

CTEIB4

AMX goes Instabus®



Bedienungsanleitung für das Comm-Tec EIB-Gateway AXG-EIB und der Ansteuersoftware CTEIB4

Version 4
Stand: 1.Juni 2002

© COMM-TEC
Siemensstr. 14, 73066 Uhingen
Tel.: +49/7161/3000-0
Fax: +49/7161/3000-400

Inhaltsverzeichnis

INHALTSVERZEICHNIS	2
VORWORT.....	4
ZIELSETZUNG	4
ROLLE DES EIB-INSTALLATEURS	4
EIB AUS DER SICHT DES AMX-PROGRAMMIERERS	4
EIB-BESONDERHEITEN BEIM EINSATZ DES GATEWAYS	5
VORWORT ZUR REVISION 4.....	6
SYSTEMBESCHREIBUNG.....	7
HARDWAREKOMPONENTEN	7
SOFTWAREKOMPONENTEN	8
GRUPPEN, ADRESSEN UND FORMATE	10
TRIGGER.....	11
INSTALLATION UND INBETRIEBNAHME.....	12
INSTALLATION DES GATEWAYS	12
INSTALLATION DER RS232-BOX/-KARTE	12
VERBINDUNG GATEWAY-BOX	13
VERBINDUNG DER BOX MIT DEM MASTER	13
EINSPIELEN DES BOX-PROGRAMMS	13
GRUPPENADRESSEN IN DIE BOX EINTRAGEN	14
VERSORGUNG DER BOX AUS DEM HAUPTPROGRAMM	14
KOMMUNIKATION MIT DER BOX	15
SYSTEM CALLS	15
SETZEN VON WERTEN	16
RÜCKMELDUNGEN	16
AKTIVE ABFRAGEN (POLLING)	16
BEFEHLE AN DIE BOX.....	17
DATENKONVERTIERUNG	17
TERMINAL-MODUS	18
KANÄLE UND LEVELS.....	18
BEFEHLSSATZ DER BOX.....	19
ADD – GRUPPENADRESSE EINTRAGEN.....	19
ADR – FORMAT DER ADRESSDARSTELLUNG UMSCHALTEN	19
BOXCH – ABBILDUNG DER LOGISCHEN KANÄLE DER BOX AUF 1-BIT AKTOREN	19
BURST - BEGRENZUNG DER SENDEGESCHWINDIGKEIT	19
DELTA – DIFFERENTIALÜBERTRAGUNG EIN/AUS	20
LIST – ANZEIGE DER BESTEHENDEN VERKNÜPFUNGEN	20
NOPOLL - LÖSCHEN EINES POLL-TRIGGERS	21
POLL - AKTIVE WERTABFRAGE.....	21
RESEND - ALLE WERTE ERNEUT SCHICKEN.....	21
RESET - GATEWAY-RESET AUSLÖSEN	21
SET - GRUPPENADRESSE SCHREIBEN	21
START - STARTEN DES UPDATE-PROZESSES.....	21
STATUS – STATUSINFORMATION ANZEIGEN	22
STOP – ANHALTEN DES UPDATE-PROZESSES	22
UPLOAD - GATEWAY-FILTERTABELLE NEU LADEN	22
WATCH - ÜBERWACHUNG EINER ADRESSE	22
WHEN .. POLL - SETZEN EINES POLL-TRIGGERS.....	23

ANHANG.....	24
VERGLEICH „ALTE“ CALLS – NEUE CALLS.....	24
WERTE FÜR DIM4-AKTOREN.....	25
BEISPIEL: EINTRAGEN DER GRUPPEN	26
BEISPIEL: HAUPTPROGRAMM.....	27
LOKALE CALLS ALS ERSATZ DER „ALTEN“ INITIALISIERUNG	29
FAQS.....	30
FEHLERMELDUNGEN	31

Vorwort

Zielsetzung

Das Softwarepaket CTEIB4 dient der Ankopplung von AMX-Steuerungen an den "European Installation Bus" EIB ("Instabus"). Es stellt eine einfach zu verwendende Schnittstelle für den Entwickler zur Verfügung und bietet ein hohes Maß an Komfort im Umgang mit dem Bus.

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.

CTEIB4 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*.

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 das Telegramm sendet. Eine Zieladresse ist eine Gruppen-Adresse, eine Adresse, die eine *Funktion* beschreibt.

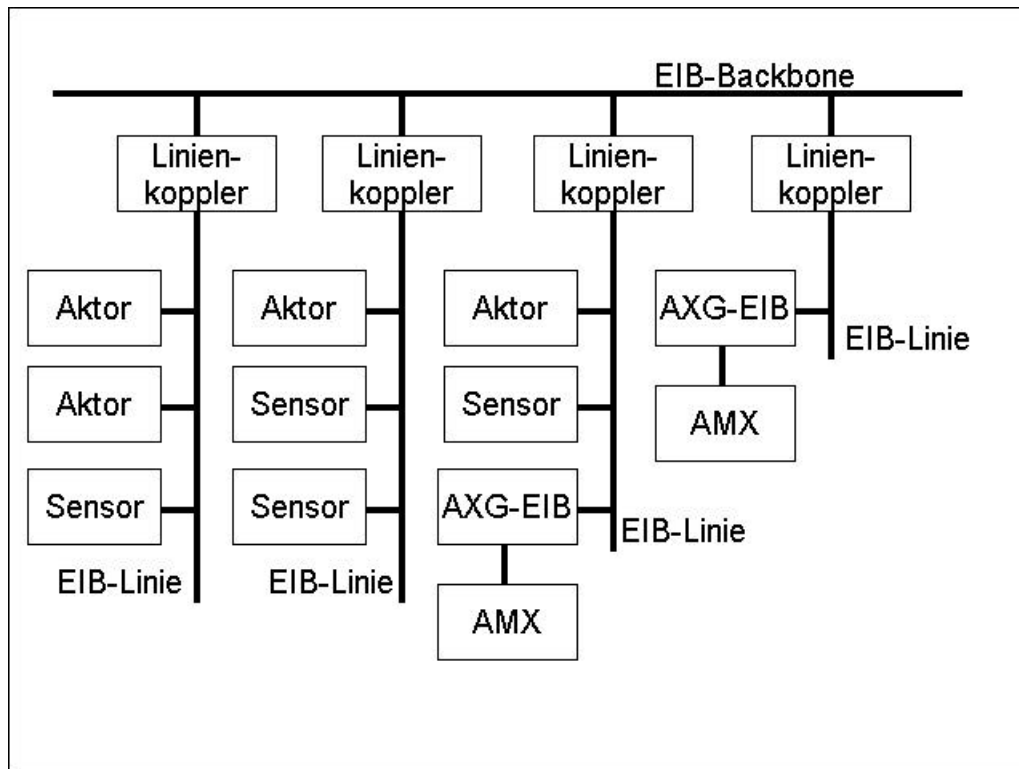
Jedes Gerät am EIB hat also genau eine Hardware-Adresse, kann aber durchaus mehrere Gruppenadressen haben. Ebenso ist es möglich, daß *mehrere* Geräte auf die selbe Gruppenadresse reagieren.

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.

EIB-Besonderheiten beim Einsatz des Gateways



Prinzipiell ist das EIB-Gateway AXG-EIB zwar ein gewöhnliches EIB-Gerät, kann also an jeder Stelle mit dem EI-Bus verbunden werden. Im Gegensatz zu einfachen Aktoren und Sensoren ist es aber unter Umständen für sehr viele (bis zu 1530) 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 EI-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 die - 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 – der Gateway-Busankoppler sollte also die Gelegenheit haben, *alle* Telegramme mitzulesen

Problem: die erste Quittung reicht, und die muss nicht unbedingt vom Gateway stammen!

Tatsächlich kann es vorkommen, dass ein an der Szene beteiligter Aktor ein Telegramm quittiert, das Gateway aber nichts davon mitbekommt, weil es noch mit der Verarbeitung des vorherigen beschäftigt ist... Sollte das vorkommen, gibt es zwei Taktiken:

1. Der Szenebaustein wird so programmiert, daß nicht die maximale Busbandbreite (ca. 50 Telegramme/Sekunde) ausgenutzt wird, sondern nur etwa zehn Telegramme pro Sekunde gesendet werden.
2. Das Gateway bekommt eine eigenen Linie, wird also von den „Zielaktoren“ isoliert. Dann kann der Linienkoppler sicherstellen, daß alle Pakete *an* das Gateway auch *vom* Gateway quittiert werden (sofern der Busankoppler des Linienkopplers schnell genug ist ;-)).

Vorwort zur Revision 4

Die grundlegende technische Überarbeitung des Gateways hatte zur Folge, dass die Software der AXB-232++ (AXC-232++) angepasst wurde.

Hauptkriterien für die neue Software waren

- ein möglichst reibungsloser Umstieg auf das neue Produkt, unter Berücksichtigung der bereits erworbene Kenntnisse im Umgang mit den früheren Revisionen
- die Möglichkeit bereits bestehende Installationen mit dem neuen Gateway auszustatten
- Kompatibilität in Kombination mit Controllern der NetLinx-Generation.

Sie können weiterhin Ihre AMX – Anbindung an EIB-System wie bisher in gewohnter Weise realisieren. Die 232++ - interne Software erkennt automatisch das angeschlossene Gateway - Sie müssen sich nicht um geänderte Speicherverwaltung oder Ähnliches kümmern.

Einzige Neuerung : die Datenfelder mit den aktuellen Switch-, Dim4-, 1Byte-, 2Byte-, 3Byte-, 4Byte- Werten müssen als Integer-Arrays deklariert werden (siehe Beispiel Seite 28).

Die mitgelieferten SYSTEM_CALLs haben sich in ihrer Funktion nicht geändert, zum Betrieb mit einem Gateway der neuen Generation müssen allerdings die der neuen Revision verwendet werden.

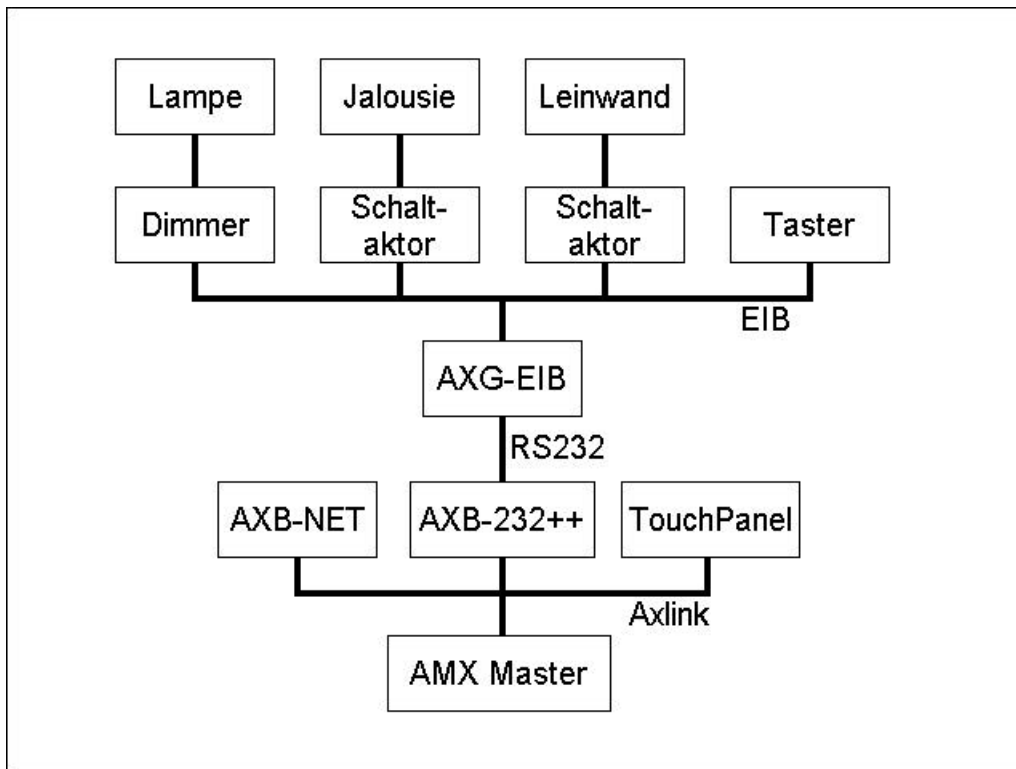
Weiterhin kann das Gateway im laufenden Betrieb nicht gewechselt werden, ohne den Master (und natürlich die 232++) zurückzusetzen.

Die Software-Version 4 inclusive der dazugehörigen SYSTEM_CALLs arbeitet mit allen Gateways zusammen.

Uhingen, Juni 2002

Systembeschreibung

Hardwarekomponenten



Zur Anbindung eines AMX-Systems an EIB sind mehrere Hardware-Komponenten notwendig:

- ein AMX-Master, in dem das Hauptprogramm Steueranweisungen erzeugt und Rückmeldungen vom EIB entgegennimmt (nachfolgend „Master“ genannt)
- ein EIB-Gateway AXG-EIB zur physikalischen Verbindung mit dem EIB, enthält den Busankoppler BA und einen konfigurierbaren Paketfilter (nachfolgend „Gateway“)
- eine intelligente RS232-Schnittstelle AXB-232++ (Busdevice) oder AXC-232++ (Einschubkarte) zur Verwaltung der EIB-Meldungen und –Befehle (nachfolgend „Box“).

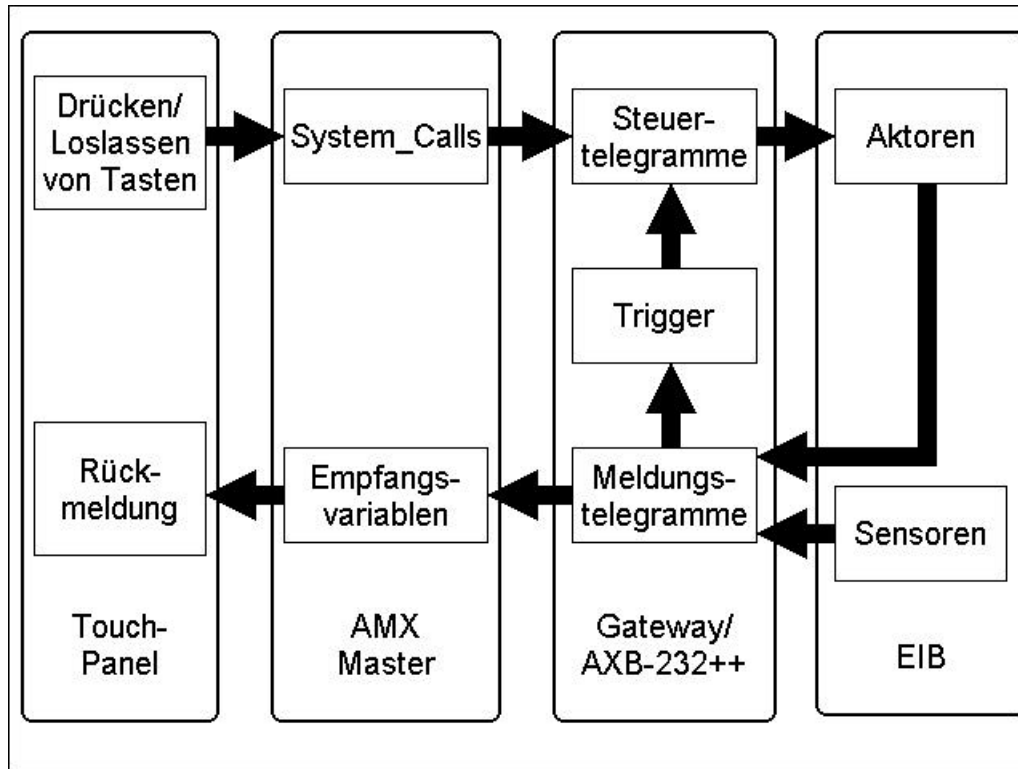
Die kompletten Konfigurationsdaten des EIB liegen in der Box, nicht zwingend im Hauptprogramm. Es ist zwar möglich, sämtliche Adressen beim Neustart des Systems neu in die Box zu laden (per System_Call), zwingend notwendig ist das zum Betrieb aber nicht. Es genügt, wenn das Hauptprogramm die Bedeutung der einzelnen "Kanäle" (Typ/Nr) kennt.

Bei jeder Änderung der Adressdaten und/oder Typen aktualisiert die Box automatisch die Daten des Paketfilters im Gateway AXG-EIB. Dadurch ist gewährleistet, dass alle verwendeten Meldungen auch passieren dürfen (sofern sie EIB-seitig erzeugt werden...), und dass kein Datenmüll aus nicht (mehr) verwendeten Gruppen das System belastet. Bei jeder neuen Kontaktaufnahme mit dem Gateway wird die Gateway-intern generierte Prüfsumme mit einer gespeicherten Version in der Box verglichen und gegebenenfalls eine Neukonfiguration des Gateways eingeleitet.

Das Verfahren ähnelt AMX-seitig also mehr dem Umgang mit einem TouchPanel: sämtliche Aktionen laufen über Kanal-Nummern, Level-Nummern und Text-Nummern mit fester Bedeutung ab; das Aussehen der einzelnen Knöpfe und Seiten spielt für das Programm keine Rolle.

Ebenso wie beim TouchPanel werden aber auch hier die Anwendungsdaten in einem zwar batteriegepufferten, dennoch aber flüchtigen Speicher gehalten. Es empfiehlt sich daher, nach jeder Änderung der Konfiguration eine maschinenlesbare Version aufzubewahren, um im Notfall den Betrieb schnell wieder aufnehmen zu können.

Softwarekomponenten



Das Programmpaket CTEIB3 umfaßt ein Steuerprogramm für die RS232-Box AXB-232++ und mehrere Programmbibliotheken (System_Calls), die in eigene (Haupt-) Programme eingebunden werden können.

Das Steuerprogramm wird in die Box (oder die Einschubkarte) geladen und deren Schnittstelle mit dem Gateway AXG-EIB verbunden. Das Programm ist fertig übersetzt (kompiliert), und muß zur Installation nur noch eingespielt werden. Sämtliche Einstellungen werden vom AMX-Master aus vorgenommen.

AMX-seitig stellt sich das Gateway durch die Verwendung einer RS232-Box als "ganz normales" Busdevice dar, man könnte sich also auch "ganz normal" über Send_Command/Send_String-Befehle unterhalten.

Durch die Telegrammstruktur des EIB und die Vielfalt der möglichen Meldungen und Befehle sind jedoch die Anforderungen an die Empfangs- und Sendelogik inklusive Paket-Verarbeitung relativ hoch. Nicht zuletzt aus Performancegründen (Reaktionszeit) sollte daher auf die mitgelieferten Bibliotheksfunktionen zurückgegriffen werden.

Sie stellen eine Reihe von Funktionen bereit, die einfach - per System_Call-Aufruf - in eigene Programme eingebunden werden können. Durch einen modularen Aufbau werden nur die wirklich benötigten Programmteile eingebunden, um die Ressourcen des Masters (Speicher, Rechenzeit, Buslast) möglichst zu schonen.

Datenmodell

EIB stellt mehrere EIS-genormte Datentypen bereit, die verschiedenartigste Aufgaben wahrnehmen sollen. Neben simplen Ein/Aus-Schalt-Typen gibt es spezielle Datentypen für vorzeichenlose Zahlen in verschiedener Länge, vorzeichenbehaftete Zahlen, Fließkommawerte, Prozentangaben, Datum, Uhrzeit, Windrichtungen und alle möglichen sonstigen meßbaren Dinge.

Das braucht in der (AMX-)Praxis kein Mensch, und CTEIB4 verwendet ein vereinfachtes Datenmodell, das die häufigsten Anwendungsfälle abdeckt, gleichzeitig aber völlig transparent ist, um beliebige Zugriffe zu erlauben.

Insgesamt stehen 6 verschiedene Datentypen zur Verfügung:

Switch	Binärer Schalter, Ein/Aus
Dim4	4-Bit Dimm-Aktor. Häufig verwendeter Ansteuertyp für Dimmer
1Byte	Sämtliche EIB-Datentypen mit einem Byte (8 Bit) Länge
2Byte	2-Byte-EIB-Typen (16 Bit)
3Byte	3-Byte-EIB-Typen (24 Bit)
4Byte	4-Byte-EIB-Typen (32 Bit)

Die beiden erstgenannten sind definierte EIB-Typen, die wohl mindestens 50% der Funktionen der meisten Anlagen "erschlagen" dürften.

Alle anderen EIB-Typen werden auf die Typen 1Byte..4Byte abgebildet - je nach Länge. Ein Element vom Typ 1Byte kann also eine 8 Bit Integer-Zahl sein (-128..+127), aber auch eine Prozentangabe (0..100).

Sämtliche Werte werden dabei 1:1 durchgereicht; wenn Sie sich also einen 32-Bit Fließkommawert nach IEEE-Norm (EIS9) vom EIB besorgen wollen – kein Problem.

Was dann aber die gelieferten "Roh"-Bytes bedeuten, ist Sache Ihrer Anwendungssoftware. CTEIB3 ist ein reines Transportmedium, eine Interpretation der Daten findet nicht statt.

Der einzige Datentyp, der sich ohne "Vorbehandlung" in der AMX benutzen lässt, ist der Typ 8 Bit unsigned, also eine vorzeichenlose Ganzzahl im Bereich 0..255 - ein 1Byte-Typ. Praktischerweise, denn das ist genau der Typ, den die meisten Dimmer als Helligkeitswert zu Ansteuerung verwenden, bzw. melden.

Um beliebige andere EIB-Datentypen zu verwenden, müssen Sie sich also Methoden basteln, die jeweiligen Typen in Strings der entsprechenden Länge zu wandeln - möglichst in beide Richtungen, und möglichst mit Prüfung auf sinnvolle Werte...

Gruppen, Adressen und Formate

Jede Komponente eines EIB-Systems, auf den von der AMX aus zugegriffen werden soll, ist EIB-seitig durch eine Adresse definiert, die Gruppenadresse. Über diese 15 Bit lange Adresse wird auf dem EIB signalisiert, welche(r) Akteur(en) angesprochen werden.

Das Gateway verwendet EIB-seitig nur die Gruppenadressen, eine vollständige und aktuelle Liste der relevanten Gruppenadressen ist also zur Einrichtung des Systems unbedingt notwendig.

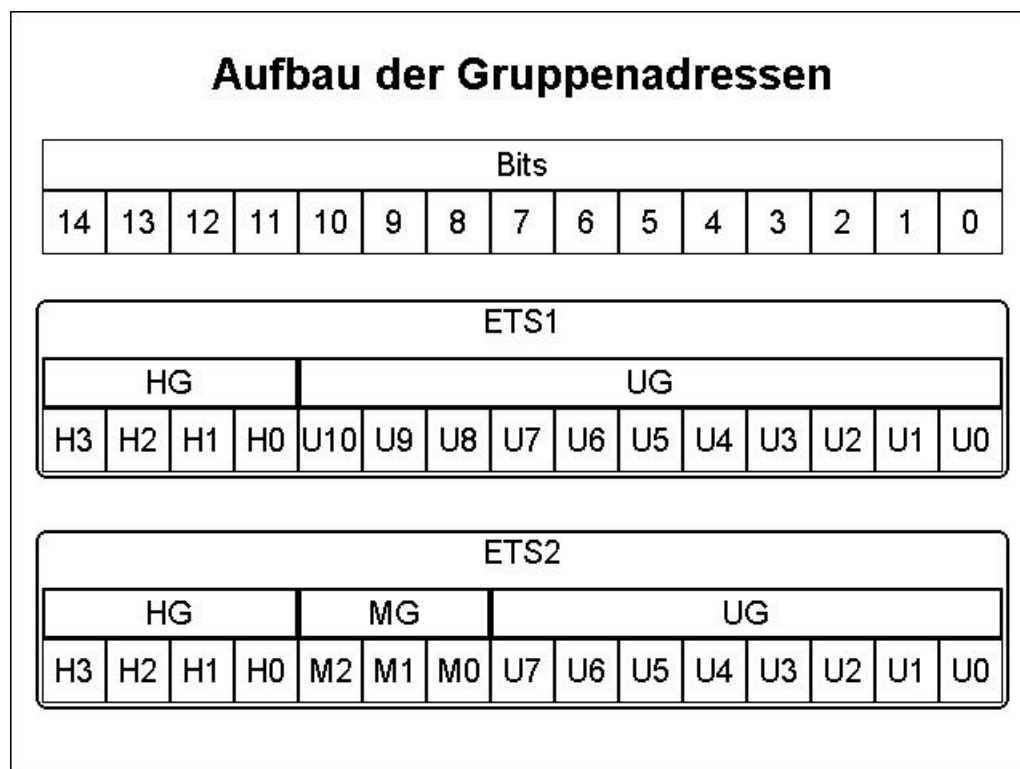
Das ETS1-Format stellt Gruppenadressen durch ein (dezimales) Zahlenpaar HG/UG dar, wobei die Hauptgruppe HG im Bereich 0..15 liegen muß, die Untergruppe UG im Bereich 0..2047.

Beispiel: 1/1027

Mit Einführung der ETS2-Software wurde ein neues Adressformat spezifiziert, das ein Zahlentripel HG/MG/UG zur Darstellung nutzt (MG: Mittelgruppe, HG: 0..15, MG: 0..7, UG: 0..255).

Beispiel: 1/4/3

Ob die Gruppenadressen dabei in der alten ETS1-Notation (2-gruppig) oder in der neuen ETS2-Notation (3-gruppig) vorliegen, ist letztlich egal - die Box verarbeitet beide Varianten. Der Unterschied liegt lediglich in der Notation der Adressen.



Die beiden Formen können recht einfach in die jeweils andere Form umgerechnet werden:

Die Hauptgruppe (0..15) bleibt gleich:

$$HG_{ETS1} = HG_{ETS2}$$

Die 11 UG-Bits des ETS1-Formats verteilen sich im Verhältnis 3:8 auf MG und UG im ETS2-Format:

$$MG_{ETS2} = UG_{ETS1} \text{ div } 256$$

$$UG_{ETS2} = UG_{ETS1} \text{ mod } 256$$

bzw.

$$UG_{ETS1} = 256 \cdot MG_{ETS2} + UG_{ETS2}$$

Bei der Angabe einer Adresse erkennt die Box selbst das Format, und schaltet die Formatierung der Ausgabe auf die zuletzt verwendete Form. Sollte eine Liste in der jeweils anderen Notation gewünscht werden, kann mit dem Befehl „ADR 2“ oder „ADR 3“ (s.u.) auf die entsprechende Darstellung umgeschaltet werden.

Trigger

Es kann beim Umgang mit dem EIB gelegentlich vorkommen, daß Wertänderungen einzelner Aktoren nicht automatisch über den Bus mitgeteilt werden.

Ein typisches Beispiel sind sog. Kombiaktoren zur Lichtsteuerung, die typischerweise vier Gruppenadressen aufweisen:

- 1 Bit Schaltfunktion Ein/Aus
- 4 Bit Dimmfunktion
- 1 Byte Wertstellung
- 1 Byte Wertmeldung

Wohlgemerkt: alle vier Gruppenadressen beziehen sich auf ein und denselben Lichtkreis!

Wenn nun über eine der ersten drei Adressen auf den Dimmer zugegriffen wird, ändert sich die Helligkeit der angeschlossenen Lampe, und damit der Wert der vierten Gruppenadresse.

D.h., daß die 1Byte-Wertstellung keinesfalls immer den aktuellen Helligkeitswert enthalten muß, obwohl diese Adresse zum Dimmen mit Absolutwerten verwendet wird.

Die einzige Möglichkeit, an den aktuellen Helligkeitswert heranzukommen, besteht also darin, die vierte Adresse ausdrücklich nach ihrem aktuellen Wert zu fragen, also die Adresse zu pollen.

Wenn man nun aber möglichst immer den aktuellen Helligkeitswert der Lampe braucht - etwa um eine Bargraph-Anzeige auf einem TouchPanel anzusteuern - müßte bei jeder Änderung eines der ersten drei Werte der Wert der vierten Adresse abgefragt werden.

Das kann man natürlich "zu Fuß" programmieren, und bei jedem Feststellen einer Änderung per System_Call 'EIB Poll ...' diesen Wert anfragen. Das kann allerdings erheblichen Aufwand bedeuten: für jede interessante Gruppenadresse muß ein alter Vergleichswert gespeichert, und dieser bei jedem Programmdurchlauf verglichen werden.

Wesentlich eleganter läßt sich das lösen, indem Trigger verwendet werden.

Dazu verwaltet die Box für jede Gruppenadresse einen Verweis, welche Adresse bei einer Wertänderung gepollt werden soll. Bei der Definition der Adressen (oder auch später) wird also festgelegt, welche andere Adresse (in diesem Fall die zweite 1Byte-Adresse) abgefragt werden soll. Durch den Einbau dreier "WHEN...POLL"-Anweisungen (s.u.) ist also gewährleistet, daß immer der aktuelle Helligkeitswert zur Verfügung steht.

Natürlich kann unabhängig davon trotzdem eine Abfrage per System_Call ausgelöst werden.

Derartige Konstruktionen, bei denen mehrere Aktoren auf eine "Summe" wirken, deren Wert aber ausdrücklich angefragt werden muß, scheint es in EIB-Systemen recht häufig zu geben, etwa in Szenen-Bausteinen.

Wenn Sie also damit zu kämpfen haben, daß manche Rückmeldungen einfach nicht stimmen oder "hängenbleiben", kann ein wohlplazierter Poll-Trigger manchmal Wunder wirken...

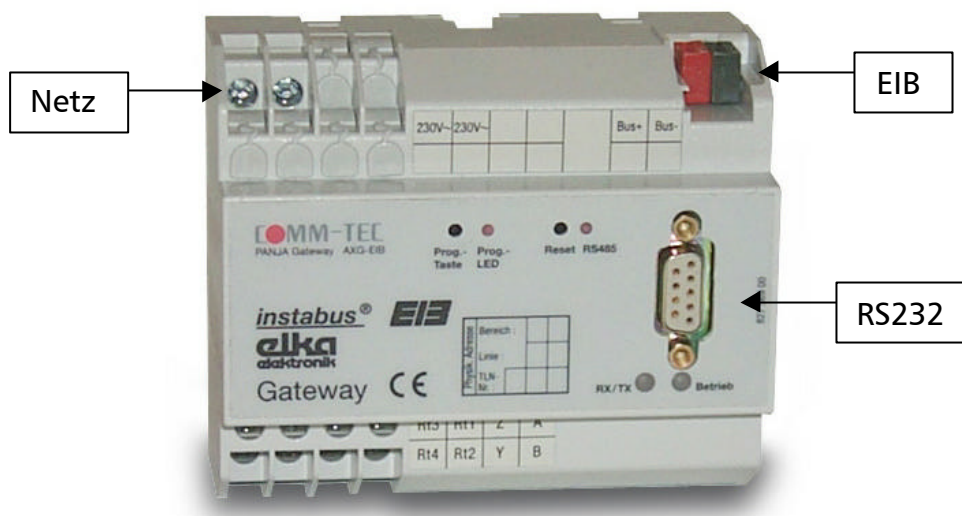
Seien Sie aber bitte mit dem Einsatz aktiver Abfragen eher zurückhaltend – die Bandbreite des EIB steht *allen* angeschlossenen Geräten *gemeinsam* zur Verfügung.

Gegen gelegentliches Abfragen einzelner Werte ist sicher nichts einzuwenden; alle erreichbaren Gruppen im Sekundentakt zu pollen ist aber Ressourcenverschwendung, und dürfte einigen Ärger nach sich ziehen...

Installation und Inbetriebnahme

Installation des Gateways

Das EIB-Gateway AXG-EIB wird auf einer Standard-Hutschiene im Schaltschrank montiert. Die rot-schwarzen Klemmen "Bus+" und "Bus-" werden mit dem EIB verbunden. Die Klemmen "230V~" werden mit der Netzversorgung verbunden.



Falls das Gateway EIB-seitig eine Hardware-Adresse zugewiesen bekommen soll, kann das über die "Prog-Taste" am Gateway und die ETS-Software geschehen. Bei Auslieferung des Gateways ist keine Adresse zugewiesen, das Gateway ist also am EIB nicht zu "sehen". In den meisten Fällen ist das auch nicht notwendig; bei Zugriff auf Aktoren, die "hinter" einem Linienkoppler sitzen, kann das aber notwendig sein. Diese Zuweisung sollte auf jeden Fall vom EIB-Installateur vorgenommen werden!

Installation der RS232-Box/-Karte

Bei Verwendung einer AXB-232++ wird diese in der Nähe (max. 30m entfernt) des Gateways montiert (z.B. mit dem 19"-Befestigungssatz AC-RK), und über die DIP-Schalter an der Frontseite eine eindeutige Device-ID eingestellt.

Wird eine Einschubkarte AXC-232++ verwendet, legen die Server-Karte und die Slot-Nr des verwendeten Einschubs des Kartenträgers die Device-ID fest.

In beiden Fällen muß die RS232-Schnittstelle auf die Parameter des Gateways eingestellt werden. Die Übertragung zum/vom Gateway erfolgt immer mit 9600bps, 8 Datenbits, 1 Stopbit, keine Parität (9600/8/N/1), alle Schalter ausser Nr. 7 müssen also auf ON stehen.

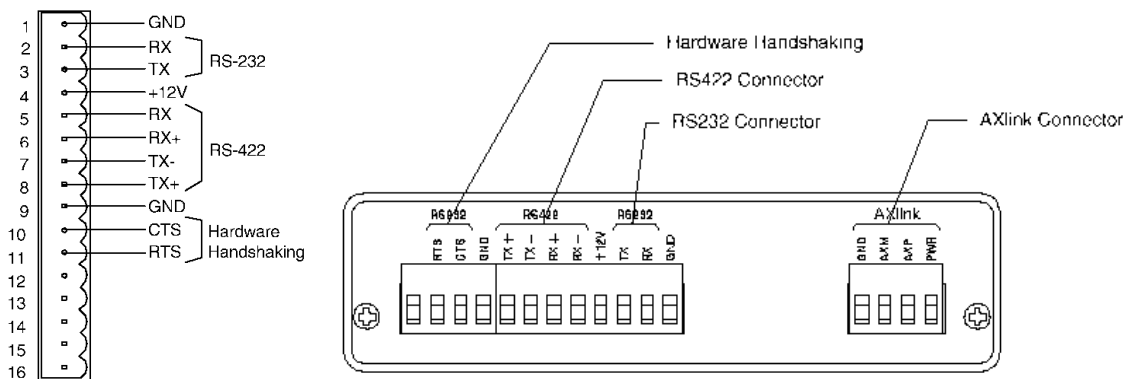
Verbindung Gateway-Box

Zur Verbindung muß ein 5-adriges Kabel verwendet werden, da das Gateway zwingend mit Hardware-Handshake (RTS/CTS) arbeitet.

Anschlußbelegung:

AXC-232++

AXB-232++



Gateway (SubD9)

Pin 2

Pin 3

Pin 4

Pin 5

Pin 6

Pin 7

Pin 8

AXB-232++/AXC-232++ (Phoenix)

Rx

Tx

GND

RTS

CTS

Verbindung der Box mit dem Master

Die Verbindung zum AMX-Master erfolgt über eine Axlink-Leitung mit einer maximalen Länge von 900m.

Einspielen des Box-Programms

Das Steuerprogramm für die RS232-Box/-Karte liegt nicht als Access-Quellcode bei, sondern als binäre Datei, die direkt in die Box gespielt werden kann. Zum Einspielen wird NetLinx Studio verwendet.

Diese Datei heisst **CTEIB4_AX.TOK**

Vorgehensweise :

- 1 Fügen Sie die Datei CTEIB4_AX.TOK zu ihrem Projekt hinzu
- 2 Markieren Sie die Datei mit einem Mausklick
- 3 Weisen Sie der Datei die Adresse Ihrer AXB/AXC-232++ zu
- 4 Starten Sie den download

Funktionsprüfung: Wenn das Einspielen geklappt hat, und die RS232-Verbindung zum Gateway fertig verkabelt ist (RTS/CTS), sollten an der Box die beiden roten LEDs TX und RX hektisch blinken, und nach ein paar Sekunden fast durchgehend leuchten.

Gruppenadressen in die Box eintragen

Sämtliche Gruppenadressen, auf die zugegriffen werden soll, müssen zunächst der Box bekannt gemacht werden.

Nachdem geklärt ist, welche Gruppenadressen und Datentypen den einzelnen Funktionen zugeordnet sind, werden alle Funktionen auf die sechs unterstützten Datentypen "verteilt" und durchnummeriert.

Von jedem der sechs Datentypen verwaltet das Programm je 255 Stück, es kann also mit maximal 1530 Gruppenadressen Kontakt aufgenommen werden.

Die RS232-Box/-Karte bekommt nun die Liste aller Adressen, Datentypen und Nummern gesendet und speichert sie. Alle Befehle von der AMX zum EIB und alle Meldungen vom EIB zur AMX laufen von nun an über die Angabe von Typ und Nummer.

Das hat zum einen den Vorteil, dass nicht bei jedem Zugriff umständlich die Adresse übermittelt werden muß, zum anderen spart das kostbaren Hauptspeicher im Master.

Die Übermittlung kann entweder aus einem Axxess-Programm heraus erfolgen - durch Verwendung des System_Calls 'EIB Send Command', oder direkt im Terminal-Modus der Box (s.u.).

Nach Eintragen aller Adressen "programmiert" die Box das Gateway automatisch; der früher notwendige Schritt, das Gateway über eine separate Windows-Software einzurichten, entfällt somit.

Ein Beispielprogramm zum Laden der Box mit den Adressen findet sich im Anhang.

Versorgung der Box aus dem Hauptprogramm

Durch Einbinden der mitgelieferten System_Calls in das Hauptprogramm werden Steuerfunktionen am EIB ausgelöst, und Meldungen über Zustandsänderungen in die entsprechenden Variablen verteilt.

Damit der Axxess-Compiler die System_Calls auch einbinden kann, müssen sie vor Benutzung in ein Verzeichnis auf der Festplatte kopiert werden. In *welches* Verzeichnis sie kopiert werden müssen, hängt davon ab, wie der Compiler installiert wurde.

Bei einer Standardinstallation von der AMX-CD unter Windows95/98 oder NT liegen sämtliche System_Calls im Verzeichnis „C:\Programme\Gemeinsame Dateien\AMXShare\SYCs“.

Kopieren Sie also bitte alle *.LIB-Dateien aus dem Verzeichnis \syscall der mitgelieferten Diskette dorthin.

Kommunikation mit der Box

System_Calls

Dem Paket CTEIB4 liegen insgesamt 16 System_Calls bei, die in das jeweilige Hauptprogramm eingebunden werden können.

Zum Einlesen der häufigsten Datentypen (und auch um die Umstellung bestehender Programme zu vereinfachen) dient der System_Call 'EIB Read Standard'. Er liest „auf einen Schlag“ sämtliche Switch-, Dim4- und 1Byte-Werte aus dem Empfangspuffer.

Um alle sechs Datentypen einzulesen, wird der System_Call 'EIB Read All' verwendet.

Für jeden der sechs Datentypen gibt es jeweils einen Call zum Setzen einer Adresse, z.B. System_Call 'EIB Set Switch', sowie einen zum aktiven Abfragen (pollen) einer Adresse, z.B. System_Call 'EIB Poll Switch'.

Der fünfzehnte Call - System_Call 'EIB Send Command' - dient zum Absetzen von Befehlen an die Box, etwa zur Definition der Gruppenadressen.

Der System_Call 'EIS5 to String' rechnet den Datentyp EIS5 – ein 2-Byte Fließkommawert – in die entsprechende Dezimaldarstellung um.

Auf der Diskette liegen im Verzeichnis \SYSCALL folgende Dateien:

<u>Dateiname</u>	<u>Klartextname</u>	<u>Bedeutung</u>
CTEIB401_AX.LIB	'EIB Read Standard'	Switch-, Dim4- und 1Byte-Werte einlesen
CTEIB402_AX.LIB	'EIB Read All'	Alle Werte einlesen
CTEIB403_AX.LIB	'EIB Set Switch'	Switch-Wert setzen
CTEIB404_AX.LIB	'EIB Set Dim4'	Dim4-Wert setzen
CTEIB405_AX.LIB	'EIB Set 1Byte'	1Byte-Wert setzen
CTEIB406_AX.LIB	'EIB Set 2Byte'	2Byte-Wert setzen
CTEIB407_AX.LIB	'EIB Set 3Byte'	3Byte-Wert setzen
CTEIB408_AX.LIB	'EIB Set 4Byte'	4Byte-Wert setzen
CTEIB409_AX.LIB	'EIB Poll Switch'	Switch-Wert abfragen
CTEIB410_AX.LIB	'EIB Poll Dim4'	Dim4-Wert abfragen
CTEIB411_AX.LIB	'EIB Poll 1Byte'	1Byte-Wert abfragen
CTEIB412_AX.LIB	'EIB Poll 2Byte'	2Byte-Wert abfragen
CTEIB413_AX.LIB	'EIB Poll 3Byte'	3Byte-Wert abfragen
CTEIB414_AX.LIB	'EIB Poll 4Byte'	4Byte-Wert abfragen
CTEIB415_AX.LIB	'EIB Send Command'	Absetzen von Befehlen an die Box
CTEIB416_AX.LIB	'EIS5 to String'	Umrechnung EIS5 in String

Setzen von Werten

Zum Setzen der Werte auf dem EIB dienen folgende sechs Calls:

```
System_Call 'EIB Set Switch' (Dev, Wert, Nr)
System_Call 'EIB Set Dim4' (Dev, Wert, Nr)
System_Call 'EIB Set 1Byte' (Dev, Wert, Nr)
System_Call 'EIB Set 2Byte' (Dev, Wert[2], Nr)
System_Call 'EIB Set 3Byte' (Dev, Wert[3], Nr)
System_Call 'EIB Set 4Byte' (Dev, Wert[4], Nr)
```

Parameter:

<i>Dev</i>	Device-ID der Box
<i>Wert</i>	Zu setzender Wert
<i>Nr</i>	Laufende Nummer des Aktors in der Box

Sie können an jeder Stelle im Programm eingesetzt werden (analog zu Send_String).

Rückmeldungen

Das Einlesen der Wertänderungen von der Box ins Hauptprogramm geschieht mit den folgenden Calls:

```
System_Call 'EIB Read Standard' (Puffer[255], SW[255], D4[255], B1[255])
System_Call 'EIB Read All' (Puffer[255], SW[255], D4[255], B1[255], B2[255][2],
                             B3[255][3], B4[255][4])
```

Parameter:

<i>Puffer[]</i>	Empfangspuffer von der RS232-Box/-Karte
<i>SW[]</i>	Tabelle mit den aktuellen Switch-Werten
<i>D4[]</i>	Tabelle mit den aktuellen Dim4-Werten
<i>B1[]</i>	Tabelle mit den aktuellen 1Byte-Werten
<i>B2[][2]</i>	Tabelle mit den aktuellen 2Byte-Werten
<i>B3[][3]</i>	Tabelle mit den aktuellen 3Byte-Werten
<i>B4[][4]</i>	Tabelle mit den aktuellen 4Byte-Werten

Es ist dringend anzuraten, einen der Empfangs-Calls direkt im Block DEFINE_PROGRAM aufzurufen, um im Puffer anstehende Pakete unverzüglich abzuarbeiten.

Ausserdem sorgen die beiden Empfangs-Calls dafür, dass Klartextmeldungen von der Box im Terminalfenster ausgegeben werden – auch wenn Sie keine Rückmeldungen benötigen sollten, ist daher der Einbau eines dieser Calls anzuraten.

Aktive Abfragen (Polling)

Aktive Abfragen können an jeder Stelle im Programm eingebaut werden.

```
System_Call 'EIB Poll Switch' (Dev, Nr)
System_Call 'EIB Poll Dim4' (Dev, Nr)
System_Call 'EIB Poll 1Byte' (Dev, Nr)
System_Call 'EIB Poll 2Byte' (Dev, Nr)
System_Call 'EIB Poll 3Byte' (Dev, Nr)
System_Call 'EIB Poll 4Byte' (Dev, Nr)
```

Parameter:

<i>Dev</i>	Device-ID der Box
<i>Nr</i>	Laufende Nummer des Aktors

Nach dem Absetzen einer Abfrage läuft das Programm weiter; beim Eintreffen der Wert-Meldung wandert diese über die Empfangs-Calls in die entsprechende Variable.

Befehle an die Box

Zum Absetzen eines Befehls dient der folgende Call:

System_Call 'EIB Send Command' (Dev, Befehl[50])

Parameter:

Dev Device-ID der Box

Befehl Zeichenkette, die von der Box interpretiert werden soll

Sämtliche Befehle lassen sich wahlweise per System_Call 'EIB Send Command' oder direkt im Terminal (s.u.) verwenden.

Datenkonvertierung

2-Byte EIB-Typen werden recht häufig zur Darstellung von Analogwerten, wie z.B. Temperaturen verwendet. Dabei kommt der EIS-Datentyp EIS5 zum Einsatz, der in 16 Bit einen Fließkommawert codiert. Die 16 Bit teilen sich dabei wie folgt auf:

1 Bit Vorzeichen (S, 0..1)

4 Bit Exponent (E, 0..15)

11 Bit Mantisse (M, 0..2047)

Der eigentliche Wert ergibt sich aus $-1^S * M * 0,01 * 2^E$

⇒ Wertebereich: +/- 670760,96

Da die AMX-AXCESS nun mal keine Float-Typen kennt, die Werte aber oft zumindest dargestellt werden sollen (z.B. in einem Textfeld auf dem TouchPanel), bietet der System_Call 'EIS5 to String' die Möglichkeit, einen 2-Byte-EIS5-Typ in die entsprechende Zifferndarstellung umzurechnen.

SYSTEM_CALL 'EIS5 to String' (Bytes[2], String[10])

Parameter:

Bytes[2] Binärer Wert, der umgerechnet werden soll

String[10] Dezimaldarstellung, z.B. '-670760,96'

Der umgekehrte Weg, eine AMX-Variable (16 Bit, 0..65535) in einen EIS5-Typ umzurechnen, wird nicht unterstützt, da hierbei Rundungsverluste auftreten könnten, und nur ein eingeschränkter Wertebereich zur Verfügung steht.

Terminal-Modus

Da bei der Installation eines EIB-Gateways damit zu rechnen ist, daß noch einige Adressänderungen anfallen, wäre es natürlich recht lästig, alle notwendigen Befehle jedesmal in ein Testprogramm einbauen zu müssen. Andererseits hat die AXB-232++ außer der RS232-Schnittstelle (die ja vom Gateway belegt ist) nur den Axlink-Zugang, um Konfigurationsdaten zu übermitteln. Über die Axlink-Verbindung zum Master kann aber nicht beliebig zugegriffen werden, da hier jederzeit Pakete mit dem Hauptprogramm ausgetauscht werden können. Es fehlt also eine Möglichkeit, "am Master vorbei" mit der Box zu sprechen.

Tatsächlich stellt jeder AMX-Master einen derartigen Mechanismus bereit. Im Terminalfenster kann per Befehl die Eingabe von der Tastatur des angeschlossenen PCs direkt auf ein Busdevice umgeleitet werden, und die Ausgabe von einem Busdevice auf das Terminalfenster erfolgen. Die Verbindung zwischen Hauptprogramm und Box wird also zeitweise gekappt. Der Befehl dazu lautet PASS gefolgt von der Device-ID der Box, also z.B. "PASS 85".

Da das die Box nicht mitbekommt, sendet sie weiterhin (binär verschlüsselt) Werte an das Hauptprogramm, man muß also die "Aufmerksamkeit" der Box auf sich ziehen, um nicht mit Zeichenschrott überflutet zu werden.

Die Box lauert deshalb ständig darauf, dass im Datenstrom das geheime Aufmerksamkeitszeichen "JUHU" vorkommt, und schaltet daraufhin die Rückmeldungen zeitweise ab.

Umgekehrt können sich während der Unterhaltung mit der Box EIB-seitig Werte ändern. Um diese Änderungen an das Hauptprogramm zu übermitteln, muß die Box wissen, wann sie direkten Kontakt mit dem Master hat, und wann die Umleitung aktiv ist.

Dieser Rückschritt geschieht automatisch nach Ablauf von 40 Sekunden nach dem letzten Befehl, beim nächsten Setzen eines Wertes, oder bei der nächsten Abfrage.

Der Abbau der Verbindung mit der Box, wird dem Master mit einer speziellen Tastenkombination mitgeteilt:

+ + Esc Esc (zweimal Plus, zweimal Escape)

Daraufhin sollte der Master-Prompt ">" erscheinen.

Das prinzipielle Vorgehen sieht also folgendermaßen aus:

- I. Sprung in das Terminalfenster (Strg-T)
- II. Einschalten der Echo-Funktion des Masters (blind „ECHO ON“ eingeben)
- III. Umleiten der Box auf das Terminal (PASS <Id>)
- IV. Aufmerksamkeit der Box auf sich lenken (JUHU)
- V. Unterhaltung mit der Box (List, Add, etc.)
- VI. Aufhebung der Umleitung (+ + Esc Esc)
- VII. Terminalfenster verlassen (F10)

Kanäle und Levels

Für zwei Datentypen gibt es zusätzliche Zugriffsmechanismen:

Sämtliche Schalttypen (Switch) werden auf die Kanäle der Box abgebildet; beim Einschalten erfolgt ein PUSH[EIB,Nr] von der Box, beim Ausschalten analog ein RELEASE[EIB,Nr]. Ebenso kann mit den Befehlen ON[], OFF[], TO[] und PULSE[] auf Schaltaktoren zugegriffen werden.

Die ersten acht 1Byte-Adressen stehen als Levels zur Verfügung (Levels 1..8); bei jeder Änderung des Wertes einer dieser Gruppenadressen wird der aktuelle Wert als Level an das Hauptprogramm übermittelt, um z.B. eine Bargraph-Anzeige anzusteuern.

Der umgekehrte Weg, mittels SEND_LEVEL auf die Aktoren zuzugreifen, ist nicht möglich.

Befehlssatz der Box

Um nicht eine Vielzahl von System_Calls zur Initialisierung der Box herumschleppen zu müssen, erfolgt die Befehlsgebung an die Box und die Rückmeldung eventueller Fehler und Statusmeldungen im Klartext. Meldungen von der Box werden von den Empfangs-Calls automatisch in das Terminalfenster umgeleitet, (Text-)Befehle an die Box werden mit dem System_Call 'EIB Send Command' abgesetzt.

Befehle bestehen aus einem oder mehreren Wörtern, die durch Leerzeichen (Space, \$20) getrennt werden, Groß- und Kleinschreibung ist dabei egal.

Es sind nur die Zeichen {'0'..'9','A'..'Z','a'..'z',' ','/','\$'} erlaubt.

Zur kompakten Bezeichnung der Datentypen gelten folgende Abkürzungen:

SW	Switch
D4	Dim4
1B	1Byte
2B	2Byte
3B	3Byte
4B	4Byte

Folgende Befehle stehen zur Verfügung

ADD – Gruppenadresse eintragen

Der ADD-Befehl trägt eine Gruppenadresse (neu) ein, schafft also die Verbindung zwischen EIB-Gruppenadresse und dem Typ/Nummer-Paar in der AMX. Sind Gruppenadresse oder Typ/Nr bereits verwendet, wird der alte Eintrag ersetzt.

Syntax: ADD <Typ> <Nr> <Adresse>

Bsp.:	ADD SW 1 08/15	Schalter Nr. 1 hat Adresse 08/15
	ADD 4B 255 15/1234	4-Byte-Aktor Nr. 255 hat Adresse 15/1234
	ADD SW 2 1/2/3	auch die ETS2-Notation ist möglich

ADR – Format der Adressdarstellung umschalten

Mit ADR kann das Darstellungsformat für Gruppenadressen zwischen der alten ETS1-Notation (HG/UG) und der neuen ETS2-Notation (HG/MG/UG) umgeschaltet werden.

Die Umschaltung wird nur für LIST-Ausgaben benötigt; bei der Adressangabe schaltet die Box automatisch auf die zuletzt verwendete Notation um.

Syntax: ADR <Format>

Bsp.:	ADR 2	Ausgabe in ETS1-Notation HH/UUUU
	ADR 3	Ausgabe in ETS2-Notation HH/M/UUU

BOXCH – Abbildung der logischen Kanäle der Box auf 1-Bit Aktoren

Kurzschlüsse auf dem Bus (AXLink) sowie ein kurzzeitiges Zusammenbrechen der Spannungsversorgung z.B. durch Anstecken von Busdevices können dazu führen, dass die im Master verwalteten Kanäle der 232++ kurzzeitig zurückgesetzt werden. Die Folge hiervon ist ein Abschalten der programmierten 1-Bit Aktoren.

Mit BOXCH kann die direkte Verknüpfung von 1-Bit Aktoren und logischen Kanälen der Box aufgehoben werden. Selbstverständlich lassen sich die 1-Bit Aktoren mit dem Standard SYSTEM_CALL 'EIB Set Switch'(<Parameter>) weiterhin schalten.

ACHTUNG! Dieser Befehl behebt nur die Symptome einer unsachgemäss durchgeführten Installation, die Ursachenbekämpfung ist Sache des System-Integrators.

Syntax: BOXCH <Zustand>

Bsp.:	BOXCH ON	(default) Switches werden auf die Kanäle der Box abgebildet
	BOXCH OFF	keine direkte Verknüpfung

BURST - Begrenzung der Sendegeschwindigkeit

Normalerweise sendet die Box alle anstehenden Daten so schnell wie möglich an das Hauptprogramm. Bei sehr grossen Programmen im Master kann die Zykluszeit bis zum nächsten

Abarbeiten des Empfangspuffers aber zu gross sein => Pufferüberlauf.

Mit BURST kann nun die Sendegeschwindigkeit der Box an das Hauptprogramm auf maximal 10 Nachrichten/Sekunde begrenzt werden, um dem Hauptprogramm genügend Zeit zur Pufferabarbeitung zu geben. Bei vielen kurz aufeinander folgenden Paketen steigt dadurch natürlich die Reaktionszeit...

Syntax: BURST <Zustand>

Bsp.: BURST ON (default) So schnell wie möglich alles senden
BURST OFF Maximal zehn Pakete pro Sekunde senden

DEL – Entfernen einer Gruppenadresse

Mit dem DEL-Befehle kann eine Gruppenadresse wieder aus dem Speicher entfernt werden.

Syntax: DEL <Typ> <Nr>

Bsp.: DEL SW 4 Schalter Nr. 4 entfernen
DEL 3B 123 3-Byte Nr. 123 entfernen

DELTA – Differentialübertragung Ein/Aus

Mit DELTA kann eingestellt werden, ob die Box nur dann ein Paket an das Hauptprogramm sendet, wenn sich der Wert des Aktors geändert hat, oder ob jedesmal ein Paket gesendet wird, wenn überhaupt ein Paket empfangen wird – egal ob der Wert gleich bleibt oder nicht.

Normalerweise werden nur Wertänderungen mitgeteilt, um möglichst wenig übertragen zu müssen. Es kann in Ausnahmefällen(!) aber sinnvoll sein, Meldungen über *alle* empfangenen Wert-Pakete an das Hauptprogramm zu übermitteln.

Die RESEND-Logik (s.u.) bleibt davon unberührt.

Syntax: DELTA <Zustand>

Bsp.: DELTA ON (default) Nur Wertänderungen an das Hauptprogramm senden
DELTA OFF Alle empfangenen Werte weiterleiten

LIST – Anzeige der bestehenden Verknüpfungen

LIST erzeugt eine Klartextliste der verwendeten Gruppenadressen und aktuellen Werte je Typ.

Syntax: LIST <Typ> <StartNr> [<EndNr>]

Bsp.: LIST SW 1 Wie ist Schalter Nr. 1 definiert?

=> Ausgabe:

:l sw 1

Switch #1: 1/0/1 Val: \$0

LIST D4 1 255

Zeige Definition aller 4-Byte-Typen

=> Ausgabe:

:list d4 1 255

Dim4 #1: 1/0/112 Val: \$0 => polls 1Byte #2

Dim4 #2: 1/0/122 Val: \$0 => polls 1Byte #4

NO POLL - Löschen eines Poll-Triggers

Mit NO POLL werden sämtliche automatischen Abfragen einer Adresse gelöscht.

Syntax: NO POLL <Typ> <Nr>

Bsp.: NO POLL SW 13
NO POLL 1B 217

POLL - Aktive Wertabfrage

Mit dem POLL-Kommando wird ein Telegramm auf den EIB abgesetzt, um den aktuellen Wert eines Aktors zu erfragen (dessen Lese-Flag dazu natürlich gesetzt sein muss).

Dieser Befehl ist nur für den Terminal-Modus gedacht; für aktive Abfragen aus dem Hauptprogramm verwenden Sie bitte die entsprechenden System_Calls.

Syntax: POLL <Typ> <Nr>

Bsp.: POLL SW 17 Aktuellen Wert von Schalter Nr. 17 erfragen
POLL 2B 128 Welchen Wert hat 2-Byte Nr. 128 ?

RESEND - Alle Werte erneut schicken

Nach einem RESEND-Befehl sendet die Box die aktuellen Werte aller eingetragenen Gruppen an das Hauptprogramm, um den Speicherinhalt zu synchronisieren.

Diese Funktion wird auch automatisch nach Verlassen des Terminal-Modus ausgelöst.

Syntax: RESEND

RESET - Gateway-Reset auslösen

Durch einen Gateway-Reset werden alle anstehenden Pakete im Gateway gelöscht, der Busankoppler initialisiert, und alle markierten Gruppenadressen abgefragt (siehe WHEN...POLL)

Bsp.: RESET

SET - Gruppenadresse schreiben

Mit dem SET-Befehl kann im Terminal-Modus direkt auf einzelne Adressen zugegriffen werden, um die Funktion zu testen. Je nach angesprochenem Typ müssen bis zu vier (Byte-)Werte entweder als Dezimalzahlen oder – mit vorangestelltem \$-Zeichen – als Hexadezimalzahlen übergeben werden.

Für den Schreibzugriff aus dem Hauptprogramm heraus verwenden Sie bitte die entsprechenden System_Calls.

Syntax: SET <Typ> <Nr> Wert [<Wert2> [<Wert3> [<Wert4>]]]

Bsp.: SET SW 1 0 Schalter Nr. 1 ausschalten
SET D4 3 \$E Dimmer 3 auf 14 setzen
SET 4B 217 1 2 3 4 4-Byte-Wert Nr. 217 auf {1,2,3,4} setzen

START - Starten des Update-Prozesses

Durch ein START-Kommando wird die mit STOP angehaltene Übertragung der Werte des angegebenen Typs wieder freigegeben. Standardmässig werden alle 6 Typen gesendet.

Syntax: START <Typ>

Bsp.: START SW Ab sofort wieder Schalter-Werte senden
START 3B 3-Byte-Werte wieder senden

STATUS – Statusinformation anzeigen

STATUS erzeugt eine Liste mit den wichtigsten Zustandsinformationen zum Betrieb des Gateways.

Syntax: STATUS
 => Ausgabe:
 :status
 Gateway Version: 2.15
 EEPROM Check: \$E922
 ADR mode: 3 groups (H/MM/UUU)
 DELTA mode: ON (report changes)
 BURST mode: ON (send full speed)
 WATCH mode: OFF
 BOXCH mode: ON
 No errors reported.

STOP – Anhalten des Update-Prozesses

Mit dem STOP-Kommando kann das Senden der Werte eines bestimmten Typs an das Hauptprogramm abgeschaltet werden, um z.B. in Ruhe Wertänderungen nachzuvollziehen, ohne mit Paketen überschüttet zu werden.

Syntax: STOP <Typ>

Bsp.: STOP SW Keine Schalter-Werte mehr senden
 STOP 4B Keine 4-Byte-Werte mehr senden

UPLOAD - Gateway-Filtertabelle neu laden

Mit dem UPLOAD-Befehl kann die Filtertabelle im Gateway erneut geschrieben werden. Dies ist normalerweise nicht notwendig, da die Box automatisch erkennt, ob sich Gruppenadressen geändert haben, oder ein neues (noch nicht programmiertes) Gateway angeschlossen wurde.

Syntax: UPLOAD

WATCH - Überwachung einer Adresse

Mit WATCH kann der Zustand einer Gruppenadresse überwacht werden, um die Funktion eines Aktors oder Sensors während der Installation zu prüfen. Wird ein Telegramm der überwachten Adresse empfangen, erzeugt die Box eine Klartextmeldung mit dem alten und dem neuen Wert.

Syntax: WATCH <Typ> <Nr>
 bzw.
 WATCH OFF

Bsp.: WATCH SW 4 Überwache Schalter Nr. 4
 WATCH 2B 23 Überwache 2-Byte-Wert Nr. 23
 WATCH OFF Überwachung abschalten

Ausgabe:

```
:watch sw 4
OK, watching SW #4
Change: SW #4: $0 => $1
Change: SW #4: $1 => $0
Change: SW #4: $0 => $1
Change: SW #4: $1 => $0
:watch off
WATCH mode set off
```

Mit WHEN...POLL kann definiert werden, dass bei Wertänderungen eines Aktors automatisch ein anderer Aktor abgefragt („gepollt“) werden soll. Das ist z.B. bei Kombiaktoren hilfreich, bei denen der Schreibzugriff auf eine Adresse (Dimmen Auf/Ab) den Wert einer anderen Adresse ändert (Helligkeit), ohne dass diese ihren Wert automatisch kundtut.

Syntax: WHEN <Typ> <Nr> POLL <Typ> <Nr>
 oder
 WHEN START POLL <Typ> <Nr>

23

Anhang

Vergleich „alte“ Calls – neue Calls

Mit Einführung von CTEIB3 haben sich u.a. auch die Namen und Funktionsweisen der System_Calls geändert. Um nun bestehende Projekte problemlos auf CTEIB3 umzurüsten, ist es daher nötig, den Aufruf der „alten“ System_Calls 'CTCT9xxx' gegen die neuen System_Calls auszutauschen.

Die häufigsten Aufrufe – Setzen von Werten und Einlesen der Rückmeldungen – folgen dabei den gleichen Konventionen zur Übergabe der Parameter; in den meisten Fällen genügt es, den Namen des Calls mit der Suchen/Ersetzen-Funktion (<Alt>-R in Axxess) zu ändern.

Im folgenden nun eine Übersicht aller System_Calls der EIB-Software Version 1.411 mit den Entsprechungen in CTEIB3.

Alte Calls (Version 1.411 ff.)	Neuer Call (CTEIB3)
Gateway-Reset	
'CTCT9000' (DEV)	'EIB Send Command' (DEV,'RESET')
Switch-Verarbeitung	
'CTCT9010' (DEV,ADR[7],NR)	'EIB Send Command' (DEV, ""ADD SW ',itoa(NR),' ',ADR[]"
'CTCT9011' (DEV,VALUE,NR)	'EIB Set Switch' (DEV,VALUE, NR)
'CTCT9019' (DEV,NR)	'EIB Poll Switch' (DEV,NR)
Dim4-Verarbeitung	
'CTCT9020' (DEV,ADR[7],NR)	'EIB Send Command' (DEV, ""ADD D4 ',itoa(NR),' ',ADR[]"
'CTCT9021' (DEV,VALUE,NR)	'EIB Set Dim4' (DEV, VALUE, NR)
'CTCT9022' (DEV,DIR,STEP,NR)	--
'CTCT9023' (DEV,PANEL,BTN_ON, BTN_OFF,SW_NR,D4_NR)	--
'CTCT9029' (DEV,NR)	'EIB Poll Dim4' (DEV, NR)
1Byte-Verarbeitung	
'CTCT9030' (DEV,ADR[7],NR)	'EIB Send Command' (DEV, ""ADD 1B ',itoa(NR),' ',ADR[]"
'CTCT9031' (DEV,VALUE,NR)	'EIB Set 1Byte' (DEV,VALUE, NR)
'CTCT9039' (DEV,NR)	'EIB Poll 1Byte' (DEV,NR)
Rückmeldungen	
'CTCT9100' (PUFFER[255], SW[255],D4[255],D8[255])	'EIB Read Standard' (PUFFER[255], SW[255],D4[255],D8[255])

Zu beachten ist vor allem die Behandlung der 4Bit-Dim-Aktoren (Dim4), die grundsätzlich nur direkt gesetzt werden können; die alten Calls CTCT9021..CTCT9023 werden also alle auf den neuen Call 'EIB Set Dim4' abgebildet.

Zum Eintragen der Gruppenadressen (CTCT90x0) können statt des sperrigen Ausdrucks auch Calls verwendet werden, die den „alten“ Aufruf umsetzen.

Beispiele dafür finden sich im Anhang, als auch als Block-Files (*.AXB) im Beispiel-Verzeichnis.

Werte für Dim4-Aktoren

Der Call 'EIB Set Dim4' reicht die übergebenen Werte direkt an den Aktor weiter – die zurückgemeldeten Werte entsprechen nun den tatsächlichen Aktor-Werten (im Bereich 0..15), die auch gesendet wurden.

Die Bedeutung der 16 möglichen Werte ist dabei durch den EIB-Datentyp EIS2 festgelegt:

Bits 0..2 legen die Dimmgeschwindigkeit fest (0: Stop, 7: Max)

Bit 3 legt die Dimmrichtung fest (0: Abwärts, 1: Aufwärts)

Daraus ergeben sich folgende Bedeutungen:

0	Stop	8	Stop
1	Langsam dunkler	9	Langsam heller
2	...	10 (\$A)	...
3	...	11 (\$B)	...
4	dunkler	12 (\$C)	heller
5	...	13 (\$D)	...
6	...	14 (\$E)	...
7	Schnell dunkler	15 (\$F)	Schnell heller

Beispiel: Eintragen der Gruppen

Das folgende Beispielprogramm zeigt die Übertragung der Gruppeneinstellungen in die Box. In einer Schleife wird alle 200ms jeweils ein Befehl mit dem System_Call 'EIB Send Command' an die Box geschickt, um entweder eine Gruppenadresse mit einer Nummer und einem Typ zu verknüpfen, oder eine automatische Wertabfrage („Poll-Trigger“) zu definieren. Es genügt völlig, den Ladevorgang einmalig bei der Installation des Systems auszuführen, da die Box die Daten in ihrem lokalen Speicher vorhält. Lediglich bei einer Änderung der Konfiguration muß die Box auf den aktuellen Stand gebracht werden. Für diesen Zweck ist ein unabhängiges kleines Ladeprogramm, das kurzzeitig in den Master geladen wird handlicher. Es wird davon abgeraten, die Initialisierung in die DEFINE_START – Sektion des Hauptprogrammes zu integrieren, da SYSTEM_CALLS in der Mainline bereits arbeiten, während noch versucht wird, Adressen ins Gateway einzutragen.

```
PROGRAM_NAME='D - Boxload Demo zum Eintragen der Adressen'
(* DATE:01/11/00 TIME:11:47:35 *)
DEFINE_DEVICE
    EIB = 17 (* Adresse AXB-232++ / AXC-232++ *)

DEFINE_VARIABLE
    X          (* Zaehler *)
    Cmd[50]    (* Aktueller Befehl *)

DEFINE_START
    X = 0      (* bei Reset anhalten *)
    WAIT 20    (* 2 Sekunden warten *)
    X = 1      (* Erste Gruppe eintragen *)

DEFINE_PROGRAM
    WAIT 2      (* alle 200ms pruefen *)
    IF(X)      (* Etwas einzutragen ? *)
    {
        SELECT
        {
            ACTIVE(X = 1): Cmd = 'ADD SW 1 1/1' (* Schalter 1: Adresse 1/1 *)
            ACTIVE(X = 2): Cmd = 'ADD SW 2 1/2'
            ACTIVE(X = 3): Cmd = 'ADD SW 3 1/0/3'
                               (* entspricht der Adresse 1/3 unter ETS1 *)
            ACTIVE(X = 4): Cmd = 'ADD SW 4 1/22' (* Dimmer Ein/Aus *)
            ACTIVE(X = 5): Cmd = 'ADD D4 1 1/23' (* Relativ dimmen *)
            ACTIVE(X = 6): Cmd = 'ADD 1B 1 1/24' (* Dimmer setzen *)
            ACTIVE(X = 7): Cmd = 'ADD 1B 2 1/25' (* aktueller Dimmerwert *)

            ACTIVE(X = 8): Cmd = 'WHEN START POLL 1B 2' (* nach Reset lesen *)
            ACTIVE(X = 9): Cmd = 'WHEN SW 4 POLL 1B 2' (* bei Schalter 4 lesen*)
            ACTIVE(X = 10): Cmd = 'WHEN D4 1 POLL 1B 2' (* bei Dimmen lesen *)
            ACTIVE(X = 11): Cmd = 'WHEN 1B 1 POLL 1B 2' (* bei Setzen sowieso *)

            ACTIVE(1): Cmd = "" (* Schluss *)
        }
    }

    IF(Cmd <> "") (* Etwas zu Senden ? *)
    {
        SYSTEM_CALL 'EIB Send Command' (EIB,Cmd) (* an die Box *)
        X = X + 1 (* Naechster *)
    }
    ELSE (* Ende erreicht *)
        X = 0 (* Zaehler aus *)
    }
```

Beispiel: Hauptprogramm

```
PROGRAM_NAME='D - Demo-Hauptprogramm zum EIB-Interface'
(*   DATE:01/11/00   TIME:10:56:35   *)

DEFINE_DEVICE
    EIB = 17 (* Device-Adresse der AXB-232++/AXC-232++ *)
    TP  = 128 (* Irgendein Bediengerat, z.B. AXT-EL+ *)

DEFINE_CONSTANT
    Sw_Licht1    = 1 (* laufende Switch-Nummer Deckenlicht *)
    Sw_Licht2    = 2 (* laufende Switch-Nummer Pultlicht *)
    Sw_Licht3    = 3 (* laufende Switch-Nummer Pultlicht *)
    Sw_Dimmer    = 4 (* Dimmer Ein/Aus *)

    D4_Dimmer    = 1 (* relativ Dimmen *)

    B1_Dim_Set   = 1 (* Dimmer setzen *)

DEFINE_CONNECT_LEVEL
    (EIB,2,TP,1) (* Helligkeit auf Bargraph 1 anzeigen *)

DEFINE_VARIABLE
    EIB_Puffer[255]      (* Empfangspuffer der Box *)
    INTEGER EIB_Switch[255] (* Aktuelle Werte der EIB-Switches *)
    INTEGER EIB_Dim4[255]  (* Aktuelle Werte Dim4 *)
    INTEGER EIB_1Byte[255] (* Aktuelle Werte 1Byte *)

DEFINE_START
    CREATE_BUFFER EIB,EIB_Puffer (* Empfangspuffer verbinden *)

DEFINE_PROGRAM
    (* Rueckmeldungen einlesen *)
    SYSTEM_CALL 'EIB Read Standard' (EIB_Puffer,EIB_Switch,EIB_Dim4,EIB_1Byte)

    (* Licht1 schalten, je ein Knopf fuer 'Ein' und 'Aus' *)
    PUSH[TP,1] (* Licht1 ein *)
        SYSTEM_CALL 'EIB Set Switch' (EIB,1,Sw_Licht1)

    PUSH[TP,2] (* Licht1 aus *)
        SYSTEM_CALL 'EIB Set Switch' (EIB,0,Sw_Licht1)

    [TP,1] = [EIB,Sw_Licht1]      (* Rueckmeldung 'ein' *)
    [TP,2] = not [EIB,Sw_Licht1] (* Rueckmeldung 'aus' *)

    (* Licht2 schalten, ein Knopf schaltet zwischen 'Ein' und 'Aus' um *)
    PUSH[TP,3] (* Licht2 ein/aus *)
        SYSTEM_CALL 'EIB Set Switch' (EIB,not [EIB,Sw_Licht2],Sw_Licht2)

    [TP,3] = [EIB,Sw_Licht2]      (* Rueckmeldung 'ein' *)

    (* Licht3 schalten, Knopf schaltet Licht fuer 10 Sekunden an *)
    PUSH[TP,4] (* Licht3 fuer 10 Sekunden an *)
        {
            CANCEL_WAIT 'Licht3'
            SYSTEM_CALL 'EIB Set Switch' (EIB,1,Sw_Licht3)
            WAIT 100 'Licht3'
            SYSTEM_CALL 'EIB Set Switch' (EIB,0,Sw_Licht3)
        }

    [TP,4] = [EIB,Sw_Licht3]      (* Rueckmeldung 'ein' *)

    (* Dimmersteuerung *)
    PUSH[TP,5] (* Dimmer aus *)
        SYSTEM_CALL 'EIB Set Switch' (EIB,0,Sw_Dimmer)
```

```

PUSH[TP,6] (* Dimmer auf 25% *)
    SYSTEM_CALL 'EIB Set 1Byte' (EIB,64,B1_Dim_Set)

PUSH[TP,7] (* Dimmer auf 50% *)
    SYSTEM_CALL 'EIB Set 1Byte' (EIB,128,B1_Dim_Set)

PUSH[TP,8] (* Dimmer auf 75% *)
    SYSTEM_CALL 'EIB Set 1Byte' (EIB,192,B1_Dim_Set)

PUSH[TP,9] (* Dimmer auf 100% *)
    SYSTEM_CALL 'EIB Set Switch' (EIB,1,Sw_Dimmer)

[TP,5] = (EIB_1Byte[B1_Dim_Val] = 0)    (* Feedback 0% *)
[TP,6] = (EIB_1Byte[B1_Dim_Val] = 64)   (* Feedback 25% *)
[TP,7] = (EIB_1Byte[B1_Dim_Val] = 128)  (* Feedback 50% *)
[TP,8] = (EIB_1Byte[B1_Dim_Val] = 192)  (* Feedback 75% *)
[TP,9] = (EIB_1Byte[B1_Dim_Val] = 255)  (* Feedback 100% *)

PUSH[TP,10] (* Heller *)
    SYSTEM_CALL 'EIB Set Dim4' (EIB,10,D4_Dimmer)

PUSH[TP,11] (* Dunkler *)
    SYSTEM_CALL 'EIB Set Dim4' (EIB,2,D4_Dimmer)

RELEASE[TP,10] (* Hell genug *)
RELEASE[TP,11] (* Dunkel genug *)
    SYSTEM_CALL 'EIB Set Dim4' (EIB,0,D4_Dimmer)

[TP,10] = ((EIB_Dim4[D4_Dimmer] >= 9) and (EIB_Dim4[D4_Dimmer] <= 15))
[TP,11] = ((EIB_Dim4[D4_Dimmer] >= 1) and (EIB_Dim4[D4_Dimmer] <= 7))

```

Lokale Calls als Ersatz der „alten“ Initialisierung

In vielen bestehenden Programmen werden bei jedem Neustart sämtliche Gruppenadressen in die Box übertragen, wobei üblicherweise die System_Calls CTCT9010, CTCT9020 und CTCT9030 zum Einsatz kommen.

Diese tragen jeweils eine Gruppenadresse für einen Switch- Dimm4- oder 1Byte-Aktor ein, tun also im Prinzip das gleiche wie ein „ADD“-Befehl (im Terminal-Modus oder per „EIB Send Command“).

Aus einem

```
SYSTEM_CALL 'CTCT9010' (EIB, '0/815', 1)
```

würde etwa

```
SYSTEM_CALL 'EIB Send Command' (EIB, 'ADD SW 1 0/815')
```

Es hat sich herausgestellt, daß der Schreibaufwand bei der Umstellung relativ hoch ist, da sich die notwendigen Parameter stark unterscheiden. Bei einigen wenigen Gruppen ist das nicht weiter tragisch, wenn aber mehrere Hundert Adressen gesetzt werden müssen, kommt man mit der Suchen/Ersetzen-Funktion des Axxess-Compilers (Alt-R) nicht weit.

Um diesen Aufwand zu begrenzen, bietet es sich an, innerhalb des Hauptprogramms eigene Define_Calls mit den Namen CTCT9010..CTCT9030 anzulegen, die die „alten“ Parameter umsetzen:

```
DEFINE_CALL 'CTCT9010' (DEVICE,ADRESS[7],COUNTER)
  IF(DEVICE AND (DEVICE <= $FF)) (* Device OK ? *)
    IF(COUNTER AND (COUNTER <= $FF)) (* Counter OK ? *)
      SYSTEM_CALL 'EIB Send Command'
        (DEVICE, "'ADD SW ', ITOA(COUNTER), ' ', ADRESS")

DEFINE_CALL 'CTCT9020' (DEVICE,ADRESS[7],COUNTER)
  IF(DEVICE AND (DEVICE <= $FF)) (* Device OK ? *)
    IF(COUNTER AND (COUNTER <= $FF)) (* Counter OK ? *)
      SYSTEM_CALL 'EIB Send Command'
        (DEVICE, "'ADD D4 ', ITOA(COUNTER), ' ', ADRESS")

DEFINE_CALL 'CTCT9030' (DEVICE,ADRESS[7],COUNTER)
  IF(DEVICE AND (DEVICE <= $FF)) (* Device OK ? *)
    IF(COUNTER AND (COUNTER <= $FF)) (* Counter OK ? *)
      SYSTEM_CALL 'EIB Send Command'
        (DEVICE, "'ADD 1B ', ITOA(COUNTER), ' ', ADRESS")
```

Dann besteht die komplette Umstellung nur noch darin, aus dem Wort „System_Call“ das „System_“ herauszulöschen, also etwa:

```
CALL 'CTCT9010' (EIB, '0/815', 1)
```

FAQs

Frage:

Nach dem Einspielen der Box-Software und dem Auflegen der Verkabelung blinkt zwar die Tx-LED, Rx bleibt aber dunkel.

Antwort:

Sehr wahrscheinlich stimmen die RS232-Parameter nicht. Stellen Sie am 8-poligen DIP-Schalter 9600 bps, 8 Datenbits, 1 Stopbit und keine Parität ein. Also alle Schalter auf „On“ außer Nr. 7.

Frage:

Weder Rx noch Tx leuchten – die Software ist in der Box, und ich habe auch die Handshake-Leitungen aufgelegt. Trotzdem tut sich nichts an meiner AXB-232++.

Antwort:

Vorsicht Fallstrick! An der Phoenix-Kontaktleiste der AXB-232++ ist der Kontakt ganz links nicht belegt. Vielleicht sind die Handshakes (RTS und CTS) um einen Kontakt „nach links“ gerutscht...

Frage:

Das Dimmen eines Lichtkreises klappt wunderbar, ich bekomme aber keine Rückmeldung vom EIB.

Antwort:

Möglicherweise handelt es sich um einen Kombiaktor, der eine eigene Gruppenadresse zum Lesen des aktuellen Wertes zur Verfügung stellt (s. Abschnitt „Trigger“).

Grundsätzlich kann es aber bei jeder EIB-Adresse nötig sein, ausdrücklich nach dem aktuellen Wert zu fragen – nicht jeder Aktor/Sensor erzeugt bei einer Wertänderung automatisch ein Telegramm.

Frage:

Beim Reset des Gateways werden viele Schaltfunktionen ausgelöst, ohne dass die AMX aktiv zu schalten versucht.

Antwort:

Das liegt wahrscheinlich daran, dass eine Adresse aktiv abgefragt („gepollt“) wird, die Bestandteil einer Gruppensdefinition ist, z.B. ein Szenenbaustein.

Sobald dieser nämlich seinen aktuellen Zustand („Szene x aktiv“) auf den Bus absetzt, interpretieren die Aktoren, die Bestandteil dieser Szene sind, dieses Telegramm als Befehl, die entsprechende Szene aufzurufen... Abhilfe: keine Szenenbausteine abfragen.

Frage:

Nach Aufstecken eines Panels fallen plötzlich alle 1-Bit Aktoren ab

Antwort:

Das kann daran liegen, dass durch Verwendung eines falschen Kabels oder einer unsachgemäß durchgeführten Installation die Spannungsversorgung kurzzeitig zusammenbricht. Die Kanäle der Box werden kurz abgeschaltet, die Verknüpfung von Boxkanälen und 1-Bit Aktoren führt zu einem Abschalten der eingeschalteten 1-Bit Aktoren.

Abhilfe : Installation überprüfen !

Temporär : Verwendung des Softwareschalters BOXCH

Fehlermeldungen

In einigen Situationen erzeugt die Box Meldungen, die die Empfangs-System_Calls (EIB Read ...) in das Terminalfenster umleiten. Sie werden durch die vorangestellte Zeichenkette „EIB:“ kenntlich gemacht, und enthalten Klartextmeldungen, die im folgenden alphabetisch sortiert sind:

- **"Adress <Adresse> not used"** Das Gateway hat den Zustandswechsel einer Adresse gemeldet, die nicht verwendet wird. Abhilfe: Gateway neu laden („UPLOAD“)
- **"BA Busy: No connection to EIB"** Das Gateway hat keine Verbindung zum EIB
- **"BA Error: Internal Error"** Interner Fehler des Gateway-Busankopplers
- **"BA-Layer: EIB problem"** Probleme beim Zugriff auf EIB
- **"Bad ADD address: <Adresse>"** Die im Befehl ADD angegebene Gruppenadresse ist ungültig
- **"Bad ADD number: <Nr>"** Die im Befehl ADD angegebene Nummer ist ungültig (0 oder >255)
- **"Bad ADD parm count"** Der ADD-Befehl ist unvollständig oder enthält zu viele Argumente
- **"Bad ADD type: <Typ>"** Im Befehl ADD angegebener Typ ist ungültig (SW, D4, 1B, 2B, 3B, 4B)
- **"Bad ADR parm count"** Der ADR-Befehl enthält keines oder mehr als ein Argument
- **"Bad DEL number: <Nr>"** Die im Befehl DEL angegebene Nummer ist ungültig (0 oder >255)
- **"Bad DEL parm count"** Der DEL-Befehl ist unvollständig oder enthält zu viele Argumente
- **"Bad DEL type: <Typ>"** Im Befehl ADD angegebener Typ ist ungültig (SW, D4, 1B, 2B, 3B, 4B)
- **"Bad HELP parm count"** Der HELP-Befehl enthält mehr als ein Argument
- **"Bad LIST parm count"** Die Parameter-Anzahl des LIST-Befehls ist falsch
- **"Bad LIST start <Nr>"** Die LIST-Befehl enthält eine ungültige Start-Nummer (0 oder >255)
- **"Bad LIST type: <Typ>"** Im Befehl LIST angegebener Typ ist ungültig (SW, D4, 1B, 2B, 3B, 4B)
- **"Bad NOPOLL parm count"** Die Parameter-Anzahl des NOPOLL-Befehls ist falsch
- **"Bad POLL number: <Nr>"** Die im Befehl POLL angegebene Nummer ist ungültig (0 oder >255)
- **"Bad POLL parm count"** Die Parameter-Anzahl des POLL-Befehls ist falsch
- **"Bad POLL trigger - <Typ> <Nr> unused"** Der im WHEN...POLL-Befehl angegebene Trigger wird nicht verwendet
- **"Bad POLL type: <Typ>"** Im Befehl POLL angegebener Typ ist ungültig (SW, D4, 1B, 2B, 3B, 4B)
- **"Bad Poll <Typ> #<Nr> unused"** Das im Befehl POLL angegebene Element wird nicht verwendet
- **"Bad SET - <Typ> <Nr> unused"** Das im Befehl SET angegebene Element wird nicht verwendet
- **"Bad SET <Typ> - illegal value"** Die angegebenen Werte sind nicht zulässig
- **"Bad SET <Typ> - wrong parm len"** Die Parameter-Anzahl des SET-Befehls passt nicht zum angegebenen Typ
- **"Bad SET number <Nr>"** Die im Befehl SET angegebene Nummer ist ungültig (0 oder >255)
- **"Bad SET parm count"** Die Parameter-Anzahl des SET-Befehls ist falsch
- **"Bad SET type <Typ>"** Der im Befehl SET angegebene Typ ist ungültig (SW, D4, 1B, 2B, 3B, 4B)
- **"Bad WHEN - POLL expected"**
- **"Bad WHEN - START expected"** Der Aufbau des WHEN...POLL –Befehls ist falsch
- **"Bad WHEN dest number: <Nr>"**
- **"Bad WHEN number: <Nr>"** Die im Befehl WHEN...POLL angegebene Nummer ist ungültig (0 oder >255)
- **"Bad WHEN dest type: <Typ>"**
- **"Bad WHEN type: <Typ>"** Der im Befehl WHEN...POLL angegebene Typ ist ungültig (SW, D4, 1B, 2B, 3B, 4B)
- **"Bad WHEN parm count"** Die Parameter-Anzahl des WHEN...POLL -Befehls ist falsch
- **"CRC: Gateway EEPROM CRC-Error"** Interner Prüfsummenfehler des Gateways
- **"Gateway Timeout"** Das Gateway hat mehr als zwei Sekunden nicht geantwortet (Verkabelung?)
- **"NAK: No EIB Device at <Adresse>"** Auf die Adresse reagiert kein Gerät am EIB
- **"Poll queue overflow"** Die Box hat mehr Poll-Anfragen, als das Gateway bearbeiten kann
- **"Send buffer overflow"** Der Sendepuffer der Box ist voll
- **"Term Timeout"** Im Terminal-Modus wurde 40sec lang kein Befehl eingegeben
- **"Term overflow... Bye"** Der aktuelle (Terminal-)Befehl ist länger als 80 Zeichen
- **"Tx: Transmit buffer overflow"** Der Sendepuffer des Gateways ist voll
- **"Unknown: <Befehl>"** Die Box kennt den angegebenen Befehl nicht