XSU - X25 Schnittstellen Umsetzer

Systemdokumentation

Autor: **Heimo Schön**
Version: **1.04**
Ausgabestand: **26.05.2009**

Version	Autor	Datum	Änderung
00.c	HS	07.10.2000	vorläufig endgültige Version zur Auslieferung der ersten Projekte
1.00	HS	14.03.2004	Portierung des Dokuments von Win-Word nach OpenOffice; redaktionelle Überarbeitung; technische Korrektur aller Änderungen im XSU seit Auslieferung
1.01	HS	-	redaktionelle Zwischenversion
1.02	HS	30.01.2009	Überarbeitung nach Portierung auf neue Hardware (bis V1.01 waren Advantech-Rechner im Einsatz und die LRP-Plattform)
1.03	HS	20.04.2009	Änderungen/Aktualisierungen für Generic-XSU Dokumentation, Kabel beschrieben Anpassungen und Korrekturen an neue Plattform eingearbeitet, Bild ergänzt
1.04	HS	18.05.2009	Korrekturen und Ergänzungen eingepflegt,die sich aus der Inbetriebnahme der Parallelausleuchtung im Verschiebebahnhof Linz ergeben haben
	HS	26.5.2009	Abweichung Doku/Code bei der Zugnummernlogik behoben

1. Inhaltsverzeichns

2. Allgemeines

2.1. Vorwort

Der XSU wurde entwickelt um Windows-Applikationen, z.B. EBO-Parallelausleuchtungen oder Nebenstreckenstellwerke, an eine ÖBB-X.25 Gegenstelle (RZÜ oder ESTW) anzubinden. Wenn in diesem Dokument EBO genannt wird, dann kann implizit auch jede andere Applikation verwendet werden, die mit einer „echten ÖBB X25 Gegenstelle“ kommunizieren möchte. Der XSU erfüllt die Anforderung des ÖBB X25 Pflichtenhefts und ermöglicht der Applikation (in diesem Dokument EBO genannt), mit einer X25 Gegenstelle zu kommunizieren.

Im Kapitel 4.4. Telegrammaufbau Seite 21 finden Sie eine Telegrammbeschreibung die der XSU von der EBO-Applikation erwartet.

2.2. Überblick

Der XSU X25 Schnittstellenumsetzer soll einer EBO ermöglichen mit ÖBB-X25 Gegenstellen zu kommunizieren. Der XSU dient in erster Linie dazu, selbständig das Protokoll entsprechend [1] Pflichtenheft X25 – Schnittstelle abzuwickeln. Die Schnittstelle zur EBO wird an das Plichtenheft X25-Schnittstelle angelehnt und stellt ein Subset der Funktionen des ÖBB-Pflichtenhefts X25-Schnittstelle dar.

Die EBO soll durch den Einsatz des XSU von den harten technischen Anforderungen des ÖBB X25 Pflichtenhefts befreit werden. Der XSU wickelt das X25 Protokoll zur X25-Gegenstelle selbständig ab und ist seinerseits wiederum vom betrieblichen Wissen der von ihm verwalteten (gespeicherten) und weitergeleiteten Daten befreit. Durch diese Gewaltentrennung können EBO und XSU die jeweils ihnen übertragenen Funktionen selbständig durchführen:

- XSU wickelt das X25-Protokoll ab und hat kein Wissen von den Dateninhalten
- EBO erzeugt die Daten und verarbeitet Telegramme von der Gegenstelle und muss sich nicht um Verbindungsaufbau, Verbindungsüberwachung, Generalabfragen, usw. kümmern

2.3. Zweck des XSU

Der XSU übernimmt in der Richtung zur X.25 Gegenstelle (Elektronisches Stellwerk oder RZÜ) die Verantwortung für den vollen Telegrammverkehr entsprechend dem ÖBB X25 Pflichtenheft. Der XSU macht den Verbindungsaufbau, die Verbindungsüberwachung, die Generalabfrage, usw. All das macht der XSU unabhängig davon, ob die X.25-Applikation auf der V.24 Seite gerade aktiv ist, oder nicht. In Richtung X.25 ändert nach dem Verlust der V.24 Seite, nur das Verhalten auf empfangene Kommandos. Die X.25 Verbindung wird jedoch nicht abgebaut oder gestört. Die X.25 Gegenstelle hat daher keinen Grund einen AFE (Anlagenfehler) oder andere Störungen anzuzeigen, egal was auf der EBO Schnittstelle gerade passiert.

Auf der Seite zur EBO kann daher ein Büro-PC mit Büro-Betriebssystem eingesetzt werden. Dieser PC muss nicht mehr die harten Anforderungen des ÖBB-Pflichtenhefts erfüllen, sondern kann wie folgt vereinfacht werden:

- Lebenstakte sind nur noch alle 15 Sekunden erforderlich
- bei Ausfällen des PCs bleibt die X25 Verbindung zwischen XSU und Gegenstelle (ESTW, RZUE, usw.) weiter aufrecht (keine Anzeige von Anlagenfehlern AFE in der X25-Gegenstelle)
- Kommandos werden bereits vom XSU quittiert und dem PC nur noch zugestellt ohne das eine Quittung erforderlich ist
- Der XSU ist selbstlernend und bedarf keiner Projektierung

- Der XSU speichert alle Elementmeldungen und im Falle einer Generalabfrage der EBO-Anwendung werden die gespeicherten Elemente erneut gesendet (keine Verbindungsabbau/aufbau mit dem ESTW erforderlich)
- Zugnummernmeldung werden in der GA korrigiert
- X25-Leitungszustand der X.25 wird an EBO gemeldet
- Wiederinbetriebnahmehandlung wird vom XSU selbständig abgewickelt

Der XSU ist entwickelt für extrem hohe Verfügbarkeit. UPTIMEs von mehreren Jahren ohne Reboot sind der Regelfall.

2.4. Funktionale Beschränkung des XSU

Der XSU ist nur für bestimmte Hauptgruppen ausgelegt (muss bei ganz neuen oder Sonderhauptgruppen u.U. projektspezifisch angepasst werden – weiter unten werden alle Hauptgruppen beschrieben die vom XSU schon heute verarbeitet werden – es sind alle Standardhauptgruppen enthalten).

Der XSU darf nur in einkanaligen nicht sicherheitsrelevanten Verbindungen eingesetzt werden.

Wird der XSU dennoch in sicherheitsrelevantem Umfeld eingesetzt, muss ein Verfahren existieren, das alle möglichen Fehler des XSU (Telegrammverfälschung, -verlust, uvm.) sicher offenbart und gefärliche Zustände verhindert. Es darf aus einem Fehlverhalten des XSU kein Zustand entstehen, der betrieblich oder technisch eine Gefährdung darstellt. Der Anlagenbetreiber/Hersteller der den XSU in ein sicheres System integriert ist für diese Sicherheitsnachweisführung verantwortlich. Wir beraten Sie gerne.

Der XSU ist ausgelegt für hohe Verfügbarkeit und wurde auch entwickelt unter Beachtung vieler Mechanismen die zufällige Fehler offenbaren (z.B. Saubere Trennung der Prozesse durch interne Kommunikation über Unix-Pipes mit Überwachung der übertragenen Daten, usw.), dennoch sind keine Maßnahmen ergriffen worden (Zweikanaligkeit, Diversität, usw.) die eine signaltechnische Sicherheit oder eine Eigensicherheit gewährleisten würden. Wenn Sie eine sichere Lösung benötigen, dann fragen Sie bei uns an – wir helfen gerne weiter.

2.5. ISO Schichtenmodell

Annäherung an das ISO (OSI) 7-Schichten Modell:

Schicht 7	ÖBB-Anwendung	EBO	X.25-Gegenstelle
Schicht 6	-	-	-
Schicht 5	-	-	-
Schicht 4	Buffern/Speichern	XSU	-
Schicht 3	CCITT X.25	PAD	-
Schicht 2b	CCITT LAP-B	PAD	-
Schicht 2a	CCITT X.21	Modem	Modem
Schicht 1	CCITT X.21	Modem	Modem

Abbildung: XSU Einbettung in das ISO (OSI) 7-Schichten Modell

Dieses Dokument beschreibt den Rechner XSU, der zwischen EBO und X25-Gegenstellen die Funktion eines Layer-4 (ISO-Schicht 4) übernimmt.

2.6. Betriebssystem

Das Rechnersystem XSU ist auf einer Linux Plattform aufgebaut. Um Linux auf Hardware mit niedrigem Stromverbrauch und ohne bewegte Teile (keine Harddisk, keine Lüfter) lauffähig zu machen, wird auf ein embedded Linux zurückgegriffen, das von EXD im Laufe des letzten Jahrzehnts entwickelt wurde. Dieses

EXD-Basissystem enthält einen Linux-Kern (dzt. 2.6.27) mit unterlagertem RTAI Realtime Microkernel (dzt. 3.4) und einer Runtime-Umgebung in der alle Dinge enthalten sind, die das EXD-System zum Betrieb benötigt. Darin eingebettet ist die XSU Applikation. Dieses EXD System ist grundsätzlich auf jeder x86 Architektur lauffähig. Wir empfehlen bestimmte Typen, können aber auch Anpassungen an die von Ihnen gewünschte Hardware vornehmen.

3. Funktionale Beschreibung

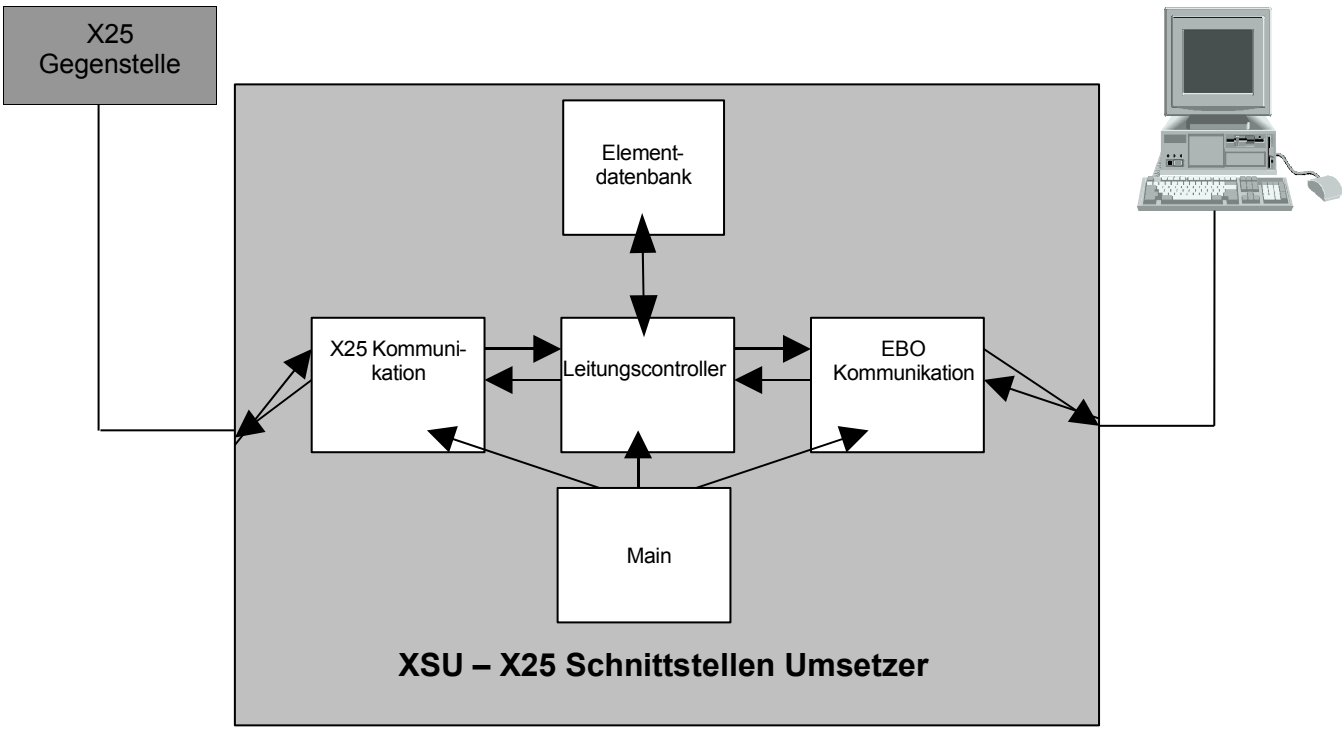
3.1. Architektur der Applikation

Das Rechnersystem XSU ist auf einer Linux Plattform aufgebaut. In der Applikationsebene sind die folgenden Funktionsblöcke implementiert:

1. Leitungs-Statemachine
2. Elementdatenbank
3. X25 Kommunikation zur X25 Gegenstelle (ESTW, RZÜ,...)
4. Kommunikation zum X25-Teilnehmer (EBO)
5. Hauptprogramm

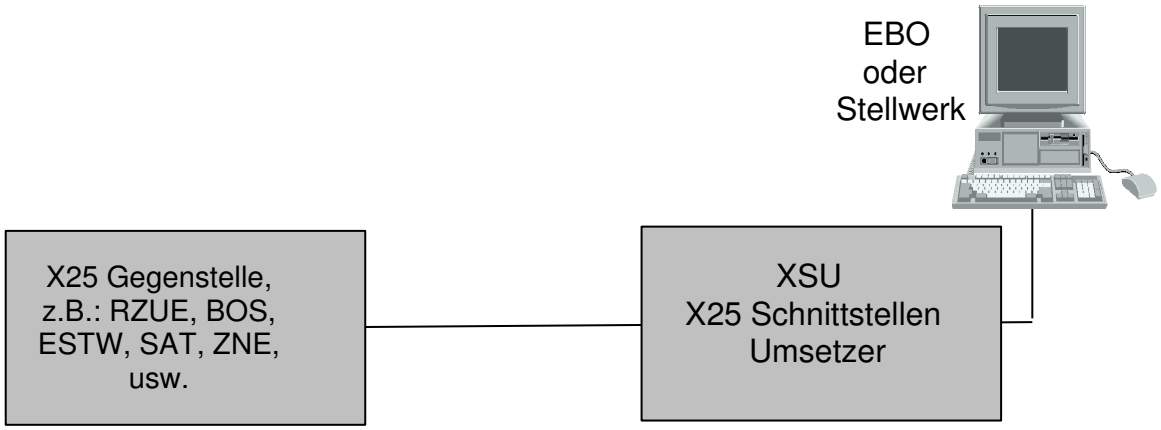
Dies läßt sich durch folgendes Modul-Blockschaltbild darstellen:

3.2. Modulübersicht

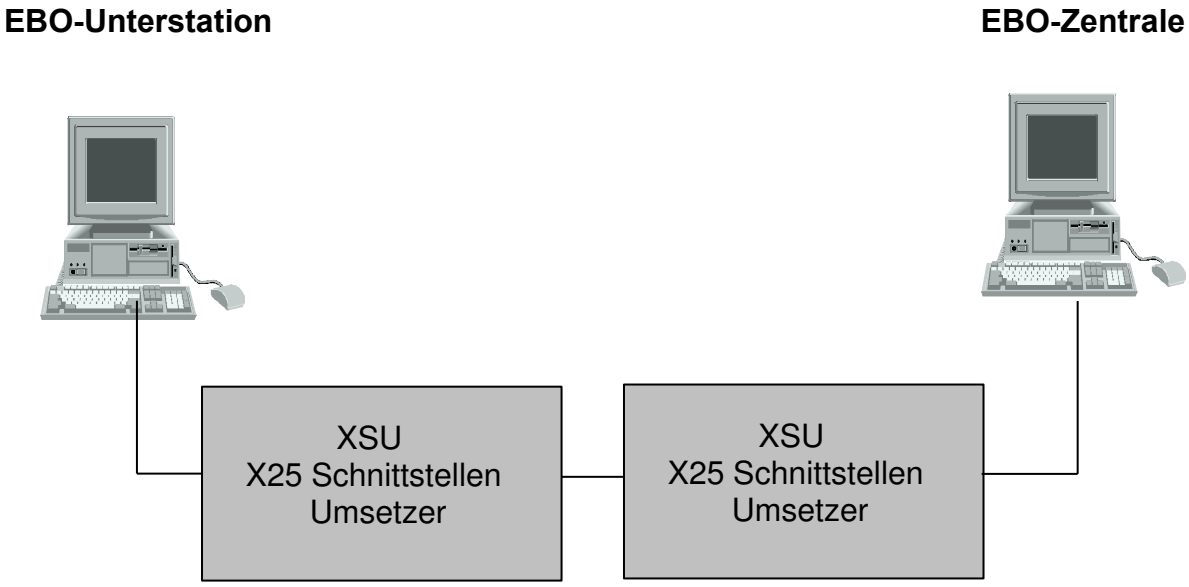


3.3. Konfigurationsvarianten:

3.3.1. EBO über XSU verbunden mit einer X25 Gegenstelle



3.3.2. EBO-Zentrale verbunden mit EBO-Unterstation über XSU (Rücken an Rücken)



3.4. technische Daten der XSU-Schnittstellen

3.4.1. Serielle Schnittstellen

Bei den im vorigen Kapitel gezeigten Varianten gilt zu beachten, daß ein XSU zwei Schnittstellen besitzt:

V.24 Schnittstelle zum X25-Teilnehmer EBO 19.200 baud, no parity, 8 Datenbit, 2 Stopbit
V.24 Schnittstelle zur X.25 Gegenstelle ESTW 19.200 baud, no parity, 8 Datenbit, 2 Stopbit

Bei der Schnittstelle zur EBO kann direkt mit einem ausgekreuzten V.24 Kabel (Pin 2 und 3 ausgekreuzt und Pin 5 1:1 verbunden) verbunden werden. Bei Enternungen bis ca. 10 Meter empfiehlt sich die Verwendung geschirmter Kabel, wobei der Schirm einseitig mit dem PIN 1 verbunden werden sollte. Bei größeren Längen sollte ein V.24 Standleitungsmodem eingesetzt werden, ode eine vergleichbare Technik z.B. mit Glasfaserumsetzern herangezogen werden.

Die Schnittstelle zur X25 Gegenstelle ist jedoch noch nicht geeignet für die direkte Verbindung und ist durch ein X.25 PAD umzusetzen auf die bei den ÖBB verwendete CCITT X.21 Schnittstelle. Das PAD ist NICHT Teil des Lieferumfangs. Die XSU Schnittstelle ist so vorberetiet, daß die bei den ÖBB üblichen PADs eingesetzt werden können.

3.4.2. Stromversorgung

Die Stromversorgung des XSU erfolgt mit einem fest montierten Steckernetzgerät. Der XSU hat eine Stromaufnahme von ca. 2,9 Watt. Inklusive Steckernetzgerät bewegt sich die Stromaufnahme deutlich unter 10 Watt.

3.4.3. Netzwerkschnittstelle

An der Netzwerkschnittstelle kann jedes CAT-5 Kabel mit 10/100 Mbit angeschlossen werden. Der XSU wird üblicherweise mit IP-Adresse 192.168.1.11 ausgeliefert. Auf dieser IP-Adresse „lauscht“ der XSU am Port 80 (HTTP-Port) und kann über dieses Port mit einem handelsüblichen Web-Browser bedient und konfiguriert werden. Mit Internetexplorer erreichen Sie den XSU über die Adresse

http://192.168.1.11

Wie Sie Ihren PC konfigurieren müssen, zeigt Ihnen Ihr Netzwerkadministrator. Die IP-Adresse und viele andere Parameter des XSU sind über das Webinterface konfigurierbar.

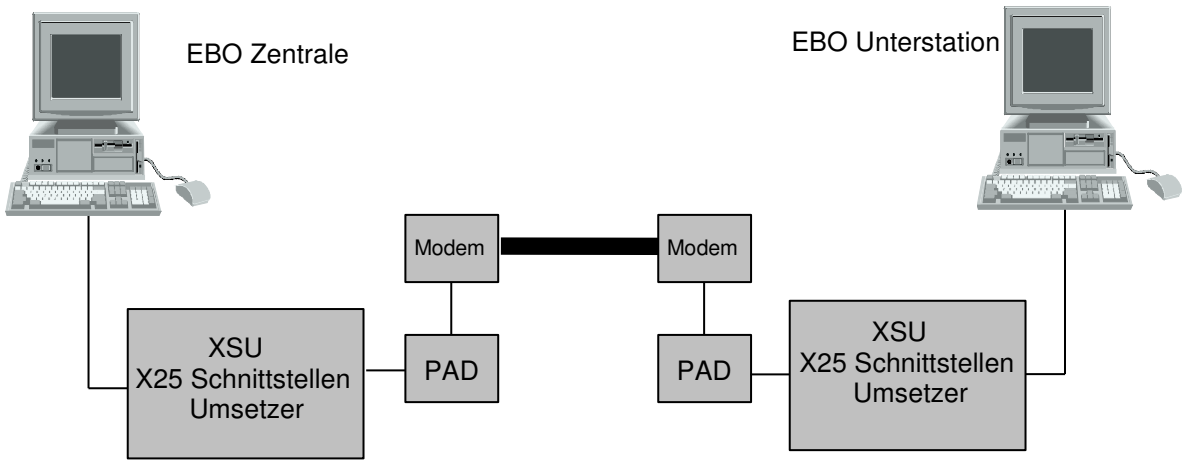
3.5. PAD – Paket-Assembler-Disassembler

3.5.1. Allgemeines zu PADs

Paket Assembler-Disassembler werden benötigt um den kontinuierlichen Datenstrom der von den V24-Teilnehmern eintrifft (z.B. von der EBO), in X.25 Pakete aufzubrechen. Dieses Aufbrechen ist derzeit nur bei den Teilnehmern im X25-Datennetz der ÖBB notwendig, die grundsätzlich und immer mit X.25 Paketen operieren (derzeit nur ALCATEL ESTW Elektra). Bei allen anderen Teilnehmern sind PADs somit nur dann erforderlich, wenn man zusätzlich auch noch eine Leitungsverlängerung mit X.21 Modems erreichen möchte (schreibt ÖBB vor, sobald eine Leitung über mehrere 10 Meter weit geführt werden muss und/oder das Gebäude verläßt). Aus diesem Grund sind PADs vorgeschrieben, bei allen Datenverbindungen die über Raumgrenzen hinweggehen und eine Übersetzung/Umwandlung von V.24 auf X.21 gefordert ist. Solange die Datenverbindung im eigenen Relais/Fernmelderaum bleibt, kann auch ohne PADs (und somit auch ohne Modem) direkt (ausgekreuztes V.24 Kabel oder V.24 Modemleitung) verbunden werden (Ausnahme: wenn die X25-Gegenstelle ein ESTW Elektra-1 ist – dann ist ein PAD immer erforderlich).

3.5.2. PAD Beispielkonfiguration

Die oben gezeigte Konfiguration mit Zentrale und Unterstation sieht somit nach Vervollständigung mit PAD und Modem wie folgt aus (wurde oben zum Verständnis vereinfacht dargestellt):



3.5.3. XSU-PAD Schnittstelle

Der Paket-Assembler-Disassembler besitzt zwei Schnittstellen. Eine X.21 Schnittstelle an der das Modem angeschlossen wird. An dieser Schnittstelle wird vom PAD CCITT X.25 Protokollverkehr abgewickelt. Die zweite Schnittstelle des PAD ist eine V.24 Schnittstelle mit 19.200 baud, auf nach CCITT X.3 Protokoll verfahren wird. Die Funktion des PAD ist, den Datenstrom von der X.3 Schnittstelle in X.25 Pakete zu verpacken und über die X.21 Schnittstelle und die Modemstrecke zu versenden. Empfangene X.25 Pakete werden entpackt und an der X.3 Schnittstelle an den XSU übergeben.

Die Steuerung mit der das Senden eines X.25 Pakets ausgelöst werden kann, ist entsprechend CCITT X.3 Protokoll festgelegt über die folgenden Möglichkeiten:

- maximale im PAD eingestellte Telegrammlänge (128 byte) ist erreicht
- Ablauf der im PAD eingestellten Timeout-Zeit

Die Möglichkeit das Senden von Telegrammen über die maximale Telegrammlänge auszulösen ist entsprechen [1] Pflichtenheft X25 – Schnittstelle nicht zulässig, da in diesem Fall verstümmelte Telegramme in X.25 Pakete verpackt werden. Damit bleibt für ÖBB Anwendungen nur die Steuerung über Timeout offen. Die bei ÖBB eingesetzten PADs (zur Zeit TA-Omega) sind auf den kürzest möglichen Timeout von 20 ms eingestellt. Um zu gewährleisten, daß die Timeoutzeit immer überschritten wird, wurde im XSU ein Sendetimeout von 50 ms gewählt. Der XSU wartet beim Senden an die X.25 Gegenstelle zwischen zwei Paketen immer die Timeoutzeit von 50 ms ab. Um im Falle langer Generalabfragen (GA) zu gewährleisten, daß Lebenstakte regelmäßig (alle 2 Sekunden) gesendet werden, ist im XSU an der X25 Schnittstelle eine Sendequeue implementiert, in der Lebensakte am vorderen Ende einsortiert werden und Elementdaten am Ende der Queue angefügt werden.

Mit diesen Maßnahmen ist der XSU für den Einsatz an PAD-Schnittstellen vorgerüstet und getestet. Die Anwenderapplikation an der EBO Schnittstelle ist von dieser Anforderung entbunden.

3.6. Funktionsübersicht

3.6.1. Mainprozeß

Der Mainprozeß beinhaltet folgende Funktionen:

- Starten aller Child-Prozesse
- Überwachung aller Child Prozesse
- Diagnosesubsystem

3.6.2. Leitungsstatemachine

Diese Statemachine ist für die Verwaltung der Zuständer der beiden Leitungen verantwortlich und entscheidet über die folgende Zustände

- Meldungsweiterleitung
  - Erkennung von zulässigen Meldungen
  - Weiterleitung an die Gegenstelle falls diese erreichbar ist
- Kommandoverarbeitung
  - Erkennung zulässiger Kommandos (zur Zeit nur Kommando Zugnummer geplant)
  - Selbständiges Aussenden einer negativen Kommandoquittung wenn Gegenstelle nicht erreichbar
  - Weiterleitung des Kommandos an die Gegenstelle wenn die Gegenstelle erreichbar ist
  - Empfangene Kommandoquittungen werden nur an die Gegenstelle weitergeleitet
- Generalabfrage
  - Abwicklung der Generalabfrage
  - Blockweises senden der Elemente
  - Verwaltung der Zulässigkeit der GAs
  - Verwaltung der Vollständigkeit der GAs in beiden Richtungen
- Verwaltung der Leitungszustände
  - Auswertung der Disconnects (unabhängig ob vom XSU selbst generiert oder von der Gegenstelle)
  - Anstoß der GA
  - Speichern der Zustände

3.6.3. Pipecontroller

Der Pipecontroller wird unterteilt in folgende Funktionsgruppen:

- Zugnummernlogik
  - Einbruch
  - Ausbruch
  - Löschen
  - Umsetzung bei GA
- Eintragung in die Elementdatenbank
  - Alle Meldungen ausg. Zugnummern werden ohne weitere Prüfung richtungsabhängig (von EBO oder von X25) in der Datenbank gespeichert
  - Zugnummern werden abhängig vom Steuerwort 1 gespeichert und/oder gelöscht
  - Gespeicherte Zugnummern werden immer mit Steuerwort 1 Bit Einbruch gespeichert und damit in der nächsten GA als Einbruch gesendet

- Meldungsverarbeitung
  - Speichern eintreffender Meldungen in der Elementdatenbank
  - Weiterleiten von Meldungen wenn Elementdatenbank sich ändert

- GA-Logik
  - Auswertung der Lebenstakttimeouts der Kommunikationsprozesse
  - KAZD-Anstoss an Elementdatenbank
  - Aussenden aller Elemente bei GA-Request von Gegenstelle

3.6.4. Elementdatenbank

Die Elementdatenbank speichert alle X25 Meldungen die ihr vom Leitungskontroller vorgelegt werden. Die Elementdatenbank speichert alle Meldungshaupgruppen, ausgenommen Lebenstakte und Quittungen. Die Elementdatenbank beherrscht folgede Funktionen:

- Elementdatenbank in allen Richtungen initialisieren (Reboot)
- Zustand der Einträge (für GA) verwalten
- Elementdatenbankeinträge generieren bei neuen Elementen
- Elementzustand in Elementdatenbank speichern
- Elementzustand aus Elementdatenbank auslesen
- Alle Elemente aus der Elementdatenbank auslesen (GA)

3.6.5. Kommunikationsmodule

Kommunikation zum X25-Teilnehmer EBO

- Verbindungsaufbau (vereinfachte GA)
- LC-Mechanismus / Verbindungsüberwachung
- Sendeprozeß mit Sendequeue
- Empfangsprozeß

Kommunikation x25-Gegenstelle (ESTW, RZUE, usw)

- Verbindungsaufbau (GA-Statemaschine)
- LC-Mechanismus / Verbindungsüberwachung
- Sendeprozeß mit Sendequeue
- Empfangsprozeß
- Telegrammfolgenummern Generator / Überwachung
- CRC Generator / Überwachung

3.6.6. Systemeinbettung

Die Systemeinbettung der Applikation erfolgt über ein Applikations Startscript, das aus /etc/inittab heraus vom Betriebssystem gestartet wird. Das Startscript und die Applikation liegen im Directory /home/heimo.

3.7. Verwendete Hardware

Um eine hohe Verfügbarkeit und Lebensdauer garantieren zu können wird auf eine 486 HW eingesetzt, da zur Zeit der 486 Prozessor bei hoher Rechenleistung bereits niedrige Temperaturen und niedrigen Stromverbrauch garantiert, da die ständige Weiterentwicklung in der Prozessortechnologie auch in den embedded Systemen eingesetzt wird.

Für die X25 Übertragungsseite wird zur Umsetzung von der V.24 Übertragung auf das X.25 Protokoll mit X.21 Übertragung, ein PAD der Marke TA-OMEGA verwendet.

Die Modems werden üblicherweise von der ÖBB bereitgestellt. Der Grund liegt darin, daß die ÖBB besser einschätzen kann, welche Kabelqualitäten verwendet werden und daher besser den Modemtyp abstimmen kann. Die Schnittstelle für den X.25 Lieferanten ist daher üblicherweise ein 15-poliger Stecker (female/Buchsen) mit X.21 Protokoll, an der Rückseite eines Modems. Der Lieferant liefert neben seiner Sicherungstechnik, einen XSU, ein PAD und alle erforderlichen Kabel bis zum Modem.
Hinweis: die X.21 Schnittstelle kann bis zu 100 m lang sein.

3.8. Elementarten

3.8.1. Allgemeines zu den Hauptgruppen

Die Hauptgruppen Lebenstakt, Kommandoquittung, Kommando Generalabfrage und Quittung Generalabfrage, werden von der Statemachine und den Kommunikationsprozessen verarbeitet.

Lebenstaktgeneratoren versorgen die beiden Leitungen mit Lebenstakten. Der XSU sendet daher auf allen Leitungen im zwei Sekundentakt Lebenstakte die nur in den Lebenstaktpausen nach einem Disconnect für 7 Sekunden unterbrochen werden um die Gegenstelle ebenfalls in den Verbindungsabbruch zu zwingen.

Meldung Zugnummer (HGR 53) werden vom Leitungscontroller in der Zugnummernlogik verarbeitet. Kommando Zugnummer (HGR 174) wird wie alle anderen Kommandos nur weitergeleitet ohne zu speichern oder Zustände zu übersetzen.

Kommandos, ausgenommen Kommando Generalabfrage (255), werden vom XSU automatisch beantwortet (mit ACK wenn die Gegenstelle verfübar ist und mit NACK wenn die Gegenstelle im Moment nicht verfübar ist) und an die Gegenstelle ohne Änderung weitergeleitet. Insbesondere Kommando Zugnummer (HGR 174) wird nicht gespeichert für die nächste GA. Jeder Teilnehmer ist daher bei einem Verbindungsausfall dazu verplichtet, alle Zugnummernfelder zu löschen.

Alle anderen Meldungen (die nicht Lebenstakt, Kommandoquittung, Quittung Generalabfrage oder Zugnummer sind) werden vom Leitungscontroller richtungsabhängig in der Elementdatenbank gespeichert und im Fall einer GA wieder aus der Elementdatenbank entnommen und gesendet.

3.8.1.1. Meldungen

Meldung 0: Verbindungsabbruch	verwendet der XSU in beiden Richtungen *1)
Meldung 1: Weiche,DKW,EKW	wird weitergeleitet und in der Datenbank gespeichert
Meldung 2: Gleiskreuzung	wird weitergeleitet und in der Datenbank gespeichert
Meldung 3: Flachkreuzung	wird weitergeleitet und in der Datenbank gespeichert
Meldung 4: Sperrschuh (elektr. Fernbedient)	wird weitergeleitet und in der Datenbank gespeichert
Meldung 5: Schlüsselsperre im Bahnhofsbereich	wird weitergeleitet und in der Datenbank gespeichert
Meldung 6: Schlüsselsperre auf der freien Strecke	wird weitergeleitet und in der Datenbank gespeichert
Meldung 7: Verschlußmelder (SpDrS)	wird weitergeleitet und in der Datenbank gespeichert
Meldung 10: Startmeldungen	wird weitergeleitet und in der Datenbank gespeichert
Meldung 11: Zielmeldungen	wird weitergeleitet und in der Datenbank gespeichert
Meldung 12: Schutzwegmeldungen	wird weitergeleitet und in der Datenbank gespeichert
Meldung 15: Startpunkt	wird weitergeleitet und in der Datenbank gespeichert
Meldung 16: Hauptsignal	wird weitergeleitet und in der Datenbank gespeichert
Meldung 17: Vorsignal	wird weitergeleitet und in der Datenbank gespeichert

Meldung 18: Schutzsignal	wird weitergeleitet und in der Datenbank gespeichert
Meldung 19: Verschubsignal	wird weitergeleitet und in der Datenbank gespeichert
Meldung 20: Geschwindigkeitsanzeiger	wird weitergeleitet und in der Datenbank gespeichert
Meldung 21: Geschwindigkeitsvoranzeiger	wird weitergeleitet und in der Datenbank gespeichert
Meldung 22: Bremsprobesignal und Signal Abfahrt	wird weitergeleitet und in der Datenbank gespeichert
Meldung 30: nicht isoliertes Gleis	wird weitergeleitet und in der Datenbank gespeichert
Meldung 31: isoliertes Gleis	wird weitergeleitet und in der Datenbank gespeichert
Meldung 32: isoliertes Streckengleis	wird weitergeleitet und in der Datenbank gespeichert
Meldung 33: Freimeldeabschnitt/Schienenkontakt	wird weitergeleitet und in der Datenbank gespeichert
Meldung 40: Streckenblock	wird weitergeleitet und in der Datenbank gespeichert
Meldung 41: Nahbedienung	wird weitergeleitet und in der Datenbank gespeichert
Meldung 42: EKSA	wird weitergeleitet und in der Datenbank gespeichert
Meldung 43: Hinweisschilder / AWS	wird weitergeleitet und in der Datenbank gespeichert
Meldung 44: Zustimmungen	wird weitergeleitet und in der Datenbank gespeichert
Meldung 45: Bahnhofssperre	wird weitergeleitet (ohne speichern in der Datenbank)
Meldung 46: Hilfssperren	wird weitergeleitet und in der Datenbank gespeichert
Meldung 47: Stromversorgung	wird weitergeleitet und in der Datenbank gespeichert
Meldung 48: Summenmeldungen	wird weitergeleitet (ohne speichern in der Datenbank)
Meldung 49: Summenmeldungen für Bereiche	wird weitergeleitet und in der Datenbank gespeichert
Meldung 50: Störungsmeldungen	wird weitergeleitet (ohne speichern in der Datenbank)
Meldung 51: Ausschlußtexte	wird weitergeleitet (ohne speichern in der Datenbank)
Meldung 52: Protokolltext	wird weitergeleitet (ohne speichern in der Datenbank)
Meldung 53: Zugnummernmeldung	wird weitergeleitet und in der Datenbank gespeichert (Sonderbehandlung)
Meldung 54: SAS / AWS	wird weitergeleitet und in der Datenbank gespeichert
Meldung 55: Fahrscheinautomat/Gepäckschließfach (Konzept)	wird weitergeleitet und in der Datenbank gespeichert
Meldung 56: Abstoß/Umsetzbewegung	wird weitergeleitet und in der Datenbank gespeichert
Meldung 57: Fernbedienung	wird weitergeleitet (ohne speichern in der Datenbank)
Meldung 58: Selbststellbetriebsverzeichnis	wird weitergeleitet (ohne speichern in der Datenbank)
Meldung 60: Selbststellbetrieb	wird weitergeleitet und in der Datenbank gespeichert
Meldung 61: Hilfssperrenbedienmeldung	wird weitergeleitet und in der Datenbank gespeichert
Meldung 70: Universalelement	nyi (not yet implemented)
Meldung 84: Reserviert für die Fa. Siemens	nyi (not yet implemented)
Meldung 85: Reserviert für die Fa. Siemens	nyi (not yet implemented)
Meldung 86: Reserviert für die Fa. Alcatel	nyi (not yet implemented)
Meldung 87: Reserviert für die Fa. Alcatel	nyi (not yet implemented)
Meldung 88: Reserviert für die Fa. Alcatel	nyi (not yet implemented)
Meldung 89: Reserviert für die Fa. Siemens	nyi (not yet implemented)
Meldung 90: E-Dienst:Störungen-Bahnhof/Bereich (Konzept)	nyi (not yet implemented)
Meldung 91: E-Dienst:Schaltermeldungen (Konzept)	nyi (not yet implemented)
Meldung 92: E-Dienst:Schaltgruppen (Konzept)	nyi (not yet implemented)
Meldung 93: E-Dienst:Reserve	nyi (not yet implemented)
Meldung 94: E-Dienst:Reserve	nyi (not yet implemented)
Meldung 100: RZÜ:Fahrstraßen	wird weitergeleitet und in der Datenbank gespeichert)
Meldung 122: Bedienabbruch	wird weitergeleitet (ohne speichern in der Datenbank)
Meldung 123: Druckquittierungen	wird weitergeleitet (ohne speichern in der Datenbank)
Meldung 124: Quittung Generalabfrage	verwendet der XSU in beiden Richtungen *1)
Meldung 125: Datum und Uhrzeit	wird weitergeleitet (ohne speichern in der Datenbank)
Meldung 126: Lebenstaktframe	verwendet der XSU in beiden Richtungen *1)
Meldung 127: Kommandoquittung	verwendet der XSU in beiden Richtungen *1)

*1) kein Speichern und kein weiterleiten dieser Hauptgruppe

3.8.1.2. Kommandos

Kommando 129: Weiche,DKW,EKW	wird automatisch quittiert und weitergeleitet
Kommando 130: Gleis / Flachkreuzung	wird automatisch quittiert und weitergeleitet
Kommando 131: Sperrschuh (elektr. fernbedient)	wird automatisch quittiert und weitergeleitet
Kommando 132: Schlüsselsperre im Bahnhofsbereich	wird automatisch quittiert und weitergeleitet
Kommando 133: Schlüsselsperre auf der freien Strecke	wird automatisch quittiert und weitergeleitet
Kommando 134: Verschlußmelder (SpDrS)	wird automatisch quittiert und weitergeleitet
Kommando 140: Regelfahrstraßen	wird automatisch quittiert und weitergeleitet
Kommando 141: Umwegfahrstraßen	wird automatisch quittiert und weitergeleitet
Kommando 150: Hauptsignal	wird automatisch quittiert und weitergeleitet
Kommando 151: Vorsignal	wird automatisch quittiert und weitergeleitet
Kommando 152: Schutzsignal	wird automatisch quittiert und weitergeleitet
Kommando 153: Verschubsignal	wird automatisch quittiert und weitergeleitet
Kommando 154: Startpunkt	wird automatisch quittiert und weitergeleitet
Kommando 160: Gleisfreimeldung	wird automatisch quittiert und weitergeleitet
Kommando 165: Streckenblock	wird automatisch quittiert und weitergeleitet
Kommando 166: Nahbedienung	wird automatisch quittiert und weitergeleitet
Kommando 167: EKSA	wird automatisch quittiert und weitergeleitet

Kommando 168: Hinweisschilder	wird automatisch quittiert und weitergeleitet
Kommando 169: Zustimmungen	wird automatisch quittiert und weitergeleitet
Kommando 170: Bahnhofssperre	wird automatisch quittiert und weitergeleitet
Kommando 171: Hilfssperren	wird automatisch quittiert und weitergeleitet
Kommando 172: Stromversorgung	wird automatisch quittiert und weitergeleitet
Kommando 173: Fehlerabfragen des Stör-/Fehlerspeichers	wird automatisch quittiert und weitergeleitet
Kommando 174: Zugnummer	wird automatisch quittiert und weitergeleitet
Kommando 175: SAS / AWS	wird automatisch quittiert und weitergeleitet
Kommando 176: Abstoß/Umsetzbewegungen	wird automatisch quittiert und weitergeleitet
Kommando 177: Fernbedienung	wird vom XSU intern verwendet und NICHT weitergeleitet
Kommando 178: Selbststellbetrieb - Bedienung	wird automatisch quittiert und weitergeleitet
Kommando 179: Quittierungspflichtige Ausdrucke	wird automatisch quittiert und weitergeleitet
Kommando 180: Selbststellbetriebsverzeichnis	wird automatisch quittiert und weitergeleitet
Kommando 182: Selbststellbetriebstelegramm an Nachbaranlage	wird automatisch quittiert und weitergeleitet
Kommando 190: Universalelement	nyi (not yet implemented)
Kommando 234: Reserviert für die Fa. Siemens	nyi (not yet implemented)
Kommando 235: Reserviert für die Fa. Siemens	nyi (not yet implemented)
Kommando 236: Reserviert für die Fa. Alcatel	nyi (not yet implemented)
Kommando 237: Reserviert für die Fa. Alcatel	nyi (not yet implemented)
Kommando 238: Reserviert für die Fa. Alcatel	nyi (not yet implemented)
Kommando 239: Reserviert für die Fa. Siemens	nyi (not yet implemented)
Kommando 240: E-Dienst: Notgruppe, Quittierungen (Konzept)	nyi (not yet implemented)
Kommando 241: E-Dienst:Schalterbetätigung (Konzept)	nyi (not yet implemented)
Kommando 242: E-Dienst:Reserve	nyi (not yet implemented)
Kommando 243: E-Dienst:Reserve	nyi (not yet implemented)
Kommando 244: E-Dienst:Reserve	nyi (not yet implemented)
Kommando 254: Datum und Uhrzeit	wird automatisch quittiert und weitergeleitet
Kommando 255: Generalabfrage	wird vom XSU intern verwendet und NICHT weitergeleitet

3.9. Projektierung und Konfiguration

Für die zentrale XSU-Funktion sind keine Projektierungen vorgesehen und auch nicht erforderlich (abgesehen von technischen Funktionen die nicht mit der XSU-Funktion zusammenhängen, wie z.B. IP-Adressen-Einstellungen usw.). Um dies zu erreichen sind die in den folgende Kapiteln beschriebenen Maßnahmen notwendig. Der XSU wird so gestaltet, daß er bei Einsatz auf einer anderen Anlage oder in einer anderen externen Konfiguration, sich automatisch an die neuen Gegebenheiten anpasst und keine Software oder Projektierungsänderungen notwendig sind.

3.9.1. Elementdatenbank

Die Elementdatenbank ist nach dem Hochlauf des XSU leer. Bei jeder eintreffenden Element-Meldung in einer GA oder im laufenden Betrieb, in irgend einer Richtung (von EBO oder von X25 Gegenstelle), die noch nicht in der Elementdatenbank gespeichert ist, wird ein neuer Datenbank-Record angelegt. Als Schlüsselfelder sind in der Datenbank die Hauptgruppe und die Elementnummer verwendet worden. Jedes Meldetelegramm wird eindeutig, durch die Kombination von Hauptgruppe und Elementnummer.

Solange kein GA-Ende von der Gegenstelle (X25 oder EBO) empfangen wurde, darf diese Richtung (X25 oder EBO) nicht als Elementspeicher für eine GA in der “jeweils anderen” Richtung verwendet werden, da noch nicht garantiert werden kann, das die Elementdatenbank vollständig mit Elementzuständen „geladen“ ist Das selbe anders formuliert: eine GA in Senderichtung zur X25-Gegenstelle wird erst begonnen, wenn von der EBO ein GA-Ende empfangen wurde und eine GA in Senderichtung zur EBO wird erst begonnen, wenn ein GA-Ende von der X25-Gegenstelle empfangen wurde. Eintreffende Kommandos Generalabfrage werden daher mit Kommandoquittungen mit gesetztem Bit ABLEHNUNG beantwortet, solange die GA von der EBO noch nicht in beiden Richtungen beendet ist.

3.9.2. Betriebsstellenbezeichnung

Die Sende- und Empfangsbetriebsstelleninformation werden erst dann im XSU gesetzt, wenn das erste Lebenstakttelegramm von der EBO empfangen wird. Ab diesem Zeitpunkt werden für alle vom XSU generierten Telegramm auf allen Leitungen nur noch diese Betriebsstelleninformationen verwendet.

3.9.3. XSU Rücken an Rücken

Um zwei XSU Rücken an Rücken stellen zu können, darf der XSU die Betriebsstelleninformation von der X25 Leitung nicht speichern. Es muss die Betriebsstelleninformation von der EBO verwendet werden. Um während der Zeit, bis das erste EBO Telegramm eintrifft keine Konflikte auf der X.25 Leitung zu haben (ständiger Abbruch der Gegenstelle wegen falscher Betriebsstelle), sendet der XSU auf der X.25 Seite erst dann Lebenstakte aus, nachdem die Betriebsstellendaten aus dem ersten Lebenstakt von der EBO gespeichert wurden.

Dieses Feature ist im XSU bereits implementiert. Das bedeutet, der XSU sendet nach dem ersten Einschalten erst dann Lebenstakte auf der X25 Leitung aus, wenn er einen Lebenstakt auf der V24 Leitung erhalten hat.

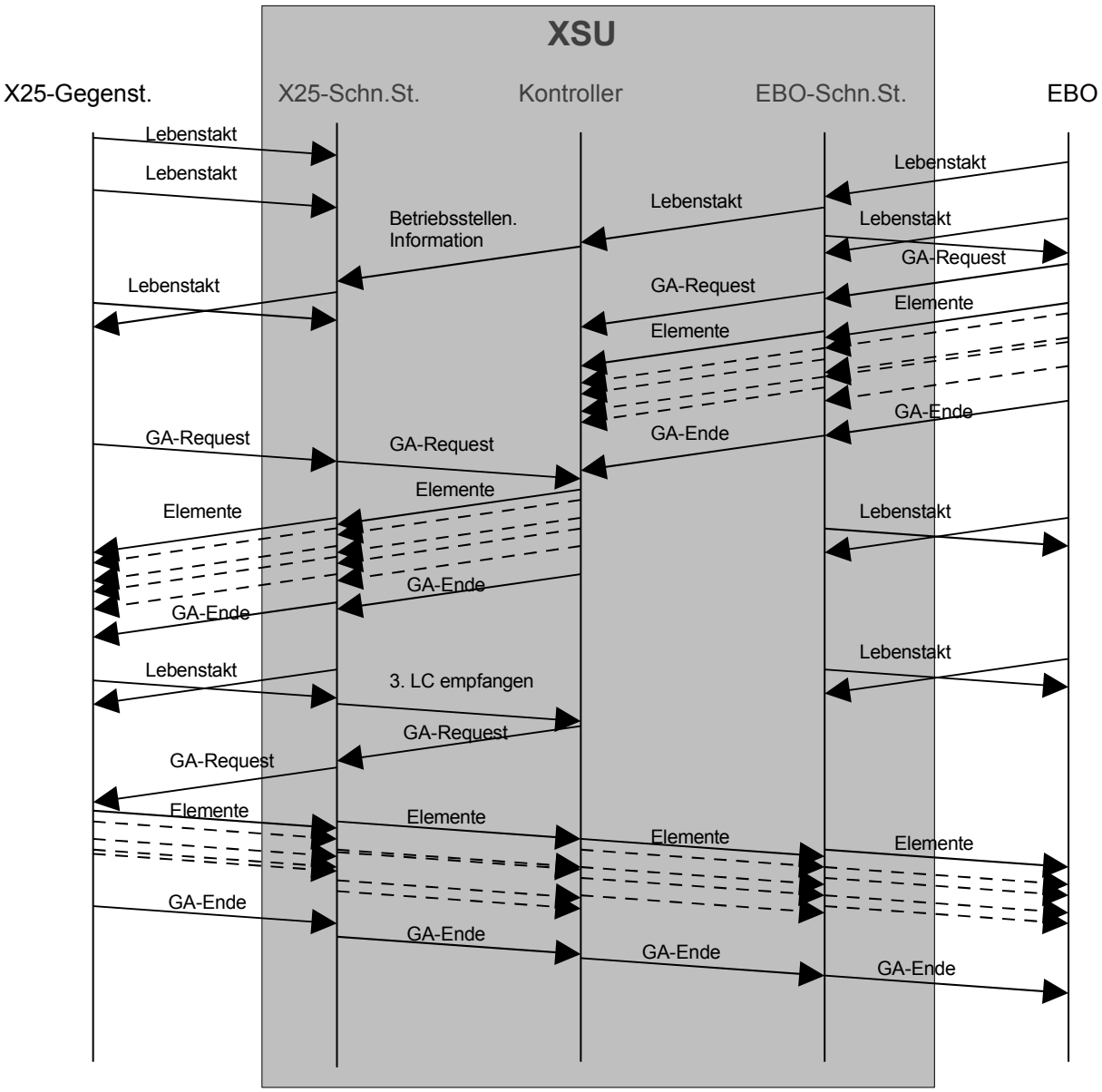
Dieses Feature wurde bei der Parallelausleuchtung wieder ausgebaut, weil dort die Forderung des Kunden besteht, daß der XSU auch dann eine Verbindung zu ESTW aufbaut, wenn die Windows-Applikation an der EBO-Schnittstelle gar nicht mehr vorhanden ist. Damit ist seit 2009 der Rücken-an-Rücken Betrieb nicht mehr möglich. Sollten er erforderlich sein, bitte kontaktieren Sie uns.

4. Kommunikations Szenarien

In diesem Kapitel werden Telegrammszenarien dokumentiert.

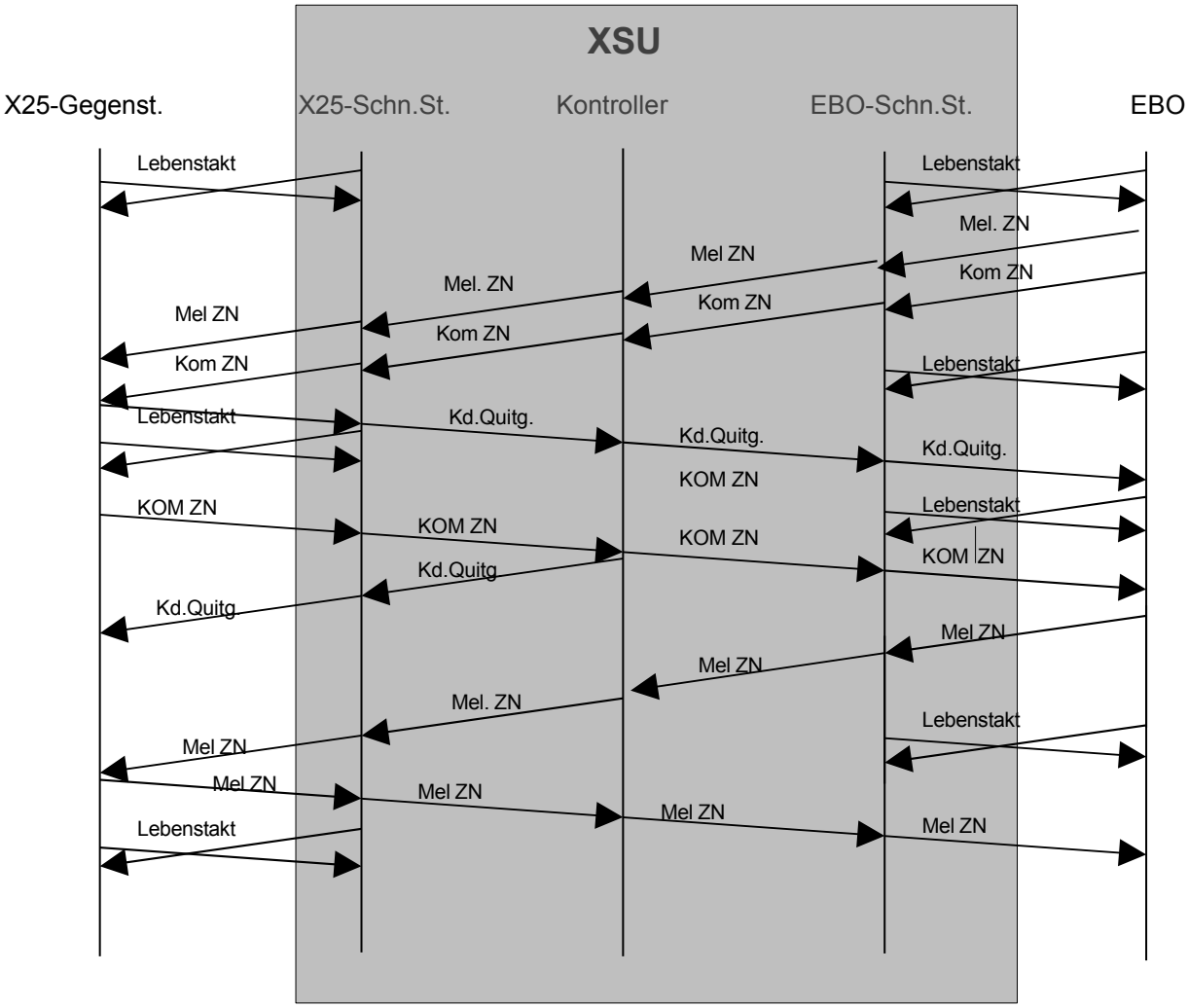
4.1. Verbindungsaufbau

Das Kapitel Verbindungsaufbau zeigt, daß der XSU erst mit den Lebenstakten zur X25-Gegenstelle beginnt, wenn mindestens ein Lebenstakt von der EBO empfangen wurde. Erst mit diesem ersten Lebenstakt kennt der XSU die Betriebsstelleninformationen mit denen er mit der X25-Gegenstelle kommunizieren soll. Weiters wird gezeigt, wie der XSU die Elementdaten von der EBO speichert und im Falle einer GA mit der X25-Gegenstelle aus der Datenbank im Kontroller an die X25 Gegenstelle sendet. Der letzte Block ist eine empfangene GA von der X25 Gegenstelle die an die EBO weitergeleitet wird.



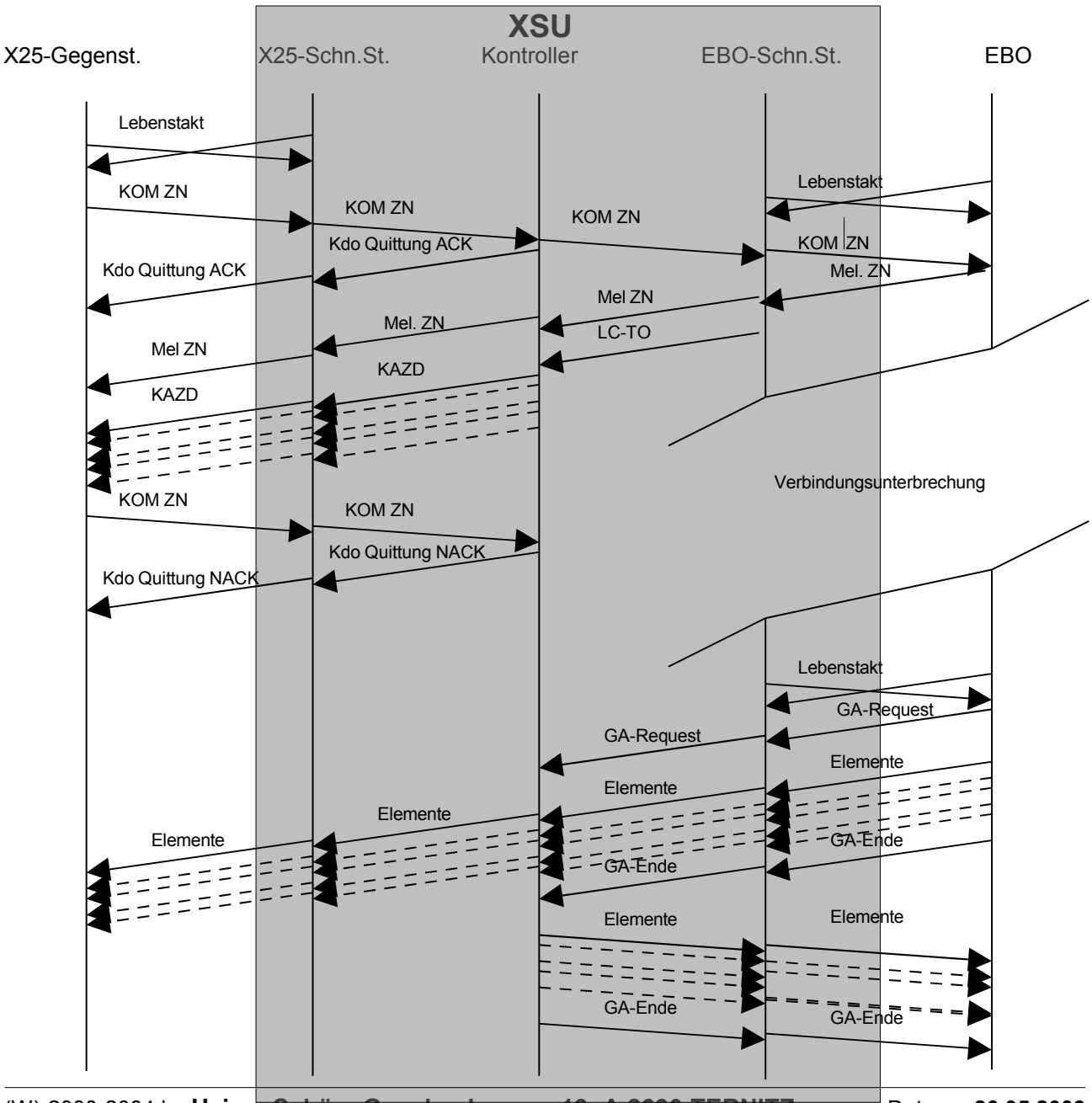
4.2. Zugnummerntelegramme

Das nachfolgende Szenario zeigt wie Zugnummernmeldungen und Kommandos vom X25-Teilnehmer EBO an die X25-Gegenstelle geleitet weden und Kommandoquittungen von der X25-Gegenstelle wieder zurückfließen. Weiters wird gezeigt wie Kommando Zugnummer von der X25-Gegenstelle vom XSU selbtständig mit positiver Kommandoquittung beantwortet wird und nur noch das Kommandotelegramm selbst an die EBO geleitet wird.



4.3. Verbindungsunterbrechung zur EBO

Das nachfolgende Szenario zeigt eine Verbindungsunterbrechung zwischen der EBO und dem XSU. Mit dem Erkennen der Unterbrechung (LC-TO = Lifecycletimeout im XSU) wird vom XSU selbständig jedes Element mit Bit KAZD (Keine aktuellen Zustandsdaten) an die X25-Gegenstelle gemeldet (automatische KAZD-Ausleuchtung durch den XSU). Eintreffende Kommandos werden mit NACK (no acknowledge) abgelehnt.
Nach dem Wiederaufbau der Verbindung (Eintreffen eines GA-Request von der EBO), werden die eintreffenden Elementmeldungen an die X25-Gegenstelle geleitet und die EBO wird aus der Elementdatenbank des XSU mit Elementen versorgt (ohne die X25 Gegenstelle neu befragen zu müssen). Eintreffende Elementmeldungen von der X25 Gegenstelle während der Verbindungsunterbrechung zu EBO werden im XSU gespeichert (nicht dargestellt – gilt implizit).



4.4. Telegrammaufbau

Der Telegrammaufbau auf der X25 Verbindung XSU zur X.25 Gegenstelle entspricht vollinhaltlich dem Dokument [1] Pflichtenheft X25 – Schnittstelle. Die Schnittstelle zur EBO ist identisch, jedoch werden die Felder TFN und CRC nicht verwendet. Es werden von der EBO nur die Felder:

- Länge
- Hauptgruppe
- Sendebetriebstelle
- Empfangsbetriebstelle
- Datenbytes

Im Telegramm versorgt. Das Telegrammfolgenummern Byte und die zwei Byte CRC werden zwar im Telegramm bereits mitgesendet, werden aber beim Empfang im XSU nicht ausgewertet. Die so von der EBO empfangenen Telegramme werden zwischengespeichert (Elementdatenbank und/oder Sendebuffer) und erst beim Senden über die X25 Leitung werden diese Bytes ausgefüllt.

Damit ergibt sich folgender Telegrammaufbau für Telegramme die von der EBO an den XSU gesendet werden:

1 Byte	TFN	(wird vom XSU nicht ausgewertet – kann von EBO leer bleiben)
1 Byte	Länge	
1 Byte	Hauptgruppe	
4 Byte	Sende BST	
4 Byte	Empfangs BST	
n Byte	Nutzdaten	
2 Byte	CRC	(wird vom XSU nicht ausgewertet – kann von EBO leer bleiben)

Die Telegramme vom X25-Teilnehmer EBO an den XSU entsprechen somit dem [1] Pflichtenheft X25 – Schnittstelle, die EBO ist aber von der Berechnung und Verwaltung der TFN (Telegrammfolgenummer) und der CRC (Checksumme entsprechend dem X25-Generatorpolynom) befreit. Diese drei Byte (1 Byte TFN und 2 Byte CRC) werden vom XSU nicht ausgewertet und können daher jeden beliebigen Inhalt haben. Sie sind aber einzufügen, da der XSU beim Zerlegen der Telegramme diese benötigt um den nächsten Telegrammbeginn zu finden.

Die EBO sendet somit nur die Länge (diese ist für alle Hauptgruppen im [1] Pflichtenheft X25 – Schnittstelle festgelegt), die Hauptgruppe entsprechend [1] Pflichtenheft X25 – Schnittstelle die Sende- und Empfangs Betriebsstelle (je 4 Byte entsprechend [1] Pflichtenheft X25 – Schnittstelle) und die Nutzdaten (wie im [1] Pflichtenheft X25 – Schnittstelle festgelegt).

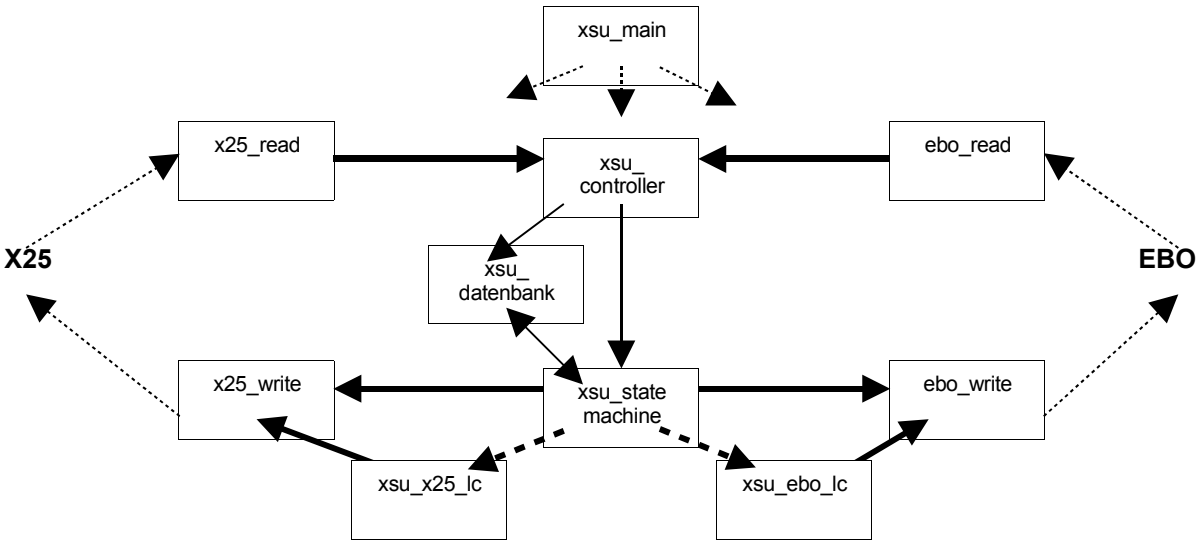
5. Softwarebeschreibung

In diesem Kapitel werden die Funktionen der einzelnen Applikationsmodule dokumentiert.

5.1. Allgemeines

Die interne Kommunikation im XSU erfolgt ausschließlich über pipes. Die Software wird bewußt nicht monolithisch ausgeführt, sondern jeder Prozeß besitzt nur einen read. Ein read eines Prozesses kann entweder von einem device (/dev/ttyS0 oder /dev/ttyS1), oder aus einer pipe erfolgen (es wird bewusst auf select und mehrfache reads verzichtet). Jeder Prozeß führt an den empfangenen Daten die spezifizierten Arbeiten durch und entscheidet ob und an welchen Prozeß die Daten weiterzusenden sind. Aus Gründen der Performance und vor allem um vor memcpy Fehlern bei der Implementierung zu schützen, wird der Empfangsbuffer beim read nach Möglichkeit wieder zum Senden verwendet. Diese und einige andere Implemtentierungsregeln dienen der Erfhöhung der Verfügbarkeit der Software um diverse Fehler die gerne in der Programmiersprache C gemacht werden, schon bei der Implementierung ausschließen zu können.

5.1.1. Prozesse und Kommunikationswege in der XSU Applikation



Pipe

Pipe die nur einmal benutzt wird (startlc)

Prozess Call

System Call

5.2. Main

5.2.1. Starten/Überwachen der Applikationsprozesse

Der Prozeß xsu.c startet mit fork() die Subprozesse:

xsu_parser.c
x25_read.c
x25_write.c
xsu_x25_lc.c
ebo_read.c
ebo_write.c
xsu_ebo_lc.c
xsu_out_router.c
xsu_datenbank.c
xsu_controller.c

Wird ein Prozeß beendet (egal ob der Prozeß sich selbst beendet oder der Prozeß zum Verlassen gezwungen wird (Signal)), stellt xsu_main.c sicher, daß alle anderen Prozesse auch beendet werden. Der Prozess xsu.c beendet sich selbst erst nachdem alle anderen Prozesse niedergefahren wurden.

Vor dem Starten der Prozesse generiert xsu_main.c folgende pipes:

pipe_to_main
pipe_to_controller
pipe_to_x25_lc
pipe_to_x25_write
pipe_to_ebo_lc
pipe_to_ebo_write
pipe_to_out_contr

und öffnet die seriellen Schnittstellen, die vom Startscript als Dateinamen übergeben wurden.

5.3. Leitungsstatemachine

5.3.1. Allgemeines

Die Leitungsstatemachine empfängt vom Controller Telegramme über eine pipe. Die Telegramme werden in einer einheitlichen Telegrammstruktur zwischen den Prozessen ausgetauscht und in einem ersten Schritt auf Zulässigkeit geprüft und gegebenenfalls weitergeleitet. Die Statemachine entscheidet auch über die Notwendigkeit und/oder Zulässigkeit einer GA und steuert somit den Datenfluß im XSU und damit auch den Datenfluß an den externen Schnittstellen. Die Statemachine verwaltet den Zustand aller externen Leitungen für beide Richtungen (senden und empfangen).

Die Statemachine ist implementiert im Modul xsu_statemachine.c und wird über zwei Call-Schnittstellen aufgerufen, je nachdem ob die Statemaschine für die X25-Gegenstelle oder für die EBO benutzt werden soll:

```
p_x25_statemachine(vf_data);
p_ebo_statemachine(vf_data);
```

Der Aufruf erfolgt vom Modul xsu_out_contr.c jeweils beim Empfang eines Telegramms von der X25 oder der EBO Schnittstelle. Die Statemachine entscheidet dann anhand des empfangenen Telegramms welche Stateübergänge durchzuführen sind, welche weitere Aktionen durchzuführen sind, was mit den Daten zu geschehen hat (speichern und/oder weiterleiten).

5.3.2. Zustände der EBO-Schnittstelle

v_ebo_ga_rece_running	[true/false]
v_ebo_ga_rece_done	[true/false]
v_ebo_ga_send_running	[true/false]
v_ebo_ga_send_done	[true/false]
v_x25_ga_rece_running	[true/false]
v_x25_ga_rece_done	[true/false]
v_x25_ga_send_running	[true/false]
v_x25_ga_send_done	[true/false]
v_x25_win_done	[true/false]

Diese Variablen werden abhängig vom Zustand der beiden Leitungen gesetzt oder gelöscht und bilden somit den Zustand aller externen Leitungen ab.

5.3.3. Prozeduren

Folgende Routinen sind in der Statmachine static implementiert:

5.3.3.1. P_make_disconnect

Erzeugen eines Disconnect Frames abhängig von der Richtung in die der Frame gesendet wird, wird jeweils die Betriebsstelleninformation eingetragen und HGR=0 Länge 10 in den Telegrammbuffer eingetragen

5.3.3.2. P_store_bst

Speichern der Betriebsstelleninformationen die beim ersten Lebenstakt von der EBO empfangen wurde. Alle weiteren Telegramme werden mit diesen Betriebsstelleninformationen erzeugt.

5.3.3.3. P_make_x25_lc
Erzeugen der Telegrammdaten von X25 Lebenstakttelegrammen.

5.3.3.4. P_make_x25_fernbedienung
Erzeugen der Telegrammdaten von X25 Meldung Fernbedienung Telegrammen

5.3.3.5. P_make_x25_ga_request
Erzeugen der Telegrammdaten von X25 Kommando GA-Request Telegrammen

5.3.3.6. P_make_x25_ga_quittung
Erzeugen der Telegrammdaten von X25 Meldung Quittung Generalabfrage

5.3.3.7. P_make_x25_kommandoquittung
Erzeugen der Telegrammdaten von X25 Kommandoquittungen

5.3.3.8. P_make_ebo_lc
Erzeugen der Telegrammdaten von EBO Lebenstakttelegrammen.

5.3.3.9. P_make_ebo_ga_request
Erzeugen der Telegrammdaten von EBO Kommando GA-Request Telegrammen

5.3.3.10. P_make_ebo_ga_quittung
Erzeugen der Telegrammdaten von EBO Meldung Quittung Generalabfrage

5.3.3.11. P_make_ebo_kommandoquittung
Erzeugen der Telegrammdaten von EBO Kommandoquittungen

5.3.3.12. P_make_x25_ga_request
Erzeugen der Telegrammdaten von X25 Kommando GA-Request Telegrammen

5.3.3.13. P_make_x25_ga_quittung
Erzeugen der Telegrammdaten von X25 Meldung Quittung Generalabfrage

5.3.3.14. P_set_ebo_betriebsstelle
Erzeugen der Telegrammdaten mit EBO-Betriebsstelleninformationen

5.3.3.15. P_set_x25_betriebsstelle
Erzeugen der Telegrammdaten mit X25-Betriebsstelleninformationen

5.3.3.16. P_disconnect
Erzeugen der Telegrammdaten in Disconnect Telegrammen. Selbständiges aussenden des Disconnect Telegramms wenn die Verbindung noch aufrecht war. Rücksetzen aller Zustandsvariablen der betroffenen Leitung. Rücksetzen der Lebenstaktzähler, sodaß die statemachine wieder mit einer GA beginnt. Die Routine bekommt die zu stoppende Leitung als Parameter übergeben.

5.3.3.17. P_x25_send_ga_block
Um auf der X25 Leitung während der GA keinen Lebenstakttimeout zu provozieren, wird die GA blockweise gesendet. Diese Routine liest jeweils 10 Telegrame aus der Datenbank und sendet diese an die X25 Leitung. Danach beendet sich diese Funktion wieder. Wird beim Auslesen eines Blocks kein ungesendetes Telegramm mehr in der Datenbank gefunden, dann werden die GA_done-Variablen auf TRUE und GA_running auf FALSE gesetzt.

5.3.3.18. P_ebo_send_ga_block

Dto. Wie p_x25_send_ga_block für blockweises Senden an die EBO.

5.3.3.19. P_x25_statemachine

Die X25 Statemachine ist ein CASE über die empfangenen Hauptgruppen. Die nachfolgenden Kapitel beschreiben die Reaktionen auf die einzelnen Telegramme:

5.3.3.19.1. C_mel_disconnect

P_disconnect wird aufgerufen.

5.3.3.19.2. C_mel_quittung_ga

Wird ACK empfangen werden die Statevariablen so gesetzt daß die GA in Empfangsrichtung laufend (running) markiert ist. Wird ENDE empfangen, dann werden die Statevariablen auf eine beendete Empfangs-GA gesetzt. Wird NAK empfangen, wird gespeichert das keine GA läuft und es wird p_disconnect ausgesendet. Bei GA-Halt und Datenverlust wird nur eine Diagnose ausgegeben, aber keine weitere Aktion durchgeführt.

5.3.3.19.3. C_mel_kommando_quittung

Kommandoquittungen werden nur diagnostiziert und nicht an die EBO weitergeleitet. Sie werden also verworfen.

5.3.3.19.4. Meldungen

Alle Hauptgruppen die Elemente melden (Weichen, Gleise, Signale, ...) werden weitergeleitet an die jeweils andere Schnittstelle.

5.3.3.19.5. C_mel_lebenstakt

Eintreffende Lebenstakte werden gezählt. Wird X25_SEND_GA_LEVEL überschritten, dann wird ein GA-Request ausgesendet. Kommt keine GA zustande, dann wird bei erreichen von X25_SEND_GA_RETRY_LEVEL ein weiterer GA-Request ausgesendet.

Läuft bereits eine GA, dann wird p_x25_send_ga_block aufgerufen um den nächsten Block mit X25 Telegrammen zu senden.

5.3.3.19.6. Kommandos

Alle Hauptgruppen die Kommandos zu Elementen sind (Weichen, Gleise, Signale, ...) werden weitergeleitet wenn auf der EBO Leitung bereits die GA in beiden Richtungen fertig ist. Gleichzeitig wird eine Kommandoquittung ACK an die X25 Gegenstelle gesendet.
Ist die EBO GA noch nicht fertig, dann wird eine Kommandoquittung NAK an die X25 Gegenstelle gesendet und das Kommando wird verworfen.

5.3.3.19.7. C_kom_fernbedienung

Das Kommando Fernbedienung wird mit Kommandoquittung beantwortet, wenn das Bit 3 im Steuerwort 1 gesetzt ist. Alle anderen Bits werden mit NACK abgewiesen.

5.3.3.19.8. C_kom_datum_und_uhrzeit

Dieses Kommando wird wie ein Element Kommando behandelt (Beantwortung mit ACK und Weiterleitung an EBO)

5.3.3.19.9. C_kom_ga

Das Kommando GA wird mit Kommandoquittung NACK und GA-Quittung NACK abgewiesen bis die GA mit der EBO in beiden Richtungen abgeschlossen ist.

Ist die GA mit der EBO schon abgeschlossen, dann wird GA-ACK und Kommandoquittung ACK ausgesendet und in der Datenbank werden alle gespeicherten Elemente als ungesendet markiert. Danach wird die Routine p_x25_send_ga_block aufgerufen und die GA wird als laufend markiert.

5.3.3.19.10. ELSE

Alle anderen Hauptgruppen werden diagnostiziert und dann verworfen.

5.3.3.20. P_ebo_statemachine

Die EBO Statemachine ist ein CASE über die empfangenen Hauptgruppen. Die nachfolgenden Kapitel beschreiben die Reaktionen auf die einzelnen Telegramme:

5.3.3.20.1. C_mel_disconnect

P_disconnect wird aufgerufen.

5.3.3.20.2. C_mel_quittung_ga

Wird ACK empfangen werden die Statevariablen so gesetzt daß die GA in Empfangsrichtung laufend (running) markiert ist. Wird ENDE empfangen, dann werden die Statevariablen auf eine beendete Empfangs-GA gesetzt. Wird NAK empfangen, wird gespeichert das keine GA läuft und es wird p_disconnect ausgesendet. Bei GA-Halt und Datenverlust wird nur eine Diagnose ausgegeben, aber keine weitere Aktion durchgeführt.

5.3.3.20.3. C_mel_kommando_quittung

Kommandoquittungen werden nur diagnostiziert und nicht an die X25 Schnittstelle weitergeleitet. Sie werden also verworfen.

5.3.3.20.4. Meldungen

Alle Hauptgruppen die Elemente melden (Weichen, Gleise, Signale, ...) werden mit umgekehrter Betriebsstelle weitergeleitet.

5.3.3.20.5. C_mel_lebenstakt

Der erste eintreffende Lebenstakt von der EBO führt zum Aufruf der Routine p_store_bst. Danach werden die beiden Lebenstaktgeneratoren gestartet. Hierzu werden mit p_make_x25_lc und p_make_ebo_lc die Lebenstakttelegrammbuffer aufgebaut und mit p_startlc die Prozesse aktiviert. (Anm.: Die Lebenstaktgeneratoren werden von xsu.c geforked und von p_startlc via pipe aktiviert).

Eintreffende Lebenstakte werden gezählt. Wird EBO_SEND_GA_LEVEL überschritten, dann wird ein GA-Request ausgesendet. Kommt keine GA zustande, dann wird bei erreichen von EBO_SEND_GA_RETRY_LEVEL ein weiterer GA-Request ausgesendet.

Läuft bereits eine GA, dann wird p_x25_send_ga_block aufgerufen um den nächsten Block mit X25 Telegrammen zu senden.

5.3.3.20.6. Kommandos

Alle Hauptgruppen die Kommandos zu Elementen sind (Weichen, Gleise, Signale, ...) werden weitergeleitet wenn auf der EBO Leitung bereits die GA in beiden Richtungen fertig ist. Gleichzeitig wird eine Kommandoquittung ACK an die EBO gesendet.
Ist die EBO GA noch nicht fertig, dann wird eine Kommandoquittung NAK an die EBO Gegenstelle gesendet und das Kommando wird verworfen.

5.3.3.20.7. C_kom_fernbedienung

Das Kommando Fernbedienung wird mit Kommandoquittung NACK beantwortet.

5.3.3.20.8. C_kom_datum_und_uhrzeit

Beantwortung mit ACK und Weiterleitung an X25.

5.3.3.20.9. C_kom_ga

Das Kommando GA wird mit GA-ACK und Kommandoquittung ACK beantwortet und in der Datenbank werden alle gespeicherten Elemente als ungesendet markiert. Danach wird die Routine p_ebo_send_ga_block aufgerufen und die GA wird als laufend markiert.

5.3.3.20.10. ELSE

Alle anderen Hauptgruppen werden diagnostiziert und dann verworfen.

5.4. Elementdatenbank

5.4.1. Allgemeines

Die Elementdatenbank ist für das Richtungsabhängige Speichern von X25 Telegrammen verantwortlich. Die Elementdatenbank besitzt auch die Logik welche Hauptgruppen zu speichern sind und vor allem hat die Elementdatenbankein Gefühl dafür wo in einem Telegramm die Elementnummer gespeichert ist.

Die Elementdatenbank speichert Daten abhängig von Richtung (ebo oder x25) und geschlüsselt nach Hauptgruppe und Elementnummer. Weiters speichert die Elementdatenbank ob ein Telegramm schon in einer GA gesendet wurde oder nicht.

Zusätzlich muß die Elementdatenbank wissen, wo in einem Telegramm die Datenbytes gespeichert sind, damit die Datenbank entscheiden kann ob sich ein Telegramm geändert hat oder nicht.

Telegramme die der Elementdatenbank übergeben werden müssen zuerst nach Richtung, Hauptgruppe und Elementnummer gesucht werden, um zu entscheiden ob ein neuer Eintrag in der Datenbank angelegt werden soll oder nicht.

Die Datenbank hat keine Logik hinsichtlich der gespeicherten Daten. Zum Beispiel gleiche Hauptgruppe mit gleicher Elementnummer in beiden Richtungen speichert die Datenbank ohne Warnung.

5.4.2. Prozeduren

5.4.2.1. P_db_create_shm

Diese Routine erzeugt die Datenbank. P_db_create_shm wird von xsu.c beim Start der Applikation aufgerufen

5.4.2.2. P_db_destroy_shm

Diese Routine wird beim Stoppen der Applikation von xsu.c aufgerufen um das shared-memory wieder freizugeben.

5.4.2.3. P_db_next_free

Diese Routine sucht in der Datenbank den nächsten freien Eintrag um neue Telegramme zu speichern. Die globale Variable db_next_free enthält den ersten freien Datenbankeintrag oder c_DB_SIZE wenn die Datenbank voll ist.

5.4.2.4. P_db_store_frame

Das übergebene Telegramm wird an der übergebenen Position gespeichert

5.4.2.5. P_db_known_hgr

Diese Funktion gibt TRUE zurück wenn eine Hauptgruppe einer Hauptgruppe ist von der die Datenbank weiß wie sie zu speichern ist.

5.4.2.6. P_db_elem

Diese Routine gibt die Elementnummer eines Telegramms zurück (Elementnummern stehen bei unterschiedlichen Hauptgruppen an unterschiedlichen Stellen).

5.4.2.7. P_db_compare

Diese Routine vergleicht ein übergebenes Telegramm mit dem gespeicherten Telegramm dessen Index übergeben wurde.

5.4.2.8. P_db_init

Diese Routine initialisiert die gesamte Datenbank in der angegebenen Richtung. Diese Routine wird beim Einrichten der Datenbank beim Applikationsstart benötigt.

5.4.2.9. P_db_insert

Diese Routine speichert ein übergebenes Telegramm in der übergebenen Richtung in der Datenbank. Die Routine sucht ob diese Hauptgruppen/Elementnummernkombination in dieser Richtung schon gespeichert ist. Wenn kein passender Eintrag gefunden wird, dann wird ein neuer Record benutzt (p_db_next_free). Die Routine gibt true oder false zurück, jenachdem ob sich Daten geändert haben.

5.4.2.10. P_db_delete

Löscht ein Telegramm (Hauptgruppe/Elementnummernkombination) in der übergebenen Richtung aus der Datenbank.

5.4.2.11. P_db_get_next_new

Diese Routine gibt den nächsten noch nicht als gesendet markierten Frame zurück.

5.4.2.12. P_db_set_all_new

Diese Routine setzt alle in der angegebenen Richtung gespeicherten Frames als noch nicht in der GA gesendete Frames.

5.5. Pipecontroller xsu_out_contr.c

5.5.1. Allgemeines

Der Pipecontroller ist für den Empfang der Telegramme von p_x25_read und p_ebo_read zuständig. Der Pipecontroller diagnostiziert die empfangenen Telegramme und reicht ALLE EMPFANGENE TELEGRAMME weiter an die Statemachinen p_ebo_statemachine oder p_x25_statemachine (je nachdem von welcher Leitung die Telegramme empfangen wurden). Aufgrund der Hauptgruppe (Lebenstakt, GA-Request, GA-Quittungen, usw.) entscheidet die Statemaschine über weitere durchzuführende Aktionen.

5.5.2. Meldungsverarbeitung (HGR 1 bis 127)

Alle übergebenen Meldungen werden an p_db_insert übergeben. P_db_insert speichert dann die übergebenen Telegramme in der Elementdatenbank (ausgenommen Kommando und Meldung Zugnummer – siehe nachfolgende Kapitel).

5.5.3. Kommandoverarbeitung (HGR 128 bis 255)

Es erfolgt kein Speichern von Kommandos in der Elementdatenbank!

5.5.4. Sonderbehandlung von Zugnummernmeldungen (HGR 53)

Meldungen der Hauptgruppe 53, das sind Zugnummern, werden zusätzlich einer Sonderbehandlung unterzogen, die wie folgt abhängig vom Steuerwort 1 abgearbeitet wird:

Meldung 53 Steuerwort 1

```
Bit 1:    Taufe, Einwahl
                  Start-Element:     wird ignoriert
                  Ziel-Element:      ZN wird mit dieser El.Nr. gespeichert
Bit 2:    Einbruch
                  Start-Element:     wird ignoriert
                  Ziel-Element:      ZN wird mit dieser El.Nr. gespeichert
Bit 3:    Löschen
                  Start-Element:     ZN mit dieser El.Nr. wird gelöscht
                  Ziel-Element:      wird ignoriert
Bit 4:    Zugausbruch
                  Start-Element:     ZN mit dieser El.Nr. wird gelöscht
                  Ziel-Element:      wird ignoriert
Bit 5:    Handfortschaltung
                  Start-Element:     ZN mit dieser El.Nr. wird gelöscht
                  Ziel-Element:      ZN wird mit dieser El.Nr. gespeichert
Bit 6:    Automatikfortschaltung
                  Start-Element:     ZN mit dieser El.Nr. wird gelöscht
                  Ziel-Element:      ZN wird mit dieser El.Nr. gespeichert
Bit 7:    Kennung
                  Start-Element:     wird ignoriert
                  Ziel-Element:      wird ignoriert
```

```
Bit 8:    Zustandsänderung
                  Start-Element:     wird ignoriert
                  Ziel-Element:      wird ignoriert
```

Meldung 53 Steuerwort 2 und 3

```
  Die Bits in den Steuerworten 2 und 3 werden beim Speichern von Zugnummern
  gelöscht. Die Bits in den Steuerworten 2 und 3 werden beim Empfang nicht
  ausgewertet und lösen somit keine Sondernbehandlungen aus.
```

Bei der Speicherung von Zugnummern werden alle drei Steuerworte gelöscht und das Bit 2 im Steuerwort 1 wird gesetzt. In einer nachfolgenden GA (also bei späteren lesenden Zugriffen auf die Elementdatenbank) werden daher alle Zugnummern mit Bit Einbruch gesendet (das heißt alle drei Steuerworte leer, bis auf Bit 2 in Steuerwort 1 auf HIGH).

5.5.5. Zusammenfassung der Sonderbehandlung von Zugnummernmeldungen

Ist in der Zugnummernmeldung im Steuerwort 1 das Bit 1, 2, 5 oder 6 gesetzt, dann wird das Telegramm in der Meldungsdatenbank mit Bit 2 Einbruch gespeichert. Alle anderen Bits in den Steuerworten 1, 2 und 3 werden gelöscht.

Ist in der Zugnummernmeldung im Steuerwort 1 das Bit 3 oder 4 gesetzt, dann wird das Zugnummerntelegramm, das unter der gerade erhaltenen Zielelementnummer gespeichert ist, aus der Elementdatenbank gelöscht.

Ist in der Zugnummernmeldung im Steuerwort 1 das Bit 5 oder 6 gesetzt, dann wird das Zugnummerntelegramm, das mit der im gerade erhaltenen Startelementnummer gespeichert ist, aus der Elementdatenbank gelöscht.

5.5.6. Kommando Zugnummer (HGR 174)

Zugnummernkommandos werden in beiden Richtungen (von X25 an EBO und von EBO an X25) jeweils an die andere Schnittstelle weitergeleitet. Es erfolgt weder eine Sonderbehandlung noch eine Speicherung der Telegramme. Behandlung wie alle anderen Kommandos (z.B. Weichen, Gleise,usw.).

5.6. X25 Kommunikation

5.6.1. Allgemeines

Die X25 Kommunikation besteht aus drei Prozessen. Der x25_read.c ist für das Empfangen von Telegrammen von der X25 Schnittstelle und die Überwachung des regelmäßigen Eintreffens von Lebenstakten verantwortlich. Der Prozeß x25_write.c sendet Telegramme an die X25 Schnittstelle aus und der Prozess xsu_x25_lc.c generiert regelmäßig X25 Lebenstakttelegramme.

5.6.2. X25_READ

Der Prozeß x25_read.c empfängt Telegramme von der Gegenstelle. X25_read.c überprüft die Telegrammfolgenummer und die CRC eintreffender Telegramme. Sind TFN und CRC in Ordnung werden die Telegramme einer weiteren Verarbeitung unterzogen. Wenn TFN oder CRC falsch sind, dann wird der SET-Wert LCTO an den xsu_controller gesendet.

Ist ein empfangenes Telegramm ein Lebenstakt, dann wird ein 6 Sekunden Timeout gestartet. Treffen Lebenstakte in Abständen kleiner 6 Sekunden ein, wird der Timeout nie ablaufen. Wenn der Timeout abläuft, wird LCTO an den controller gesendet.

Immer wenn LCTO an den Controller gesendet wird, darf die TFN des nächsten emfpfangenen Telegramms einen beliebigen Wert haben. Wurde ein Telegramm empfangen, dann wird diese Möglichkeit wieder abgeschaltet und TFNen müssen kontinuierlich um 1 inkrementiert empfangen werden.

5.6.3. X25_WRITE

Der Prozeß x25_write.c verwirft alle empfangenen Meldungen und Kommandos bis zum Eintreffen eines Telegramms mit der Kennung FIRSTLC vom xsu_controller.c. Wird FIRSTLC empfangen, dann wird ein 2 Sekunden Timeout gestartet. Mit jedem Ablauf des 2 Sekunden Timeouts wird ein Lebenstakt ausgesendet und der Timeout erneut gestartet. Die mit dem Telegramm FIRSTLC empfangene Sender-Empfänger Betriebsstellenkennung wird für die Generierung der Lebenstakttelegramme herangezogen.

X25_write.c empfängt Meldungen und Kommandos vom x25_controller.c. Von xsu_datenbank.c werden Elementmeldungen empfangen.

Wird ein Telegramm mit Hauptgruppe 0 empfangen wird der 2 Sekunden Timeout neu gestartet mit einer 6 Sekunden Wartezeit und das Telegramm mit Hauptgruppe 0 wird ausgesendet. In diesen 6 Sekunden werden alle Telegramme die via pipe eintreffen, verworfen. Nach Ablauf der 6 Sekunden wird wieder der Timeout mit 2 Sekunden gestartet und es werden wieder Lebenstakte ausgesendet.

5.7. EBO Kommunikation

5.7.1. Allgemeines

Die EBO Kommunikation besteht aus drei Prozessen. Der ebo_read.c ist für das Empfangen von Telegrammen von der EBO Schnittstelle und die Überwachung des regelmäßigen Eintreffens von Lebenstakten verantwortlich. Der Prozeß ebo_write.c sendet Telegramme an die EBO Schnittstelle aus und der Prozeß xsu_ebo_lc.c generiert regelmäßig (alle 10 Sekunden) Lebenstakttelegramme.

5.7.2. EBO_READ

Der Prozeß ebo_read.c empfängt Telegramme vom X25-Teilnehmer an der EBO-Schnittstelle. ebo_read.c überprüft weder die Telegrammfolgenummer noch die CRC eintreffender Telegramme. Das TFN-Byte und die 2-byte CRC werden zwar im Telegramm erwartet aber von ebo-read nicht ausgewertet.

Ist ein empfangenes Telegramm ein Lebenstakt, dann wird ein **30 Sekunden Timeout** gestartet. Treffen Lebenstakte in Abständen kleiner 30 Sekunden ein, wird der Timeout nicht ablaufen. Wenn der Timeout abläuft, wird LCTO an den controller gesendet.

5.7.3. EBO_WRITE

Der Prozeß ebo_write.c verwirft alle empfangenen Meldungen und Kommandos bis zum Eintreffen eines Telegramms mit der Kennung FIRSTLC vom xsu_controller.c. Wird FIRSTLC empfangen, dann wird ein 10 Sekunden Timeout gestartet. Mit jedem Ablauf des 10 Sekunden Timeouts wird ein Lebenstakt ausgesendet und der Timeout erneut gestartet. Die mit dem Telegramm FIRSTLC empfangene Sender-Empfänger Betriebsstellenkennung wird für die Generierung der Lebenstakttelegramme herangezogen.

ebo_write.c empfängt Meldungen und Kommandos vom xsu_controller.c. Von xsu_datenbank.c werden Elementmeldungen empfangen.

Wird ein Telegramm mit Hauptgruppe 0 empfangen, dann wird dieses Telegramm ohne weitere Bearbeitung ausgesendet. Telegramme mit der Hauptgruppe 0 könen vom X25-Teilnehmer an der EBO-Schnittstelle benutzt werden, um den Ausfall der Schnittstelle zu erkennen. Mit dem Eintreffen einer Meldung GA-Ende kann der Teilnehmer an der EBO-Schnittstelle erkennen, daß die Verbindung wieder voll funktionsfähig ist. Beide Telegramme können aber auch ignoriert werden. Es hängt stark von der Funktion des Teilnehmers ab. Ist der Teilnehmer verpflichtet den Zustand der X25-Schnittstelle anzuzeigen, dann kann er die Disconnect und GA-Anfang und GA-Ende Telegramme auswerten, andernfalls kann der Teilnehmer diese Telegramme auch ignorieren.

6. Hardwarebeschreibung

Dieses Kapitel beschreibt die XSU Hardware. Hierzu zählen die Beschreibung des XSU-Rechners und der XSU Kabel.

6.1. XSU-Rechner

Die XSU-Hardware basiert auf der EXD-Plattform. Die EXD-Plattform benutzt als zentralen Kern den Rechner MSM586 des schweizer Herstellers Digitallogic. Der Rechnerkern ist ein PC/104 Board mit einem AMD ELAN SC520 Chip, der auf einem Intel 486 beruht und mit 133 Mhz getaktet wird. Der Speicher ist mit 128 MB bestückt. Das Betriebssystem ist auf der 512 MB großen Industrial-CF-Card hinterlegt und bootet von dort.

6.2. XSU-System

Das System ist als embedded-system ausgelegt und wird bei Anlegen der Betriebsspannung automatisch gebootet. Die Stromversorgung kann jederzeit unterbrochen werden, ohne das ein Beschädigung des Geräts oder der Daten befürchtet werden muss. Die Hochlaufzeit beträgt ca. 2 Minuten. Eine LED zeigt an, daß Betriebsspannung angelegt ist.

6.3. XSU-Schnittstellen

Der XSU besitzt folgende Schnittstellen:

6.3.1. XSU-Stromversorgung

Der XSU ist mit einem Steckernetzgerät ausgestattet, daß für eine maximale Belastung von 15 Watt ausgelegt ist. Die tatsächliche Stromaufnahme des XSU liegt bei ca. 4 bis 5 Watt.

6.3.2. XSU-Netzwerkanschluß

Am XSU befindet sich ein Netzwerkanschluß, an dem der XSU auf folgende Netzwerkprotokolle reagiert:

HTML	Zugang mit Internet-Explorer, Firefox, usw. http://192.168.1.11

User: heimo
Passwort: psolla

SSH, SCP root@192.168.1.11 password: psolla
heimo@192.168.1.11 password: psolla

Windows-Fileserver:
Der XSU fungiert als Windows-Dateiserver. Sie können auf die internen Laufwerke des XSU zugreifen, indem Sie in der Windows-Netzwerkumgebung nach dem Rechner suchen (Funkton „Computer suchen“ im Windows-Explorer und dort nach der IP-Adresse des XSU suchen) und sich mit Username: exd und Password: exd anmelden

6.3.3. EBO-Schnittstelle

An der EBO-Schnittstelle wird die kundenseitige X25-Teilnehmer-Applikation angeschlossen.

An der EBO-Schnittstelle kann an einem 9-poligen SUB-D Stecker eine V.24 Schnittstelle angeschlossen werden. Die Kommunikation erfolgt mit 19.200 baud. Die Steckerbelegung ist identisch mit der Belegung eines jeden handelsüblichen Pcs oder der handelsüblichen USB-Serial-Converter.

Pin 2 – Receive Data
Pin 3 – Transmit Data
Pin 5 – Signal-GND

6.3.4. X25-Schnittstelle

An der X25-Schnittstelle wird die X25-Gegenstelle angeschlossen. Üblicherweise wird an der X25-Schnittstelle ein X.25-PAD-Kabel angeschlossen.

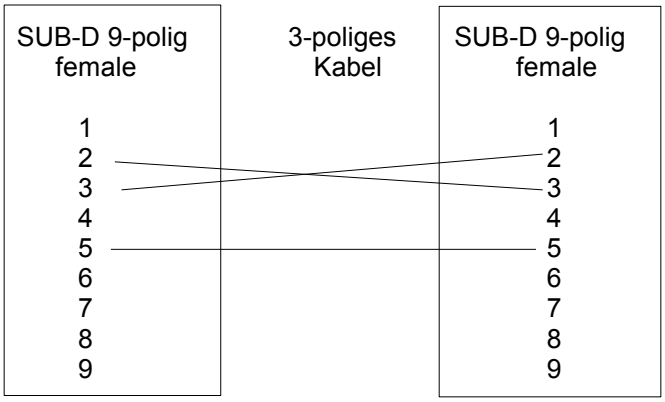
An der X25-Schnittstelle kann an einem 9-poligen SUB-D Stecker eine V.24 Schnittstelle angeschlossen werden. Die Kommunikation erfolgt mit 19.200 baud. Die Steckerbelegung ist identisch mit der Belegung eines jeden handelsüblichen Pcs oder der handelsüblichen USB-Serial-Converter.

Pin 2 – Receive Data
Pin 3 – Transmit Data
Pin 5 – Signal-GND

6.4. XSU-Kabel

6.4.1. Kabel an der EBO-Schnittstelle

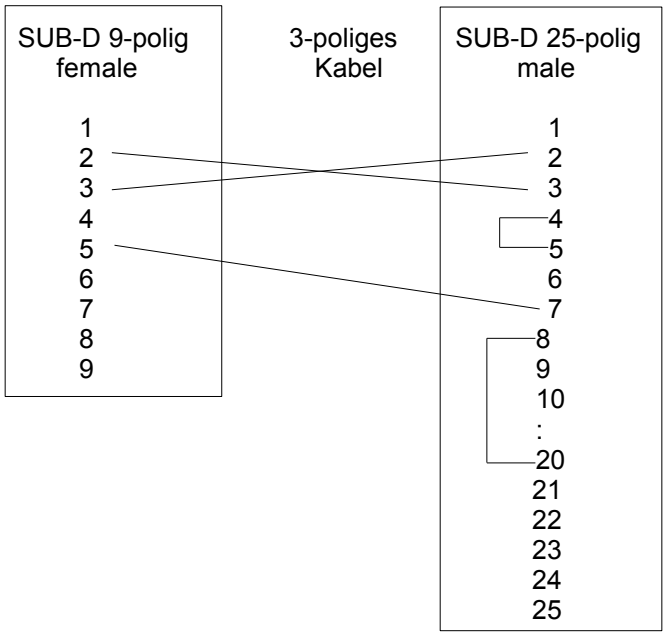
XSU-Kabel zwischen EBO-Schnittstelle und X25-Anwendung (Windows-PC)



Dieses Kabel befindet sich nicht im Lieferumfang und sollte nicht länger als 3 bis 4 Meter sein und in jedem Fall geschirmt sein. Der Schirm darf nur einseitig mit dem Gehäuse des SUB-D Steckers verbunden werden. Auf keinen Fall Signal-GND auf Pin 5 verwenden.

6.4.2. Kabel an der X25-Schnittstelle

XSU-Kabel zwischen EBO-Schnittstelle und X25-Anwendung (Windows-PC)



Das Kabel das am PAD angeschlossen wird benötigt im 25-poligen Stecker zwei Drahtbrücken zwischen den PINs
4 und **5**
8 und **20**

Dieses Kabel befindet sich nicht im Lieferumfang und sollte nicht länger als 3 bis 4 Meter sein und in jedem Fall geschirmt sein. Der Schirm darf nur einseitig mit dem Gehäuse des SUB-D Steckers verbunden werden. Auf keinen Fall Signal-GND auf Pin 5 verwenden.

7. Dokumentenverzeichnis

[1] Pflichtenheft X25 – Schnittstelle
ÖBB Pflichtenheft von GD-ST vom 22.03.93 Version 2.0 Anh.2

[2] RAD-PAD Minihowto
http://www.exd.at/RADPAD-HOWTO.html

[3] PCM5820-LRP Minihowto
http://www.exd.at/PCM5820-HOWTO.html

[4] SBC-LXA Minihowto
http://www.exd.at/SBC-LXA-HOWTO.html

[5] DATUS-PAD – HOWTO
http://www.exd.at/DATUS-HOWTO.html

[6] XSU – X25 Schnittstellenumsetzer
http://www.exd.at/xsu.html