Erläuterungen zur Vector CAN Databasis

Übersicht

Die beiliegende Vector CAN Datenbasis (*.dbc) für Netzgeräte oder elektronische Lasten mit CAN-Schnittstelle IF-AB-CAN dient zur Einbindung in Software der Firma Vector, wie z. B. CANalyzer oder CANoe. Durch die Datenbasis werden die wichtigsten für das jeweilige Gerät verfügbaren Befehle und Umrechnungsfaktoren definiert.

Einschränkungen

Es sind, aufgrund der Struktur der Datenbasis, nicht alle Befehle implementiert, die ein Gerät anbietet. Falls diese trotzdem benötigt werden, können Sie durch den Anwender selbst umgesetzt werden, indem CAPL benutzt wird. Bitte beachten Sie dazu die Modbus-Registerliste zu der Serie des Gerätes und die Programmieranleitung.

Was wird benötigt?

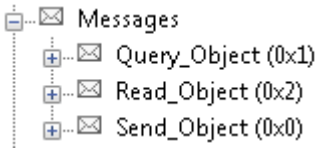
Ein Gerät mit CAN-Schnittstelle IF-AB-CAN, eine dazu gehörige Datenbasis (z.B. ELR9080-510.dbc), Vector-Software oder kompatible Software, CAN-Hardware.

Was ist auf der Seite des Gerätes zu tun?

Der Anwender muss lediglich die zu verwendende Basis-ID (siehe Geräte-Handbuch bzw. Schnittstellen-Handbuch) am Gerät einstellen, abgesehen von den allgemeinen CAN-Einstellungen. Bei mehreren Geräten am gleichen Bus müssen die Basis-IDs unterschiedlich sein.

Was ist auf der PC-Seite zu tun?

Der Anwender benötigt den Datenbasis-Editor CANdb++, um die Datenbasis seinen eigenen Gegebenheiten anzupassen. Normalerweise ist nur die Anpassung der Botschafts-IDs der drei Haupt-Botschaften nötig, die standardmäßig auf 0, 1 und 2 festgelegt sind, passend zur Basis-ID 0x0 nach Auslieferung des Gerätes oder Zurücksetzen auf Standardwerte.



Hierbei gilt:
Query_Object = Basis-ID + 1
Read_Object = Basis-ID + 2
Send_Object = Basis-ID,
wobei die Basis-ID die am Gerät eingestellte ist.

Beispiel: das Gerät wurde auf Basis-ID 500 eingestellt. Dann müssen **Query_Object (501), Send_Object (500)** und **Read_Object (502)** sein.

Es gibt außerdem eine gewisse Anzahl weiterer Botschaften für die zyklische Datenübertragung (siehe auch Handbuch des Gerätes, Einstellungen zur CAN-Schnittstelle am Bedienteil). Mehr dazu unten.

Objektdefinition und Kompatibilität

Der Dateninhalt der Nachrichten ist angelehnt an ModBus RTU. Bei ModBus sind Objekte immer sog. Register mit 16 Bit Breite. Daher gilt: Objektnummer = Registernummer. Mit dem Gerät wird eine Programmieranleitung mitgeliefert, zu der Registerlisten für diverse Serien gehören. Diese Listen, die Programmieranleitung und die DBC bilden die nötige Referenz für die Kommunikation mit dem Gerät über CAN. Die CAN-Schnittstelle wird von folgenden Geräteserien unterstützt:

- ELR 9000
- PSI 9000 (alle)
- EL 9000 B
- PSE 9000

Wie werden die drei Haupt-Botschaften benutzt?

Mit diesen drei Botschaften wird die gesamte nicht-zyklische Gerätekommunikation erledigt.

Hinweis: das Setzen von Sollwerten und/oder Status wird als Steuerung des Gerätes betrachtet und erfordert vorherige Umschaltung in Fernsteuerbetrieb.

*Verwendung von **Send_Object***

Name	Send_Object
ID	Basis-ID
Zweck	Zum Senden von Sollwerten (U, I, P usw.) und Status (Eingang/Ausgang ein/aus usw.) Zugehörige ModBus-Funktionen: WRITE Single Coil, WRITE Single Register, WRITE Multiple Registers
Länge	4-8 Bytes (Registernummer und Argument)

Send_Object dient zum Setzen von Sollwerten oder Zuständen. Das bedingt, daß a) ausgewählt werden muß, welcher Sollwert/Zustand gesetzt werden soll und b) wird der Sollwert/Zustand selbst benötigt. Die Auswahl des zu setzenden Sollwertes/Zustandes wird über das Signal Mux_Send erledigt, das mit einer Tabelle verknüpft ist um symbolische Befehle auszuwählen, wie z. B. „Set_Voltage_Value“.

*Verwendung von **Query_Object***

Name	Query_Object
ID	Basis-ID + 1
Zweck	Zur Abfrage von Sollwerten, Istwerten und Status Zugehörige ModBus-Funktionen: READ Holding Registers, READ COILS
Länge	2 Bytes (Registernummer)

Mit „Query_Object“ werden in erster Linie Status und Istwerte abgefragt, es kann aber auch jeder Sollwert und die meisten gesetzten Gerätezustände und –modi ausgelesen werden. Diese Botschaft sendet lediglich die Objektnummer im Datenbereich der Botschaft. Die erwartete Antwort kommt dann auf der ID der Botschaft „Read_Object“, siehe unten.

*Verwendung von **Read_Object***

Name	Read_Object
ID	Basis-ID + 2
Zweck	Auf dieser ID sendet das Gerät Antworten auf Anfragen und Kommunikationsfehler
Länge	3…8 Bytes (Registernummer und Daten)

Diese Botschaft dient nur zum Einsortieren der vom Gerät eingehenden Nachrichten (Antworten). Üblicherweise wird jeder Befehl, der bei **Query_Object** angefragt wird, auch in **Read_Object** vorhanden sein und kann somit ausgewertet werden.

Wie sind aus dem Gerät gelesene Zustände und anderes zu interpretieren?

- Soll- und Istwerte werden durch Faktoren in der DBC in Realwerte umgerechnet
- Status wie „Ausgang/Eingang ein/aus“ sind durch Wertetabellen beschrieben
- Fehlercodes sind durch Wertetabellen beschrieben

Hinweise

- Sollten in den typischen Vector-Panels angezeigte Istwertfelder rot unterlegt sein, bedeutet das, der zugehörige Maximalwert wurde überschritten, obwohl er nicht überschritten werden kann. Das ist ein Umrechnungsfehler, der durch abgerundete Umrechnungsfaktoren entsteht. Um das zu umgehen, braucht nur die letzte Stelle des Umrechnungsfaktors um 1 verringert werden
- Für jeden Gerätetyp muss eine extra Datenbasis vorhanden sein, damit die Faktoren für die Umrechnung der Soll- und Istwerte in phys. Werte passen
- Jedes Gerät besitzt mehrere IDs: eine Basis-CAN-ID (insg. 3 IDs), eine Broadcast-ID, eine weitere Basis-ID für zyklisches Lesen (insg. 5 IDs) und noch eine Basis-ID für zyklisches Schreiben (insg. 2 IDs)
- Die separat einstellbare Broadcast-ID dient zum gleichzeitigen Senden von Werten/Status an mehrere Geräte im Bus, die dann alle auf dieselbe, vom Anwender bestimmte Broadcast-ID eingestellt sein müssen.
- Die verschiedenen ID´s dürfen nicht gleich sein und sich untereinander nicht mit einer der anderen ID´s des Gerätes überschneiden!
- Die zeitliche Aufeinanderfolge von mehreren Nachrichten darf nicht zu gering sein. Es wird empfohlen, nicht öfter als alle 5ms eine Anfrage zu senden oder einen Wert zu schreiben.

Aufbau der normalen CAN-Telegramme

Send_Object

Basis-ID und Broadcast-ID: WRITE Single Coil und WRITE Single Register
(6 Bytes, die übrigen Datenbytes sind 0)

Bytes 0+1	Byte 2	Bytes 3+4
Register	Reg.Anz.	Datenwort
0…65534	1	Zu schreibender Wert

Basis-ID: WRITE Multiple Registers
Wird verwendet für Daten ab einer Registerlänge von 2. Die Kennung 0xFF identifiziert die erste Nachricht, 0xFE die zweite, usw...

Bytes 0+1	Byte 2	Byte 3	Bytes 4-7
Startreg.	Reg.Anz.	Kennung	Datenbytes
0…65534	2…123	0xFF, 0xFE…	n zu schreibende Werte

Query_Object

Basis ID + 1: READ Holding Registers und READ Coils

Bytes 0+1
Register
0…65534

Read_Object

Basis-ID + 2: Antworten (wenn Anzahl angefragter Register 1-3)

Bytes 0+1	Byte 2-3
Register	Daten 2 Bytes
0…65534	Angefragte Reg.

oder

Bytes 0+1	Byte 2-5
Register	Daten 4 Bytes
0…65534	Angefragte Reg.

oder

Bytes 0+1	Byte 2-7
Register	Daten 6 Bytes
0…65534	Angefragte Reg.

Basis-ID + 2: Antworten (wenn Anzahl angefragter Register >3)

Bytes 0+1	Byte 2	Byte 3-7
Register	Kennung	Daten 5 Bytes
0…65534	0xFF, 0xFE…	Angefragte Reg.

Basis-ID + 2: Fehlermeldungen

Bytes 0+1	2
Register	
65535	Fehlercode

Zyklische Daten

< ab KE-Firmware 2.13 in PSI/ELR 9000 und EL 9000 B)

Neben der normalen, PC-gesteuerten CAN-Kommunikation kann das Gerät zyklisch bestimmte Daten senden. Ist der zyklische Datenverkehr aktiviert, übermittelt das Gerät in der eingestellten Zykluszeit seinen Status, die aktuellen Ist- und Sollwerte und die konfigurierten Limits an den CAN Bus (über die Basis-IDs für zyklische Daten).
Über die Base-ID Senden können die Sollwerte U, I, P und R und eine Auswahl an Kontrollbefehlen zyklisch und in kompakterem Format an das Gerät übermittelt werden.

Empfohlene Botschafts-Namen für die Integration in die Datenbasis:
Send_Mode = Basis-ID Senden
Send_Set_Values = Basis-ID Senden +1
Read_Status = Basis-ID Lesen
Read_Act_Values = Basis-ID Lesen + 1
Read_Set_Values = Basis-ID Lesen + 2
Read_Limits_1 = Basis-ID Lesen + 3
Read_Limits_2 = Basis-ID Lesen + 4

Die Lese-Zykluszeiten können am Gerät zwischen 20ms und 5000ms eingestellt werden. Deaktiviert werden die jeweiligen Lesenachrichten indem die zugehörige Zykluszeit auf 0 gesetzt wird.

Aufbau der zyklischen CAN-Telegramme

Send_Modes

Basis-ID Senden: Steuerung/Modi
(2 Bytes, die übrigen Datenbytes sind 0)

Bytes 0-1
Control
Zu schreibender Wert

Für den Aufbau des 16-Bit-Wertes siehe Programmieranleitung „Programming ModBus & SCPI“, Abschnitt 8 für CAN.

Send_Set_Values

Basis-ID Senden + 1: Sollwerte (Uset, Iset, Pset, Rset)
(8 Bytes)

Bytes 0-1	Bytes 2+3	Bytes 4+5	Bytes 6+7
Uset	Iset	Pset	Rset
Zu schreibender Wert	Zu schreibender Wert	Zu schreibender Wert	Zu schreibender Wert

Read_Status

Basis-ID Lesen: Status
(4 Bytes, die übrigen Datenbytes sind 0)

Bytes 0+1+2+3
Status
Wert

Für den Aufbau des 32-Bit-Wertes siehe Programmieranleitung „Programming ModBus & SCPI“, Abschnitt 8 für CAN.

Read_Actual_Values

Basis-ID Lesen + 1: Istwerte (Uactual, Iactual, Pactual)
(6 Bytes, die übrigen Datenbytes sind 0)

Bytes 0+1	Bytes 2+3	Bytes 4+5
Uactual	Iactual	Pactual
Wert	Wert	Wert

Read_Set_Values

Basis-ID Lesen + 2: Sollwerte (Uset, Iset, Pset, Rset)
(8 Bytes)

Bytes 0+1	Bytes 2+3	Bytes 4+5	Bytes 6+7
Uset	Iset	Pset	Rset
Wert	Wert	Wert	Wert

Read_Limits1

Basis-ID Lesen + 3: Einstellgrenzen 1 (Imax, Imin, Umax, Umin)
(8 Bytes)

Bytes 0+1	Bytes 2+3	Bytes 4+5	Bytes 6+7
Imax	Imin	Umax	Umin
Wert	Wert	Wert	Wert

Read_Limits2

Basis-ID Lesen + 4: Einstellgrenzen 2 (Pmax, Rmax)
(4 Bytes, die übrigen Datenbytes sind 0)

Bytes 0+1	Bytes 2+3
Pmax	Rmax
Wert	Wert