücke		--	X	--	X	--	--	--	--

Ausgänge: Betriebsarten (Extended-Seite) (32 Ausgänge)

19297

= diese Konfiguration ist voreingestellt

Ausgänge	mögliche Betriebsart		einstellen mit FB	FB-Eingang	Wert	
					dez	hex
Q00_E ...Q15_E	OUT_DIGITAL_H	plus	SET_OUTPUT_MODE_E	MODE	1	0001
	OUT_DIGITAL_L	minus	SET_OUTPUT_MODE_E	MODE	2	0002
	Diagnose	bei OUT_DIGITAL_H via Strommessung	SET_OUTPUT_MODE_E	DIAGNOSTICS	TRUE	
	Überlastschutz	bei OUT_DIGITAL_H mit Strommessung	SET_OUTPUT_MODE_E	PROTECTION	TRUE	
	Strommessbereich	keine Strommessung	SET_OUTPUT_MODE_E	CURRENT_RANGE	0	00
		2 A	SET_OUTPUT_MODE_E	CURRENT_RANGE	1	01
		4 A	SET_OUTPUT_MODE_E	CURRENT_RANGE	2	02
Q16_E ...Q31_E	OUT_DIGITAL_H	plus	SET_OUTPUT_MODE_E	MODE	1	0001
	Diagnose	bei OUT_DIGITAL_H	SET_OUTPUT_MODE_E	DIAGNOSTICS	FALSE	
	Strommessbereich	2 A	SET_OUTPUT_MODE_E	CURRENT_RANGE	1	01

Details → Kapitel **Ausgänge Q00_E...Q31_E: zulässige Betriebsarten (→ S. 240)**

Betriebsarten mit folgendem Funktionsbaustein einstellen:

OUTPUT_BRIDGE_E	= **OUTPUT_BRIDGE** (→ S. 151) für die Extended-Seite
OUTPUT_CURRENT_CONTROL_E	= **OUTPUT_CURRENT_CONTROL** (→ S. 155) für die Extended-Seite
PWM1000_E	= **PWM1000** (→ S. 158) für die Extended-Seite
SET_OUTPUT_MODE_E	= **SET_OUTPUT_MODE** für die Extended-Seite

Ausgänge Q00_E...Q31_E: zulässige Betriebsarten

19904

Betriebsart		Q00_E	Q01_E	Q02_E	Q03_E	Q04_E	Q05_E	Q06_E	Q07_E
OUT_DIGITAL_H	plus	X	X	X	X	X	X	X	X
OUT_DIGITAL_L	minus	--	X	--	X	--	--	--	--
Diagnose	bei OUT_DIGITAL_H via Strommessung	X	X	X	X	X	X	X	X
Überlastschutz	bei OUT_DIGITAL_H mit Strommessung	X	X	X	X	X	X	X	X
Strommessbereich	2 A	X	X	X	X	X	X	X	X
	4 A	X	X	X	X	--	--	--	--
PWM		X	X	X	X	X	X	X	X
PWMi		X	X	X	X	X	X	X	X
H-Brücke		--	X	--	X	--	--	--	--

Betriebsart		Q08_E	Q09_E	Q10_E	Q11_E	Q12_E	Q13_E	Q14_E	Q15_E
OUT_DIGITAL_H	plus	X	X	X	X	X	X	X	X
OUT_DIGITAL_L	minus	--	X	--	X	--	--	--	--
Diagnose	bei OUT_DIGITAL_H via Strommessung	X	X	X	X	X	X	X	X
Überlastschutz	bei OUT_DIGITAL_H mit Strommessung	X	X	X	X	X	X	X	X
Strommessbereich	2 A	X	X	X	X	X	X	X	X
	4 A	X	X	X	X	--	--	--	--
PWM		X	X	X	X	X	X	X	X
PWMi		X	X	X	X	X	X	X	X
H-Brücke		--	X	--	X	--	--	--	--

Betriebsart		Q16_E	Q17_E	Q18_E	Q19_E	Q20_E	Q21_E	Q22_E	Q23_E
OUT_DIGITAL_H	plus	X	X	X	X	X	X	X	X
Diagnose	bei OUT_DIGITAL_H via Spannungsmessung	X	X	X	X	X	X	X	X

Betriebsart		Q24_E	Q25_E	Q26_E	Q27_E	Q28_E	Q29_E	Q30_E	Q31_E
OUT_DIGITAL_H	plus	X	X	X	X	X	X	X	X
Diagnose	bei OUT_DIGITAL_H via Spannungsmessung	X	X	X	X	X	X	X	X

7.3 Fehler-Tabellen

Inhalt

19606

7.3.1 Fehlermerker

19608

→ Kapitel **Systemmerker** (→ S. 214)

7.3.2 Fehler: CAN / CANopen

19610
19604

→ Systemhandbuch "Know-How ecomat*mobile*"
→ Kapitel **CAN / CANopen: Fehler und Fehlerbehandlung**

EMCY-Codes: CANx

13094

Die Angaben für CANx gelten für jede der CAN-Schnittstellen.

EMCY-Code Objekt 0x1003		Objekt 0x1001	herstellerspezifische Informationen					
Byte 0 [hex]	Byte 1 [hex]	Byte 2 [hex]	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Beschreibung
00	80	11	---	---	---	---	---	CANx Monitoring SYNC-Error (nur Slave)
00	81	11	---	---	---	---	---	CANx Warngrenze (> 96)
10	81	11	---	---	---	---	---	CANx Empfangspuffer Überlauf
11	81	11	---	---	---	---	---	CANx Sendepuffer Überlauf
30	81	11	---	---	---	---	---	CANx Guard-/Heartbeat-Error (nur Slave)

EMCY-Codes: E/As, System (Standard-Seite)

2668

Die folgenden EMCY-Meldungen werden in folgenden Fällen automatisch versendet:
- als CANopen-Master: wenn **CANx_MASTER_EMCY_HANDLER** (→ S. 83) zyklisch aufgerufen wird
- als CANopen-Slave: wenn **CANx_SLAVE_EMCY_HANDLER** (→ S. 93) zyklisch aufgerufen wird

EMCY-Code Objekt 0x1003		Objekt 0x1001	herstellerspezifische Informationen					
Byte 0 [hex]	Byte 1 [hex]	Byte 2 [hex]	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Beschreibung
00	21	03	I07...I00	I15...I08				Leiterbruch Eingänge
08	21	03	I07...I00	I15...I08				Kurzschluss Eingänge
10	21	03	I07...I00	I15...I08				Überstrom 0…20 mA
00	23	03	Q07…Q00	Q15…Q08				Leiterbruch Ausgänge
08	23	03	Q07…Q00	Q15…Q08				Kurzschluss Ausgänge
00	31	05						Versorgungsspannung VBBs
00	33	05						Ausgangsspannung VBBo
08	33	05						Ausgangsspannung VBBr
00	42	09						Übertemperatur

EMCY-Codes: E/As, System (Extended-Seite)

13095

Die folgenden EMCY-Meldungen werden in folgenden Fällen automatisch versendet:
- als CANopen-Master: wenn **CANx_MASTER_EMCY_HANDLER** (→ S. 83) zyklisch aufgerufen wird
- als CANopen-Slave: wenn **CANx_SLAVE_EMCY_HANDLER** (→ S. 93) zyklisch aufgerufen wird

EMCY-Code Objekt 0x1003		Objekt 0x1001	herstellerspezifische Informationen					
Byte 0 [hex]	Byte 1 [hex]	Byte 2 [hex]	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Beschreibung
01	21	03	I07_E ...I00_E	I15_E ...I08_E				Leiterbruch Eingänge
09	21	03	I07_E ...I00_E	I15_E ...I08_E				Kurzschluss Eingänge
11	21	03	I07_E ...I00_E	I15_E ...I08_E				Überstrom 0…20 mA
01	23	03	Q07_E …Q00_E	Q15_E …Q08_E	Q23_E …Q16_E	Q31_E …Q24_E		Leiterbruch Ausgänge
09	23	03	Q07_E …Q00_E	Q15_E …Q08_E	Q23_E …Q16_E	Q31_E …Q24_E		Kurzschluss Ausgänge
10	33	05						Ausgangsspannung VBB1
11	33	05						Ausgangsspannung VBB2
12	33	05						Ausgangsspannung VBB3
13	33	05						Ausgangsspannung VBB4
18	33	05						Versorgung Relais VBBrel

8 Begriffe und Abkürzungen

A

Adresse

Das ist der „Name“ des Teilnehmers im Bus. Alle Teilnehmer benötigen eine unverwechselbare, eindeutige Adresse, damit der Austausch der Signale fehlerfrei funktioniert.

Anleitung

Übergeordnetes Wort für einen der folgenden Begriffe:
Montageanleitung, Datenblatt, Benutzerinformation, Bedienungsanleitung, Gerätehandbuch, Installationsanleitung, Onlinehilfe, Systemhandbuch, Programmierhandbuch, usw.

Anwendungsprogramm

Software, die speziell für die Anwendung vom Hersteller in die Maschine programmiert wird. Die Software enthält üblicherweise logische Sequenzen, Grenzwerte und Ausdrücke zum Steuern der entsprechenden Ein- und Ausgänge, Berechnungen und Entscheidungen.

Architektur

Spezifische Konfiguration von Hardware- und/oder Software-Elementen in einem System.

B

Baud

Baud, Abk.: Bd = Maßeinheit für die Geschwindigkeit bei der Datenübertragung. Baud ist nicht zu verwechseln mit "bits per second" (bps, Bit/s). Baud gibt zwar die Anzahl von Zustandsänderungen (Schritte, Takte) pro Sekunde auf einer Übertragungsstrecke an. Aber es ist nicht festgelegt, wie viele Bits pro Schritt übertragen werden. Der Name Baud geht auf den französischen Erfinder J. M. Baudot zurück, dessen Code für Telexgeräte verwendet wurde.
1 MBd = 1024 x 1024 Bd = 1 048 576 Bd

Bestimmungsgemäße Verwendung

Das ist die Verwendung eines Produkts in Übereinstimmung mit den in der Anleitung bereitgestellten Informationen.

Bootloader

Im Auslieferungszustand enthalten ecomat*mobile*-Controller nur den Bootloader.
Der Bootloader ist ein Startprogramm, mit dem das Laufzeitsystem und das Anwendungsprogramm auf dem Gerät nachgeladen werden können.
Der Bootloader enthält Grundroutinen...
- zur Kommunikation der Hardware-Module untereinander,
- zum Nachladen des Laufzeitsystems.

Der Bootloader ist das erste Software-Modul, das im Gerät gespeichert sein muss.

Bus

Serielle Datenübertragung mehrerer Teilnehmer an derselben Leitung.

C

CAN

CAN = **C**ontroller **A**rea **N**etwork
CAN gilt als Feldbussystem für größere Datenmengen, das prioritätengesteuert arbeitet. Es gibt mehrere höhere Protokolle, die auf CAN aufsetzen, z. B. 'CANopen' oder 'J1939'.

CAN-Stack

CAN-Stack = Software-Komponente, die sich um die Verarbeitung von CAN-Telegramme kümmert.

CiA

CiA = CAN in Automation e.V.
Anwender- und Herstellerorganisation in Erlangen, Deutschland. Definitions- und Kontrollorgan für das CANopen-Protokoll.
Homepage → www.can-cia.org

CiA DS 304

DS = **D**raft **S**tandard
CANopen-Geräteprofil für sichere Kommunikation

CiA DS 401

DS = **D**raft **S**tandard
CANopen-Geräteprofil für digitale und analoge E/A-Baugruppen

CiA DS 402

DS = **D**raft **S**tandard
CANopen-Geräteprofil für Antriebe

CiA DS 403

DS = **D**raft **S**tandard
CANopen-Geräteprofil für Bediengeräte

CiA DS 404

DS = **D**raft **S**tandard
CANopen-Geräteprofil für Messtechnik und Regler

CiA DS 405

DS = **D**raft **S**tandard
CANopen-Spezifikation der Schnittstelle zu programmierbaren Steuerungen (IEC 61131-3)

CiA DS 406

DS = **D**raft **S**tandard
CANopen-Geräteprofil für Drehgeber / Encoder

CiA DS 407

DS = **D**raft **S**tandard
CANopen-Anwendungsprofil für den öffentlichen Nahverkehr

COB-ID

COB = **C**ommunication **Ob**ject = Kommunikationsobjekt
ID = **Id**entifier = Kennung
ID eines CANopen-Kommunikationsobjekts
Entspricht dem Identifier der CAN-Nachricht, mit der das Kommunikationsobjekt über den CAN-Bus gesendet wird.

CODESYS

CODESYS® ist eingetragene Marke der 3S – Smart Software Solutions GmbH, Deutschland.
'CODESYS for Automation Alliance™' vereinigt Firmen der Automatisierungsindustrie, deren Hardware-Geräte alle mit dem weit verbreiteten IEC 61131-3 Entwicklungswerkzeug CODESYS® programmiert werden.
Homepage → www.codesys.com

CSV-Datei

CSV = **C**omma **S**eparated **V**alues (auch: **C**haracter **S**eparated **V**alues)
Eine CSV-Datei ist eine Textdatei zur Speicherung oder zum Austausch einfach strukturierter Daten.
Die Dateinamen-Erweiterung lautet `.csv`.

Beispiel: Quell-Tabelle mit Zahlenwerten:

Wert 1.0	Wert 1.1	Wert 1.2	Wert 1.3
Wert 2.0	Wert 2.1	Wert 2.2	Wert 2.3
Wert 3.0	Wert 3.1	Wert 3.2	Wert 3.3

Daraus entsteht folgende CSV-Datei:

```
Wert 1.0;Wert 1.1;Wert 1.2;Wert 1.3
Wert 2.0;Wert 2.1;Wert 2.2;Wert 2.3
Wert 3.0;Wert 3.1;Wert 3.2;Wert 3.3
```

D

Datentyp

Abhängig vom Datentyp können unterschiedlich große Werte gespeichert werden.

Datentyp	min. Wert	max. Wert	Größe im Speicher
BOOL	FALSE	TRUE	8 Bit = 1 Byte
BYTE	0	255	8 Bit = 1 Byte
WORD	0	65 535	16 Bit = 2 Bytes
DWORD	0	4 294 967 295	32 Bit = 4 Bytes
SINT	-128	127	8 Bit = 1 Byte
USINT	0	255	8 Bit = 1 Byte
INT	-32 768	32 767	16 Bit = 2 Bytes
UINT	0	65 535	16 Bit = 2 Bytes
DINT	-2 147 483 648	2 147 483 647	32 Bit = 4 Bytes
UDINT	0	4 294 967 295	32 Bit = 4 Bytes
REAL	$-3{,}402823466 \cdot 10^{38}$	$3{,}402823466 \cdot 10^{38}$	32 Bit = 4 Bytes
ULINT	0	18 446 744 073 709 551 615	64 Bit = 8 Bytes
STRING			number of char. + 1

DC

Direct **C**urrent = Gleichstrom

Diagnose

Bei der Diagnose wird der "Gesundheitszustand" des Gerätes geprüft. Es soll festgestellt werden, ob und gegebenenfalls welche →Fehler im Gerät vorhanden sind.

Je nach Gerät können auch die Ein- und Ausgänge auf einwandfreie Funktion überwacht werden:
- Drahtbruch,
- Kurzschluss,
- Wert außerhalb des Sollbereichs.

Zur Diagnose können Konfigurations-Dateien herangezogen werden, die während des "normalen" Betriebs des Gerätes erzeugt wurden.
Der korrekte Start der Systemkomponenten wird während der Initialisierungs- und Startphase überwacht.
Zur weiteren Diagnose können auch Selbsttests durchgeführt werden.

Dither

to dither (engl.) = schwanken / zittern.
Dither ist ein Bestandteil der →PWM-Signale zum Ansteuern von Hydraulik-Ventilen. Für die elektromagnetischen Antriebe von Hydraulik-Ventilen hat sich herausgestellt, dass sich die Ventile viel besser regeln lassen, wenn das Steuersignal (PWM-Impulse) mit einer bestimmten Frequenz der PWM-Frequenz überlagert wird. Diese Dither-Frequenz muss ein ganzzahliger Teil der PWM-Frequenz sein.

DLC

Data **L**ength **C**ode = bei CANopen die Anzahl der Daten-Bytes in einer Nachricht.
Für →SDO: DLC = 8

DRAM

DRAM = **D**ynamic **R**andom **A**ccess **M**emory.
Technologie für einen elektronischen Speicherbaustein mit wahlfreiem Zugriff (Random Access Memory, RAM). Das speichernde Element ist dabei ein Kondensator, der entweder geladen oder entladen ist. Über einen Schalttransistor wird er zugänglich und entweder ausgelesen oder mit neuem Inhalt beschrieben. Der Speicherinhalt ist flüchtig: die gespeicherte Information geht bei fehlender Betriebsspannung oder zu später Wiederauffrischung verloren.

DTC

DTC = **D**iagnostic **T**rouble **C**ode = Fehler-Code
Beim Protokoll J1939 werden Störungen und Fehler über zugeordnete Nummern – den DTCs – verwaltet und gemeldet.

E

ECU

(1) **E**lectronic **C**ontrol **U**nit = Steuergerät oder Mikrocontroller
(2) **E**ngine **C**ontrol **U**nit = Steuergerät eines Motors

EDS-Datei

EDS = **E**lectronic **D**ata **S**heet = elektronisch hinterlegtes Datenblatt, z.B. für:
- Datei für das Objektverzeichnis im CANopen-Master,
- CANopen-Gerätebeschreibungen.

Via EDS können vereinfacht Geräte und Programme ihre Spezifikationen austauschen und gegenseitig berücksichtigen.

Embedded Software

System-Software, Grundprogramm im Gerät, praktisch das →Laufzeitsystem.
Die Firmware stellt die Verbindung her zwischen der Hardware des Gerätes und dem Anwendungsprogramm. Die Firmware wird vom Hersteller der Steuerung als Teil des Systems geliefert und kann vom Anwender nicht verändert werden.

EMCY

Abkürzung für Emergency (engl.) = Notfall
Nachricht im CANopen-Protokoll, mit der Fehler gemeldet werden.

EMV

EMV = **E**lektro-**M**agnetische **V**erträglichkeit.
Gemäß der EG-Richtlinie (2004/108/EG) zur elektromagnetischen Verträglichkeit (kurz EMV-Richtlinie) werden Anforderungen an die Fähigkeit von elektrischen und elektronischen Apparaten, Anlagen, Systemen oder Bauteilen gestellt, in der vorhandenen elektromagnetischen Umwelt zufriedenstellend zu arbeiten. Die Geräte dürfen ihre Umgebung nicht stören und dürfen sich von äußerlichen elektromagnetischen Störungen nicht ungünstig beeinflussen lassen.

Ethernet

Ethernet ist eine weit verbreitete, herstellerneutrale Netzwerktechnologie, mit der Daten mit einer Geschwindigkeit von 10 bis 10 000 Millionen Bit pro Sekunde (Mbps) übertragen werden können. Ethernet gehört zu der Familie der sogenannten „bestmöglichen Datenübermittlung“ auf einem nicht exklusiven Übertragungsmedium. 1972 entwickelt, wurde das Konzept 1985 als IEEE 802.3 spezifiziert.

EUC

EUC = **E**quipment **U**nder **C**ontrol (kontrollierte Einrichtung).
EUC ist eine Einrichtung, Maschine, Gerät oder Anlage, verwendet zur Fertigung, Stoffumformung, zum Transport, zu medizinischen oder anderen Tätigkeiten (→ IEC 61508-4, Abschnitt 3.2.3). Das EUC umfasst also alle Einrichtungen, Maschinen, Geräte oder Anlagen, die →Gefährdungen verursachen können und für die sicherheitsgerichtete Systeme erforderlich sind.
Falls eine vernünftigerweise vorhersehbare Aktivität oder Inaktivität zu durch das EUC verursachten Gefährdungen mit unvertretbarem Risiko führt, sind Sicherheitsfunktionen erforderlich, um einen sicheren Zustand für das EUC zu erreichen oder aufrecht zu erhalten. Diese Sicherheitsfunktionen werden durch ein oder mehrere sicherheitsgerichtete Systeme ausgeführt.

Fehlanwendung

Das ist die Verwendung eines Produkts in einer Weise, die vom Konstrukteur nicht vorgesehen ist. Eine Fehlanwendung führt meist zu einer →Gefährdung von Personen oder Sachen.
Vor vernünftigerweise, vorhersehbaren Fehlanwendungen muss der Hersteller des Produkts in seinen Benutzerinformationen warnen.

FiFo

FIFO (**F**irst **I**n, **F**irst **O**ut) = Arbeitsweise des Stapelspeichers: Das Datenpaket, das zuerst in den Stapelspeicher geschrieben wurde, wird auch als erstes gelesen. Pro Identifier steht ein solcher Zwischenspeicher (als Warteschlange) zur Verfügung.

Flash-Speicher

Flash-ROM (oder Flash-EPROM oder Flash-Memory) kombiniert die Vorteile von Halbleiterspeicher und Festplatten. Die Daten werden allerdings wie bei einer Festplatte blockweise in Datenblöcken zu 64, 128, 256, 1024, ... Byte zugleich geschrieben und gelöscht.

Vorteile von Flash-Speicher

- Die gespeicherten Daten bleiben auch bei fehlender Versorgungsspannung erhalten.
- Wegen fehlender beweglicher Teile ist Flash geräuschlos, unempfindlich gegen Erschütterungen und magnetische Felder.

Nachteile von Flash-Speicher

- Begrenzte Zahl von Schreib- bzw. Löschvorgängen, die eine Speicherzelle vertragen kann:
  - Multi-Level-Cells: typ. 10 000 Zyklen
  - Single-Level-Cells: typ. 100 000 Zyklen
- Da ein Schreibvorgang Speicherblöcke zwischen 16 und 128 kByte gleichzeitig beschreibt, werden auch Speicherzellen beansprucht, die gar keiner Veränderung bedürfen.

FRAM

FRAM, oder auch FeRAM, bedeutet **Fe**rroelectric **R**andom **A**ccess **M**emory. Der Speicher- und Löschvorgang erfolgt durch eine Polarisationsänderung in einer ferroelektrischen Schicht.
Vorteile von FRAM gegenüber herkömmlichen Festwertspeichern:
- nicht flüchtig,
- kompatibel zu gängigen EEPROMs, jedoch:
- Zugriffszeit ca. 100 ns,
- fast unbegrenzt viele Zugriffszyklen möglich.

Heartbeat

Heartbeat (engl.) = Herzschlag.
Die Teilnehmer senden regelmäßig kurze Signale. So können die anderen Teilnehmer prüfen, ob ein Teilnehmer ausgefallen ist.

HMI

HMI = **H**uman **M**achine **I**nterface = Mensch-Maschine-Schnittstelle

ID – Identifier

ID = **Id**entifier = Kennung

Name zur Unterscheidung der an einem System angeschlossenen Geräte / Teilnehmer oder der zwischen den Teilnehmern ausgetauschten Nachrichtenpakete.

IEC 61131

Norm: Grundlagen Speicherprogrammierbarer Steuerungen
- Teil 1: Allgemeine Informationen
- Teil 2: Betriebsmittelanforderungen und Prüfungen
- Teil 3: Programmiersprachen
- Teil 5: Kommunikation
- Teil 7: Fuzzy-Control-Programmierung

IEC-User-Zyklus

IEC-User-Zyklus = SPS-Zyklus im CODESYS-Anwendungsprogramm.

IP-Adresse

IP = **I**nternet **P**rotocol = Internet-Protokoll.
Die IP-Adresse ist eine Nummer, die zur eindeutigen Identifizierung eines Internet-Teilnehmers notwendig ist. Zur besseren Übersicht wird die Nummer in 4 dezimalen Werten geschrieben, z. B. 127.215.205.156.

ISO 11898

Norm: Straßenfahrzeuge – CAN-Protokoll
- Teil 1: Bit-Übertragungsschicht und physikalische Zeichenabgabe
- Teil 2: High-speed medium access unit
- Teil 3: Fehlertolerante Schnittstelle für niedrige Geschwindigkeiten
- Teil 4: Zeitgesteuerte Kommunikation
- Teil 5: High-speed medium access unit with low-power mode

ISO 11992

Norm: Straßenfahrzeuge – Austausch von digitalen Informationen über elektrische Verbindungen zwischen Zugfahrzeugen und Anhängefahrzeugen
- Teil 1: Bit-Übertragungsschicht und Sicherungsschicht
- Teil 2: Anwendungsschicht für die Bremsausrüstung
- Teil 3: Anwendungsschicht für andere als die Bremsausrüstung
- Teil 4: Diagnose

ISO 16845

Norm: Straßenfahrzeuge – Steuergerätenetz (CAN) – Prüfplan zu Konformität

J1939

→ SAE J1939

Klemme 15

Klemme 15 ist in Fahrzeugen die vom Zündschloss geschaltete Plusleitung.

Laufzeitsystem

Grundprogramm im Gerät, stellt die Verbindung her zwischen der Hardware des Gerätes und dem Anwendungsprogramm.
→ Kapitel **Software-Module für das Gerät** (→ S. 39)

LED

LED = **L**ight **E**mitting **D**iode = Licht aussendende Diode.
Leuchtdiode, auch Luminiszenzdiode, ein elektronisches Element mit hoher, farbiger Leuchtkraft auf kleinem Volumen bei vernachlässigbarer Verlustleistung.

Link

Ein Link ist ein Querverweis zu einer anderen Stelle im Dokument oder auf ein externes Dokument.

LSB

Least **S**ignificant **B**it/Byte = Niederwertigstes Bit/Byte in einer Reihe von Bit/Bytes.

MAC-ID

MAC = **M**anufacturer‘s **A**ddress **C**ode
= Hersteller-Seriennummer.
→ID = **Id**entifier = Kennung
Jede Netzwerkkarte verfügt über eine so genannte MAC-Adresse, ein unverwechselbarer, auf der ganzen Welt einzigartiger Zahlencode – quasi eine Art Seriennummer. So eine MAC-Adresse ist eine Aneinanderreihung von 6 Hexadezimalzahlen, etwa "00-0C-6E-D0-02-3F".

Master

Wickelt die komplette Organisation auf dem →Bus ab. Der Master entscheidet über den zeitlichen Buszugriff und fragt die →Slaves zyklisch ab.

MMI

MMI = **M**ensch-**M**aschine-**I**nterface
→ HMI (→ S. 248)

MRAM

MRAM = **M**agnetoresistive **R**andom **A**ccess **M**emory
Die Informationen werden mit magnetischen Ladungselementen gespeichert. Dabei wird die Eigenschaft bestimmter Materialien ausgenutzt, die ihren elektrischen Widerstand unter dem Einfluss magnetischer Felder ändern.
Vorteile von MRAM gegenüber herkömmlichen Festwertspeichern:
- nicht flüchtig (wie FRAM), jedoch:
- Zugriffszeit nur ca. 35 ns,
- unbegrenzt viele Zugriffszyklen möglich.

MSB

Most **S**ignificant **B**it/Byte = Höchstwertiges Bit/Byte einer Reihe von Bits/Bytes.

NMT

NMT = **N**etwork **M**anagement = Netzwerk-Verwaltung (hier: im CANopen-Protokoll).
Der NMT-Master steuert die Betriebszustände der NMT-Slaves.

Node

Node (engl.) = Knoten. Damit ist ein Teilnehmer im Netzwerk gemeint.

Node Guarding

Node (engl.) = Knoten, hier: Netzwerkteilnehmer
Guarding (engl.) = Schutz
Parametrierbare, zyklische Überwachung von jedem entsprechend konfigurierten →Slave. Der →Master prüft, ob die Slaves rechtzeitig antworten. Die Slaves prüfen, ob der Master regelmäßig anfragt. Somit können ausgefallene Netzwerkteilnehmer schnell erkannt und gemeldet werden.

Obj / Objekt

Oberbegriff für austauschbare Daten / Botschaften innerhalb des CANopen-Netzwerks.

Objektverzeichnis

Das **Ob**jekt**v**erzeichnis OBV enthält alle CANopen-Kommunikationsparameter eines Gerätes, sowie gerätespezifische Parameter und Daten.

OBV

Das **Ob**jekt**v**erzeichnis OBV enthält alle CANopen-Kommunikationsparameter eines Gerätes, sowie gerätespezifische Parameter und Daten.

OPC

OPC = **O**LE for **P**rocess **C**ontrol = Objektverknüpfung und -einbettung für Prozesssteuerung
Standardisierte Software-Schnittstelle zur herstellerunabhängigen Kommunikation in der Automatisierungstechnik
OPC-Client (z.B. Gerät zum Parametrieren oder Programmieren) meldet sich nach dem Anschließen am OPC-Server (z.B. Automatisierungsgerät) automatisch bei diesem an und kommuniziert mit ihm.

operational

Operational (engl.) = betriebsbereit
Betriebszustand eines CANopen-Teilnehmers. In diesem Modus können →SDOs, →NMT-Kommandos und →PDOs übertragen werden.

PC-Karte

→ PCMCIA-Karte

PCMCIA-Karte

PCMCIA = Personal Computer Memory Card International Association, ein Standard für Erweiterungskarten mobiler Computer.
Seit der Einführung des Cardbus-Standards 1995 werden PCMCIA-Karten auch als PC-Karte (engl.: PC Card) bezeichnet.

PDM

PDM = **P**rocess and **D**ialog **M**odule = **P**rozess- und **D**ialog-**M**onitor.
Gerät zur Kommunikation des Bedieners mit der Maschine / Anlage.

PDO

PDO = **P**rocess **D**ata **O**bject = Nachrichten-Objekt mit Prozessdaten.
Die zeitkritischen Prozessdaten werden mit Hilfe der "Process Data Objects" (PDOs) übertragen. Die PDOs können beliebig zwischen den einzelnen Knoten ausgetauscht werden (PDO-Linking).
Zusätzlich wird festgelegt, ob der Datenaustausch ereignisgesteuert (asynchron) oder synchronisiert erfolgen soll. Je nach der Art der zu übertragenden Daten kann die richtige Wahl der Übertragungsart zu einer erheblichen Entlastung des →CAN-Bus führen.
Dem Protokoll entsprechend, sind diese Dienste nicht bestätigte Dienste: es gibt keine Kontrolle, ob die Nachricht auch beim Empfänger ankommt. Netzwerkvariablen-Austausch entspricht einer "1-zu-n-Verbindung" (1 Sender zu n Empfängern).

PDU

PDU = **P**rotocol **D**ata **U**nit = Protokoll-Daten-Einheit.
Die PDU ist ein Begriff aus dem →CAN-Protokoll →SAE J1939. Sie bezeichnet einen Bestandteil der Zieladresse (PDU Format 1, verbindungsorientiert) oder der Group Extension (PDU Format 2, nachrichtenorientiert).

PES

Programable **e**lectronic **s**ystem = Programmierbares elektronisches System ...
- zur Steuerung, zum Schutz oder zur Überwachung,
- auf der Basis einer oder mehrerer programmierbarer Geräte,
- einschließlich aller Elemente dieses Systems, wie Ein- und Ausgabegeräte.

PGN

PGN = **P**arameter **G**roup **N**umber = Parameter-Gruppennummer
PGN = 6 Null-Bits + 1 Bit reserviert + 1 Bit Data Page + 8 Bit PDU Format (PF) + 8 PDU Specific (PS)
Die Parameter-Gruppennummer ist ein Begriff aus dem →CAN-Protokoll →SAE J1939.

PID-Regler

Der PID-Regler (proportional–integral–derivative controller) besteht aus folgenden Anteilen:
- P = Proportional-Anteil
- I = Integral-Anteil
- D = Differential-Anteil (jedoch nicht beim Controller CR04nn, CR253n).

Piktogramm

Piktogramme sind bildhafte Symbole, die eine Information durch vereinfachte grafische Darstellung vermitteln (→ Kapitel **Was bedeuten die Symbole und Formatierungen?** (→ S. 7)).

Pre-Op

Pre-Op = PRE-OPERATIONAL mode (engl.) = Zustand vor 'betriebsbereit'.
Betriebszustand eines CANopen-Teilnehmers. Nach dem Einschalten der Versorgungsspannung geht jeder Teilnehmer automatisch in diesem Zustand. Im CANopen-Netz können in diesem Modus nur →SDOs und →NMT-Kommandos übertragen werden, jedoch keine Prozessdaten.

Prozessabbild

Mit Prozessabbild bezeichnet man den Zustand der Ein- und Ausgänge, mit denen die SPS innerhalb eines →Zyklusses arbeitet.

- Am Zyklus-Beginn liest die SPS die Zustände aller Eingänge in das Prozessabbild ein. Während des Zyklusses kann die SPS Änderungen an den Eingängen nicht erkennen.
- Im Laufe des Zyklusses werden die Ausgänge nur virtuell (im Prozessabbild) geändert.
- Am Zyklus-Ende schreibt die SPS die virtuellen Ausgangszustände auf die realen Ausgänge.

PWM

PWM = Puls-Weiten-Modulation
Bei dem PWM-Ausgangssignal handelt es sich um ein getaktetes Signal zwischen GND und Versorgungsspannung.
Innerhalb einer festen Periode (PWM-Frequenz) wird das Puls-/Pausenverhältnis variiert. Durch die angeschlossene Last stellt sich je nach Puls-/Pausenverhältnis der entsprechende Effektivstrom ein.

ratiometrisch

Ratio (lat.) = Verhältnis
Messungen können auch ratiometrisch erfolgen = Verhältnismessung. Wenn das Ausgangssinal eines Sensors proportional zu seiner Versorgungsspannung ist, kann durch ratiometrische Messung (= Messung im Verhältnis zur Versorgung) der Einfluss von Schwankungen der Versorgung reduziert, im Idealfall sogar beseitigt werden.
→ Analogeingang

RAW-CAN

RAW-CAN bezeichnet das reine →CAN-Protokoll, das ohne ein zusätzliches Kommunikationsprotokoll auf dem CAN-Bus (auf ISO/OSI-Schicht 2) arbeitet. Das CAN-Protokoll ist international nach →ISO 11898-1 definiert und garantiert zusätzlich in →ISO 16845 die Austauschbarkeit von CAN-Chips.

remanent

Remanente Daten sind gegen Datenverlust bei Spannungsausfall geschützt.
Z.B. kopiert das →Laufzeitsystem die remanenten Daten automatisch in einen →Flash-Speicher, sobald die Spannungsversorgung unter einen kritischen Wert sinkt. Bei Wiederkehr der Spannungsversorgung lädt das Laufzeitsystem die remanenten Daten zurück in den Arbeitsspeicher. Dagegen sind die Daten im Arbeitsspeicher einer Steuerung flüchtig und bei Unterbrechung der Spannungsversorgung normalerweise verloren.

ro

ro = read only (engl.) = nur lesen
Unidirektionale Datenübertragung: Daten können nur gelesen werden, jedoch nicht verändert.

RTC

RTC = **R**eal **T**ime **C**lock = Echtzeituhr
Liefert (batteriegepuffert) aktuell Datum und Uhrzeit. Häufiger Einsatz beim Speichern von Fehlermeldungsprotokollen.

rw

rw = read/write (engl.) = lesen und schreiben
Bidirektionale Datenübertragung: Daten können sowohl gelesen als auch verändert werden.

SAE J1939

Das Netzwerkprotokoll SAE J1939 beschreibt die Kommunikation auf einem →CAN-Bus in Nutzfahrzeugen zur Übermittlung von Diagnosedaten (z.B.Motordrehzahl, Temperatur) und Steuerungsinformationen.
Norm: Recommended Practice for a Serial Control and Communications Vehicle Network
- Teil 2: Agricultural and Forestry Off-Road Machinery Control and Communication Network
- Teil 3: On Board Diagnostics Implementation Guide
- Teil 5: Marine Stern Drive and Inboard Spark-Ignition Engine On-Board Diagnostics Implementation Guide
- Teil 11: Physical Layer – 250 kBits/s, Shielded Twisted Pair
- Teil 13: Off-Board Diagnostic Connector
- Teil 15: Reduced Physical Layer, 250 kBits/s, Un-Shielded Twisted Pair (UTP)
- Teil 21: Data Link Layer
- Teil 31: Network Layer
- Teil 71: Vehicle Application Layer
- Teil 73: Application Layer – Diagnostics
- Teil 81: Network Management Protocol

SD-Card

Eine SD Memory Card (Kurzform für **S**ecure **D**igital Memory Card; deutsch: Sichere digitale Speicherkarte) ist ein digitales Speichermedium, das nach dem Prinzip der →Flash-Speicherung arbeitet.

SDO

SDO = **S**ervice **D**ata **O**bject = Nachrichten-Objekt mit Servicedaten.
Das SDO dient dem Zugriff auf Objekte in einem CANopen-Objektverzeichnis. Dabei fordern 'Clients' die gewünschten Daten von 'Servern' an. Die SDOs bestehen immer aus 8 Bytes.
Beispiele:
- Automatische Konfiguration aller →Slaves über SDOs beim Systemstart.
- Auslesen der Fehlernachrichten aus dem →Objektverzeichnis.

Jedes SDO wird auf Antwort überwacht und wiederholt, wenn sich innerhalb der Überwachungszeit der Slave nicht meldet.

Selbsttest

Testprogramm, das aktiv Komponenten oder Geräte testet. Das Programm wird durch den Anwender gestartet und dauert eine gewisse Zeit. Das Ergebnis davon ist ein Testprotokoll (Log-Datei), aus dem entnommen werden kann, was getestet wurde und ob das Ergebnis positiv oder negativ ist.

Slave

Passiver Teilnehmer am Bus, antwortet nur auf Anfrage des →Masters. Slaves haben im Bus eine eindeutige →Adresse.

Steuerungskonfiguration

Bestandteil der CODESYS-Bedienoberfläche.

► Programmierer teilt dem Programmiersystem mit, welche Hardware programmiert werden soll.
> CODESYS lädt die zugehörigen Bibliotheken.
> Lesen und schreiben der Peripherie-Zustände (Ein-/Ausgänge) ist möglich.

stopped

stopped (engl.) = angehalten
Betriebszustand eines CANopen-Teilnehmers. In diesem Modus werden nur →NMT-Kommandos übertragen.

Symbole

Piktogramme sind bildhafte Symbole, die eine Information durch vereinfachte grafische Darstellung vermitteln (→ Kapitel **Was bedeuten die Symbole und Formatierungen?** (→ S. 7)).

Systemvariable

Variable, auf die via IEC-Adresse oder Symbolname aus der SPS zugegriffen werden kann.

Target

Das Target enthält für CODESYS die Hardware-Beschreibung des Zielgeräts, z.B.: Ein- und Ausgänge, Speicher, Dateiablageorte.
Entspricht einem elektronischen Datenblatt.

TCP

Das **T**ransmission **C**ontrol **P**rotocol ist Teil der Protokollfamilie TCP/IP. Jede TCP/IP-Datenverbindung hat einen Sender und einen Empfänger. Dieses Prinzip ist eine verbindungsorientierte Datenübertragung. In der TCP/IP-Protokollfamilie übernimmt TCP als verbindungsorientiertes Protokoll die Aufgabe der Datensicherheit, der Datenflusssteuerung und ergreift Maßnahmen bei einem Datenverlust. (vgl.: →UDP)

Template

Template (englisch = Schablone) ist eine Vorlage, die mit Inhalten gefüllt werden kann.
Hier: Eine Struktur von vorkonfigurierten Software-Elementen als Basis für ein Anwendungsprogramm.

UDP

UDP (**U**ser **D**atagram **P**rotocol) ist ein minimales, verbindungsloses Netzprotokoll, das zur Transportschicht der Internetprotokollfamilie gehört. Aufgabe von UDP ist es, Daten, die über das Internet übertragen werden, der richtigen Anwendung zukommen zu lassen.
Derzeit sind Netzwerkvariablen auf Basis von →CAN und UDP implementiert. Die Variablenwerte werden dabei auf der Basis von Broadcast-Nachrichten automatisch ausgetauscht. In UDP sind diese als Broadcast-Telegramme realisiert, in CAN als →PDOs.
Dem Protokoll entsprechend, sind diese Dienste nicht bestätigte Dienste: es gibt keine Kontrolle, ob die Nachricht auch beim Empfänger ankommt. Netzwerkvariablen-Austausch entspricht einer "1-zu-n-Verbindung" (1 Sender zu n Empfängern).

Verwendung, bestimmungsgemäß

Das ist die Verwendung eines Produkts in Übereinstimmung mit den in der Anleitung bereitgestellten Informationen.

Watchdog

Der Begriff Watchdog (englisch; Wachhund) wird verallgemeinert für eine Komponente eines Systems verwendet, die die Funktion anderer Komponenten beobachtet. Wird dabei eine mögliche Fehlfunktionen erkannt, so wird dies entweder signalisiert oder geeignete Programm-Verzweigungen eingeleitet. Das Signal oder die Verzweigungen dienen als Auslöser für andere kooperierende Systemkomponenten, die das Problem lösen sollen.

Zykluszeit

Das ist die Zeit für einen Zyklus. Das SPS-Programm läuft einmal komplett durch.
Je nach ereignisgesteuerten Verzweigungen im Programm kann dies unterschiedlich lange dauern.

9 Index

A

B

C

T

U

V

W

Z

10 Notizen • Notes • Notes

11 ifm weltweit • ifm worldwide • ifm à l'échelle internationale

Stand: 2017-12-18 8310

ifm electronic gmbh • Friedrichstraße 1 • 45128 Essen
www.ifm.com • E-Mail: info@ifm.com
Service-Hotline: 0800 16 16 16 4 (nur Deutschland, Mo...Fr, 07.00...18.00 Uhr)

ifm Niederlassungen • Sales offices • Agences

D	Niederlassung Nord • 31135 Hildesheim • Tel. 05121 7667-0 Niederlassung West • 45128 Essen • Tel. 0201 36475 -0 Niederlassung Mitte-West • 58511 Lüdenscheid • Tel. 02351 4301-0 Niederlassung Süd-West • 64646 Heppenheim • Tel. 06252 7905-0 Niederlassung Baden-Württemberg • 73230 Kirchheim • Tel. 07021 8086-0 Niederlassung Bayern • 82178 Puchheim • Tel. 089 80091-0 Niederlassung Ost • 07639 Tautenhain • Tel. 036601 771-0
AE	ifm electronic FZC • Saif Zone, Sharjah • Tel. +971- 6-5573601
AR	ifm electronic s.r.l. • 1107 Buenos Aires • Tel. +54 11 5353-3436
AT	ifm electronic gmbh • 1120 Wien • Tel. +43 / 1 / 617 45 00
AU	ifm efector pty ltd. • Mulgrave Vic 3170 • Tel. +61 1300 365 088
BE	ifm electronic n.v./s.a. • 1731 Zellik • Tel. +32 2 481 0220
BG	ifm electronic eood • 1202 Sofia • Tel. +359 2 807 59 69
BR	ifm electronic Ltda. • 03337-000 Sao Paulo / SP • Tel. +55-11-2672-1730
CA	ifm efector Canada inc. • Mississauga, ON L5N 2X7 • Tel. +1 855-436-2262
CH	ifm electronic ag • 4624 Härkingen • Tel. +41 / 800 88 80 33
CL	ifm electronic SpA • Oficina 5041 Comuna de Conchalí • Tel. +56-2-32239282
CN	ifm electronic (Shanghai) Co. Ltd. • 201203 Shanghai • Tel. +86 21 3813 4800
CZ	ifm electronic, spol. s.r.o. • 140 00 Praha 4 • Tel. +420 267 990 211
DK	ifm electronic a/s • 2605 Brøndby • Tel. +45 70 20 11 08
ES	ifm electronic s.a. • 08820 El Prat de Llobregat • Tel. +34 93 479 30 80
FI	ifm electronic oy • 00440 Helsinki • Tel . +358 75 329 5000
FR	ifm electronic s.a. • 93192 Noisy-le-Grand Cedex • Tél. +33 0820 22 30 01
GB	ifm electronic Ltd. • Hampton, Middlesex TW12 2HD • Tel. +44 / 20 / 8213 0000
GR	ifm electronic monoprosopi E.P.E. • 15125 Amaroussio • Tel. +30 210 61 800 90
HU	ifm electronic kft. • 9028 Györ • Tel. +36-96 / 518-397
IN	ifm electronic India Private Limited • Kolhapur, 416234 • Tel. +91 / 231 / 267 27 70
IE	ifm electronic (Ireland) Ltd. • Dublin 22 • Tel. +353 / 1 / 461 32 00
IT	ifm electronic s.r.l. • 20864 Agrate Brianza (MB) • Tel. +39 39-6899982
JP	efector co., ltd. • Chiba-shi, Chiba 261-7118 • Tel. +81 043-299-2070
KR	ifm electronic Ltd. • 04420 Seoul • Tel. +82 2-790-5610
MX	ifm efector S. de R.L. de C.V. • San Pedro Garza Garcia, N.L. 66269 • Tel. +52-81-8040-3535
MY	ifm electronic Pte. Ltd • 47100 Puchong, Selangor • Tel. +603 8066 9853
NA	ifm elctronic (pty) Ltd • 25 Dr. W. Kulz Street Windhoek • Tel. +264 61 300984
NL	ifm electronic b.v. • 3843 GA Harderwijk • Tel. +31 341-438 438
NZ	ifm efector pty ltd • 930 Great South Road Penrose, Auckland • Tel. +64 / 95 79 69 91
PL	ifm electronic sp. z o.o. • 40-106 Katowice • Tel. +48 32 70 56 400
PT	ifm electronic s.a. • 4410-137 São Félix da Marinha • Tel. +351 223 71 71 08
RO	ifm electronic s.r.l • Sibiu 557260 • Tel. +40 269 224 550
RU	ifm electronic • 105318 Moscow • Tel. +7 495 921-44-14
SG	ifm electronic Pte Ltd • 609 916 Singapore • Tel. +65 6562 8661
SK	ifm electronic s.r.o. • 831 06 Bratislava • Tel. +421 244 872 329
SE	ifm electronic ab • 412 50 Göteborg • Tel. +46 31-750 23 00
TR	ifm electronic Ltd. Sti. • 34381 Sisli, Istanbul • Tel. +90 212 210 50 80
TW	ifm electronic • Kaohsiung City, 806, Taiwan R.O.C. • Tel. +886 7 3357778
UA	TOV ifm electronic • 02660 Kiew • Tel. +380 44 501-85-43
US	ifm efector inc. • Malvern, PA 19355 • Tel. +1 800-441-8246
VN	ifm electronic Vietnam Co., Ltd. • 700000 Ho Chi Minh City • Tel. +84-28-2253.6715
ZA	ifm electronic (Pty) Ltd. • 0157 Pretoria • Tel. +27 12 450 0412