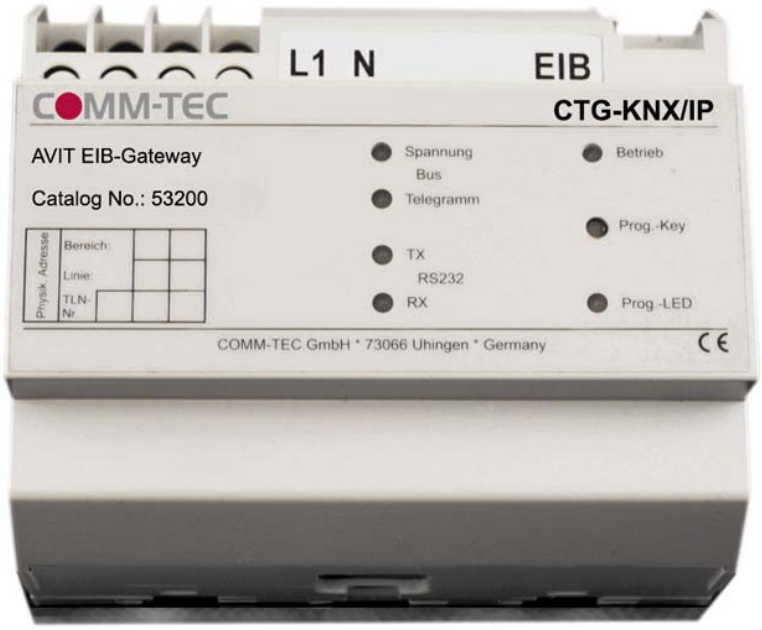
Benutzer-Handbuch

für das
COMM-TEC EIB Gateway
CTG-KNX/IP (v1.0.1)



Inhaltsverzeichnis

Siemensstrasse 14
D-73066 Uhingen
Tel.: +49 (0)7161/3000-0
Fax: +49 (0)7161/3000-400

1 Vorwort

1.1 Zielsetzung

Das Softwaremodul dient der Ankopplung von AVIT und AMX NetLinx-Steuerungen an den "European Installation Bus" EIB ("Instabus"). Es stellt eine einfach zu verwendende Schnittstelle für den Entwickler zur Verfügung.

1.2 Rolle des EIB-Installateurs

Es kann nicht genug betont werden: beim Anschluß an ein EIB-System ist solides EIB-Fachwissen und der enge Kontakt zu einem kundigen EIB-Installateur dringend anzuraten. Ein falsch gesetztes Lese-Flag in einem Aktor oder ein restriktiv programmierter Linienkoppler können ohne gute Analyse-Tools wirklich schwer zu finden sein.

Das Netlinx Modul kann keinesfalls ein EIB-System konfigurieren. Das Paket dient zur Kontaktaufnahme mit einem funktionsfähigen EIB, und kann nur auf Buselemente zugreifen, deren Benutzung auch gestattet ist. Schon aus diesem Grund sollte mit den EIB-Installateuren geklärt werden, ob alle Buskomponenten die gewünschten Funktionen auch ausführen ***dürfen***.

1.3 EIB aus der Sicht des AMX-Programmierers

Analog zu AMX-Systemen ist der European Installation Bus – wie der Name schon sagt – ein Bussystem: alle Komponenten sind im Prinzip an der selben Leitung angeschlossen und teilen sich die verfügbare Bandbreite. Der Bus selbst ist ein zweiadriges Kabel, das sowohl eine Versorgungsspannung von 24V bereitstellt, als auch dem Datentransport zwischen den Geräten dient.
Im Gegensatz zu AMX ist ein EIB-System aber dezentral organisiert – es gibt keinen speziellen Master, der die Kommunikation steuert, sondern jedes Gerät darf an jedes andere Gerät Daten senden. Ein ausgeklügeltes Protokoll stellt dabei sicher, daß immer nur ein Gerät gleichzeitig sendet, und Kollisionen weitestgehend vermieden werden.

Sämtliche Kommunikation erfolgt dabei in Form sogenannter Telegramme. Ein Telegramm ist ein Datenpaket, das aus folgenden Komponenten besteht:

- Absenderkennung – Hardware-Adresse des sendenden Gerätes
- Zieladresse – Gruppenadresse der empfangenden Geräte
- Nutzdaten

Ein Telegramm kann dabei an mehrere Zielgeräte gleichzeitig gesendet werden, z.B. um sämtliche Lampen in einem Raum gleichzeitig auszuschalten. Es besteht also ein grundlegender Unterschied zwischen Quell- und Zieladressen:
Eine Quelladresse ist eine Hardware-Adresse, also die Adresse des *Gerätes* das ein Telegramm sendet. Eine Zieladresse ist eine Gruppen-Adresse, eine Adresse, die eine *Funktion* beschreibt.

Jedes Gerät am EIB hat also genau eine Hardware-Adresse, kann aber durchaus mehrere Gruppenadressen haben. Ebenso ist es möglich und auch üblich, daß *mehrere* Geräte auf die selbe Gruppenadresse reagieren.

Beide Adressarten werden vom EIB-Installateur vergeben – die Hardware-Adressen beschreiben Art und Anzahl der verwendeten Geräte, und werden während der Planung und Installation zugeteilt. Für Hardware-Adressen interessiert sich das Gateway überhaupt nicht.

Die wesentlich wichtigeren Adressen für AMX sind die Gruppenadressen.
Sie legen die Funktionen fest, die eine EIB-Installation ausführen kann. Funktionen werden also schlicht dadurch ausgelöst, daß einer Gruppenadresse ein bestimmter Wert gesendet wird.

Das folgende Diagramm gibt einen grafischen Überblick zum Ablauf der Kommunikation vom projektbezogenen NetLinx Source Code über das NetLinx COMM-Modul bis hin zum EIB Gateway.

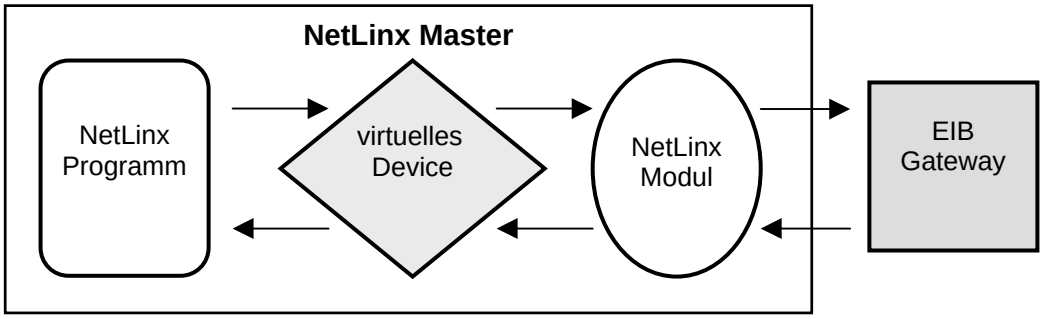
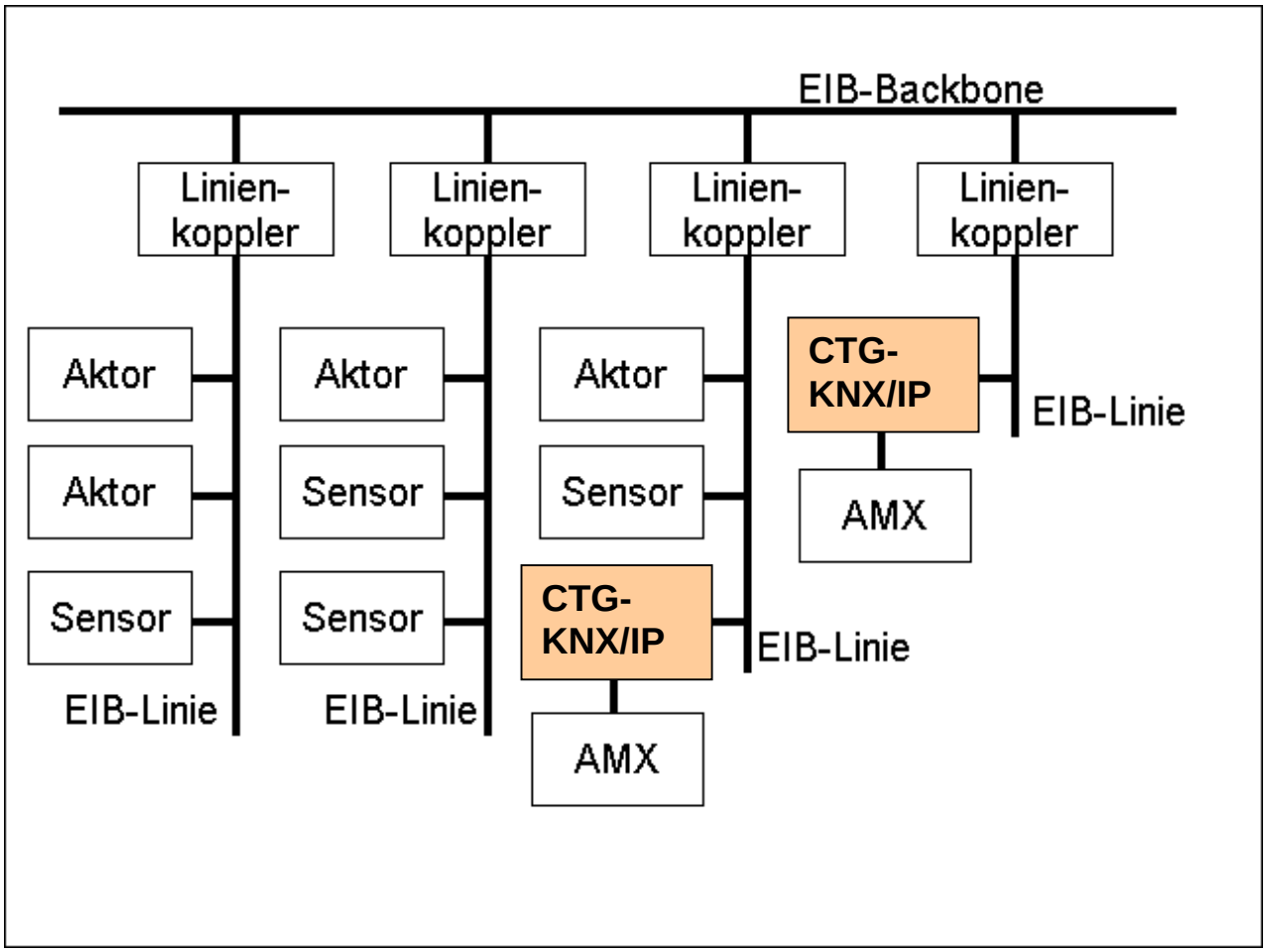


Abbildung 1 - Ablauf der Kommunikation

1.4 EIB-Besonderheiten beim Einsatz des Gateways



Prinzipiell ist das EIB-Gateway CTG-KNX/IP zwar ein gewöhnliches EIB-Gerät, kann also an jeder Stelle mit dem EIB verbunden werden. Im Gegensatz zu einfachen Aktoren und Sensoren ist es aber unter Umständen für sehr viele (bis zu 3000) Gruppenadressen zuständig – ein normaler Dimmer reagiert z.B. nur auf vier Adressen.
Daher ist unbedingt darauf zu achten, daß das Gateway eine reelle Chance hat, auf alle Bus-Telegramme zu reagieren, die von Interesse sind. Insbesondere beim Einsatz von Linienkopplern ist eine sorgfältige Planung im Vorfeld unabdingbar. Folgende Überlegungen sollten berücksichtigt werden:

Zum Einen müssen Bustelegramme das Gateway natürlich erreichen können. Sind Linienkoppler „zwischen“ Gateway und zu steuernden Komponenten eingefügt, müssen die Filtertabellen der Koppler so programmiert sein, daß auch wirklich alle relevanten Telegramme durchgereicht werden.

Zum Anderen sind ältere Busankoppler träge – nach jedem empfangenen Telegramm benötigt ein EIB-Gerät eine gewisse Zeit, bis das nächste Telegramm aufgenommen werden kann.
Speziell Szenenbausteine können aber eine Flut von Telegrammen erzeugen, die an alle - an der Szene beteiligten - Aktoren gesendet werden. Da das normalerweise *verschiedene* Geräte sind, fällt die „Totzeit“ der Busankoppler nicht ins Gewicht – jeder Koppler hat genügend Zeit, sich zu erholen, bevor ein neues Telegramm an ihn empfangen wird.

Nicht so beim Gateway: üblicherweise sind *sämtliche* beteiligten Gruppenadressen einer Szene von Interesse – das Gateway benötigt keine Ruhezeit und kann problemlos alle Telegramme mitlesen, auch bei hoher Buslast. Bei der Ansteuerung ist jedoch darauf zu achten, dass die angesteuerten Aktoren genug Zeit haben !

2 Hardware

2.1 Anschlüsse, Anzeige- und Bedienelemente

Anschlüsse :

LEDs :	Busspannung	blinkt, wenn die Busspannung unter 15V sinkt
	Bustelegramm	zeigt Telegramme auf dem EIB an
	TX	zeigt Datenübertragung vom Gateway zu AVIT / AMX
	RX	zeigt Datenübertragung von AVIT / AMX zum Gateway
	Betrieb	leuchtet, wenn Spannungsversorgung anliegt
	Prog.-LED	blinkt, wenn das Gateway EIB-seitig in den Programmiermodus gesetzt ist
Taster :	Prog.-Key	setzt das Gateway in den Programmiermodus (EIB-seitig)

2.2 Installation

2.2.1 Verbindung Gateway - TCP/IP Schnittstelle

Das Gateway wird mit einem handelsüblichen Netzwerkkabel mit dem Netzwerk verbunden.

3 Programmierung

3.1 Modul einbinden

Zunächst muss dem Programm mitgeteilt werden, welche IP-Adresse das Gateway hat.
Dies geschieht im Online_Event des virtuellen Devices :

```
DATA_EVENT[vdvEIB]
{
      ONLINE :
      {
            SEND_COMMAND vdvEIB,'IP=172.16.100.123'
      }
}
```

(siehe auch Anhang)

Der Port für das Gateway ist fest auf 10002 eingestellt und kann nicht geändert werden.
Die IP-Adresse wird mit der beiliegenden Software des Gateways gesetzt.
Im Auslieferungszustand ist die Ethernet Adresse des Gateways 192.168.1.106

Die Schnittstelle zum Modul und zur Kommunikation mit dem EIB Gateway stellt das virtuelle Device `vdvEIB` zur Verfügung. Die Kommunikation mit dem Gateway erfolgt ***AUSSCHLIESSLICH*** über dieses virtuelle Device.
Eine komplette Dokumentation der verfügbaren Kommandos und der String-Rückmeldungen ist im Abschnitt „Die NetLinx-Modul Schnittstelle“ zu finden.

Das Kommunikations-Modul (COMM-Modul) übersetzt zwischen der Standard Kommando Schnittstelle auf NetLinx Seite und dem Protokoll für die EIB Anbindung.

Das Kommunikations-Modul für die EIB Anbindung wird mit folgender Source-Code Zeile eingebunden:

```
DEFINE_MODULE 'CTEIB7_mod' EIB7(   vdvEIB,
                                   dvEIB,
                                   lEIB_Value)
```

Die Parameter sind die Folgenden:

- **vdvEIB** - ist das virtuelles Device für die Kommunikation mit dem Modul
- **dvEIB** - ist die physikalische Schnittstelle zur EIB Anbindung (Gateway)
- **lEIB_Value** - ist das zentrale Wertearray der EIB Aktoren (Typ LONG !)

Dieses Rückmeldearray muss in DEFINE_VARIABLE deklariert werden.

```
DEFINE_VARIABLE
...
LONG lEIB_Value[3000]
...
```

Das Modul kümmert sich auch um die richtige Initialisierung der seriellen Schnittstelle, es ist also nicht notwendig, z.B. die Baudrate zu setzen.

Um Aktoren auf dem Bus ansprechen / auswerten zu können, müssen die EIB Gruppenadressen dem Programm bekannt gemacht werden. Dies geschieht in einer externen Datei, in welcher

Gruppenadresse, Typ, Pollverknüpfungen und zusätzliche Eigenschaften auf eine “AMX Nummer“ zwischen 1 und 3000 abgebildet werden. Die Kommunikation mit den betreffenden Aktoren erfolgt dann nur noch über diese Nummer.

Diese Tabelle wird mit folgender Source-Code Zeile eingebunden :

```
#INCLUDE 'EIB_Table.axi'
```

Ein Beispiel für eine solche Datei, sowie ein Beispielprogramm finden sie im Anhang A.

Zusätzlich sollte die Datei EIB_Tools.AXI eingebunden werden um einen einfachen Zugriff auf die häufig verwendeten Funktionen zu haben. (siehe Anhang B)

```
#INCLUDE 'EIB_Tools.axi'
```

3.2 Rückmeldungen auswerten

Die Rückmeldungen sollten alle in einem Data_Event ausgewertet werden. Pro Rückmeldung wird ***ein*** DATA_EVENT ausgelöst – es steht also genau ***eine*** Rückmeldung in DATA.TEXT !
Kommen mehre Rückmeldungen, werden entsprechend viele Events ausgelöst.

3.3 Die NetLinx-Modul Schnittstelle

3.3.1 Kommandos zum Modul

Kommandos an das Modul erfolgen immer per SEND_COMMAND an das virtuelle Device.

Das Modul kennt folgende Kommandos:

Kommando	Beschreibung

ADD=<Nr>:
<Typ>:
<GrpAdr>
[:Flags]
	Hinzufügen einer EIB-Gruppenadresse zur Liste
	Hinweis : Flags sind optional
	
	<Nr> - 1 .. 3000 = AMX Nummer des Aktors
	<Typ> - Aktortyp (Switch, Dim4, 1Byte, 2Byte, 3Byte, 4Byte, Text, HEXText)
	<GrpAdr> - EIB Gruppenadresse in 2 oder 3 gruppiger Darstellung
	<Flags> - EIS5 – Wert wird zusätzlich als ASCII Float Wert gemeldet. Der EIB Wert wird nach EIS5 Standard gewandelt. (nur gültig für 2Byte Aktoren)
	- Time – Wert wird zusätzlich als ASCII Zeit gemeldet (hh:mm:ss) (nur gültig für 3Byte Aktoren)
	- Date – Wert wird zusätzlich als ASCII Datum gemeldet (Achtung AMX Format: MM/DD/YY) (nur gültig für 3Byte Aktoren)
	- PS – Aktor wird beim ***Start*** vom AMX System automatisch gepollt
	
	Mehrere Flags werden durch Kommas getrennt
	Bsp.:
	`SEND_COMMAND vdvEIB,'ADD=13:Switch:1/0/11'`
	`SEND_COMMAND vdvEIB,'ADD=17:1Byte:4/7/12:PS'`
	`SEND_COMMAND vdvEIB,'ADD=45:2Byte:3/0/11:EIS5'`
	`SEND_COMMAND vdvEIB,'ADD=12:3Byte:2/1/101:TIME,PS'`

DATE=<Nr>: <Datum>	Setzen des Datums Hinweis : Nur gültig für Aktoren vom Typ 3 Byte <Nr> - 1 .. 3000 = AMX Nummer des Aktors <Datum> - Datum im Format MM/DD/YY (AMX-Datumsformat) Bsp.: `SEND_COMMAND vdvEIB,'DATE=17:14/08/06`
DATE?<Nr>	Abfrage des Datumswertes Hinweis : Nur gültig für Aktoren vom Typ 3 Byte <Nr> - 1.. 3000 = AMX Nummer des Aktors Bsp.: `SEND_COMMAND vdvEIB,'DATE?17`
DEBUGON=<Level>	Einschalten der Debugmeldungen <Level> - (... demnächst ...) Bei aktivierten Debugmeldungen werden alle Aktoren im Terminal aufgelistet, die über den EIB angesprochen werden. Hierdurch ist eine einfach Diagnose möglich. Bsp.: `SEND_COMMAND vdvEIB,'DEBUGON=1'`
DEBUGOFF	Abschalten der Debugmeldungen Bsp.: `SEND_COMMAND vdvEIB,'DEBUGOFF'`
EIS5=<Nr>: <Fliesskommawert>	Setzen eines EIS5 Wertes Rechnet einen in ASCII dargestellten Fliesskommawert vor dem Versenden in einen 2Byte EIS5 Wert um. <Nr> - 1 .. 3000 = AMX Nummer des Aktors <Fliesskommawert> - Zahl im Bereich zw. –671088.64 und 670760.96 Hinweis : Nur gültig für Aktoren vom Typ 2 Byte `Bsp.: SEND_COMMAND vdvEIB,'EIS5=12:24.3'`

Kommando	Beschreibung
EIS5?<Nr>	Abfragen eines EIS5 Wertes Wandelt die 2Byte Rohdaten in einen ASCII-String mit Fliesskommadarstellung Hinweis : Nur gültig für Aktoren vom Typ 2 Byte <Nr> - 1 .. 3000 = AMX Nummer des Aktors Bsp.: `SEND_COMMAND vdvEIB,'EIS5?12'`
GET=<Nr> GET?<Nr>	Abfragen des im Modul gespeicherten Wertes eines Aktors Hinweis : Erzeugt kein Telegramm auf dem EIB. (Zur Syncronisation von Master zu Master Verbindungen sollte nur das POLL Kommando benutzt werden) Nicht für Datentypen Text und HEXText <Nr> - 1 .. 3000 = AMX Nummer des Aktors Bsp.: `SEND_COMMAND vdveib,'GET=17'`
POLL=<Nr> POLL?<Nr>	Abfrage aktueller Wert eines Aktors. (Zur Syncronisation von Master zu Master Verbindungen sollte nur das POLL Kommando benutzt werden) <Nr> - 1 .. 3000 = AMX Nummer des Aktors Bsp.: `SEND_COMMAND vdvEIB,'POLL=17'` Bsp.: `SEND_COMMAND vdvEIB,'POLL?17'`
IP=<IP Adresse> **(nur für IP-Gateway verfügbar)**	Setzen der IP-Adresse, unter der versucht wird, mit dem Gateway Verbindung aufzunehmen. <IP Adresse> - gültige IP Adresse (4 Bytes in Dezimaldarstellung) Bsp.: `SEND_COMMAND vdvEIB,'IP=192.168.1.200'`
IP=STOP **(nur für IP-Gateway verfügbar)**	Trennen der IP Verbindung <IP Adresse> - gültige IP Adresse (4 Bytes in Dezimaldarstellung) Bsp.: `SEND_COMMAND vdvEIB,'IP=STOP'`
IP? **(nur für IP-Gateway verfügbar)**	Abfrage der aktuell gesetzten IP Adresse Bsp.: `SEND_COMMAND vdvEIB,'IP?'`
POLLDELAY=<Wert>	Setzen der Pause zwischen (automatisierten) Wertabfragen <Wert> - 0-2 (default = 1) Hinweis : 0 bedeutet sehr schnell und sollte nicht verwendet werden, da das Gateway sonst eine hohe Buslast erzeugt. Für Anlagen, in denen Aktoren mit langsamen Busankopplern (BCU1) installiert sind, sollte der Wert 2 gewählt werden. Bsp.: `SEND_COMMAND vdvEIB,'POLLDELAY=2'`
SENDDELAY=<Wert>	Verzögerung zwischen den einzelnen Befehlen an den EIB. Wert ist Zeit in 1/10 Sekunden. Ein Wert von „0“ Deaktiviert die Verzögerung.
SET=<Nr>:<Wert>	Setzen eines Aktors Hinweis : Aktortyp bei Wertebereich beachten ! Das Modul begrenzt den Wertebereich automatisch auf den maximal zulässigen Bereich des angesprochenen Aktortyps. <Nr> - 1 .. 3000 = AMX Nummer des Aktors <Wert> - zu setzender Wert Bsp.: `SEND_COMMAND vdvEIB,'SET=5:1'`
STARTDELAY=<Wert>	Setzt den Zeitpunkt zu dem das Modul aktiviert wird, Zeit für Bootvorgang des Masters. <Wert> - Zeit in Sekunden (default = 30; Wert sollte nicht verändert werden) Bsp.: `SEND_COMMAND vdvEIB,'STARTDELAY=40'`
STATE?	Ausgabe des aktuellen Modulstatus im Terminal

Kommando	Beschreibung
TIME=<Nr>:<Zeit>	Setzen der Uhrzeit Hinweis : Nur gültig für Aktoren vom Typ 3 Byte <Nr> - 1 .. 3000 = AMX Nummer des Aktors <Zeit> - Zeit im Format hh:mm:ss Bsp.: `SEND_COMMAND vdvEIB,'TIME=8:13:15:00'`
TIME?<Nr>	Abfrage der Zeit Hinweis : Nur gültig für Aktoren vom Typ 3 Byte <Nr> - 1 .. 3000 = AMX Nummer des Aktors Bsp.: `SEND_COMMAND vdvEIB,'TIME?8'`
UPLOAD	Synchronisierung interne Tabelle – Gateway Schreibt die aktuelle interne Tabelle ins Gateway – **nur zu Debugzwecken**
VERSION	Ausgabe der aktuellen Version des Moduls im Terminal Bsp.: `SEND_COMMAND vdvEIB,'VERSION'`
WHEN=<Nr>:<Nr2>	Definition von Polltriggern. Hinweis : Der Trigger wird ausgelöst, wenn die erste Adresse angesprochen wird, nicht nur bei einer Wertänderung der ersten Adresse. <Nr> - 1 .. 3000 = AMX Nummer des Aktors, der das Polling auslöst <Nr2> - 1 .. 3000 = AMX Nummer des Aktors, der gepollt wird Bsp.: `SEND_COMMAND vdvEIB,'WHEN=32:12'`

3.3.2 Rückmeldungen vom Modul

Rückmeldungen vom Modul erfolgen immer als STRING.

Folgende Rückmeldungen können vom Modul generiert werden :

String	Beschreibung
DATE=<Nr>:<Wert>	Rückmeldung des Datums Hinweis : Wird als ***ZUSÄTZLICHE*** Rückmeldung geschickt, wenn beim Aktor <Nr> das DATE Flag gesetzt ist. <Nr> - 1 .. 3000 = AMX Nummer des Aktors <Wert> - Datumsstring im Format MM/DD/YY (AMX Darstellung) Bsp.: `DATE=17:08/14/06`
EIS5=<Nr>:<Wert>	Rückmeldung eines Wertes in ASCII Flieskommadarstellung. Der Aktorwert muß nach EIS5 kodiert sein. Hinweis : Wird als ***ZUSÄTZLICHE*** Rückmeldung geschickt, wenn beim Aktor <Nr> das EIS5 Flag gesetzt ist. <Nr> - 1 .. 3000 = AMX Nummer des Aktors <Wert> - Fliesskommawert (String), umgerechnet nach EIS5 Spezifikation Bsp.: `EIS5=12:20.25`
ERRORM=	Fehlermeldung vom Gateway und/oder Bus. Die Meldungen haben nur Informativen Charakter !
SET=<Nr>:<Wert>	Meldung einer ***Wertänderung*** Hinweise : Das Rückmeldearray (Typ Long) wird automatisch aktualisiert, unveränderte Werte werden mit VAL= gemeldet (siehe unten) <Nr> - 1 .. 3000 – AMX Nummer des Aktors <Wert> - neuer Wert des Aktors (Rohdaten) Bsp.: `SET=8:1`

String	Beschreibung
TIME=<Nr>:<Wert>	Rückmeldung der Zeit Hinweis : Wird als ***ZUSÄTZLICHE*** Rückmeldung geschickt, wenn beim Aktor <Nr> das Timeflag gesetzt ist. <Nr> - 1 .. 3000 = AMX Nummer des Aktors <Wert> - Zeitsstring im Format hh:mm:ss Bsp.: `Time=18:09:55:30`
VAL=<Nr>:<Wert>	Rückmeldung eines unveränderten Wertes (z.B. nach GET oder POLL) <Nr> - 1 .. 3000 AMX Nummer des Aktors <Wert> - Wert des Aktors

3.3.3 Terminalmodus

Zur Inbetriebnahme und zur Diagnose stehen umfangreiche Terminalfunktionen zur Verfügung. Die nachfolgenden Befehle können direkt in einer Terminalverbindung an das Modul abgesetzt werden. Diese Befehle werden per SEND_COMMAND an das virtuelle Device abgesetzt.
Um die Textausgaben ins Terminal sichtbar zu machen, muss diese mit dem Befehl

- **"MSG ON" aktiviert werden und**
- **der SEND_COMMAND ,'DEBUGON=1' an das virtuelle Device gesendet werden**

Gross-/Kleinschreibung wird bei der Eingabe ignoriert, zusätzliche Parameter (wenn vorhanden) werden durch ein **Leerzeichen** (hier in der Dokumentation mit **"_"** gekennzeichnet) abgetrennt. Selbstverständlich sind im Terminalmodus auch die Befehle der regulären Kommandoschnittstelle verfügbar (s.o., z.B. ADD=, SET=, ...)

Diese so erzeugten Rückmeldungen/Hinweise dienen ausschließlich zu Diagnosezwecken, und sollten zur Laufzeit abgeschaltet werden. Die ausgegebenen Meldungen werden durch die vorangestellte Zeichenkette "EIB:" kenntlich gemacht

Kommando	Beschreibung
ADR_<Wert>	Festlegung des Ausgabeformats der EIB Gruppenadresse (Haupt-/Mittel-/Untergruppe ODER Hauptgruppe/Untergruppe) <Wert> - 2/3 Bsp.: `SEND_COMMAND vdvEIB,'ADR 3'`
DEL_<Wert>	Löschen eines Aktors aus der Tabelle <Wert> - 1 .. 3000 – AMX Nummer Bsp.: `SEND_COMMAND vdvEIB,'DEL 3'`
HELP /?	Ausgabe der verfügbaren Terminalkommandos Bsp.: `SEND_COMMAND vdvEIB,'HELP'`
LIST	Auflistung aller eingetragenen Aktoren mit AMX Nummer, EIB Gruppenadresse, aktuellem Wert, eventuell eingetragener Flags und daraus resultierende zusätzliche Rückmeldewerte. Auflistung der Summe einzelner Typen, Summe aller Aktoren Bsp.: `SEND_COMMAND vdvEIB,'LIST'`
LIST_<Typ>	Auflistung aller eingetragenen Aktoren eines Types mit AMX Nummer, EIB Gruppenadresse, aktuellem Wert, eventuell eingetragener Flags und daraus resultierende zusätzliche Rückmeldewerte. Summe aller Aktoren eines Typs <Typ> - Datentyp, wobei SW oder SWITCH – 1Bit Aktoren D4 oder DIM4 – 4Bit Aktoren 1B oder 1BYTE – 1Byte Aktoren 2B oder 2BYTE – 2Byte Aktoren 3B oder 3BYTE – 3Byte Aktoren 4B oder 4BYTE – 4Byte Aktoren Bsp.: `SEND_COMMAND vdvEIB,'LIST 1B'`
LIST_<Nr>	Auflistung EINES Aktors (AMX Nummer) mit EIB Gruppenadresse, aktuellem Wert, eventuell eingetragener Flags und daraus resultierende zusätzliche Rückmeldewerte. <Nr> - 1 .. 3000 – AMX Nummer des Aktors Bsp.: `SEND_COMMAND vdvEIB,'LIST 17'`

Kommando	Beschreibung
LIST_<Nr>-<Nr2>	Auflistung der Aktoren im Bereich von <Nr> bis <Nr2> (AMX Nummern) mit EIB Gruppenadresse, aktuellem Wert, eventuell eingetragener Flags und daraus resultierende zusätzliche Rückmeldewerte. <Nr> - 1 .. 3000 – AMX Nummer des Aktors (Start) <Nr2> - 1 .. 3000 – AMX Nummer des Aktors (Ende) Bsp.: `SEND_COMMAND vdvEIB,'LIST 17-24'`
LIST_FLAGS	Auflistung ALLER Aktoren, denen Flags in der Tabelle zugeordnet wurden mit EIB Gruppenadresse, aktuellem Wert, den eingetragenen Flags und daraus resultierenden zusätzlichen Rückmeldewerten. Bsp.: `SEND_COMMAND vdvEIB,'LIST FLAGS'`
LIST_LOAD_[<Dateiname>]	Liest die Einträge der mit LIST_SAVE geschriebenen Tabelle von der CF zurück. Die aktuelle Tabelle wird durch die Eingelesene ersetzt. Die Angabe eines Dateinamens ist optional. Wird kein Dateinamen angegeben, wird der Default Dateiname benutzt. Default Dateiname : EIBTableNX.TXT (In einer Terminal Verbindung mit dem Master können durch Eingabe von "list" (kein SEND_COMMAND an das virtuelle Device !!!) die bereits auf der CF vorhandenen Dateien gelistet werden) Bsp.: `SEND_COMMAND vdvEIB,'LIST LOAD'` Bsp.: `SEND_COMMAND vdvEIB,'LIST LOAD MyTable.txt'`
LIST_POLL	Auflistung aller Polltrigger mit AMX Nummer und EIB Gruppenadresse Bsp.: SEND_COMMAND vdvEIB,'LIST POLL'
LIST_SAVE_[<Dateiname>]	Schreibt die aktuelle EIB Tabelle, inklusive der Polltrigger, als Textdatei auf die CF. Diese Datei kann mit einem einfachen Texteditor bearbeitet werden. Die Einträge entsprechen dem Aufbau der regulären Tabelle. Somit kann eine Tabelle zwischengespeichert werden, Modifikationen an der aktuellen Tabelle vorgenommen werden (z.B. Aktoren loeschen, hinzufügen,) und danach mit LIST LOAD (siehe oben) wieder rekonstruiert werden. Die Angabe eines Dateinamens ist optional. Wird kein Dateinamen angegeben wird der Default Dateiname benutzt. Default Dateiname : EIBTableNX.TXT (In einer Terminal Verbindung mit dem Master können durch Eingabe von "list" (kein SEND_COMMAND an das virtuelle Device !!!) die bereits auf der CF vorhandenen Dateien gelistet werden) Bsp.: `SEND_COMMAND vdvEIB,'LIST SAVE'` Bsp.: `SEND_COMMAND vdvEIB,'LIST SAVE MyTable.txt'`
LIST_SUM	Auflistung der Summe aller Typen, Summe aller Aktoren Bsp.: `SEND_COMMAND vdvEIB,'LIST SUM'`
LIST_WATCH	Auflistung des aktuell beobachteten Aktors mit EIB Gruppenadresse, aktuellem Wert, eventuell eingetragener Flags und daraus resultierende zusätzliche Rückmeldewerte. Bsp.: `SEND_COMMAND vdvEIB,'LIST WATCH'`
LIST_GAPS	Auflistung der freien (nicht benutzen) AMX Nummern. Bsp.: `SEND_COMMAND vdvEIB,'LIST GAPS'`
SEARCH <Gruppenadresse>	Suchen einer EIB Gruppenadresse Hinweis : Hier werden 2- und 3-gruppige Darstellungen akzeptiert. (Achtung ! Die Adressen 7 / 715 und 7 / 2 / 203 sind z.B. identische EIB Gruppenadressen) Bsp.: `SEND_COMMAND vdvEIB,'SEARCH 1/0/101'`
STATUS	Auflistung allgemeiner Statusinformationen zu : AMX Hardware, EIB Modul, Gateway, aktive EIB Tabelle Bsp.: `SEND_COMMAND vdvEIB,'STATUS'`

Kommando	Beschreibung
UPLOAD	Schreiben der Tabelle ins Gateway Bsp.: `SEND_COMMAND vdvEIB,'UPLOAD'`
WATCH <Nr>	Aktivierung der Beobachtungsfunktion für einen Aktor. Alle Wertänderungen werden im Terminal mitprotokolliert mit EIB Gruppenadresse, aktuellem Wert, den eingetragenen Flags und daraus resultierenden zusätzlichen Rückmeldewerten. <Nr> - 0 .. 3000 – AMX Nummer, wobei 0 der Deaktivierung der Beobachtung entspricht Bsp.: `SEND_COMMAND vdvEIB,'WATCH 12'`
WATCH_OFF	Deaktivierung der Beobachtung eines Aktors

3.3.4 Channels und Levels

Alle Adressen stehen als Channels zur Verfügung. Der aktuelle Wert wird auf den betreffenden Channels des virtuellen Devices abgebildet. Bei Wertmeldung "0" ist der Kanal aus, bei allen anderen Werten ein.

Channel	Beschreibung
1 .. 3000	Abbildung der Werte (ungeachtet des EIB-Typs)
3001	Gateway wurde erkannt, Modul ist bereit für ADD= Befehle
3002	Gateway und Modul arbeiten korrekt

Die ersten 200 Adressen (Aktoren in der EIB-Tabelle) stehen als Levels zur Verfügung. Bei jeder Wertänderung wird der aktuelle Wert als Level an das Programm übermittelt, um z.B. eine Bargraph-Anzeige anzusteuern.

Level	Beschreibung
1 .. 200	Abbildung der Werte der ersten 200 Gruppenadressen (ungeachtet des EIB-Typs)

4 Datentypen

'Switch'	-	Wert '0' oder '1'	(z.B. AUS – EIN)
'4 Bit'	-	Wert '0' bis '15'	(z.B. relatives Dimmen – Richtung,Bereich)
'1 Byte'	-	Wert '0' bis '255'	(z.B. Wertsetzung absolut)
'2 Byte'	-	Wert '0' bis '65535'	(z.B. Fliesskommawert in EIS5 Notation)
'3 Byte'	-	3 Byte	(z.B. Datum oder Zeit)
'4 Byte'	-	4 Byte	
'Text'	-	1 bis 14 ASCII Zeichen, String wird automatisch mit Leerzeichen aufgefüllt	
'HEXText'	-	1 bis 14 Byte Hexadezimalwert in ASCII-Notation	

5 Anhang A – Beispielprogramm

5.1 Hinweise zur EIB-Tabelle

Alle Aktoren, die geschaltet/gesetzt/überwacht werden sollen, müssen dem Gateway bekannt gemacht werden. Dies geschieht durch den Aufbau einer Tabelle.

Wird diese Tabelle mit einer Hilfsdatei (z.B. ‚EIB_Table.AXI') wie im Beispielprogramm generiert, muss lediglich beachtet werden, dass auch alle Einträge bearbeitet werden, also die letzte Switch_Case Anweisung (default :) auch als letztes abgearbeitet wird. Es darf also keine Lücken im Zähler geben. Bei den eigentlichen Einträgen hingegen muss keinerlei Reihenfolge beachtet werden.

Empfehlung : Verwendung der Variante mit den Hilfsfunktionen (**Beispiel 1,** siehe unten). Bei dieser Variante werden weniger Tippfehler entstehen und der Compiler kann bereits diverse Prüfungen auf Richtigkeit vornehmen.

Polltrigger werden nur eingetragen, wenn die pollende und die gepollte Adresse bereits eingetragen sind. Wir empfehlen die Polltrigger am Ende einzutragen.

Alternativ kann die Tabelle auch von der Compact Flash des Masters geladen werden. Die Syntax ist identisch mit dem SEND_COMMAND „ADD=" Kommando. Das Send Command und die Device Adresse entfallen hierbei. (**Beispiel 3,** siehe unten).

5.2 Hinweise zur Programmierung

Zur Ansteuerung stehen **vordefinierte Funktionen** zur Verfügung, die SEND_COMMANDs an das virtuelle Device generieren.

Wir empfehlen nicht direkt die Send Commands zu verwenden sondern immer die Funktionen der EIB_Tools.AXI Include-Datei einzusetzen (Anhang B). Der Compiler hat hierdurch die Möglichkeit Tippfehler bereits beim Compilevorgang abzufangen. Zusätzlich wird einem Schreibarbeit abgenommen.

5.2.1 Beispiel 1 – Aufbau der EIB Tabelle mit Funktionen aus EIB_Tools.AXI

Hinweis : Im folgenden Beispiel werden bei den cases 14,15,17 und 18 zusaetzliche Flags mit String-Konstanten eingetragen. Diese Konstanten werden ebenfalls in der Datei EIB_Tools.AXI deklariert. Eine alternative Notation wäre (z.B. fuer Case 15)

```
EIBAdd(vdvEIB, 15, eib2Byte, '1/0/201', 'EIS5,PS')
```

```
PROGRAM_NAME='EIB_Table'

DEFINE_VARIABLE
INTEGER nCounter


DEFINE_START

nCounter = 0
WAIT_UNTIL ([vdvEIB, 3001])   // Startverzögerung, Modul meldet bereit für ADD
{
    nCounter = 1   // Startfreigabe
}


#INCLUDE 'EIB_Tools.axi'

DEFINE_PROGRAM

WAIT 1
{
    IF(nCounter)
    {
        SWITCH(nCounter)
        {
            CASE  1: EIBAdd(vdvEIB,  1, eibSwitch, '1/0/1', "")                // Licht 1 setzen
            CASE  2: EIBAdd(vdvEIB,  2, eibSwitch, '1/0/2', "")                // Licht 2 setzen
            CASE  3: EIBAdd(vdvEIB,  3, eibSwitch, '1/0/3', "")                // Licht 3 setzen
            CASE  4: EIBAdd(vdvEIB,  4, eibSwitch, '1/0/11', "eibPollstart")       // Licht,  Zustand
            CASE  5: EIBAdd(vdvEIB,  5, eibSwitch, '1/0/12', "eibPollstart")       // Licht 2 Zustand
            CASE  6: EIBAdd(vdvEIB,  6, eibSwitch, '1/0/13', "eibPollstart")       // Licht 3 Zustand

            CASE  7: EIBAdd(vdvEIB,  7, eibSwitch, '1/0/21', "")               // Lichtszenen,1+2
            CASE  8: EIBAdd(vdvEIB,  8, eibSwitch, '1/0/22', "")               // Lichtszenen 3+4

            CASE  9: EIBAdd(vdvEIB,  9, eibSwitch, '1/0/31', "")               // Jalousie, Auf/Ab
            CASE 10: EIBAdd(vdvEIB, 10, eibSwitch, '1/0/32', "")               // Jalousie, Lamellen

            CASE 11: EIBAdd(vdvEIB, 11, eibSwitch, '1/0/111', "")              // Dimmer,  Ein/Aus
            CASE 12: EIBAdd(vdvEIB, 12, eibDim4,   '1/0/112', "")              // Dimmer relativ
            CASE 13: EIBAdd(vdvEIB, 13, eib1Byte,  '1/0/113', "")              // Dimmer absolut
            CASE 14: EIBAdd(vdvEIB, 14, eib1Byte,  '1/0/114', "eibPollstart")  // Dimmer Wert

            CASE 15: EIBAdd(vdvEIB, 15, eib2Byte,  '1/0/201', "eibEIS5, ',',eibPollstart")
            CASE 16: EIBAdd(vdvEIB, 16, eib1Byte,  '1/0/203', "")              // Analogwert 2

            CASE 17: EIBAdd(vdvEIB, 17, eib3Byte,  '1/0/205', "eibTIME, ',',eibPollstart")// Zeit
            CASE 18: EIBAdd(vdvEIB, 18, eib3Byte,  '1/0/206', "eibDATE, ',',eibPollstart")// Datum

            CASE 19: EIBWhenPoll(vdvEIB, 1, 2)                                 // Polltrigger
            CASE 20: EIBWhenPoll(vdvEIB, 1, 3)                                 // Polltrigger

            CASE 21: EIBWhenPoll(vdvEIB, 11, 14)                               // Polltrigger
            CASE 22: EIBWhenPoll(vdvEIB, 12, 14)                               // Polltrigger
            CASE 23: EIBWhenPoll(vdvEIB, 13, 14)                               // Polltrigger
            CASE 24: EIBWhenPoll(vdvEIB, 14, 11)                               // Polltrigger

            DEFAULT: nCounter = 0
        }
        IF (nCounter)  nCounter ++
    }
}
```

5.2.2 Beispiel 2 – Aufbau der EIB Tabelle mit SEND_COMMANDS

```
PROGRAM_NAME='EIB_Table'

DEFINE_VARIABLE
INTEGER nCounter


DEFINE_START

nCounter = 0
WAIT_UNTIL ([vdvEIB, 3001])   // Startverzögerung, Modul meldet bereit für ADD
{
    nCounter = 1       // Startfreigabe
}



DEFINE_PROGRAM

WAIT 1
{
    IF(nCounter)
    {
        SWITCH(nCounter)
        {
            CASE  1 : SEND_COMMAND vdvEIB, "'ADD=1:Switch:1/0/1'"               // Licht 1 setzen
            CASE  2 : SEND_COMMAND vdvEIB, "'ADD=2:Switch:1/0/2'"               // Licht 2 setzen
            CASE  3 : SEND_COMMAND vdvEIB, "'ADD=3:Switch:1/0/3'"               // Licht 3 setzen
            CASE  4 : SEND_COMMAND vdvEIB, "'ADD=4:Switch:1/0/11:PS'"           // Licht 1 Zustand
            CASE  5 : SEND_COMMAND vdvEIB, "'ADD=5:Switch:1/0/12:PS'"           // Licht 2 Zustand
            CASE  6 : SEND_COMMAND vdvEIB, "'ADD=6:Switch:1/0/13:PS'"           // Licht 3 Zustand

            CASE  7 : SEND_COMMAND vdvEIB, "'ADD=7:Switch:1/0/21'"              // Szene 1+2
            CASE  8 : SEND_COMMAND vdvEIB, "'ADD=8:Switch:1/0/22'"              // Szene 3+4

            CASE  9 : SEND_COMMAND vdvEIB, "'ADD=9:Switch:1/0/31'"              // Jalousie Auf/Ab
            CASE 10 : SEND_COMMAND vdvEIB, "'ADD=10:Switch:1/0/32'"             // Jalousie Lamellen

            CASE 11 : SEND_COMMAND vdvEIB, "'ADD=11:Switch:1/0/111'"            // Dimmer, Ein/Aus
            CASE 12 : SEND_COMMAND vdvEIB, "'ADD=12:Dim4:1/0/112'"              // Dimmer relativ
            CASE 13 : SEND_COMMAND vdvEIB, "'ADD=13:1Byte:1/0/113'"             // Dimmer absolut
            CASE 14 : SEND_COMMAND vdvEIB, "'ADD=14:1Byte:1/0/114:PS'"          // Dimmer Wert lesen

            CASE 15 : SEND_COMMAND vdvEIB, "'ADD=15:2Byte:1/0/201:EIS5,PS'"     // Analogwert
            CASE 16 : SEND_COMMAND vdvEIB, "'ADD=16:1Byte:1/0/203'"             // Analogwert

            CASE 17 : SEND_COMMAND vdvEIB, "'ADD=17:3Byte:1/0/205:Time,PS'"     // Zeit
            CASE 18 : SEND_COMMAND vdvEIB, "'ADD=18:3Byte:1/0/206:Date,PS'"     // Datum

            CASE 19 : SEND_COMMAND vdvEIB, "'WHEN=1:2'"                         // Polltrigger
            CASE 20 : SEND_COMMAND vdvEIB, "'WHEN=1:3'"                         // Polltrigger

            CASE 21 : SEND_COMMAND vdvEIB, "'WHEN=11:14'"                       // Polltrigger
            CASE 22 : SEND_COMMAND vdvEIB, "'WHEN=12:14'"                       // Polltrigger
            CASE 23 : SEND_COMMAND vdvEIB, "'WHEN=13:14'"                       // Polltrigger
            CASE 24 : SEND_COMMAND vdvEIB, "'WHEN=14:11'"                       // Polltrigger

            DEFAULT:  nCounter = 0
        }
        IF (nCounter)  nCounter ++
    }
}
```

5.2.3 Beispiel 3 – Eintragen aus einer Datei

Die Adresstabelle kann auch aus einer Datei auf der CF-Karte gelesen und generiert werden. Das Einlesen der Datei kann z.B. in der ONLINE-Sektion der Schnittstelle gestartet werden.
(Beschreibung siehe Kapitel “Terminal-Modus“)

```
...
DATA_EVENT[dvEIB]
{
  ONLINE :
  {
    SEND_COMMAND vdvEIB,‘LIST LOAD MyTable.txt‘
  }
}
...
```

Hinweis: Kommentare am Ende einer Zeile müssen mindestens durch ein Leerzeichen getrennt sein, und werden - analog zur Programmierung - mit “***//***“ eingeleitet.
Es ist nur ein Befehl pro Zeile erlaubt. Führende Leerzeichen werden Ignoriert.
Zeilen, die mit einem “***//***“ (Doppelslash) beginnen, werden gesamt als Kommentar betrachtet und von der Steuerung nicht bearbeitet.

```
// EIB – Tabelle fuer Projekt xy
//
// ab hier : Gruppenadressen
//
ADD=1:Switch:1/0/1            // Licht 1 setzen
ADD=2:Switch:1/0/2            // Licht 2 setzen
ADD=3:Switch:1/0/3            // Licht 3 setzen
ADD=4:Switch:1/0/11:PS        // Licht 1 Zustand, beim Start abfragen
ADD=5:Switch:1/0/12:PS        // Licht 2 Zustand, beim Start abfragen
ADD=6:Switch:1/0/13:PS        // Licht 3 Zustand, beim Start abfragen
ADD=7:Switch:1/0/21           // Szene 1+2
ADD=8:Switch:1/0/22           // Szene 3+4
ADD=9:Switch:1/0/31           // Jalousie Auf/Ab
ADD=10:Switch:1/0/32          // Jalousie Lamellen
ADD=11:Switch:1/0/111         // Dimmer Ein/Aus
ADD=12:Dim4:1/0/112           // Dimmer relativ
ADD=13:1Byte:1/0/113          // Dimmer absolut
ADD=14:1Byte:1/0/114:PS       // Dimmer Wert lesen, beim Start abfragen
ADD=15:2Byte:1/0/201:EIS5,PS  // Analogwert, beim Start abfragen
ADD=16:1Byte:1/0/203          // Analogwert
ADD=17:3Byte:1/0/205:Time,PS  // Zeit, beim Start abfragen
ADD=18:3Byte:1/0/206:Date,PS  // Datum, beim Start abfragen
//
// ab hier : Polltrigger
//
WHEN=1:2                      // Polltrigger
WHEN=1:3                      // Polltrigger
WHEN=11:14                    // Polltrigger
WHEN=12:14                    // Polltrigger
WHEN=13:14                    // Polltrigger
WHEN=14:11                    // Polltrigger
```

5.2.4 Beispiel 4 – Hauptprogramm

```
DEFINE_DEVICE

dvEIB    =  5001:1:0
vdvEIB   =  33001:1:0

dvPanel  =  10001:1:0


DEFINE_CONSTANT
...


DEFINE_VARIABLE

VOLATILE LONG lEIB_Value[3000]   // Rückmeldearray für max 3000 Adressen
...


DEFINE_START
...


#INCLUDE 'EIB_Table.AXI'
DEFINE_MODULE 'CTEIB6_mod' GATEWAY_1 (vdvEIB, dvEIB, lEIB_Value)


DEFINE_EVENT
...

// nur für Ethernet-Gateway :

DATA_EVENT[vdvEIB]
{
  ONLINE :
  {
    SEND_COMMAND vdvEIB,'IP=172.16.100.123'  // Setzen der IP-Adresse
  }
}



// fuer alle Varianten

BUTTON_EVENT[dvPanel,1]
{
   PUSH :
   {
      EIBSet(vdvEIB,1,1)         // Licht 1 EIN
      EIBSet(vdvEIB,16,128)      // Analogaktor auf 50%
      EIBSet(vdvEIB,12,10)       // Dimmer heller
   }
   RELEASE :
   {
      EIBSet(vdvEIB,12,0)        // Dimmer Stop
   }
}

DEFINE_PROGRAM
...

[dvPanel,11] = lEIB_Value[4]        // Rückmeldung für Licht 1 (siehe EIB Tabelle)
[dvPanel,12] = (lEIB_Value[16] > 127)  // Taste leuchtet, wenn Aktor >= 50%
```

6 Anhang B – EIB_Tools.AXI

Wir empfehlen nicht direkt die Send Commands zu verwenden sondern immer die Funktionen dieser Include-Datei einzusetzen. Der Compiler hat hierdurch die Möglichkeit Tippfehler bereits beim Compilevorgang abzufangen. Zusätzlich wird einem Schreibarbeit abgenommen.

Diese Datei stellt auch Konstanten zum relativen Dimmen bereit :

```
EIB_DIM_UP = 9        // Heller, Brighter
EIB_DIM_DN = 1        // Dunkler, Dark
EIB_DIM_SP = 0        // Dimmen Stop, Dimming Stop
```

Folgende Funktionen stehen in der Datei EIB_Tools.AXI zur Programmierung zur Verfügung :
(eine genaue Beschreibung der Parameter finden Sie im Kapitel

EIBSet(<virtuelles Device>,<AMXNr>,<Wert>)
Funktion : Setzt Aktor <AMXNr> auf <Wert>
Das Modul begrenzt den Wertebereich automatisch auf den maximal zulässigen Bereich des angesprochenen Aktortyps.
Bsp.: `EIBSet(vdvEIB,13,1)`

EIBGet(<virtuelles Device>,<AMXNr>)
Funktion : Holt den im Modul gespeicherten Wert des Aktors <AMXNr>
Bsp.: `nVal = EIBGet(vdvEIB,13)`

EIBPoll(<virtuelles Device>,<AMXNr>)
Funktion : Pollt den Aktor <AMXNr>
Bsp.: `EIBPoll(vdvEIB,13)`

EIBAdd(<virtuelles Device>,<AMXNr>,<Typ>,<Gruppenadresse>,<Flags>)
Funktion : Fügt einen Eintrag zur EIB Tabelle hinzu (Parameterbeschreibung siehe Kap. 3.3.1)
Bsp.: `EIBAdd(vdvEIB,13,eib2Byte,'1/0/206',"eibPollstart"`

EIBWhenPoll(<virtuelles Device>,<AMXNr1>,<AMXNr2>)
Funktion : Fügt einen Polltrigger hinzu. Wertmeldung von <AMXNr1> löst einen Pollvorgang auf <AMXNr2> aus.
Bsp.: `EIBWhenPoll(vdvEIB,13,20)`

7 Anhang C – Vergleich zur Vorgängerversion

Die prinzipielle Arbeitsweise der Hard- und Software haben sich nicht geändert. Es muss nach wie vor eine Filtertabelle generiert werden, die ins Gateway geladen wird. Über diese Tabelle werden die ein- und ausgehenden Telegramme gefiltert. Die Ansteuerung der Aktoren, bzw. die Rückmeldungen von Sensoren erfolgt über das Setzen von Werten, welche bestimmten Adressen zugeordnet sind.

- um die von der AMX erzeugte Buslast auf dem EIB zu begrenzen, werden
  - beim Pollen ca. 10 Telegramme pro Sekunde abgesetzt,
  - bei der Ansteuerung ca. 35 Telegramme (bisher ca. 6-10 Telegramme)
- Telegramme werden bis zur maximalen Buslast fehlerfrei gelesen
- resourcenschonende Arbeitsweise des Gateways und des Treibers. Eine permanente Kommunikation mit dem Gateway über die serielle Schnittstelle entfällt
- der Treiber wurde auf eine einfache Nutzung in vernetzten Systemen optimiert
- der Treiber erleichtert die Überprüfung der Ansteuerung auch ohne betriebsbereiten EIB. (siehe auch Terminalkommando “DEBUGON“)

7.1 Aufbau / Generierung der Filtertabelle im Gateway

Bisher : `SEND_COMMAND vdvEIB,“ADD SW 4 1/0/65“`

Beim Aufbau der Tabelle bekommen alle Aktoren/Sensoren jetzt eine eindeutige Nummer (1 .. 3000), unabhängig vom Typ. Über diese Nummer werden die Aktoren/Sensoren angesprochen. Zusätzlich können diverse Flags angegeben werden, im nachfolgenden Beispiel z.B. “PS“ . Dies bedeutet, dass der betreffende Aktor beim Start der AMX sofort gepollt wird. (Beschreibung der verfügbaren Flags siehe Kapitel ‘Kommandos zum Modul‘)

NEU : `EIBAdd(vdvEIB,4,Switch,'1/0/65',“PS“)` (empfohlen !) oder
`SEND_COMMAND vdvEIB,“'ADD=4:Switch:1/0/1:PS‘“`

7.2 Ansteuerung von Aktoren

Bisher : `SYSTEM_CALL ‘EIB_NX Set Switch‘(vdvEIB,1,4)`

NEU : `EIBSet(vdvEIB,4,1)` (empfohlen !) oder
`SEND_COMMAND vdvEIB,“‘SET=4:1‘“`

7.3 Rückmeldungen

Es gibt nur noch EIN Array, in dem die Werte ALLER Aktoren/Sensoren gespeichert werden. Zusätzlich können Rückmeldungen – je nach angegebenen Flags – zusätzlich in einer lesbaren ASCII-Darstellung ausgegeben werden, d.h. die EIB Rohdaten werden als Zeitstring, Datumsstring, Fliesskommadarstellung, ... ausgegeben.

Beispiel :

Rückmeldung eines 2Byte Wertes, welcher nach dem EIS5 Standard gewandelt wird (z.B. Temperaturwert). Der betreffende Aktor wurde beim Eintragen in die Filtertabelle mit dem Flag „EIS5“ versehen

```
EIBAdd(vdvEIB, 15, eib2Byte,  '1/0/201', "eibEIS5")
```

Das virtuelle Device gibt nun bei jeder Wertänderung (bzw. als Antwort auf einen Pollbefehl) zwei Rückmeldungen :

String 1 vom virtuellen Device (bei Wertänderung) : `'SET=15:3175'`
String 2 vom virtuellen Device : `'EIS5=15:22.54'`

oder

String 1 vom virtuellen Device (keine Wertänderung) : `'VAL=15:3175'`
String 2 vom virtuellen Device : `'EIS5=15:22.54'`

8 Anhang D – Was tun wenn es mal nicht geht ?

Die folgende Tabelle gibt einige Hinweise zur Fehlereingrenzung, falls es mal nicht funktioniert.
Dies dient zur schnellen Fehlerdiagnose **VOR ORT**.
Wir empfehlen bei der Diagnose den Debugmodus zu aktivieren um zusätzliche Fehlermeldungen im Terminal zu erhalten. Dies wird mit dem Terminalbefehl „DEBUGON“ möglich.

Fehler	Mögliche Abhilfe / Fehlereingrenzung
Keine Steuerung möglich, keine Rückmeldungen	Im Terminalmodus den Befehl „Status“ eingeben. Wird das Gateway nicht erkannt ist vermutlich das Kabel falsch / defekt oder das Gateway hat keinen Strom
Keine Steuerung möglich, keine Rückmeldungen, das Gateway wird laut „Status“ erkannt	Im Terminalmodus den Befehl „List“ eingeben. Sind alle Adressen eingetragen ? Werden hier Werte für die Rückmeldung angezeigt. Mit dem Terminalbefehl „SET“ versuchen einzelne Adressen direkt zu schalten (z.B. Licht). Funktionert dies, liegt vermutlich ein Fehler im AMX Programm vor. Ist hier ebenfalls keine Ansteuerung möglich, liegt es vermutlich an falschen Gruppenadressen. (Brennt im Nachbargebäude noch das Licht ?)
Modul meldet immer wieder „no STX at beginning of GW Feedback“	Das Gateway wird vom Modul gefunden, beim Versuch die Version auszulesen wird die Verbindung aber wieder unterbrochen. Lösung : Bitte Kabelverbindung von der AMX zum Gateway überprüfen.