

# Manual INTUS

INTUS TCL  
Programmierhandbuch  
D3000-004.13



## **Hinweiszeichen**



Dieses Symbol weist Sie auf Informationen hin, die für den Umgang mit **INTUS TCL** wichtig sind und beachtet werden müssen.

**INTUS TCL**  
Programmierhandbuch  
Stand 04.2017  
Bestell-Nr. D3000-004.13

### **PCS Systemtechnik GmbH**

Pfälzer-Wald-Str. 36, D-81539 München  
Telefon +49/ (0)89/68004-0  
Homepage: <https://www.pcs.com>

### **PCS Service-Center**

Telefon: +49/ (0)89/68004-666  
Fax: +49/ (0)89/68004-562  
E-Mail: [support@pcs.com](mailto:support@pcs.com)

**PCS, INTUS, DEXICON, „The terminal people.“ und „INTUS. The terminal.“** sind eingetragene Marken der **PCS Systemtechnik GmbH**.

Alle anderen Namen von Produkten und Dienstleistungen sind Marken der jeweiligen Firmen und Organisationen.

Die Vervielfältigung des vorliegenden Handbuchs, auch auszugsweise, ist nur mit ausdrücklicher Genehmigung der **PCS Systemtechnik GmbH** erlaubt.

Um stets auf dem Stand der Technik bleiben zu können, behalten wir uns Änderungen vor.

Copyright 2017 by **PCS Systemtechnik GmbH**

## **Inhaltsverzeichnis**

<b>Hinweiszeichen .....</b>	<b>1-2</b>
<b>Inhaltsverzeichnis .....</b>	<b>1-3</b>
<b>1 Einleitung.....</b>	<b>1-11</b>
1.1 TCL Version .....	1-11
1.2 Notwendige Vorkenntnisse .....	1-11
1.3 Weitere Handbücher .....	1-11
1.4 Datenaustausch in der INTUS TCL Terminalfamilie .....	1-12
1.5 Einsatz von TCL .....	1-13
<b>2 Einführung in TCL.....</b>	<b>2-1</b>
2.1 Datenflüsse.....	2-1
2.1.1 Datenfluss zwischen Host und Terminal .....	2-1
2.1.2 Datenfluss zwischen Kanal B und Terminal.....	2-4
2.2 TCL - Programm Konzept .....	2-6
2.3 Parallele Vorgänge in TCL .....	2-7
2.4 Ereignisse in TCL .....	2-8
2.5 Einführung in die syntaktischen Elemente von TCL .....	2-10
2.5.1 Konstanten .....	2-10
2.5.2 Datenoperatoren '&' und '<&>' .....	2-11
2.5.3 Datenoperator ',' .....	2-11
2.5.4 Datenoperator '()'.....	2-12
2.5.5 Datenoperator '+' .....	2-12
2.5.6 Datenoperator '//' .....	2-13
2.5.7 TCL-Feldnamen.....	2-13
2.5.8 Datenoperator '[]'.....	2-14
2.5.9 Datenoperator '[*]'.....	2-14
2.5.10 Datenoperator '-' .....	2-15
2.5.11 Operatoren '@' und '#' .....	2-15
2.5.12 Operator '%' .....	2-16
2.5.13 Ringpuffer.....	2-16
<b>3 TCL - Syntax und Übersicht.....</b>	<b>3-1</b>
3.1 Lexikalischer Aufbau.....	3-1
3.2 Übersicht über die Syntax .....	3-2
3.2.1 Konstanten .....	3-2
3.2.2 Zahlenwert .....	3-3
3.2.3 Feldnamen.....	3-3
3.2.4 Ringpuffer .....	3-4
3.2.5 Feldadressen.....	3-4
3.2.6 Quelloperanden.....	3-5
3.3 Übersicht über die TCL-Felder .....	3-5
3.4 Übersicht über die TCL-Anweisungen .....	3-7
<b>4 TCL-Felder.....</b>	<b>4-1</b>

4.1	AB Auftragsbeginn .....	4-1
4.2	AE Auftragsende.....	4-2
4.3	AN Auftragsnummer .....	4-2
4.4	B Barcodeleser.....	4-3
4.5	CE Ersetzungszeichen.....	4-5
4.6	CV Setupparameter.....	4-6
4.7	D Display .....	4-28
4.8	Dx DNCIN-Steuerfeld .....	4-29
4.9	DL Programmreich.....	4-31
4.10	Ex - Digitaler Eingang DI .....	4-32
4.10.1	Digitale Eingänge/Vandalenkontakte: Terminals/Zutrittskontrollmanager .....	4-33
4.10.2	Digitale Eingänge/Vandalenkontakte der abgesetzten Leser .....	4-35
4.10.3	Programmierhinweise .....	4-36
4.10.4	Taktüberwachung.....	4-37
4.11	ED Ergebnisfeld Divisionsrest.....	4-38
4.12	ER Ergebnis .....	4-38
4.13	EV Ergebnisfeld Vorzeichen .....	4-38
4.14	EZ Ergebnislänge.....	4-39
4.15	Gx Stillstandszeiten .....	4-39
4.16	Hx Hilfsfelder .....	4-40
4.17	I Induktivkartenleser und Identifikation mit Biometrieleser.....	4-41
4.18	KT Datum .....	4-44
4.19	Lx Lampe und Hupe .....	4-45
4.20	LS Leser- und Terminalstatus.....	4-47
4.21	LZ Gesamlaufzeit .....	4-56
4.22	M Magnetkartenleser und weitere Kartenleser .....	4-57
4.23	MI MONIN-Steuerfeld .....	4-61
4.24	ND Nicht definierte Stillstandszeiten .....	4-63
4.25	Ox Digitaler Ausgang DO .....	4-64
4.26	Px Parameterfelder.....	4-67
4.27	P0 Schlüsselschalterflag.....	4-67
4.28	P1 Notpuffer-Statusflag .....	4-68
4.29	P2 Maschinentaskflag .....	4-68
4.30	P3 Betriebsstatusflag.....	4-69
4.31	P10 MONOUT-Steuerfeld, Routinginformation .....	4-71
4.32	P20 Universelles Konfigurationsfeld.....	4-73
4.33	P21 Setupsteuerung.....	4-77
4.34	P22 Erweiterte Benutzerschnittstelle .....	4-78
4.35	PO Prozedurstatusflag.....	4-80

4.36	PN Personalnummer .....	4-81
4.37	S S-Feld.....	4-81
4.38	Sx Schnittstellenparameter.....	4-82
4.39	SP Offset (für die PT Anweisung) .....	4-89
4.40	ST Gesamtstillstandszeit.....	4-89
4.41	T T-Feld .....	4-89
4.42	Tx Taktüberwachung.....	4-90
4.43	TAx Taktausfallüberwachung.....	4-91
4.44	TF Tabellenfeld.....	4-92
4.45	TM Time - Uhrzeit im 12-Stunden-Format .....	4-93
4.46	TOx Taktabweichung.....	4-94
4.47	TR Trace .....	4-95
4.48	UR Uhrzeit.....	4-97
4.49	Zx Zähler zum digitalen Eingang .....	4-98
<b>5</b>	<b>TCL-Anweisungen.....</b>	<b>5-1</b>
5.1	@ Sprung .....	5-1
5.2	! Kommentar einfügen .....	5-1
5.3	# Sprungziel .....	5-2
5.4	% Unterprogrammsprung.....	5-3
5.5	A Display aktualisieren.....	5-4
5.6	AD Addition .....	5-5
5.7	C0 Stillstandsdauer berechnen.....	5-6
5.8	C1 Auftragsdauer berechnen.....	5-6
5.9	C2 Felder Gx initialisieren.....	5-6
5.10	C3 Ergebnis im ER Feld rechtsbündig ausgeben.....	5-7
5.11	C4 Trace ausgeben.....	5-7
5.12	D Dekrementieren.....	5-8
5.13	DI Division .....	5-9
5.14	E Eingabe über Tastatur und Kartenleser .....	5-10
5.14.1	Abgesetzte Eingabestationen - LBus Adressen .....	5-12
5.14.2	Gleichzeitige Eingabe von mehreren Eingabeeinheiten .....	5-13
5.14.3	Funktionstasteneingabe .....	5-15
5.14.4	Tastatureingabe .....	5-18
5.14.5	Lesereingabe .....	5-23
5.14.6	Eingaben über die seriellen Zusatzschnittstellen .....	5-27
5.14.7	Eingabefreigabe zurücknehmen .....	5-27
5.15	EO Exklusives Oder.....	5-28
5.16	F Felder füllen.....	5-28
5.17	GR Grafikfunktionen .....	5-29

5.18	I Inkrementieren.....	5-31
5.19	K Kopieren.....	5-32
5.20	KW Kopieren mit Wandeln .....	5-33
5.21	M1 Modulo 10/11 Fehlerprüfung .....	5-36
5.22	MU Multiplikation .....	5-38
5.23	OR Oder.....	5-38
5.24	P Vergleich .....	5-39
5.24.1	Standardoperationen .....	5-39
5.24.2	PIN-Code Verifikation ab TCL V6.20 und V6.70.....	5-39
5.25	PS Prüfsumme anhängen .....	5-41
5.26	PT Tabellenvergleich.....	5-42
5.27	R Reset.....	5-45
5.28	RD Ringpuffer lesen .....	5-47
5.29	RL Linksshift .....	5-48
5.30	RR Rechtsshift .....	5-49
5.31	RS Ringpuffer-Sprung .....	5-50
5.32	S Stop.....	5-51
5.33	SB Subtraktion.....	5-52
5.34	SE Senden an Host (über Notpuffer \$4) .....	5-53
5.35	SR Senden an Host (direkt über Ringpuffer \$2).....	5-55
5.36	T Timeout .....	5-57
5.37	UN Und.....	5-58
5.38	WO Schreiben in Ringpuffer ohne CR .....	5-58
5.39	WR Schreiben in Ringpuffer mit CR .....	5-59
5.40	XA Binäre Addition.....	5-60
5.41	XS Binäre Subtraktion .....	5-61
<b>6</b>	<b>Laufzeitsystem.....</b>	<b>6-1</b>
6.1	Ringpuffer .....	6-2
6.1.1	Empfangspuffer Hostschnittstelle (\$1) .....	6-2
6.1.2	Sendepuffer Hostschnittstelle (\$2).....	6-3
6.1.3	Interpreter-Anweisungspuffer (\$3) .....	6-3
6.1.4	Notpuffer (\$4) .....	6-3
6.1.5	Empfangspuffer Kanal B, C oder D (\$5, \$8 oder \$A) .....	6-3
6.1.6	Sendepuffer Kanal B, C oder D (\$6, \$9 oder \$B).....	6-3
6.1.7	Interpreter-Datenpuffer (\$7) .....	6-4
6.1.8	Sendepuffer (\$D) und Empfangspuffer (\$C) für abgesetzte Eingabestationen am LBus .....	6-4
6.2	MONIN-Prozess .....	6-5
6.2.1	Aufbau der Datensätze .....	6-5
6.2.2	Steuerung des MONIN-Prozesses.....	6-8
6.3	MONOUT-Prozess .....	6-9
6.3.1	Aufbau des Sendedatensatzes .....	6-10
6.3.1.1	Logische Satznummer.....	6-11

6.3.1.2	Statusflag .....	6-11
6.3.1.3	Routingbytes .....	6-12
6.3.1.4	Checksumme.....	6-12
6.3.1.5	Satzendezeichen.....	6-12
6.3.1.6	Leersatz.....	6-13
6.3.2	Betriebsart 1 (transparentes Senden).....	6-14
6.3.3	Betriebsart 2 (Voreinstellung).....	6-15
6.3.4	Betriebsart 3 .....	6-17
6.4	Sicherheitskonzept ab TCL Version 5.5 .....	6-19
6.4.1	Login auf der Hostschnittstelle .....	6-19
6.4.2	Verschlüsselung auf der Hostschnittstelle .....	6-20
6.4.3	Verschlüsselung des LBus .....	6-20
6.4.4	Setup mit Berechtigungsstufen .....	6-21
6.5	Taktüberwachung (Maschinentask) .....	6-22
6.5.1	Kompatibilität mit der INTUS 2000 Serie .....	6-24
6.6	Sommer-/ Winterzeitumschaltung .....	6-25
6.6.1	Felder zur Kontrolle der Uhrzeit und des Datums .....	6-25
6.6.1.1	Teifelder aktueller Zeitstatus .....	6-26
6.6.1.2	Teifelder Zeitkontrolle .....	6-29
6.6.2	Anwendungsbeispiele .....	6-31
6.6.3	Zeiteinträge in Buchungssätzen .....	6-33
<b>7</b>	<b>Ansteuerung des Displays.....</b>	<b>7-1</b>
7.1	Übersicht über die eingesetzten Displays .....	7-3
7.1.1	2-zeilige Displays.....	7-4
7.1.2	240x64 Pixel Display.....	7-5
7.1.3	320x240 Pixel Display .....	7-5
7.1.4	INTUS Graph 640x480 Display .....	7-6
7.1.5	320x240 Pixel TFT Display .....	7-6
7.1.6	480x272 Pixel TFT Display .....	7-6
7.2	Cursorsteuerung und Editorfunktionen.....	7-7
7.3	Zeichensätze und Zeichenmapping .....	7-10
7.3.1	Statische Änderung des Zeichensatzes .....	7-12
7.3.2	Temporäre Änderung des Zeichensatzes .....	7-13
7.3.3	Beispiel 1: Ausgaben mit mehreren Zeichensätzen .....	7-14
7.3.4	Beispiel 2: Ausgaben im ISO 8859-1 8-Bit Code mit Semigrafik .....	7-15
7.4	Doppelthohe und -breite Darstellung .....	7-16
7.5	Anzeige- und Ausgabeseite.....	7-16
7.6	Steuersequenzen für Bit-Grafik .....	7-17
7.6.1	Ausgabe von Bit-Grafik an Zeichenpositionen.....	7-17
7.6.2	Ausgabe von Bit-Grafik an Pixelpositionen .....	7-18
<b>8</b>	<b>TCL-Programmentwicklung .....</b>	<b>8-1</b>
8.1	Speicheraufteilung .....	8-1
8.2	TCL-Programm schreiben .....	8-2
8.3	TCL-Programm laden .....	8-4
8.3.1	Laden in den Programmreich .....	8-4
8.3.2	Laden des Defaultprogramms .....	8-5
8.3.3	Defaultprogramm .....	8-5

8.4	TCL-Programm starten .....	8-6
8.4.1	Programmstart vom Leitrechner .....	8-6
8.4.2	Programmstart nach Terminal-Reset .....	8-6
8.4.3	Anlaufmodi: Warm-, Kalt- , Neu- Eiskalt- und Comstart.....	8-6
8.4.3.1	Warmstart.....	8-6
8.4.3.2	Kaltstart und Neustart .....	8-7
8.4.3.3	Eiskaltstart .....	8-8
8.4.3.4	Comstart.....	8-8
8.5	TCL-Programm testen .....	8-8
8.5.1	Format der TCL-Fehlermeldungen .....	8-8
8.5.2	Syntaktische Fehler beheben.....	8-9
8.5.3	Semantische Fehler beheben.....	8-9
8.5.4	Laufzeitfehler beheben.....	8-10
8.5.5	Systemfehlermeldungen.....	8-11
<b>9</b>	<b>Fehlermeldungen .....</b>	<b>9-1</b>
9.1	Syntaktische Fehlermeldungen .....	9-1
9.2	Semantische Fehlermeldungen .....	9-5
9.3	Systemfehlermeldungen.....	9-8
9.3.1	Konfigurationsfehler des Programmreichs.....	9-8
9.3.2	Prozess-Fehlermeldungen .....	9-8
9.3.3	Abbruch-Fehlermeldungen .....	9-9
9.4	Fehlercodes bei Lesereingaben .....	9-11
<b>10</b>	<b>TCL Zeichensätze .....</b>	<b>10-1</b>
<b>11</b>	<b>TCL Adressen der Zutrittskontrollmanager.....</b>	<b>11-1</b>
11.1	INTUS ACM40, INTUS ACM40 AKKU .....	11-2
11.1.1	LBus 1: PP / LBus 2: PP (Voreinstellung).....	11-2
11.1.2	LBus 1: PP / LBus 2: nicht belegt.....	11-2
11.2	INTUS ACM4, INTUS ACM4 AKKU .....	11-3
11.2.1	LBus 1: PP / LBus 2: PP (Voreinstellung).....	11-3
11.2.2	LBus 1: PP / LBus 2: nicht belegt.....	11-3
11.3	INTUS ACM8, INTUS ACM8(0)e Rack .....	11-4
11.3.1	Point-to-Point Verkabelung .....	11-4
11.3.2	Multipoint Verkabelung .....	11-5
11.3.3	DI /-DO-Adressen für Systemanwendungen .....	11-5
11.4	INTUS ACM80e Wand .....	11-6
11.4.1	Point-to-Point Verkabelung .....	11-6
11.4.2	Multipoint Verkabelung .....	11-7
11.4.3	DI /-DO-Adressen für Systemanwendungen .....	11-7
11.5	INTUS ACM8e Wand .....	11-8
11.5.1	Point-to-Point Verbindung .....	11-8
11.5.2	Multipoint Verbindung .....	11-9
11.5.3	DI /-DO-Adressen für Systemanwendungen .....	11-9
11.6	INTUS 3000 ACM.....	11-10
11.6.1	LBus 1: MP / LBus 2: MP (Voreinstellung).....	11-10
11.6.2	LBus 1: PP / LBus 2: MP.....	11-11
<b>12</b>	<b>Anhang.....</b>	<b>12-1</b>

<b>Tabellenverzeichnis.....</b>	<b>12-1</b>
<b>Abbildungsverzeichnis.....</b>	<b>12-2</b>
<b>Dokumenthistorie.....</b>	<b>12-3</b>
<b>Haben Sie noch Fragen?.....</b>	<b>12-6</b>



# 1 Einleitung

## 1.1 TCL Version

Das vorliegende Handbuch beschreibt die TCL Version 6.70.

## 1.2 Notwendige Vorkenntnisse

Zum Verständnis dieses Handbuchs sind grundlegende Kenntnisse der Datenverarbeitung notwendig.

## 1.3 Weitere Handbücher

Außer dem vorliegenden Programmierhandbuch sind noch folgende Handbücher für alle INTUS Terminals und INTUS Zutrittskontrollmanager erhältlich:

- ***Installations- und Wartungshandbuch***  
Dieses Handbuch für Monteur und Elektriker beschreibt die Montage, Installation und Wartung des INTUS Terminals bzw. INTUS Zutrittskontrollmanagers. Darin finden Sie ausführliche Informationen über erforderliche Anschlüsse, Schnittstellen und die Umgebungsbedingungen.
- ***Konfiguration und Betrieb***  
Dieses Handbuch beschreibt die Inbetriebnahme, die Parametrierung und die Fehlerdiagnose des INTUS Terminals bzw. INTUS Zutrittskontrollmanagers.
- ***INTUS 3000 Präprozessor TCL (Bestellnummer D3000-007)***  
Dieses Handbuch beschreibt den TCL-Präprozessor, der die Strukturierung eines TCL-Programms unterstützt, symbolische Bezeichnungen ermöglicht und eine Cross-Referenz Liste erzeugt. Er ist für alle TCL Versionen ab TCL Version 4.2 einsetzbar.

## 1.4 Datenaustausch in der INTUS TCL Terminalfamilie

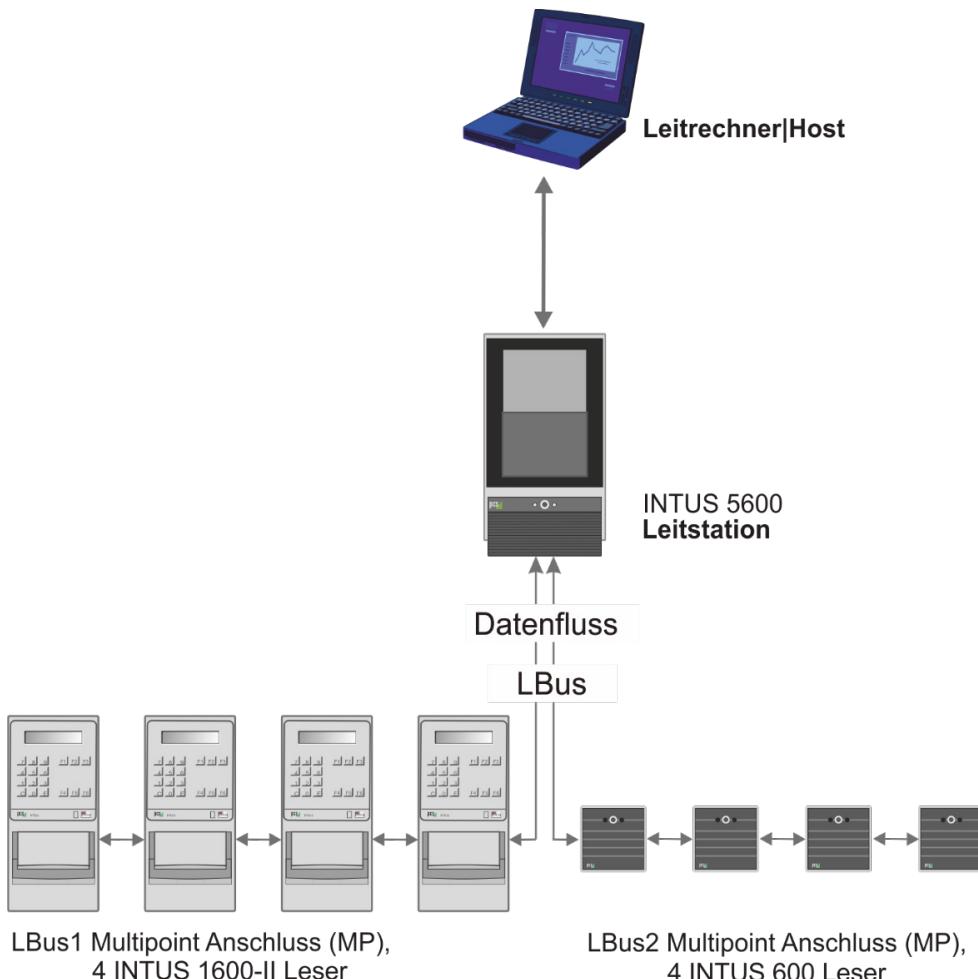


Die INTUS TCL Terminalfamilie bestand zunächst aus mehreren, unterschiedlichen INTUS 3000 Terminals und dem Zutrittskontrollmanager INTUS 3000 ACM. Es kamen weitere Zutrittskontrollmanager hinzu. Beispielhaft seien hier nur INTUS ACM40, ACM80e Rack und INTUS ACM80e Wand genannt. Die jüngsten Modelle in der Terminalfamilie führen die „5000“ im Namen, wie z.B. der INTUS 5600. Der Ausdruck „INTUS 3000 Terminal“ in diesem Dokument bezieht sich auf alle Modelle der INTUS TCL Terminalfamilie.

Die INTUS 3000 Terminals sind in den verschiedenen Varianten ideale Geräte zur Personal-Zeiterfassung, für die Betriebsdatenerfassung und zur Zutrittskontrolle. Sie besitzen ein Display zur Anzeige von Information sowie eine Tastatur für die Bedienung und im allgemeinen einen Leser für Datenträger auf Identifikations-Karten, zum Beispiel RFID-Transponder, Barcode, Magnet- oder Chipkarten.

Im Normalfall werden die INTUS 3000 Terminals über einen Kommunikationskanal an einen Leitrechner angeschlossen. Der Leitrechner wird auch als Host bezeichnet und die am Terminal befindliche Kommunikationsschnittstelle als Hostschnittstelle.

Das INTUS 3000 selbst kann Leitstation für eine Reihe von externen, abgesetzten Eingabestationen sein, die an einem LBus angeschlossen sind. Diese abgesetzten Eingabestationen verfügen über einen Leser und eventuell über ein Display und Tastatur, wie zum Beispiel das INTUS 1600.



*Abbildung 1.1 – INTUS TCL Terminalfamilie*

## 1.5 Einsatz von TCL

INTUS 3000 Terminals sind vom Anwender frei programmierbar. Die auf den Terminals ablaufende Programmiersprache TCL ist speziell darauf ausgerichtet, die Anforderungen an eine rasche Abarbeitung unabhängiger Ereignisse zu erfüllen.

Dazu kommt, dass TCL speziell auf die Bedürfnisse von Industrie-Terminals zugeschnitten ist. In die Sprache integriert sind folgende wichtige Elemente:

- Steuerung und Verarbeitung von Leser- und Tastatureingaben
- Ereignisgetriebene Abarbeitung von Änderungen an digitalen Eingängen
- Ausgaben auf Displays
- Steuerung von Leuchtdioden, Hupen und digitalen Ausgängen
- Zeitüberwachungsfunktionen zur Überwachung von mehreren, parallel laufenden Vorgängen
- gesicherte Datenübertragung zwischen Terminal und Host (Notpuffer-Konzept)

Da sich der Übersetzer und der Interpreter der TCL-Sprache in dem INTUS Terminal befindet, kann ein TCL-Programm in Quellform von praktisch jedem Hostrechner aus in das Terminal geladen werden.

Somit kann man von einem TCL-Programmiersystem im Terminal sprechen, das die Programmierung durch folgende Funktionen unterstützt:

- die Meldung eventuell auftretender Fehler kann auf verschiedene Ausgabekanäle umgesteuert werden
- Testunterstützung in Form eines ein- und ausschaltbaren Programm-Trace
- Anzeige von Zuständen und Variablen durch TCL-Kommandos, die sofort ausgeführt werden

Dieses Dokument beschreibt die aktuell mit der Terminalserie ausgelieferte Version des TCL-Programmiersystems in Form eines Referenz-Manuals.



Sollte es nach diesem Stand noch weitere Änderungen oder Erweiterungen geben, sind sie den aktuellen Release Notes zu entnehmen.



## **2 Einführung in TCL**

### **2.1 Datenflüsse**

#### **2.1.1 Datenfluss zwischen Host und Terminal**

Um eine schnelle Antwortzeit zu garantieren, laufen im TCL-Programmsystem quasi-parallele Prozesse, von denen die meisten ihre Daten in FIFO-Speicher ablegen bzw. abholen. Die FIFO-Speicher sind als Ringpuffer implementiert und werden im Folgenden so bezeichnet.

Die Abbildung 2.1 gibt eine Übersicht über die an der Host-Kommunikation beteiligten Prozesse.



Die Kenntnis der Prozesse und der zum Datenaustausch verwendeten Ringpuffer ist für die TCL-Programmierung wichtig, da der Programmierer in vielfältiger Weise in diesen Datenfluss eingreifen kann.

Zunächst einmal kann der Programmierer per TCL-Anweisung direkt Daten aus einem Ringpuffer lesen, oder er kann ihn beschreiben und damit einen Datentransfer anstoßen.

Weiterhin kann das Verhalten der verschiedenen Prozesse durch spezielle Steuerfelder beeinflusst werden.

Die Ringpuffer werden im TCL-System mit zweibuchstabigen Kurzbezeichnungen angesprochen, von denen der erste Buchstabe ein '\$' ist und der zweite Buchstabe eine Hexadezimalziffer, z.B. \$1 und \$A.

In den folgenden Diagrammen werden:

- Puffer mit einfacher Umrandung,
- Prozesse mit fetter Umrandung dargestellt.

Bei Fehlermeldungen erscheint der Name des Prozesses, der den Fehler meldet, als erstes, so dass der Benutzer daran schon die Stelle im Datenfluss des Systems erkennen kann, an der ein Fehler aufgetreten ist.

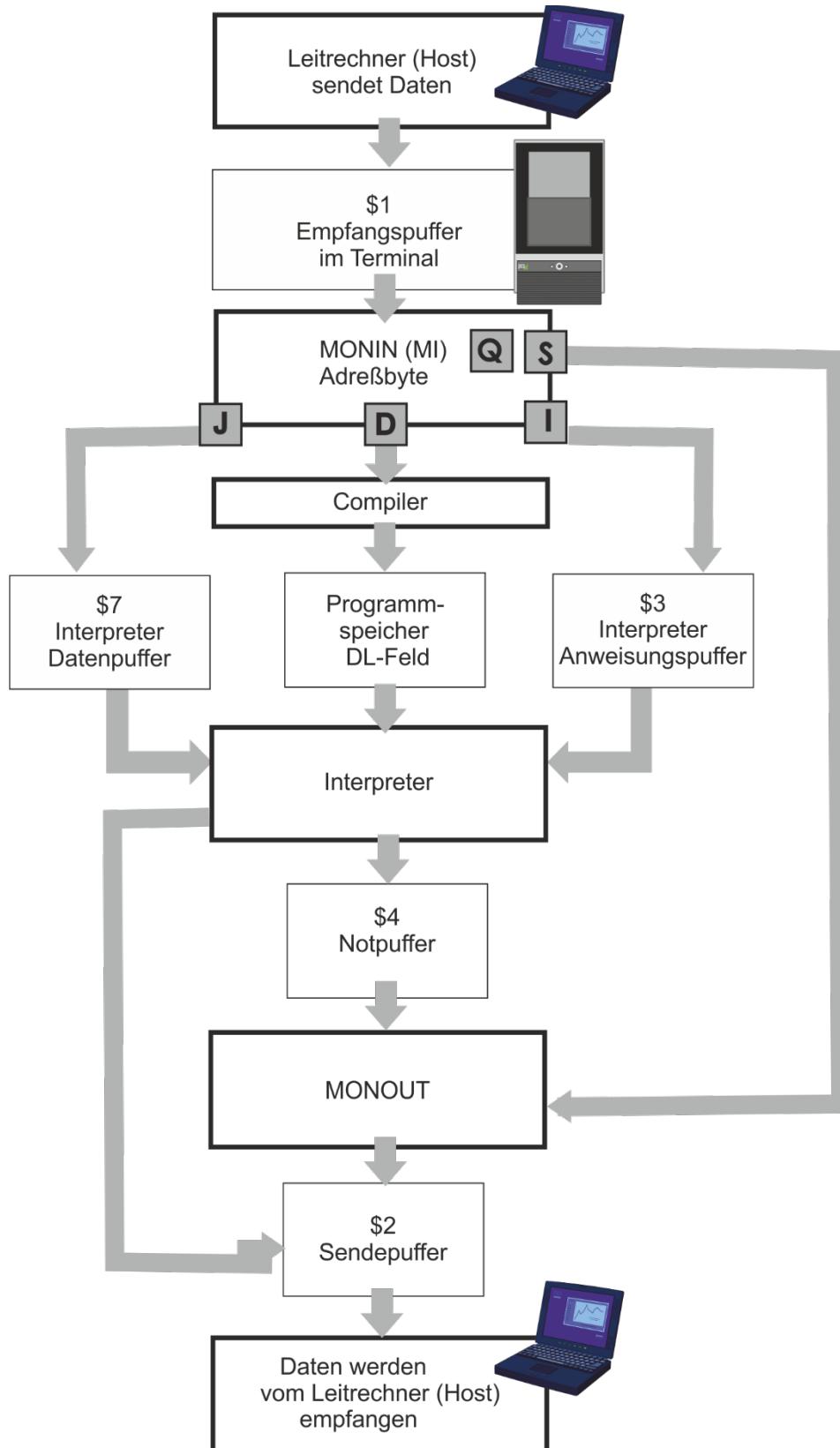


Abbildung 2.1 - Datenfluss zwischen Leitrechner und Terminal

Das Diagramm gibt den normalen Datenfluss vom Host durch das INTUS Terminal und zurück zum Host wieder.

### **MONIN-Prozess**

Die vom Host ankommenden Daten werden zunächst in dem \$1 Empfangspuffer abgelegt. Von dort entnimmt der MONIN-Prozess Datensätze, die in TCL standardmäßig durch ein CR-Zeichen, "0D", abgetrennt werden. Anhand des ersten Zeichens eines Datensatzes, dem MONIN-Adressbyte, entscheidet der MONIN-Prozess, welchen weiteren Weg ein Datensatz nimmt. In dem Diagramm sind nur die wichtigsten Adressbytes zu sehen.

Mit dem Adressbyte 'D' versehene Datensätze sind Zeilen von TCL-Programmen, die ohne das 'D' dem internen TCL-Compiler zur Übersetzung übergeben werden. Das Resultat der Übersetzung wird dem im DL-Feld befindlichen, ausführbaren TCL-Programm angehängt.

Mit dem Adressbyte 'T' versehene Datensätze werden ebenfalls dem Compiler übergeben, nach der Übersetzung jedoch dann zur direkten Ausführung in den \$3 Interpreteranweisungspuffer geschrieben.

Datensätze mit dem Adressbyte 'J' werden in den \$7 Interpreterdatenpuffer abgelegt und können von dort mit Hilfe von TCL-Anweisungen abgeholt werden.

Das Verhalten des MONIN-Prozesses lässt sich mit Hilfe des MI-Felds zusätzlich beeinflussen.

### **TCL-Interpreter**

Der TCL-Interpreter führt das im DL-Feld befindliche Programm solange aus, bis eine Stop-anweisung gefunden wird. Danach wird die nächste Anweisung aus dem \$3-Puffer gelesen. Dort werden neben den mit dem 'T'-Adressbyte versehenen TCL-Anweisungen alle Ereignissprünge, deren Mechanismus im nächsten Abschnitt erläutert wird, abgelegt.

Per TCL-Anweisung kann der Interpreter in praktisch jeden Ringpuffer schreiben und aus jedem Ringpuffer lesen. Dies birgt jedoch die Gefahr der Verklemmung, siehe auch Kapitel 6.1 „Ringpuffer“.

Daten, die durch eine gesicherte, das heißt eine quittierte Übertragung zum Host geschickt werden sollen, werden mit Hilfe einer SE Anweisung vom Interpreter in den \$4-Notpuffer abgelegt.

Der Interpreter kann auch direkt mit einer SR Anweisung in den \$2-Sendepuffer schreiben. In diesem Fall wird keine Quittung erwartet und der Benutzer muss damit rechnen, dass der Datensatz verloren geht, oder er muss den Transport über einen anderen Mechanismus sichern.

### **MONOUT-Prozess**

Aus dem \$4-Notpuffer werden die Datensätze vom MONOUT-Prozess ausgelesen, eventuell mit Satznummer, Statusflag und Checksumme versehen, und in den \$2 Sendepuffer abgelegt, von wo sie schnellstmöglich zum Host übertragen werden. Wenn der MONIN-Prozess eine Quittung für den Datensatz erhält, Adressbyte 'Q', dann gibt dieser die Quittung dem MONOUT-Prozess weiter, der daraufhin den Datensatz aus dem Notpuffer löscht und den nächsten zum Host sendet.

Das Verhalten des MONOUT-Prozesses wird durch die PO-, P1-, P3- und P10-Felder kontrolliert, die sich durch TCL-Anweisungen ändern lassen.

### **Ringpuffergrößen**

Die Ringpuffer können alle einen maximalen Datensatz von 256 Byte Länge aufnehmen, so dass längere Datensätze durch den Host aufgespalten und durch spezielle TCL-Anweisungen wieder zusammengefügt werden sollten, falls notwendig.

Der \$2 Sendepuffer ist mit 273 Bytes etwas größer, weil er neben dem 256 Byte großen Datensatz auch eventuell noch Satznummer, Routingbytes und Sonstiges aufnehmen muss. Der \$4 Notpuffer hat eine konfigurierbare Länge von 3kB bis zur modellabhängigen Speicherobergrenze, so dass sich dort auch große Mengen an Daten ablegen lassen, wenn der Host nicht erreichbar ist.

## **2.1.2 Datenfluss zwischen Kanal B und Terminal**

Das untenstehende Diagramm zeigt den Datenfluss für Daten, die an zusätzlichen (seriellen) Schnittstellen des Terminals anfallen oder dorthin ausgegeben werden, beispielweise bei Anschluss einer Waage.

Die Daten, die an der ersten Zusatzschnittstelle, auch 2. Schnittstelle oder Kanal B genannt, anfallen, werden zunächst in den Empfangspuffer \$5 abgelegt.

Im Normalfall werden sie von dort durch den DNCIN0-Prozess gelesen, der sie auf vielfältige Art und Weise weiterverarbeiten und in andere Ringpuffer und TCL-Felder ablegen kann.

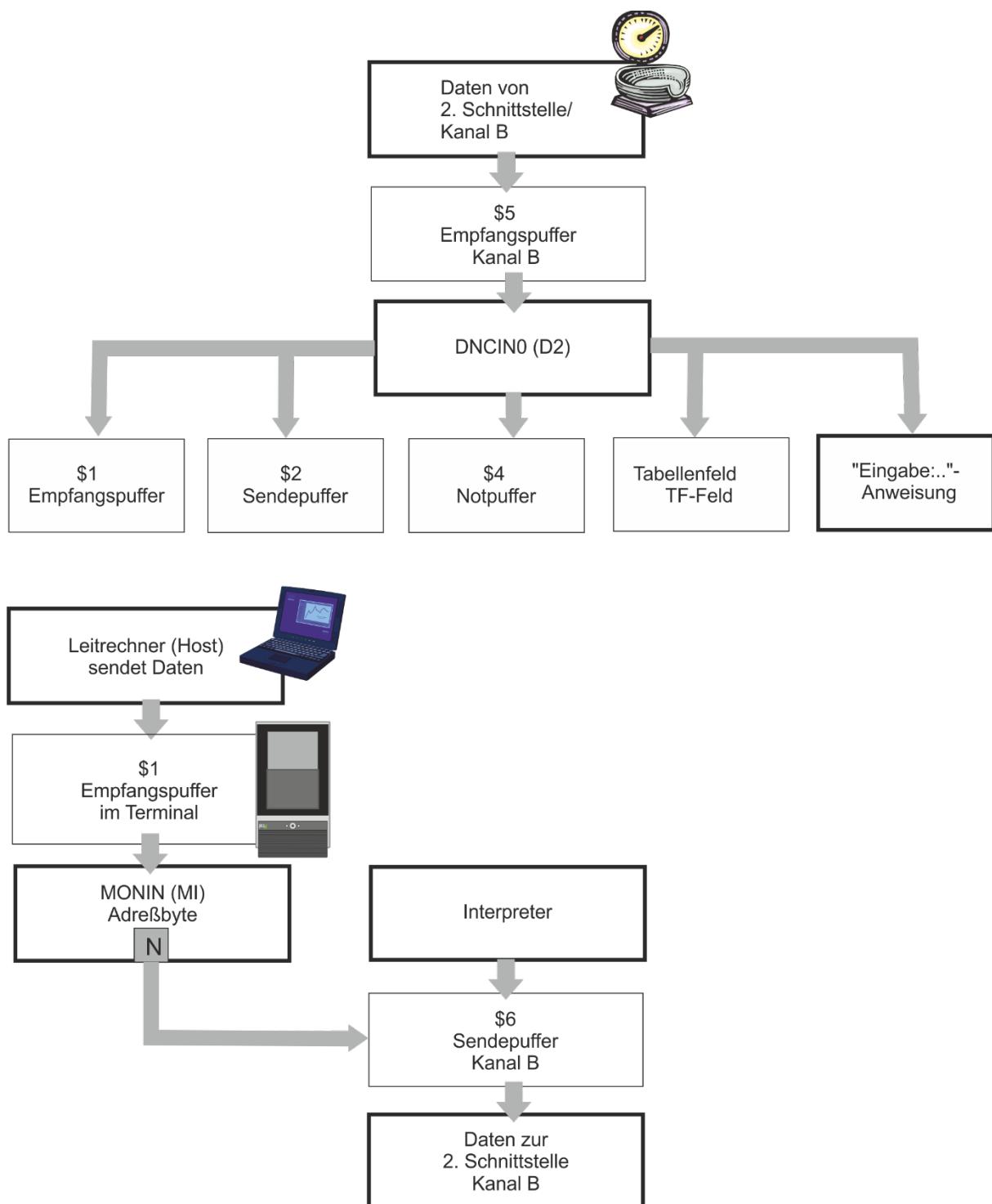
Der DNCIN-Prozess der 2. Schnittstelle, Kanal B, wird durch das D2-Feld kontrolliert.

Hier lässt sich die Weiterverarbeitung und Weiterreichung der Datensätze einstellen. Es lässt sich aber auch dort einstellen, dass er keine Daten aus dem \$5 Empfangspuffer entnehmen soll. In diesem Fall kann der Interpreter per TCL-Anweisung dort die Roh-Daten des Kanals B entnehmen.

Wenn der Interpreter in den Sendepuffer \$6 schreibt, dann werden die Daten an das an der 2. Schnittstelle angeschlossene Gerät gesandt; den gleichen Weg nehmen auch die Datensätze, die vom Host mit dem MONIN-Adressbyte 'N' versehen wurden.

Wenn es weitere Schnittstellen gibt, etwa die 3. Schnittstelle, Kanal C, oder die 4. Schnittstelle, Kanal D, dann sind die dazugehörigen Empfangspuffer \$8 bzw. \$A und die Sendepuffer \$9 bzw. \$B.

Die dazugehörigen Verarbeitungsprozesse heißen DNCIN1 und DNCIN2 mit den Steuerfeldern D3 und D4. Vom Host können die MONIN-Adressbytes 'O' und 'P' für die direkte Ausgabe über \$9 bzw. \$B verwendet werden.



*Abbildung 2.2 - Datenfluss zwischen Terminal und Gerät an Kanal B*

## 2.2 TCL - Programm Konzept

Die Reaktion auf externe, asynchrone Ereignisse ist bei TCL ein Teil des Konzepts der Programmiersprache.

- Alle eingetretenen Ereignisse werden in Form einer Sprunganweisung in den \$3-Ringpuffer eingetragen.
- Sobald der TCL-Interpreter die Ausführung eines Programmabschnitts aus dem Programmspeicher DL bei einer Stopanweisung „S;“ beendet hat, führt er die nächste Anweisung aus dem \$3-Puffer aus.
- Das TCL-Programm wird abhängig von dem Ereignis an einer vom Programmierer vorgegebenen Stelle (Sprungziel) fortgesetzt. Dies hat den Vorteil, dass ein Programmierer Ereignisse schon anhand der Programmstelle (Sprungziel), die angesprungen wird, unterscheiden und differenziert behandeln kann.

Im anschließenden Beispiel-Programm laufen die zwei Anweisungsfolgen #10/#11 und #20 völlig unabhängig als parallele Vorgänge ab.

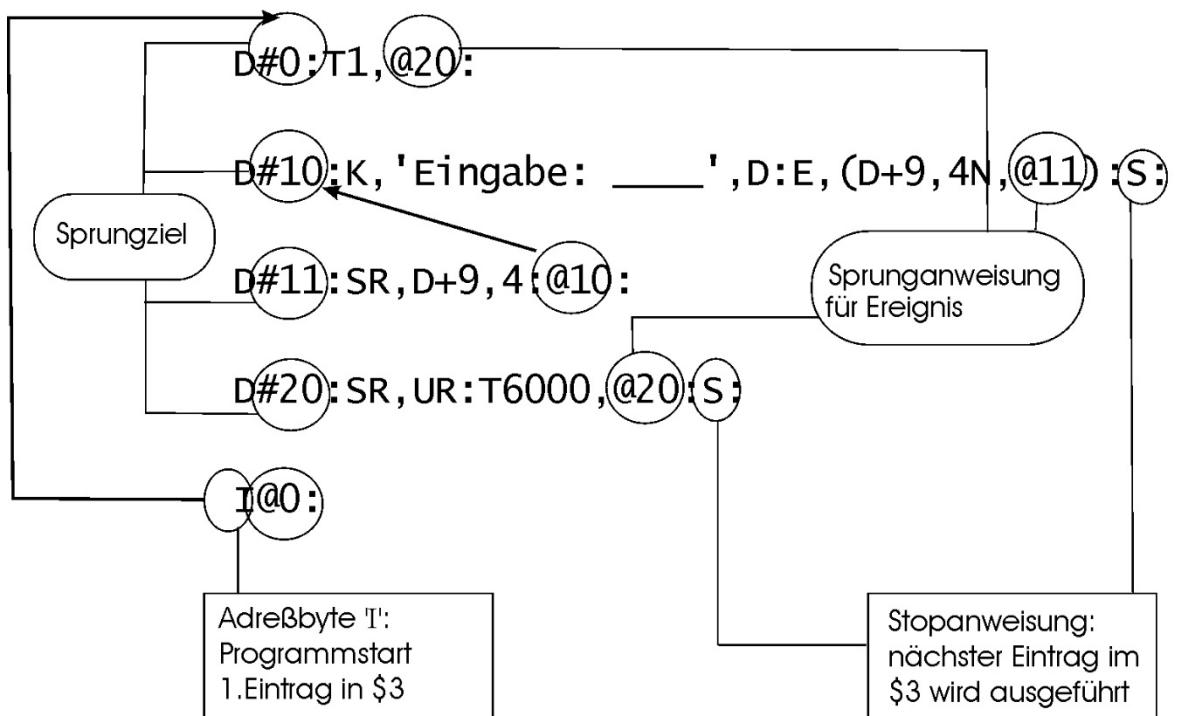


Abbildung 2.3 – Programm Konzept

## 2.3 Parallele Vorgänge in TCL

In einem System unter Last können mehrere Ereignissprünge in \$3 auf Ausführung warten. So können auch mehrere Lesungen von internen und abgesetzten Lesern auf Abarbeitung warten.

Da es nur ein Feld je Lesertyp im TCL-System gibt, das M-, I- oder B-Feld, in das Leserdaten eingetragen werden, sorgt das TCL-System für eine Sequentialisierung der quasi-parallelen Abläufe.

Die Abarbeitung der Lesungen wird vom TCL-System in folgender Weise sequentialisiert:

- die Leserdaten werden beim Eintritt eines Leser-Ereignisses in das entsprechende Leserfeld eingetragen.
- Das TCL-System garantiert, dass diese Daten nicht durch eine andere Lesung überschrieben werden, von der Sprunganweisung für ein Ereignis bis zur Stopanweisung.
- Nach dieser Stopanweisung kann das Leserfeld dann vom System verändert werden, wenn eine andere Lesung vorliegt.

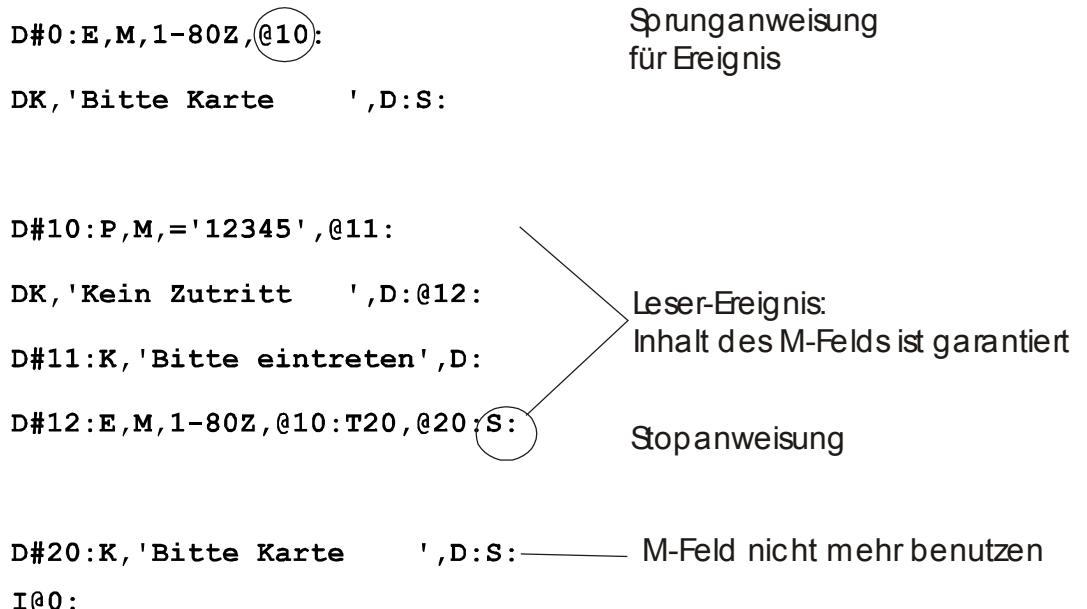


Abbildung 2.4 – Beispiel eines parallelen Vorgangs

Für den Programmierer heißt dies, dass der Inhalt der M-, I- und B-Leserfelder vom Ereignissprung bis zur nächsten Stopanweisung unverändert bleibt. Danach ist der Inhalt jedoch nicht mehr garantiert.

Es ist also fehlerhaft, eine Zeitüberwachung bei einem Leserereignis anzufordern und dann bei Eintreffen des Zeitüberwachungsergebnisses nochmals auf das Leserfeld zuzugreifen. Diese kann dann schon eine andere Lesung enthalten. Es sei denn, man stellt sicher, dass nur ein Leser im System freigegeben war und dieser über den Zeitraum der Zeitüberwachung gesperrt ist.

Eine weitere Hilfe zur Bewältigung paralleler Lesereignisse besteht darin, dass ein Leser nach der Lesung für weitere Lesungen gesperrt bleibt, bis er wieder explizit durch eine TCL-Anweisung freigegeben wird. Somit kann der Programmierer die zu einem Leser gehörende Verwaltungsinformation in einer Tabelle, die er definiert hat, dadurch schützen, dass der Leser erst nach Abschluss einer komplexen Transaktion wieder freigegeben wird.

Ähnlich wie die M-, I- und B-Lesefelder sind die Teilstücke in P20, P20+20,2 und P20+24,2, die - neben dem Sprungziel - Auskunft über die Herkunft eines Ereignisses geben, nur von dem Eintreffen des Ereignisses bis zur nächsten Stopanweisung für ein Ereignis gültig.

### **Vom TCL-Programm unabhängige Felder**

Es gibt einige (Teil-) Felder, die ihren Wert während der Ausführung eines TCL-Programmstücks ändern. Diese Felder liefern bei zwei aufeinanderfolgenden Zugriffen nicht unbedingt denselben Wert. Diese Felder reflektieren den aktuellen Zustand eines Hardware-signals, das sich unabhängig von einer Stopanweisung ändern kann.

Zu diesen Feldern gehört neben den Uhrzeit- und Datumsfeldern, UR, TM und KT, die erste Speicherzelle der Ex-Felder, die den Zustand der digitalen Eingänge anzeigt, sowie die dazugehörigen Zähler in den Zx-Feldern.

Da sich der Wert eines digitalen Eingangs in der ersten Speicherzelle eines Ex-Felds auch schon sofort nach einem DI-Ereignissprung geändert haben kann, sollte ein Programmierer in diesem Fall nicht den aktuellen Wert für die weitere Verarbeitung nehmen, sondern die Flankenstellung in der zweiten Speicherzelle des Ex-Felds als Basis für die weitere Entscheidung benutzen.

Weiterhin gehören zu den sich unabhängig ändernden Feldern auch Felder, die nicht vom Interpreter selbst sondern von anderen Prozessen aktualisiert werden: das PO-Feld, das P3-Feld sowie die Zustandsinformation für die Leser im LS-Feld.

## **2.4 Ereignisse in TCL**

Ein Ereignis wird ausgelöst, wenn im Programm eine Anforderung oder Freigabe in Form einer TCL-Anweisung steht oder ein Eintrag eines Sprungziels in bestimmte TCL-Felder.

Beispiele für eine Anforderung in Form einer Anweisung sind:

- die T Anweisung für die Zeitüberwachung,
- die RS-Anweisung für Ringpuffer-Ereignisse,
- die Anweisung E gibt Eingabeereignisse frei,
- weitere Anweisungen existieren, die die jeweiligen Freigaben oder Anforderungen wieder rückgängig machen (z.B. R,T , oder R,E Anweisungen).

Eine Freigabe in Form eines Eintrags in ein TCL-Feld findet sich bei den digitalen Eingängen (Ex-Felder) und den dazugehörigen Zählern (Zx-Felder) sowie an vielen anderen speziellen Stellen.

Die folgende Liste gibt eine Übersicht über die Art der Ereignisse in TCL und deren Aktivierungsmechanismen:

Ereignis	Aktivierung der Überwachung	Verhalten	Deaktivierung der Überwachung
<b>Eingaben über Tastatur oder Leser</b>	<b>E Anweisung</b>	Ereignis muss nach dem Auftreten neu angefordert werden	<b>R,E Anweisung</b>
<b>Zeitüberwachung (Timeout)</b>	<b>T Anweisung</b>	Ereignis muss nach dem Auftreten neu angefordert werden	<b>R,T Anweisung</b>
<b>Digitale Eingänge</b>	<b>Ex-Feld (Ex+1,1 und Ex+2,2) P2-Feld</b>	Ereignis wiederholt sich bis zur Deaktivierung	<b>Ex-Feld (Ex+1,1 und Ex+2,2) P2-Feld</b>
<b>Überlauf des zum digitalen Eingang gehörenden Zählers</b>	<b>Zx-Feld (Zx+14,2) P2-Feld</b>	Ereignis muss nach jedem Auftreten im Zx-Feld neu gesetzt werden	<b>Eintreten des Ereignisses</b>
<b>Taktüberwachung am digitalen Eingang</b>	<b>Tx-Feld (Tx+3,2 und Tx+8,2), TAX-Feld (TAX+3,2), Ex+1,1, P2-Feld.</b>	Ereignis wiederholt sich bis zur Deaktivierung	<b>Tx-Feld (Tx+3,2 und Tx+8,2), TAX-Feld (TAX+3,2), Ex+1,1, P2-Feld.</b>
<b>Daten in Ringpuffer</b>	<b>RS Anweisung</b>	Ereignis muss nach dem Auftreten neu angefordert werden	<b>R,R Anweisung</b>
<b>\$4-Notpuffer voll</b>	<b>für Interpreter P1-Feld (P1+1,4) für DNCIN-Prozesse Dx-Felder (Dx+3,4)</b>	Ereignis wiederholt sich bis zur Deaktivierung	<b>für Interpreter P1-Feld (P1+1,4) für DNCIN-Prozesse Dx-Felder (Dx+3,4)</b>
<b>Änderung Kommunikationsstatus mit Host</b>	<b>PO-Feld (PO+1,4) P3-Feld (P3+1,4)</b>	Ereignis wiederholt sich bis zur Deaktivierung	<b>PO-Feld (PO+1,4) P3-Feld (P3+1,4)</b>
<b>Setup</b>	<b>P21-Feld (P21+2,4)</b>	Ereignis wiederholt sich bis zur Deaktivierung	<b>P21-Feld (P21+2,4)</b>
<b>Treiberfehler</b>	<b>LS-Feld (LS+92,4)</b>	Ereignis wiederholt sich bis zur Deaktivierung	<b>LS-Feld (LS+92,4)</b>

Tabelle 2.1 – Ereignisse in TCL

Die Freigabe wird rückgängig gemacht, in dem man eine Null als Sprungziel in das jeweilige Feld einträgt; ein Ereignissprung nach #0 ist in diesem Fall nicht möglich.

Er ist auch nicht sinnvoll, weil #0 der festgelegte Programmeinsprung bei einem Warmstart nach einem Einschalten oder Reset des Terminals ist.

## 2.5 Einführung in die syntaktischen Elemente von TCL

In diesem Abschnitt werden Datentypen, Ausdrücke und Operatoren von TCL vorgestellt. Im folgenden Abschnitt wird die TCL-Syntax formal spezifiziert und eine Übersicht über die TCL-Felder und Anweisungen gegeben.

### 2.5.1 Konstanten

In TCL werden keine festumrissenen Datentypen verwendet, wie in anderen modernen und strukturierten Programmiersprachen. Die Handhabung der Konstanten und der Variablenfelder entspricht eher der Programmiersprache COBOL. Wie dort auch, werden in TCL Variablenfelder als Aneinanderreihung von Speicherzellen zu 8 Bit verstanden, hervorragend geeignet zur Aufnahme von Text in Form von Zeichenketten. Daher sind die gebräuchlichsten Konstanten Zeichenketten, die mit einfachen Hochkommata geklammert sind:

#### 'TCL Zeichenkette'

Eine Zeichenkette in TCL kann selbst keine Hochkommata enthalten; es ist im Prinzip möglich, nichtdruckbare Zeichen in eine Zeichenkette einzustreuen, jedoch sollte man davon Abstand nehmen, weil einige nichtdruckbare Zeichen zur Satztrennung verwendet werden und andere von den zwischengeschalteten Treibern nicht weitergereicht werden.

Der Zeichensatz in TCL ist konfigurierbar. Standardmäßig ist der deutsche ISO-7 Bit Code eingestellt, in dem eine gleichzeitige Darstellung von 'l' und 'Ü' nicht möglich ist; es ist jedoch auch möglich den Latin-1 ISO 8859-1 8 Bit Code einzustellen, der z.B. bei MS-Windows und beim UNIX System Standard ist. Um dies zu nutzen, muss natürlich der zur Hostschnittstelle gehörende Treiber auch für den Transport von 8-Bit Zeichen eingerichtet sein. Die Einstellung des Zeichensatzes kann über VT100 ESC-Sequenzen (siehe Abschnitt 7.3) oder über den Setup erfolgen (siehe Betriebshandbuch).

Wenn man in TCL eine Zeichenkette mit nichtdruckbaren Zeichen angeben will, verwendet man dazu die doppelten Hochkommata und schreibt die Zeichen in hexadezimaler Darstellung, wobei die hexadezimalen Ziffern A bis F groß geschrieben werden müssen. Weiterhin muss immer eine gerade Anzahl von Buchstaben/ Ziffern verwendet werden:

```
"313030"  
"64"  
"AFFEDEAD"
```

Vorsicht: Die Länge einer hexadezimalen Zeichenkette ist die Hälfte der Anzahl der hexadezimalen Ziffern. Wird ein Feld etwa mit Hilfe der Kopieranweisung K überschrieben, so beschränkt sich die Länge der Veränderung auf die Anzahl der Buchstaben in der konstanten Zeichenkette. Die beiden Anweisungen

```
K, '100',D:  
K, "313030",D:
```

haben die gleiche Wirkung: die ersten drei Speicherzellen des variablen TCL-Feldes D werden mit den Zeichen '100' überschrieben (und dann auf das Display ausgegeben). Die anderen Speicherzellen des D-Felds bleiben unberührt. Eine TCL-Zeichenkette hat also kein angehängtes Endezeichen, wie zum Beispiel das '\0'-Zeichen in der Programmiersprache C.

### **2.5.2 Datenoperatoren '&' und '<&>'**

Die Datenoperatoren '&' und '<&>' erzeugen eine Zeichenverknüpfung.

Bei einigen Anweisungen, z.B. SR und SE, kann man Zeichenketten mit Hilfe des '&-Zeichens aneinanderreihen (konkatenieren). Dabei können auch variable TCL-Felder, bzw. Teifelder mit angereiht werden. Z.B. überträgt die Anweisung

**SR, 'Ergebnis ist'&"09"&ER,8:**

den Text 'Ergebnis ist' gefolgt von einem Tabulatorzeichen und 8 Zeichen aus dem ER-Feld an den Hostrechner, wobei die SR Anweisung zur Satztrennung automatisch noch das Steuerzeichen CR ("0D") anfügt.

Bei anderen Anweisungen, die ebenfalls über die Möglichkeit der Aneinanderreihung verfügen, ist es zusätzlich notwendig, die Aneinanderreihung mit den Zeichen '<' und '>' zu klammern. Dies ist etwa bei der K Anweisung notwendig, weil bei dieser Anweisungen die Syntax sonst nicht eindeutig aufzuschlüsseln ist. Mit

**K,<'Ergebnis ist'&"09"&ER,8>,S:**

wird der Text aus dem vorangegangenen Beispiel in das S-Feld kopiert. Auch hier werden nur die ersten 21 Speicherzellen des S-Felds verändert, die restlichen Speicherzellen bleiben erhalten.

### **2.5.3 Datenoperator ','**

Zusammen mit dem Datenoperator ',' werden Feldadressen, Teifelder und Zahlenwerte erklärt.

Die variablen TCL-Felder haben sehr verschiedene Längen. Wenn ein Feld ohne eine Längenangabe verwendet wird, ist meist der Inhalt des ganzen Felds gemeint. Wenn eine Längenangabe in einem Ausdruck oder einer Anweisung angegeben werden kann, so geschieht dies durch Angabe eines Zahlenwerts nach einem Komma. Ein Zahlenwert kann ein komplizierter Ausdruck sein. Meistens ist er eine normale, dezimale Zahlenkonstante, eine Zeichenkette oder ein anderes, variables TCL-Feld in '()' geklammert. Beispiele

```
ER,8  
ER,'8'  
ER,"08"  
ER,(EZ)
```

Man beachte, dass eine normale, mit einfachen Hochkommata abgetrennte Zeichenkette hier anders behandelt wird, wie eine hexadezimale Zeichenkette etwa bei der Kopieranweisung oder Aneinanderreihung: dort wo ein Zahlenwert erwartet wird, wird der Binärwert der hexadezimalen Zeichenkette als Zahlenwert genommen, während bei der normalen Zeichenkette eine dezimale Zahl in lesbarer (druckbarer) Form erwartet wird. In diesem Sinne sind also die Teilausdrücke, die ein Teifeld angeben

```
TF,100  
TF,'100'  
TF,"64"
```

gleichwertig, was jedoch nicht für die Verwendung als Zeichenkette gilt.

Die beiden Anweisungen

```
K,"64",D:  
K,'100',D:
```

bewirken keineswegs dasselbe.

## 2.5.4 Datenoperator '()'

Wenn ein Klammeroperator, '()', auf ein variables Feld angewendet wird, dann wird sein Inhalt in einen Zahlenwert gewandelt.

Das Feld muss dazu eine dezimale Zahl in druckbarer Darstellung enthalten.

Alle arithmetischen Anweisungen in TCL, wie die AD-, I-, D-, DI-, MU- und SB-Anweisungen, arbeiten mit druckbaren, dezimalen Zahlen und erzeugen wieder Ergebnisse als druckbare, dezimale Zahlen. Sollten binäre Operatoren notwendig werden, wie etwa die OR-, EO-, RL-, RR- und UN-Anweisungen, die auch binäre Resultate liefern, dann können diese Resultate mit Hilfe der KW-Anweisung wieder in die druckbare, dezimale Darstellung gebracht werden.

Das Feld EZ ist drei Speicherzellen groß. Mit den Anweisungen

K, '010', EZ:SR, TF, (EZ):

wird der Wert 10 in druckbarer, dezimaler Darstellung in das Feld EZ geschrieben. Es füllt das ganze Feld EZ aus. Bei Verwendung der Konstanten '10' in dem Beispiel würden nur die ersten beiden Speicherzellen des Felds EZ verändert; in der letzten Speicherzelle von EZ befände sich dann noch der vorangegangene, hier unbekannte Wert.

Mit der im Beispiel folgenden SR-Anweisung werden also die ersten zehn Speicherzellen des TF-Felds an den Host gesandt. Da auch innerhalb der '()' eine Angabe einer Feldlänge zulässig ist, könnte man auch das Beispiel so umformulieren:

K, '10', EZ:SR, TF, (EZ, 2):

wobei sich allerdings ein anderer Inhalt im EZ-Feld ergibt.

## 2.5.5 Datenoperator '+'

In vielen Fällen möchte man ein Teilstück nicht bei der ersten Speicherzelle beginnen lassen, sondern an einer beliebigen Stelle. In diesem Fall hängt man dem Feldnamen einen '+'-Operator gefolgt von einem Zahlwert, genannt Offset, an, der den Beginn des Teilstückes angibt. Offset-Angaben können wiederholt werden. Man beachte, dass die erste Speicherzelle eines Felds einen Offset von 0 hat, die zweite Speicherzelle einen Offset von 1 usw. Damit sind die Teilstücke in den beiden Angaben

TF, 2

TF+0, 2

identisch. Wie bei den Längenangaben, so können auch Zeichenketten und durch '()' geklammerte, variable TCL-Felder einen Offset-Zahlenwert bilden. Das oben schon benutzte Beispiel ließe sich mit Hilfe einer Offset-Angabe so umformulieren:

K, '10', EZ+1:SR, TF, (EZ+1, 2):

wobei sich der Inhalt von EZ wiederum unterscheidet. Diesmal ist der Wert der ersten Zelle in EZ unverändert bzw. nicht bekannt.

In folgendem Beispiel wird eine Tabelle in dem TF-Feld bei Offset 1000 angelegt. Sie sei mit Datensätzen von 80 Speicherzellen Länge gefüllt. Eine vorangegangene Operation hat den Index eines Datensatzes ermittelt und in 5-Speicherzellen Länge im ER-Feld abgelegt. Dann kopiert man mit dem Programmfragment

MU, 80, ER, 5:K, TF+(ER, (EZ))+1000, S, 80:

den gewünschten Datensatz in das S-Feld. Die MU-Anweisung in dieser syntaktischen Form multipliziert den Index in ER,5 mit 80 und legt das Ergebnis mit der geringstmöglichen Länge bzw. linksbündig im ER-Feld ab. Die Länge des Ergebnisses findet sich im EZ-Feld.

### 2.5.6 Datenoperator '//'

Eine weitere Möglichkeit einen Zahlenwert in TCL anzugeben ist der '//' -Operator, der auf ein Feld angewendet, die Länge eines Felds zum Ergebnis hat. Der Teilausdruck

/T/

ist ein Zahlenwert, der der Länge des T-Felds, 255, entspricht. Der Längenoperator wird oft bei der F-Anweisung verwendet, die dazu benutzt wird, den Wert eines Felds vorzubelegen:

F,T,/T/,0:

Wenn der Längenoperator auf einen Ringpuffer angewendet wird, dann entspricht der resultierende Zahlenwert dem freien Speicher im Ringpuffer. So ermittelt

/\$4/

den verfügbaren Platz im Notpuffer. Diese Information kann bei einem System mit hohen Performance-Ansprüchen dazu verwendet werden, die bei der SE-Anweisung vorhandenen Wartezeiten zu umgehen. Achtung: die zugrundeliegende Operation sollte mindestens 8-stellig sein.

### 2.5.7 TCL-Feldnamen

Es gibt Felder, die als Listen aufgebaut sind, wobei jedes Listenelement häufig für ein Hardwaresignal steht, das es kontrolliert oder auswertbar macht. Beispiele dafür sind die Ex-, Ox-, und Lx-Felder. Es gibt aber auch die Sx- und Dx-Felder, die sich auf die Kommunikationschnittstellen des Terminals beziehen. Hier gibt eine Dezimalzahl direkt hinter dem Feldnamen das entsprechende Listenelement an. So stehen

E0

L1

O2

für den ersten digitalen Eingang des Terminals, das zweite Ausgabesignal (die erste Leuchtdiode) und der dritte digitale Ausgang. Wie beim Offset kann der Zahlenwert ein in '()' geklammertes Teilstück sein. Hier ist aber nur ein Wert möglich; wenn ein '+' folgt, so bedeutet der folgende Operand einen Offset in das Listelement:

E(ER,(EZ))+1,1

gibt ein Teilstück an, das zu dem digitalen Eingang gehört, dessen Wert in ER mit der Länge EZ steht. Genau bezeichnet dieser Teilausdruck die zweite Speicherzelle dieses Felds, welches die Flanke für einen sogenannten DI-Sprung einstellt. Da die Ox- und Lx-Felder nur eine Speicherzelle lang sind, macht dort die Angabe eines Offsets keinen Sinn.

## 2.5.8 Datenoperator '['

Der Operator '[' wählt ein zu einer abgesetzten Eingabestation gehörendes Listenfeld aus. Aus historischen Gründen ist er innerhalb der TCL-Semantik nicht eindeutig. Im Zusammenhang mit den Ex, Lx und Ox-Feldern bedeutet die in dem Operator befindliche Zahl, die LBus-Adresse des abgesetzten Lesers. Eine LBus-Adresse 0 gibt Felder an, die zum lokalen Terminal gehören.

Die LBus-Adresse ist ein Zahlenwert. Sie kann also auch ein in '()' geklammertes Teilstück sein.

Zunächst einmal sind also die beiden Feldangaben

E0  
E[0]0

identisch. Das aus der Praxis entnommene Beispiel

E[(P20+24,2)](P20+20,2)+1,1

bezeichnet die Flankeneinstellung des digitalen Eingangs, der gerade einen DI-Ereignissprung ausgelöst hat. Beim Auslösen dieses Sprungs werden im INTUS 3000 nämlich das Teilstück P20+20,2 auf die Nummer des aktiven digitalen Eingangs gesetzt, und P20+24,2 enthält die LBus-Adresse des Geräts zu dem dieser Eingang gehört.

Weitere Beispiele sind

O[1]1  
L[2]0

der zweite Ausgang am ersten abgesetzten Leser und das erste Ausgabesignal, die Hupe, am zweiten abgesetzten Leser.

Diese Schreibweise ist in TCL nicht nur für die Angabe der LBus-Adresse reserviert. Sie wird auch in dieser Form für die Angabe einer Nummer für eine Zeitüberwachung in der T-Anweisung verwendet:

T[5]20,@20:

Weiterhin hat der normale '['-Operator auch für das D-Feld eine andere Bedeutung. Er gibt hier die Zeilennummer an, wobei 0 die erste Zeile ist:

D[1]+5,1

bedeutet das sechste Zeichen in der zweiten Zeile des Displays.

## 2.5.9 Datenoperator '['\*']'

Da aber abgesetzte Leser auch über ein Display verfügen können, muss man zur Angabe der LBus-Adresse dort hinter der öffnenden Klammer des '['-Operators ein '\*'-Zeichen einfügen. Damit bedeutet

D[\*](P20+24,2)][1]+5,1

das sechste Zeichen in der zweiten Zeile des Displays, welches sich an dem Gerät befindet, dessen Adresse in den zwei Speicherzellen des Teilstücks P20+24,2 enthalten ist.

Um Verwirrungen vorzubeugen, sei angemerkt, dass auch der um den '\*' erweiterte '['-Operator innerhalb der TCL-Sprache noch eine andere Bedeutung hat. Bei Verwendung im Zusammenhang mit den B-, M- und I-Feldern innerhalb der E-Anweisung gibt er einen oder mehrere Code-Typen für eine Leserfreigabe an, während der normale '['-Operator die LBus-Adresse(n) eines abgesetzten Lesers festlegt. Neben den B-, M- und I-Leserfeldern können mit dem '['-Operator in der Eingabeanweisung auch Funktionstasten von abgesetzten Lesern bezeichnet werden, z.B. F[1]1, die die zweite Funktionstaste am ersten abgesetzten Leser angibt. Die genaue Syntax der E-Anweisung findet sich in Abschnitt 5.14.

### 2.5.10 Datenoperator '-'

In der D-, I- und innerhalb der E-Anweisung werden Wertebereiche vorgegeben. Diese Bereichsangaben bestehen aus zwei Zahlenwerten, die eine Unter- und eine Obergrenze darstellen und mit einem '-' Operator verknüpft werden. Auch hier können die Zahlenwerte in '()' geklammerte Teifelder sein:

**I,ER,(EZ),1-(TF+1000,6):**

erhöht den Wert im Teifeld ER der Länge EZ um eins, wenn der aktuelle Wert in dem Teifeld zwischen 1 und dem Wert in dem sechs Speicherzellen großen Teifeld TF+1000,6 liegt.

Dieses Beispiel gibt nebenbei Anlass zu einer Warnung: bei der Analyse einer TCL-Anweisung kommt es oft vor, dass man nur schwer zwischen einem Parameter einer Anweisung und einer Längenangabe zu einem Teifeld unterscheiden kann, denn beide werden mit Kommata abgetrennt. Wäre im Beispiel das ganze ER-Feld durch die I-Anweisung zu bearbeiten (inkrementieren) gewesen, dann hätte man auch

**I,ER,1-(TF+1000,6):**

schreiben können, die Längenangabe ist dann nicht vorhanden und es folgt gleich der zweite (Bereichs-) Parameter der I-Anweisung. Die TCL-Syntax ist aber in jedem Fall eindeutig, auch wenn sich erst recht spät bei dem Zeichen '-' herausstellt, dass die 1 keine Längenangabe sondern die Untergrenze des Bereichsparameters ist.

### 2.5.11 Operatoren '@' und '#'

Innerhalb einer TCL-Anweisung leitet das Zeichen '@' die Angabe eines Sprungs ein. Ein unveränderlicher Sprung wird als dezimale Zahl angegeben, die beiden Angaben

**@100**

**@0100**

bewirken dasselbe: wenn dieser, eventuell bedingte, Sprung ausgeführt wird, dann wird das Programm in der Zeile fortgesetzt, die mit

**#100:**

(wegen des MONIN-Adressbytes genau genommen entweder mit 'D#100:' oder 'E#100:') beginnt.

Da der Sprung ein Zahlenwert ist, ist es möglich, mit Hilfe eines in '()' geklammerten Teifelds und eines zusätzlichen Offsets einen variablen Sprung anzugeben, etwa um eine Sprungtabelle zu implementieren, z.B.:

**@(TF+(ER,(EZ))+1000,4)+1**

### **2.5.12 Operator '%'**

Während Sprünge auch als bedingte Sprünge in TCL-Anweisungen vorkommen können, gibt es Unterprogrammaufrufe nur als separate TCL-Anweisungen. Das Fragment

`%100:`

verzweigt an das Sprungziel #100 bis eine

`%:`

Anweisung ausgeführt wird, die einen Rücksprung direkt hinter die aufrufende Anweisung bewirkt. Auch hier kann ein in '()' geklammertes Teilstück zur Implementierung eines variablen Unterprogrammsprungs verwendet werden:

`%(TF+(ER,(EZ))+1000,4):`

In einem Unterprogramm sollte es keine Stopanweisung geben, da es sonst bei einer Fortführung des Programms durch einen (unerwarteten) Ereignissprung an einer ganz anderen Stelle zu einem Überlauf des Rücksprungstacks kommen kann.

### **2.5.13 Ringpuffer**

Ein TCL-Programm kann direkt in die zwischen den einzelnen TCL-Prozessen befindlichen Datenflüsse eingreifen, indem es in Ringpuffer schreibt oder daraus liest. Außerdem kann ein Programm durch einen Ereignissprung beeinflusst werden, der auftritt, wenn Daten in einen Ringpuffer geschrieben werden.

Ein Ringpuffer wird in einer Anweisung mit dem Zeichen '\$' gefolgt von einer hexadezimalen Ziffer (A-F müssen wiederum Großbuchstaben sein) angegeben. Es ist hier als Sonderfall auch möglich, eine Nummer in einem mit '()' geklammerten Teilstück anzugeben, die auf die hexadezimale Ziffer addiert wird. Die Zahl muss in dem Teilstück in druckbaren, dezimalen Ziffern vorliegen. Beispiele

`$1`

`$C`

`$1(ER,(EZ))`

Wie oben beim '//'-Operator schon erwähnt, kann der in einem Ringpuffer verfügbare Speicherplatz ermittelt werden. So gibt

`/$4/`

den freien Speicherplatz im Notpuffer an.

## 3    **TCL - Syntax und Übersicht**

Dieser Abschnitt stellt den lexikalischen Aufbau und die Grundzüge der Syntax von TCL vor. Weiterhin wird eine Übersicht der in TCL vorhandenen Felder und Anweisungen präsentiert.

Zur Beschreibung der Syntax werden die syntaktischen Regeln angelehnt an die traditionelle Backus-Naur Form angegeben, in der syntaktische Konstrukte durch einen in '<' und '>' geklammerten Begriff bezeichnet werden.

Ein Teil einer syntaktischen Regel, der in '{' und '}' eingeschlossen ist, ist optional; ist zusätzlich ein '\*' angefügt, kann sich der optionale Teil mehrfach wiederholen.

Alternative Regelteile werden durch '|' getrennt, wobei dieser Operator die geringste Priorität innerhalb des rechten, auf '::=' folgenden Teils einer Regel hat.

Auf die Verwendung weiterer Operatoren wurde verzichtet. Damit tragen alle anderen Zeichen in den Regeln syntaktische Bedeutung.

### 3.1    **Lexikalischer Aufbau**

Ein TCL-Programm ist eine ASCII-Textdatei, deren Zeilen durch CR-Zeichen, "0D", getrennt sind. Ab TCL 4.0 kann dem CR-Zeichen ein LF-Zeichen, "0A", oder ein NUL-Zeichen, "00", folgen, das ignoriert wird.

Das erste Zeichen einer Zeile ist das MONIN-Adressbyte. Es gibt die weitere Verwendung eines Datensatzes oder einer Anweisungszeile vor. TCL-Anweisungen können auf die MONIN-Adressbytes 'E', 'D' und 'T' folgen.

**Es gelten folgende Regeln:**

- Eine TCL-Anweisung hat den Aufbau <*Anweisungsname*> { , <*Parameter*> }\* :
- Der Doppelpunkt schließt eine Anweisung ab. Die gültigen Anweisungsnamen finden sich zusammen mit einer detaillierten Beschreibung in Abschnitt 5.
- In einer Zeile dürfen mehrere TCL-Anweisungen stehen.
- Eine Zeile darf wie andere MONIN Datensätze nicht mehr als 254 Bytes Nettodaten haben.
- Innerhalb einer TCL-Anweisung stehende Leerzeichen, Tabulatoren etc. haben immer eine Bedeutung. Hinter dem abschließenden Doppelpunkt einer Anweisung folgende Leerzeichen werden ab TCL Version 2.6 ignoriert.
- Eine TCL-Anweisung kann über mehrere Zeilen gehen. Die Folgezeile muss dasselbe MONIN-Adressbyte tragen wie die erste Zeile. Von dieser Möglichkeit sollte man nur Gebrauch machen, wenn man sicherstellen kann, dass die Folgezeile ohne Unterbrechung durch andere Verarbeitungsinstanzen zum Terminal geschickt werden kann.

## 3.2 Übersicht über die Syntax

In der hier folgenden Übersicht werden nur die in TCL vorkommenden Operatoren und Operanden vorgestellt. Der syntaktische Aufbau der Anweisungen findet sich im Abschnitt 5, wo auch die Wirkung der Anweisungen beschrieben ist. Dort werden die hier folgenden Syntaxdefinitionen verwendet.

### 3.2.1 Konstanten

TCL kennt Dezimalzahlen, Hexadezimalzahlen und Zeichenketten als Konstante. Diese sind wie folgt aufgebaut:

```
<ASCII-Konstante> ::= ' <ASCII-Char> { <ASCII-Char> }* '
<Zahlenkonst>      ::= <Dezimalzahl> | ' <Dezimalzahl> ' |
                           <Hexzahl>

<Dezimalzahl>       ::= <Dezimalziffer> { <Dezimalziffer> }*
<Dezimalziffer>     ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

<Hexzahl>           ::= " <Hex-Char> { <Hex-Char> }* "
<Hex-Char>          ::= <Hexziffer> <Hexziffer>
<Hexziffer>         ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
                           | A | B | C | D | E | F
```

Ein ASCII-Char ist normalerweise ein druckbares Zeichen im Codebereich von 32 bis 126 oder von 160 bis 255. TCL lässt weitere Zeichen, außer dem Satztrennzeichen CR zu, aber es kann sein, dass diese durch Treiber konfigurationsabhängig auf dem Host oder im Terminal verschluckt werden. Deshalb wird von der Verwendung nichtdruckbarer Zeichen in einer *<ASCII-Konstante>* abgeraten.

Wie in Abschnitt 2.5 *Einführung in die syntaktischen Elemente von TCL* schon erwähnt, können eine *<Hexzahl>* oder eine *<ASCII-Konstante>* implizit in einen Zahlenwert verwandelt werden. In diesem Fall sollte die *<ASCII-Konstante>* eine druckbare Dezimalzahl enthalten. Bei Umwandlung einer *<Hexzahl>*, die mehrere Bytes lang ist, ist das erste Byte das höchswertige. Dies entspricht der Motorola-Byteordnung bzw. "Little-Endian". Diese Byteordnung wird auch in allen anderen TCL-Befehlen, die binäre Operationen ausführen, verwendet.

### 3.2.2 Zahlenwert

Zahlenwerte werden in TCL zur Angabe von LBus Adressen, Offsets, Sprungzielen usw. verwendet. Sie sind nach folgenden Regeln aufgebaut.

```

<Zahlenwert>      ::= <Zahlenkomp> { + <Zahlenkomp> }*
<Zahlenkomp>      ::= <Zahlenkonst> | / <Betrag-OP> / |
                           ( <Zahlenzeiger> )
<Betrag-OP>        ::= <Feldadr> | @ | <Ringpuffer>
<Zahlenzeiger>     ::= <Feldadr> { , <Zahlenwert> } | <Zahlenkonst>

```

Der Term /@/ ermittelt den freien Bereich im TCL Programmspeicher DL. Der Term /<Ringpuffer>/ berechnet den freien Speicherplatz in einem Ringpuffer. Der Term /<Feldadr>/ gibt die (Teil-)Feldlänge an.

Der Maximalwert von <Zahlenwert> ist  $2^{32}-1 (= 4.294.967.295)$ , auch wenn im Weiteren von 10-stelligen Operanden gesprochen wird.

### 3.2.3 Feldnamen

Variable in TCL werden mit ein- oder zweibuchstabige (Feld-) Bezeichner angesprochen. Sie können durch Index-Operatoren ergänzt werden, wenn das Feld aus einem Vektor von Unterfeldern besteht. Die folgenden Festlegungen geben alle TCL-Feldbezeichner an. Der Benutzer kann keine neuen Felder hinzu deklarieren.

```

<Feld>            ::= <Linearfeld> | <Indexfeld> | <LBusindexfeld> | <Displayfeld>
<Linearfeld>       ::= AB | AE | AN | B | CE | CV | DL | ED | ER | EV | EZ | I | KT | LS | LZ | M | MI | ND | PO | PN | S | SP | ST | T | TF | TM | TR | UR
<Indexfeld>        ::= <Indexfieldname> <Index>
<Indexfieldname>   ::= G | H | P | T | TA | TO | S | D
<LBusindexfeld>   ::= <Lname> <Index> | <Lname> [ <LBusadr> ] <Zahlenkomp>
<Lname>            ::= E | Z | O | L
<LBusadr>          ::= <Zahlenwert>
<Index>             ::= <Dezimalzahl> | ( <Zahlenzeiger> )
<Displayfeld>      ::= D { [* <LBusadr> ] } { [ <Zeile> ] }
<Zeile>             ::= <Zahlenwert>

```

Die Bedeutung und Länge des jeweiligen Felds wird in Abschnitt 4 beschrieben.

Wenn anstelle einer <LBusadr> der Wert 0 angegeben wird, so gehört das Feld zu einer Hardware (deren Zustand es anzeigt oder verändert), die sich lokal auf dem INTUS 3000 Terminal befindet. Bei einem anderen Wert für <LBusadr> bezieht sich das Feld auf eine Hardware, die sich an einem abgesetzten Leser mit dieser internen Nummer befindet.

Es gibt derzeit maximal 16 abgesetzte Leser mit den LBus-Adressen 1 bis 16. Sie teilen sich auf maximal zwei LBusse auf, wobei die Leser mit den LBus-Adressen 1 bis 8 zum ersten LBus gehören und dort auch physikalisch die Adressen 1 bis 8 haben.

Die Leser mit den LBus-Adressen 9-16 befinden sich am zweiten LBus und haben dort die physikalischen Adressen 1 bis 8.

### 3.2.4 Ringpuffer

Die zur Kommunikation zwischen TCL Prozessen dienenden Ringpuffer können durch spezielle TCL-Anweisungen ausgelesen oder beschrieben werden. Im TCL Programm werden sie wie folgt angesprochen:

```
<Ringpuffer> ::= $<Rp> { ( <Zahlenzeiger> ) }
<Rp>      ::= 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
                  A | B | C | D
```

Die Bedeutung und Länge der jeweiligen Ringpuffer wird in Abschnitt 6.1 zusammenfassend beschrieben. Außerdem sei auf Abschnitt 2 *Einführung in TCL* mit Abschnitt 2.1 und Abschnitt 2.5.13 weiter oben verwiesen.

### 3.2.5 Feldadressen

Wenn ein TCL-Feld in einer Anweisung oder durch einen Operator angesprochen wird, dann geschieht dies in der allgemeinen Form als *<Feldadr>*, der die Angabe eines Teifelds mit Startzeichenposition *<Offset>* erlaubt, oder in der Form eines *<Teilfeld>*, das zusätzlich eine Längenangabe *<Anz>* ermöglicht.

```
<Feldadr>      ::= <Feld> { + <Offset> }
<Teilfeld>      ::= <Feldadr> { , <Anz> }
<Offset>        ::= <Zahlenwert>
<Anz>          ::= <Zahlenwert>
```

### 3.2.6 Quelloperanden

Mit den syntaktischen Grundelementen der vorangegangenen Abschnitte werden im Abschnitt 5 die TCL Anweisungen beschrieben. In vielen Anweisungen kommen die beiden folgenden Operandentypen vor:

**komplexer Quelloperand:**

<b>&lt;kQop&gt;</b>	::=	<b>&lt;Zeichenverknüpfung&gt;   &lt;Feldadr&gt;   &lt;Konstante&gt;</b>
---------------------	-----	---

**einfacher Quelloperand:**

<b>&lt;eQop&gt;</b>	::=	<b>&lt;Feldadr&gt;   &lt;Konstante&gt;</b>
<b>&lt;Zeichenverknüpfung&gt;</b>	::=	<b>&lt; &lt;Zfolge&gt; { &amp; &lt;Zfolge&gt; }* &gt;</b>
<b>&lt;Zfolge&gt;</b>	::=	<b>&lt;Feldadr&gt; { , &lt;Zahlenwert&gt; }   &lt;Konstante&gt; { , &lt;Zahlenwert&gt; }</b>
<b>&lt;Konstante&gt;</b>	::=	<b>&lt;ASCII-Konstante&gt;   &lt;Hexzahl&gt;</b>

**Hilfsdefinitionen:**

Um die Syntaxbeschreibungen der TCL Anweisungen leichter lesen zu können, definieren wir außerdem

<b>&lt;Anz&gt;</b>	::=	<b>&lt;Zahlenwert&gt;</b>
--------------------	-----	---------------------------

welche für eine Zeichenanzahl oder Teilfeldlänge benötigt wird, sowie

<b>&lt;Label&gt;</b>	::=	<b>&lt;Zahlenwert&gt;</b>
----------------------	-----	---------------------------

um die Sprünge und Sprungziele von anderen Zahlenwerten abzuheben.

## 3.3 Übersicht über die TCL-Felder

In TCL werden variable Werte in TCL-Feldern gespeichert. Alle Variablen sind in Form von TCL-Feldern vordefiniert. Es ist nicht möglich, Felder per Programm-Deklarationen selbst neu- oder umzudefinieren. Hier wird eine kurze Übersicht über die in TCL vorhandenen Felder gegeben; die Felder und ihre jeweilige Funktion werden in Abschnitt 4 genau beschrieben.

Die Felder lassen sich in folgende Gruppen einteilen:

### Allgemeine Felder

Diese Felder können beliebig verwendet werden.

- TF Tabellenfeld
- S S-Feld
- T T-Feld
- Px Parameterfelder (P4 - P9 und P11 - P19)
- AN Auftragsnummer
- PN Personalnummer

### Auftragsbearbeitung

Diese Felder werden von den Anweisungen C0 und C1 verwendet. In allen anderen Fällen können sie beliebig verwendet werden.

- AB Auftragsbeginn
- AE Auftragsende
- GX Stillstandszeiten
- LZ Gesamlaufzeit
- ND Nicht definierte Stillstandszeiten

- ST Gesamtstillstandszeit
- HX Hilfsfelder (H1, H2 und H3)

### **Arithmetik**

Diese Felder werden von den arithmetischen Operationen AD, SB, MU und DI verwendet. Je nach Konfiguration in P20 wird EZ auch von der RD Anweisung verwendet.

- ER Ergebnis
- ED Ergebnisfeld Divisionsrest
- EZ Ergebnislänge
- EV Ergebnisfeld Vorzeichen

### **Hilfsfeld für die Tabellenprüf-anweisung**

- SP Offset (für die PT Anweisung)

### **Felder, die Hardware des Terminals oder der abgesetzten Eingabestation repräsentieren**

- D Display
- B Barcodeleser
- M Magnetkartenleser
- I Induktivkartenleser
- KT Datum
- UR Uhrzeit
- TM Time - Uhrzeit im 12-Stunden-Format
- Ox Digitaler Ausgang DO
- Ex - Digitaler Eingang DI
- Zx Zähler zum digitalen Eingang
- Tx Taktüberwachung (Taktüberwachung optional ab TCL Version 5.02)
- TAx Taktausfallüberwachung (Taktüberwachung optional ab TCL Version 5.02)
- TOx Taktabweichung (Taktüberwachung optional ab TCL Version 5.02)
- P0 Schlüsselschalterflag (zur Zeit nicht im INTUS 3000)

### **Kontrolle der Kommunikation mit dem Host**

- MI MONIN-Steuerfeld
- P1 Notpuffer-Statusflag
- P3 Betriebsstatusflag
- P10 MONOUT-Steuerfeld, Routinginformation
- PO Prozedurstatusflag

### **Kontrolle der Kommunikation über die zusätzlichen seriellen Schnittstellen (Kanal B, C oder D)**

- Dx DNCIN-Steuerfeld

### **Treiberkontrolle**

- LS Leser- und Terminalstatus
- P2 Maschinentaskflag
- Sx Schnittstellenparameter

### **Setup und Konfiguration**

- CV Setupparameter
- P20 Universelles Konfigurationsfeld
- P21 Setupsteuerung
- P22 Erweiterte Benutzerschnittstelle
- CE Ersetzungszeichen

#### Trace

- TR Trace

#### Programmbereich

- DL Programmbereich

## 3.4 Übersicht über die TCL-Anweisungen

Der Befehlssatz von TCL ist für die Bedürfnisse der Personal-Zeiterfassung und Betriebsdatenerfassung ausgelegt. Deshalb fehlen Anweisungen, die in anderen Programmiersprachen üblich sind. Auf der anderen Seite enthält TCL Anweisungen, die z.B. für die Einhaltung von Realzeitbedingungen unerlässlich sind. Hier wird nur eine Übersicht über die vorhandenen Anweisungen gegeben; eine genaue Beschreibung findet sich in Abschnitt 5.

Die Anweisungen lassen sich in folgende Gruppen einteilen:

#### Ablaufsteuerung

- @ Sprung
- # Sprungziel (Label)
- % Unterprogrammsprung
- P Vergleich
- RS Ringpuffer-Sprung
- S Stop
- T Timeout

#### Zuweisungen

- K Kopieren
- F Felder füllen
- KW Kopieren mit Wandeln

#### Eingabe

Die Verarbeitung der Tastatur- und Lesereingabe erfolgt mit der Anweisung: E Eingabe über Tastatur und Kartenleser

#### Datenübertragung

Mit den folgenden Anweisungen ist es möglich, Daten vom Host oder von den zusätzlichen seriellen Schnittstellen (Kanal B, C oder D) zu empfangen oder Daten an diese Schnittstellen oder den Host zu senden. Zum Senden werden die Datensätze in Ringpuffer geschrieben und empfangene Datensätze können aus Ringpuffern gelesen werden.

- WR Schreiben in Ringpuffer mit CR (satzorientiertes Schreiben)
- WO Schreiben in Ringpuffer ohne CR (zeichenorientiertes Schreiben)
- RD Ringpuffer lesen
- SE Senden an Host (über Notpuffer \$4)
- SR Senden an Host (direkt über Ringpuffer \$2)

#### Arithmetikanweisungen

- I Inkrementieren
- D Dekrementieren
- AD Addition
- SB Subtraktion
- MU Multiplikation
- DI Division

- C3 Ergebnis im ER Feld rechtsbündig ausgeben
- C0 Stillstandsdauer berechnen
- C1 Auftragsdauer berechnen
- XA Binäre Addition
- XS Binäre Subtraktion
- RL Linksshift
- RR Rechtsshift
- EO Exklusives Oder (logische Operation)
- OR Oder (logische Operation)
- UN Und (logische Operation)

### Spezielle Funktionen

- R Reset
- A Display aktualisieren
- GR Grafikfunktionen
- PT Tabellenvergleich
- PS Prüfsumme anhängen
- M1 Modulo 10/11 Fehlerprüfung
- C4 Trace ausgeben
- aus Kompatibilitätsgründen: C2 Felder Gx initialisieren
- ! Kommentar einfügen

Bevor in Abschnitt 5 die Wirkung der Anweisungen im Detail beschrieben wird, werden zuvor in Abschnitt 4 die TCL-Variablenfelder, auf die die Anweisungen wirken, vorgestellt.

Werden mit Anweisungen nicht vorhandene Schnittstellen oder abgesetzte Leser angesprochen, führt dies zu keinem Fehler, aber auch zu keiner sonstigen Reaktion.

## 4 TCL-Felder

### 4.1 AB Auftragsbeginn

AB
AB+1
----
AB+12

Das AB-Feld hat eine Länge von 13 Bytes.

Abgesehen von der Anweisung C1, Auftragsdauer berechnen, kann AB beliebig verwendet werden.

Für C1 muss AB Uhrzeit und Datum vom Auftragsbeginn in folgendem Format enthalten:

Feldaufteilung	Bedeutung	Wert
AB,2	Stunde	'00'-'23'
AB+2,1	Trennzeichen	'.'
AB+3,2	Minuten	'00'-'59'
AB+5,1	Trennzeichen	'.'
AB+6,2	Sekunden	'00'-'59'
AB+8,2	Monat	'01'-'12'
AB+10,2	Monatstag	'01'-'31'
AB+12,1	Wochentag	'0' Sonntag '1' Montag ... '6' Samstag

## 4.2 AE Auftragsende

AE
AE+1
----
AE+12

Das AE-Feld hat eine Länge von 13 Bytes.

Abgesehen von der Anweisung C1, Auftragsdauer berechnen, kann AE beliebig verwendet werden.

Für C1 muss AE Uhrzeit und Datum vom Auftragsende in folgendem Format enthalten:

Feldaufteilung	Bedeutung	Wert
AE,2	Stunde	'00'-'23'
AE+2,1	Trennzeichen	'.'
AE+3,2	Minuten	'00'-'59'
AE+5,1	Trennzeichen	'.'
AE+6,2	Sekunden	'00'-'59'
AE+8,2	Monat	'01'-'12'
AE+10,2	Monatstag	'01'-'31'
AE+12,1	Wochentag	'0' Sonntag '1' Montag ... '6' Samstag

## 4.3 AN Auftragsnummer

AN
AN+1
----
AN+9

Das AN-Feld hat eine Länge von 10 Bytes.

Dieses Feld kann beliebig verwendet werden. AN kann eine Auftragsnummer aufnehmen.

## 4.4 B Barcodeleser

Wenn eine Eingabe über Barcodeleser mit der Anweisung E freigegeben wurde, werden die Kartendaten und weitere Informationen nach einer Lesung in das B-Feld eingetragen.

Feldaufteilung (88 Byte)	Feldaufteilung (115 Byte)	Bedeutung
B,80	B,107	Datenfeld
B+80,1	B+107,1	Konfiguration
B+81,1	B+108,1	Lesefehler
B+82,1	B+109,1	Codekennung
B+83,1	B+110,1	Lesernummer
B+84,3	B+111,3	Zeichenanzahl
B+87,1	B+114,1	Leserichtung

Das Barcodeleserfeld B hat eine Länge von 88 oder 115 Bytes, je nach Einstellung im Setup bzw. im CV-Feld (CV+43,1).

### Datenfeld: B,80 oder B,107

Maximal 80 bzw. 107 Zeichen können als Daten der Lesung abgelegt werden.

### Konfiguration: B+80 oder B+107

Bevor eine Lesereingabeanweisung ausgeführt wird, muss das Konfigurationsbyte belegt sein. Es gibt an, ob ein Eingabesprung auch bei Fehllesungen erfolgen soll oder nicht:

- '0' ist die Voreinstellung. Bei korrekter Lesung wird der Sprung ausgeführt. Eine Fehllesung wird ignoriert, die Eingabefreigabe ist weiter aktiv.
- '1' Der Sprung wird auch bei Fehllesung ausgeführt, die Lesefehler- und Codekennungsbytes werden entsprechend gesetzt. Sofern die Fehllesung Daten erzeugt hat, stehen diese zur weiteren Analyse zur Verfügung.

### Lesefehler: B+81 oder B+108

Wenn eine Eingabe erfolgt und das Konfigurationsbyte auf '1' gesetzt ist, kann über das Lesefehlerbyte festgestellt werden, ob die Lesung fehlerfrei war.

- '0' Gutlesung
- '1' Fehlerfall. Das Codekennungsbyte enthält den Fehlercode.

### Codekennung: B+82 oder B+109

Bei einer gültigen Lesung enthält das Codekennungsbyte den jeweiligen Code einer Lesung.

Codekennung	Barcode
'0'	Code 2/5 Matrix
'1'	Code 2/5 Interleave
'2'	Code 39
'3'	Code 2/5 Industrial
'4'	Code 93
'?'	Unbekannter Barcode

Im Fehlerfall enthält das Codekennungsbyte den Fehlercode (siehe Abschnitt 9.4 *Fehlercodes bei Lesereingaben*).

#### **Lesernummer: B+83 oder B+110**

Die Lesernummer gibt an, von welchem Leser die Eingabe erfolgte. Da das Lesernummerfeld nur eine Speicherzelle lang ist, werden für die externen Leser nicht nur Ziffern, sondern auch Buchstaben verwendet.

Lesernummer	Leser
'0'	Interner Leser
'1'-'8'	Externer Leser am LBus1
'9', 'A'-'G'	Externer Leser am LBus2
'C'	Leser an Kanal B über DNCIN0 Prozess
'D'	Leser an Kanal C über DNCIN1 Prozess
'E'	Leser an Kanal D über DNCIN2 Prozess

Um die Nummer des Lesers, an der die Eingabe erfolgte, herauszufinden, kann auch P20+24,2 verwendet werden: Wenn eine Eingabe abgeschlossen ist, und der Eingabesprung in den Interpreteranweisungspuffer \$3 eingetragen wird, wird in P20+24,2 die Lesernummer als 2-stellige ASCII Zahl hinterlegt. Diese Information ist bis zur nächsten Stopanweisung aktuell.

#### **Zeichenanzahl: B+84,3 oder B+111,3**

Das Teilstück enthält rechtsbündig die Anzahl der gelesenen Zeichen als 3-stellige ASCII-Dezimalzahl.

#### **Leserichtung: B+87 oder B+114**

Das Leserichtungsbyte gibt die Durchzugsrichtung an.

Wenn im INTUS 3000 ein interner Barcodeleser installiert ist, der über zwei verschiedene Barcodelesegeräte, also etwa Stift und Durchzugsleser, verfügt, dann kann am Leserichtungsbyte das verwendete Lesegerät erkannt werden.

Leserichtung	Bedeutung
'0'	nicht erkannt
'1'	1. Barcodeleser Durchzug vorwärts (bei INTUS 3000 und INTUS 1600 ist vorwärts von rechts nach links, bei INTUS 2000 von links nach rechts)
'2'	1. Barcodeleser Durchzug rückwärts (bei INTUS 3000 und INTUS 1600 ist rückwärts von links nach rechts, bei INTUS 2000 von rechts nach links)
'3'	2. Barcodeleser Durchzug vorwärts (bei INTUS 3000 und INTUS 1600 ist vorwärts von rechts nach links, bei INTUS 2000 von links nach rechts)
'4'	2. Barcodeleser Durchzug rückwärts (bei INTUS 3000 und INTUS 1600 ist rückwärts von links nach rechts, bei INTUS 2000 von rechts nach links)

Bei Lesern an den seriellen Zusatzschnittstellen kann die Leserichtung nicht erkannt werden.

**Inhalt des B-Feldes nach einem Terminal-Reset**

Wird der Anlaufmodus Warm- oder Kaltstart ausgeführt, ist das Datenfeld B,80 oder B,107 mit Leerzeichen ' ' gefüllt. Die Teilstücke ab B+80 oder B+107 werden mit Nullen '0' vorbesetzt.

## 4.5 CE Ersetzungszeichen

CE
CE+1

Das CE-Feld hat eine Länge von 2 Byte.

**CE,1 Ersetzungszeichen für ein gelösches Zeichen:**

In CE,1 wird das Zeichen definiert, das bei der Tastatureingabe das Zeichen ersetzt, das mit der C-Taste gelöscht wurde.

Es ist sinnvoll, ein Eingabefeld im Display mit demselben Zeichen vorzubelegen, ehe die Eingabe erfolgt.

Die Voreinstellung ist das UNDERLINE-Zeichen '\_' ("5F").

**Ab TCL Version 6.50:****CE+1,1 Ersetzungszeichen für das Zeichenecho bei verdeckter Eingabe am INTUS 5200, INTUS 5205, INTUS 5540 und INTUS 5600:**

In CE+1,1 wird das Zeichen definiert, das bei verdeckter Tastatureingabe (siehe Abschnitt 5.14.4 *Tastatureingabe*) im Display angezeigt wird.

Die Voreinstellung ist das Zeichen '\*' ("2A").

## 4.6 CV Setupparameter

Das CV-Feld enthält die eingestellten Setupparameter.

Im Betriebshandbuch ist beschrieben, wie die Parameter manuell im Setup eingestellt werden können. Anschließend muss ein Terminal-Reset durchgeführt werden.

Das CV-Feld kann auch mit der Kopieranweisung K verändert werden. Die Daten müssen hierzu als Hexzahlen ("...") angegeben werden. Nach der Kopieranweisung muss ebenfalls ein Terminal-Reset durchgeführt werden, damit die im Feld CV eingetragenen Parameter eingestellt und im Terminal wirksam werden können. Dabei wird auch die Checksumme über das Feld CV neu errechnet. Das Terminal-Reset kann durch einen Datensatz vom Host mit Addressbyte 'R' an den MONIN Prozess ausgelöst werden (WR,\$1,'R':). Diese Schreibanweisung sollte durch einen Timeout überwacht und wiederholt werden, falls der Empfangspuffer Hostschnittstelle \$1 voll war und der Datensatz verworfen wurde.

Die Voreinstellungen der einzelnen Parameter des CV-Feldes sind bei der Beschreibung des Setup im Betriebshandbuch nachzulesen.

### Beispiel:

K,"0503000001010501",CV+1:

Dadurch wird eingestellt:

Hostschnittstelle	4800 Baud
Zusatzschnittstelle Kanal B	2400 Baud
Datenformat Hostschnittstelle	7E1
Datenformat Kanal B	7E1
Notpuffer	3KB
Tabellenfeld	3KB
Logische Quittungszeit	10 s
Anlaufmodus	Kaltstart

Die Parameterwerte sind hexadezimal angegeben.

Feldaufteilung	nur lesbar	Bedeutung	Wertebereich
CV,1	*	Softwarekonfiguration identisch mit +22	Tabelle 2a
CV+1,1		Baudrate Hostschnittstelle	Tabelle 1a
CV+2,1		Baudrate Kanal B	Tabelle 1a
CV+3,1		Datenformat Hostschnittstelle	Tabelle 1b
CV+4,1		Datenformat Kanal B	Tabelle 1b
CV+5,1		Notpuffergröße (Wert * 3072 Bytes) ab TCL Version 5.01: niederwertiges Byte in CV+5,1, höherwertiges Byte in CV+107,1	"01" 3 KB ... "FF" 765 KB siehe <i>Bemerkung 2</i>
CV+6,1		Tabellenfeldgröße (Wert * 3072 Bytes) ab TCL Version 5.01: niederwertiges Byte in CV+6,1, höherwertiges Byte in CV+108,1	"01" 3 KB ... "FF" 765 KB siehe <i>Bemerkung 2</i>
CV+7,1		logische Quittungszeit für MONOUT Prozess (Wert * 2 s)	"01" 2 s .... "73" 230 s
CV+8,1		Anlaufmodus	Tabelle 2b
CV+9,1		Verkabelung der Zutrittskontrollmanager INTUS ACM8, INTUS ACM8(0)e Rack, INTUS ACM8(0)e Wand (ab TCL Version 5.5), INTUS ACM4, INTUS ACM40 und INTUS 3000 ACM (ab TCL Version 6)	<u>Bitfeld:</u> <u>Position:</u> Bit 0: reserviert Bit 1: LBus1 Bit 2: LBus2 <u>Wertebereich:</u> 0: MP, Multipoint 1: PP, Point-To-Point Default für INTUS ACM8(e) : "02" Default für INTUS ACM4(0): "06" Default für INTUS 3000 ACM: "00" Siehe Kapitel 11
CV+10,1 ... CV+16,1		Einstellungen zum Protokoll (zur Prozedur) der Hostschnittstelle (CV+31,1)	Tabelle 1e: TTY Tabelle 1f: BSC
CV+17,5		Nur INTUS 2000: Einstellungen zum HDLC-Master	
CV+22,1	*	Softwarekonfiguration identisch mit CV+0	Tabelle 2a
CV+23,1	*	Leserkonfiguration	Tabelle 2c
CV+24,1	*	Hardwarekonfiguration	Tabelle 2
CV+25,1	*	Displayvariante	Tabelle 2e

Feldaufteilung	nur lesbar	Bedeutung	Wertebereich
CV+26,1	*	Tastaturvariante	Tabelle 2f
CV+27,1	*	Digitale I/O (DI/DO)	Tabelle 2g
CV+28,1	*	Interface Kanal A/C	Tabelle 1c
CV+29,1	*	Interface Kanal B	Tabelle 1c
CV+30,1	*	RAM Bereich	Tabelle 2h
CV+31,1	Ab TCL6.0 schreibbar	Protokoll (Prozedur) Kanal A	Tabelle 1d
CV+32,1	Ab TCL6.0 schreibbar	Protokoll (Prozedur) Kanal B	Tabelle 1d
CV+33,1	Ab TCL6.0 schreibbar	Protokoll (Prozedur) Kanal C	Tabelle 1d
CV+34,1	Ab TCL6.0 schreibbar	Protokoll (Prozedur) Kanal D	Tabelle 1d
CV+35,1		Baudrate Kanal C	Tabelle 1a
CV+36,1		Baudrate Kanal D	Tabelle 1a
CV+37,1		Datenformat Kanal C	Tabelle 1b
CV+38,1		Datenformat Kanal D	Tabelle 1b
CV+39,1		Zeichensatz im Display, national oder international	Tabelle 2i
CV+41,1		8860 Modus (nicht für INTUS 3000)	"00" nein "01" ja
CV+42,1		Interface Kanal D (ab TCL Version 5.5)	Tabelle 1c
CV+43,1		Feldgröße B/M/I-Feld	"00" 115 Byte "01" 88 Byte
CV+44,1		Logische Satznummer für MONOUT Prozess	"00" nein "01" ja
CV+46,1		interner Leser	Tabelle 2j
CV+47,1		externer Leser (am LBus1) mit <LBusadr> 1	Tabelle 2l und für INTUS ACM4(0) Wiegand <i>Bemerkung 4</i>
CV+49,1		externer Leser (am LBus1) mit <LBusadr> 2	Tabelle 2l und für INTUS ACM4(0) Wiegand <i>Bemerkung 4</i>
CV+51,1		externer Leser (am LBus1) mit <LBusadr> 3	Tabelle 2l

Feldaufteilung	nur lesbar	Bedeutung	Wertebereich
CV+53,1		externer Leser (am LBus1) mit <LBusadr> 4	Tabelle 2l
CV+55,1		externer Leser (am LBus1) mit <LBusadr> 5	Tabelle 2l
CV+57,1		externer Leser (am LBus1) mit <LBusadr> 6	Tabelle 2l
CV+59,1		externer Leser (am LBus1) mit <LBusadr> 7	Tabelle 2l
CV+61,1		externer Leser (am LBus1) mit <LBusadr> 8	Tabelle 2l
CV+63,1		Anzahl der TCL Sprungziele (Wert *256 +512)	"00" 512 ... "0F" 4352
CV+64,1	*	Anzahl DIs im INTUS 3000	"00" kein DI "01"- "25" größter DI-Index + 1
CV+65,1	*	Anzahl DOs im INTUS 3000	"00" kein DO "01"- "25" größter DO-Index + 1
CV+66,1		Virtuelles Display (Bit 1) (nicht für INTUS 3000)	0 deaktivieren 1 aktivieren
CV+67,1		EEPROM-TCL laden beim Eiskalt-, Com- und Kaltstart (Bit 0)	0 nein 1 ja
		EPROM-TCL laden beim (Eis-) Kaltstart (Bit 2) (nicht für INTUS 3000)	0 nein 1 ja
CV+68,2		Terminaladresse zur Identifikation im INTUS-Net, Teilfeld wird von OEM-Partner genutzt	'00' - '99'
CV+70,6		Setup-Passwort ab TCL Version 5.5 für Ebene 1	'000000'- '999999' Default: '111111'
CV+76,1	*	INTUS 3000 Terminal-Typ	Tabelle 2k
CV+77,1	*	INTUS 3000 Layout-Version	Tabelle 2m
CV+78,2	*	aktuelle Zeilenanzahl im Display (immer aktuell)	'02',...,'25' '02' wenn ohne Display
CV+80,3	*	aktuelle Spaltenanzahl im Display (immer aktuell)	'030',...,'080' '040' wenn ohne Display
CV+83,1		externer Leser (am LBus2) mit <LBusadr> 9	Tabelle 2l und für INTUS ACM4(0) Wiegand <i>Bemerkung 4</i>

Feldauftteilung	nur lesbar	Bedeutung	Wertebereich
CV+84,1		externer Leser (am LBus2) mit <LBusadr> 10	Tabelle 21 und für INTUS ACM4(0) Wiegand <i>Bemerkung 4</i>
CV+85,1		externer Leser (am LBus2) mit <LBusadr> 11	Tabelle 21
CV+86,1		externer Leser (am LBus2) mit <LBusadr> 12	Tabelle 21
CV+87,1		externer Leser (am LBus2) mit <LBusadr> 13	Tabelle 21
CV+88,1		externer Leser (am LBus2) mit <LBusadr> 14	Tabelle 21
CV+89,1		externer Leser (am LBus2) mit <LBusadr> 15	Tabelle 21
CV+90,1		externer Leser (am LBus2) mit <LBusadr> 16	Tabelle 21
CV+91,8		Zutrittskontrollmanager: Zuordnung der Leser am LBus1 zu Steckplatz/Kanal (1 Byte je Leser, enthält Steckplatznummer)	"01"- "08" pro Eintrag, siehe <i>Bemerkung 1</i>
CV+99,8		Zutrittskontrollmanager: Zuordnung der Leser am LBus2 zu Steckplatz/Kanal (1 Byte je Leser, enthält Steckplatznummer)	"01"- "08" pro Eintrag, siehe <i>Bemerkung 1</i>
CV+107,1		Notpuffergröße, höherwertiges Byte in (3072*256) Bytes	"00"- "FF", siehe <i>Bemerkung 2</i>
CV+108,1		Tabellenfeldgröße, höherwertiges Byte in (3072*256) Bytes	"00"- "FF", siehe <i>Bemerkung 2</i>
CV+109,1	*	Display kann ISO 8859-1 Latin-I Zeichensatz (siehe Abschnitt 7)	'0' nur national '1' international
CV+110,5	*	Größe des Speicherausbaus in kB	derzeit '00000'- '04252', siehe <i>Bemerkung 3</i>
CV+115,1		(nicht verwendet)	

## Ab TCL Version 5.5:

Feldauftteilung	nur lesbar	Bedeutung	Wertebereich
CV+116,8		Setup-Passwort für Ebene 2 Siehe Abschnitt 6.4.4	ASCII-Zeichen Default: '14789632'
CV+124,8		Setup-Passwort für Ebene 3 Siehe Abschnitt 6.4.4	ASCII-Zeichen Default: '14589632'

CV+132,1		Login auf der Hostschnittstelle Siehe Abschnitt 6.4.1	'0' deaktiviert (Voreinstellung) '1' aktiviert (Login-Meldungen im MONOUT-Format) '2' aktiviert (Login-Meldungen unformatiert)
CV+133,8		Passwort für einfachen Zugriff auf der Hostschnittstelle Siehe Abschnitt 6.4.1	ASCII-Zeichen
CV+141,8		Passwort für administrativen Zugriff auf der Hostschnittstelle Siehe Abschnitt 6.4.1	ASCII-Zeichen
CV+149,8		Routing-Bytes für Login-Meldungen Siehe Abschnitt 6.4.1	ASCII-Zeichen Default: '     '(8 Leerzeichen)
CV+157,1		Ersetzungszeichen für Satznummer Siehe Abschnitt 6.4.1	ASCII-Zeichen Default: '*'
CV+159,1		Verschlüsselung auf der Hostschnittstelle Siehe Abschnitt 6.4.2	'0' deaktiviert (Voreinstellung) '1' aktiviert
CV+160,16		Schlüssel für die Hostschnittstelle Siehe Abschnitt 6.4.2	binär
CV+176,8		Verschlüsselung aktivieren für an LBus1 angeschlossene INTUS 1600-II, INTUS 600, 500, 400, INTUS 600 FP oder INTUS 350H Siehe Abschnitt 6.4.3	je Eintrag: '0' deaktiviert (Voreinstellung) '1' aktiviert
CV+184,16		Schlüssel für LBus1 Siehe Abschnitt 6.4.3	binär
CV+200,8		Verschlüsselung aktivieren für an LBus2 angeschlossene INTUS 1600-II, INTUS 600, 500, 400, INTUS 600 FP oder INTUS 350H Siehe Abschnitt 6.4.3	je Eintrag: '0' deaktiviert (Voreinstellung) '1' aktiviert
CV+208,16		Schlüssel für LBus2 Siehe Abschnitt 6.4.3	binär

CV+224,1		Einfache Adressierung für LBus1 und LBus2 einschalten (Wenn die Verkabelung des LBus point-to-point ist (CV+9,1), können alle Leser am LBus die HardwareAdresse 1 haben.)	<u>Bitfeld</u> :  <u>Position</u> : Bit 0: reserviert Bit 1: LBus1 Bit 2: LBus2  <u>Wertebereich</u> : 0: keine einfache Adressierung 1: einfache Adressierung Default für INTUS ACM8(e): "02" Default für INTUS ACM4(0): "06" Default für INTUS 3000 ACM: "00"
CV+226,2		Wartungsgruppe, der das Terminal angehört. Teilfeld ist schreibbar, aber nicht lesbar. Ausgabe von '***'	"0000"- "FFFF" Default: "0000"

**Ab TCL Version 5.9:**

Feldaufteilung	nur lesbar	Bedeutung	Wertebereich
CV+228,3	*	Terminalmodell	Tabelle 2n
CV+231,2	*	Anzahl Leserlizenzen (Zutrittskontrollmanager INTUS ACM8(0)e Rack und Wand, INTUS 5300, INTUS 5200/5500/5540/5600)	'00' – '16'
CV+233,1		Helligkeit des MagicEyes (nur INTUS 5300)	"00"- "0F" Default: "09"
CV+234,1		Backlight-Saver deaktiviert (nur INTUS 5300)	'0' Backlight wird nach 1 Stunde dunkel geschaltet (Voreinstellung) '1' Backlight immer mit voller Helligkeit (Achtung, verringert die Lebensdauer)

**Ab TCL Version 6.0:**

Feldaufteilung	nur lesbar	Bedeutung	Wertebereich
CV+235,1		Konfiguration Ethernet-Link	Tabelle 2o
CV+236,1	*	Status Ethernet-Link	Tabelle 2o
CV+237,1		Hupton INTUS 3300 / 3500	"00"- "FF"

CV+238,1		Hupenansteuerung	<p><u>Bitfeld:</u></p> <p><u>Position:</u> Bit 0: Funktionstasten Bit 1: andere Tasten</p> <p><u>Wertebereich:</u> 0: kein Keyclick 1: Keyclick</p> <p><u>Position:</u> Bit 4: Hupe (L0) während des Betriebs (unter Kontrolle des TCL-Programms)</p> <p><u>Wertebereich:</u> 0: aktiv 1: deaktiviert Voreinstellung: "00"</p>
----------	--	------------------	--

**Ab TCL Version 6.10:**

Feldauftteilung	nur lesbar	Bedeutung	Wertebereich
CV+239,1		Quittierungstaster INTUS ACM4 AKKU	"00" Meldung über E[0]3, "01" Meldung als F[0]21 Voreinstellung: "00" Mit der Meldung als Funktions-taste F[0]21 wird der INTUS ACM4 AKKU kompatibel zum INTUS ACM40 AKKU und den INTUS ACM8(e) Model- len.
CV+240,1		reserviert	nicht verändern!

**Ab TCL Version 6.50:**

Feldauftteilung	nur lesbar	Bedeutung	Wertebereich
CV+241,10		Offset für Binding zwischen Teifeld des TF-Feldes und er-weiterter Benutzerschnittstelle	'0000000000' - '9999999999'
CV+251,10		Länge für Binding zwischen Teifeld des TF-Feldes und er-weiterter Benutzerschnittstelle	'0000000000' - '9999999999'

**Ab TCL Version 6.62:**

Feldaufteilung	nur lesbar	Bedeutung	Wertebereich
CV+261,1	*	INTUS 5200: RW-Leserlizenz vorhanden Interner und externer Leser kann LBus Lese- und Schreibkommandos ausführen. Ohne RW-Leserlizenz wird der Fehler „BSCM-TX:403“ generiert	'0' nein '1' ja

Das Teilstück CV+241,21 ist in den TCL Versionen TCL 6.20 bis < 6.50 nicht verfügbar.

**Ab TCL Version 6.20 und 6.70:**

Feldaufteilung	nur lesbar	Bedeutung	Wertebereich
CV+262,1	Nicht schreibbar und lesbar	PIN-Code Verifikation Modus	'0' Algorithmus basierend auf Elliptischen Kurven (ECIES, Curve 25519)
CV+263,44	Schreibbar, nicht lesbar	PIN-Code Verifikation privater Schlüssel im Base64 Format	<ASCII-Konstante>
CV+307,1	*	Terminal stellt PIN-Code Verifikation zur Verfügung	'0' nein '1' ja

Die PIN-Code Verifikation ist syntaktisch in die P-Anweisung integriert, siehe Abschnitt 5.24.

Die nicht aufgeführten Teilstücke bei den geraden Offsets zwischen CV+48,1 und CV+62,1 hatten für ältere INTUS 2000 Releases die Bedeutung des Lesertyps der externen Leser. Schon seit TCL Version 2.6 ist der Lesertyp der externen Leser eine Kopie des Felds CV+46,1. Eine Veränderung dieser Felder wird ignoriert.

Weitere nicht aufgeführte Teilstücke sind für zukünftige Erweiterungen reserviert.

## Tabelle 1a-1f für die Kommunikationshardware

**Tabelle 1a****Baudrate CV+1, CV+2, CV+35, CV+36**

Baud	Hexzahl	Baud	Hexzahl
50	"0A"	75	"0B"
110	"08"	134	"0C"
150	"0D"	200	"0E"
300	"00"	600	"01"
1200	"02"	1800	"0F"
2400	"03"	4800	"05"
9600	"06"	19200	"07"
38400	"10"		

**Tabelle 1b****Datenformat CV+3, CV+4, CV+37, CV+38**

Datenformat	Hexzahl	Datenformat	Hexzahl
7E1	"00"	7E2	"01"
7O1	"02"	7O2	"03"
8N2	"04"	8N1	"05"
8E1	"06"	8O1	"07"
7N1	"08"	7N2	"09"
8E2	"0A"	8O2	"0B"

Datenformat Darstellung: 1.Zahl für Datenbits, E oder O oder N (für even oder odd oder no parity), 2.Zahl für Stopbits.

**Tabelle 1c**

**Interface der Schnittstellen Kanal A/C (CV+28), Kanal B (CV+29) und ab TCL Version 5.5 Kanal D (CV+42)**

Wert	Interface
"00"	V24, RS232
"01"	Für INTUS 2000: 20 mA
"02"	RS485/RS422 (4-Draht und 2-Draht für INTUS 3000), bzw. RS485/RS422 (4-Draht für INTUS 2000)
"03"	2-Draht (nur INTUS 2000)
"04"	Für INTUS 2000: Modemmodul
"05"	Für INTUS 2000: Inhouse
"07"	nicht vorhanden (auch bei Verwendung als LBus)
"08"	Für INTUS 3000: 4DI-Modul vorhanden

**Tabelle 1d**

**Protokoll (Prozedur) der Hostschnittstelle (CV+31) und der Kanäle B bis D (CV+32, CV+33, CV+34)**

<b>Wert</b>	<b>Protokoll</b>
"00"	PTY (Zeichenstrom Modus mit über Setup einstellbarer Datenflussteuerung)
"01"	BSC
"02"	Für INTUS 2000: HDLC Für INTUS 3000 Kanal B und/oder Kanal C: 4DI-Modul Behandlung
"03"	LSV (derzeit nicht für INTUS 3000)
"05"	TCP/IP (nur Hostschnittstelle)

**Tabelle 1e**

**Einstellungen für die TTY-Version**

<b>Feldaufteilung</b>	<b>Bedeutung</b>	<b>Wertebereich</b>
CV+10,1	S2/M2 (RTS/CTS) aktiv	"00" ja "01" nein
CV+13,1	XON/XOFF Hostschnittstelle	"00" ein "01" aus "02" Senden aus "03" Empfang aus

**Tabelle 1f**

**Einstellungen für die BSC-Version**

<b>Feldaufteilung</b>	<b>Bedeutung</b>	<b>Wertebereich</b>
CV+10,1	Terminal Group ID	"40" @ "41"- "5A" A-Z
CV+11,1	Terminal Device ID	"40" @ "41"- "5A" A-Z
CV+13,1	Sendeverzögerung (ms)	"00"- "FF" 0-255
CV+14,1	Daten Timeout (* 100ms)	"01"- "FF" 1-255
CV+15,1	Poll Timeout (s)	"01"- "FF" 1-255
CV+16,1	Trailing Pads	"00"- "09" 0-9

## Tabelle 2a-2k für die Terminalkonfiguration

### Tabelle 2a Softwarekonfiguration des Terminals CV+22

Der Inhalt von CV+22 und CV+0 ist identisch.

Bit	Softwarekonfiguration
0	Konfigurierbarer Treiber aktiviert (INTUS 2000)
1	Bitbus-Treiber aktiviert (INTUS 2000)
2	nationale Sprache im EEPROM geladen
3	Schlüsselschalter nicht vorhanden (INTUS 2000)
4	DHCP Protokoll verfügbar
5	Taktüberwachung verfügbar
6	Sound Modul verfügbar (INTUS 5200/5500/5540/5600) (ab TCL Version 6.60)
7	reserviert

### Tabelle 2b Anlaufmodus des Terminals CV+8

Bit	Bedeutung	Wertebereich
0-2	TCL-Anlaufmodus	"00" Warm "01" Kalt "02" Neu "03" Eiskalt "04" Comstart (ab TCL Version 6)
3-4	reserviert	
5	INTUS Sound: Audio löschen zusätzlich zum TCL-Anlaufmodus, nur für INTUS 5200, 5500, 5540, 5600 (ab TCL Version 6.60)	Zum Löschen mit "20" odern.
6	Erweiterte Benutzerschnittstelle: Masken/Logo löschen zusätzlich zum TCL-Anlaufmodus, nur für INTUS 5200, 5205, 5540, 5600 (ab TCL Version 6.50)	Zum Löschen mit "40" odern.
7	Linux-Systemstart zusätzlich zum TCL-Anlaufmodus (nur für INTUS ACM40, INTUS 3460, INTUS 3660, INTUS 5200, INTUS 5205, INTUS 5500, INTUS 5540, INTUS 5600, INTUS ACM80e Rack und Wand)	Für Linux-Systemstart mit "80" odern.

**Tabelle 2c****Leserkonfiguration des Terminals CV+23**

Werte für INTUS 2000,

CV+23 wird ab TCL 6.10 nicht mehr unterstützt, Wert ist immer "00".

<b>Bit</b>	<b>Lesertyp des internen Lesers</b>
0-4	Werte wie in CV+46 (siehe Tabelle 2j)
5	Barcode Schnittstelle aktiviert
6	Barcodeleser vorhanden/ nicht vorhanden
7	reserviert

**Tabelle 2d****Hardwarekonfiguration des Terminals CV+24**

<b>Bit</b>	<b>Hardwarekonfiguration</b>
1, 0	Buserweiterung Steckerkonfiguration (INTUS 2000) 00 : kein Anschluss 01 : 8 DI/DOs 10 : 3. und 4. serielle Schnittstelle 11 : 3. und 4. serielle Schnittstelle + 4 DI/DO's
3, 2	Layoutstand Basisplatine 00 : INTUS 2000, B-Layout 01 : INTUS 2000, C-Layout 10 : INTUS 3000, tatsächliche Layoutversion in CV+77,1 11 : INTUS timer
4	SLA Platine vorhanden (INTUS 2000) AKKU vorhanden (INTUS ACM4 Akku, INTUS ACM40 Akku)
5	DIDO Erweiterung vorhanden (INTUS 2000)
6	AS400 Controller vorhanden (INTUS 2000)
7	ETH Controller vorhanden

**Tabelle 2e**  
**Displayvariante des Terminals CV+25**

Wert	Displayvariante
"00"	INTUS 2000 25-zeilig
"01"	INTUS 3000 und INTUS 2200 2*40 Zeichendisplay
"02"	INTUS 2400 6-zeilig
"04"	INTUS Timer 1-zeilig
"05"	INTUS 2600 Grafikdisplay (25-zeilig)
"06"	INTUS 2400/LCD (8-zeilig)
"07"	INTUS 3000 240*64 Pixel Grafikdisplay 8-zeilig
"08"	INTUS 3000 320*240 Pixel Grafikdisplay
"09"	INTUS 3000 2*40 Zeichendisplay (ISO 8859-1 (Latin-1) Zeichensatz)
"0A"	INTUS 3100 , ACM4, ACM40 2*20 Zeichendisplay
"0B"	kein Display vorhanden
"0C"	INTUS Graph 640x480
"0D"	INTUS 5200/5205 320*240 Pixel TFT Display
"0E"	INTUS 5540 480*272 Pixel TFT Display

**Tabelle 2f**  
**Tastaturvariante des Terminals CV+26**

Wert	Tastaturvariante
"00"	INTUS 3000 Folientastatur "Timer-Layout" : 5 Funktionstasten oder 7 Funktionstasten bei INTUS 5300, INTUS 5500, INTUS 5540 INTUS 2000 Tastatur V1/V2
"01"	INTUS 2000 Tastatur V3
"02"	INTUS 2000 Tastatur V4
"03"	INTUS 2000 Tastatur V5
"04"	INTUS 2000 Tastatur V6
"05"	INTUS 2000 DIN
"07"	INTUS 5600 Touch, INTUS 5200/5205 Touch
"08"	INTUS 3000 Matrix-Touch Tastatur
"09"	keine Tastatur
"0A"	INTUS 3000 Folientastatur mit 6 Funktionstasten
"0B"	INTUS 5200 Touch und Folientastatur

**Tabelle 2g**  
**Digitale I/O des Terminals CV+27**

Siehe auch die Liste der Terminalmodelle: Tabelle 2n

<b>Wert</b>	<b>DI/DO Modul</b>
"00"	INTUS 3300 ethertime: 0 DI, 0 DO INTUS 2000: 6 DI, 2 DO
"01"	INTUS 3100, INTUS 3200 (ethertime): 2 DI, 1 DO INTUS 3300, INTUS 3450-plus, INTUS 3460-plus, INTUS 3500, INTUS 3600, INTUS 3660: 2 DI, 2 DO (für zusätzliche DI-Aufsteckmodule siehe CV+28 und CV+29) INTUS 5500, 5540, 5600 mit IO-Option: 2 DI, 2 DO, Steckplätze Kanal B und C (siehe CV+28, CV+29)
"03"	INTUS 3105, INTUS 3450-time, INTUS 3460-time, INTUS 5205, INTUS 5200 ohne IO-Option: 0 DI, 0 DO
"05"	INTUS 3000 ACM
"06"	INTUS ACM4, INTUS ACM40
"07"	INTUS 3400, INTUS 3450, INTUS 3450-timeplus, INTUS 3460-timeplus: 2 DI, 1 DO
"08"	INTUS ACM8, INTUS ACM8(0)e Rack, INTUS ACM8(0)e Wand
"09"	INTUS ACM4 Wiegand, INTUS ACM40 Wiegand
"0B"	INTUS 5300, INTUS 5200 mit IO-Option: 2 DI, 1 DO
"0C"	INTUS 5300 PoE: 0 DI, 0 DO INTUS 5500, 5540, 5600 ohne IO-Option: 0 DI, 0 DO, ohne Steckplätze für Kanal B und C

Ab TCL Version 4 stehen CV+64,1 und CV+65,1 mit Angaben zu den digitalen Ein- und Ausgängen zur Verfügung.

**Tabelle 2h**  
**RAM Bereich des Terminals CV+30**

<b>Wert</b>	<b>RAM Bereich</b>
"01"- "03"	*256 KB für INTUS 2000
"02"	INTUS 3000 in der Basisausstattung
"03"	INTUS 3000 mit Speichererweiterung

Die tatsächliche Speichergröße kann in CV+110,5 ausgelesen werden.

**Tabelle 2i****Nationaler oder internationaler Zeichensatz im Display CV+39**

<b>Wert</b>	<b>Zeichensatz im Display</b>
"00"	ISO 646 Großbritannien, UK 7-Bit
"01"	ISO8859-1, US-ASCII
"02"	ISO 646 Deutschland, Deutsch 7-Bit
"03"	ISO 646 Frankreich, Französisch 7-Bit
"04"	ISO 646 Spanien, Spanisch 7-Bit
"05"	ISO 646 Norwegen, Norwegisch 7-Bit
"06"	ISO8859-5
"07"	ISO8859-2
"08"	ISO8859-15

In Abschnitt 10 sind alle TCL Zeichensätze vollständig aufgelistet.

Abschnitt 7.3 enthält weitere Informationen zu Zeichensätzen und Zeichenmapping.

**Tabelle 2j****Interner Leser CV+46**

Diese Werte sind nur für INTUS 3000 gültig.

Für INTUS 2000 gibt es andere Leser bzw. andere Werte.

<b>Bit</b>	<b>Interner Leser</b>
6, 5, 4	<b>Lesertyp</b> 000: Standardleser: serielle Anbindung oder Takt-Daten Anbindung 001: freie serielle Anbindung 010: serielle Anbindung des FEIG-Lesers
3	zusätzlicher Barcode
2, 1, 0	<b>Lesermodus, abhängig vom Lesertyp</b> Standardleser: serielle Anbindung oder Takt-Daten Anbindung <ul style="list-style-type: none"> <li>000 : Magnetkarte Spur 1</li> <li>001 : Magnetkarte Spur 2, Leser mit OMRON-Emulation: Proximityleser, z.B. Legic, Hitag, Mifare Speicherchipkarten-Interface Wiegand mit Adapterplatine (4-polige Klemme/B3100-010)</li> <li>010 : Magnetkarte Spur 3</li> <li>011 : Wiegand mit Adapterplatine (6-polige Klemme/B3100-006)</li> <li>100 : Barcode</li> <li>101: Leser mit BSC Protokoll: Proximityleser, z.B. Legic advant, Mifare DESFire EV1, Hitag Speicherchipkarte, immer einstellen, wenn Lesercontroller eingebaut ist.</li> </ul> Freie serielle Anbindung <ul style="list-style-type: none"> <li>001 : serielle Leserschnittstelle frei, Kommunikation über DNCIN</li> </ul> Serielle Anbindung des FEIG-Lesers <ul style="list-style-type: none"> <li>000: Lesung: FEIG ID bevorzugt, sonst Seriennummer</li> <li>001: Seriennummernlesung</li> <li>010: FEIG ID Lesung</li> <li>011: Lesung: Seriennummer und FEIG ID</li> <li>100: Datenblocklesung</li> <li>101: Lesung: Seriennummer und Checksumme</li> </ul>

Dieser Wert wird in die Teilsteller mit geraden Offsets im Bereich von CV+48,1 bis CV+62,1 dupliziert.

**Tabelle 2k****INTUS 3000 Terminal-Typ CV+76**

Siehe auch die Liste der Terminalmodelle: Tabelle 2n

<b>Wert</b>	<b>INTUS 3000 Terminal-Typ</b>
'A'	INTUS ACM8, INTUS ACM8(0)e Rack, INTUS ACM8(0)e Wand
'B'	INTUS 3600, INTUS 3660
'P'	INTUS 3300, INTUS 3500, INTUS 3400, INTUS 3450, INTUS 3450-plus, INTUS 3460-plus, INTUS 5500, 5540, 5600 mit IO-Option
'T'	INTUS 3300 ethertime, INTUS 5500, 5540, 5600 ohne IO-Option
'Z'	INTUS 3000 ACM
'a'	INTUS ACM4, INTUS ACM40
'p'	INTUS 3100, INTUS 3200, INTUS 3450-time, INTUS 3450-timeplus, INTUS 3460-time, INTUS 3460-timeplus, INTUS 5300, INTUS 5200 mit IO-Option
't'	INTUS 3200 ethertime, INTUS 5205, INTUS 5200 ohne IO-Option

**Tabelle 2l****Externer Leser CV+47-CV+61, CV+83-CV+90**

<b>Wert</b>	<b>Externer Leser</b>
"00"	INTUS LBus Modus A (INTUS 1500)
"05"	nicht vorhanden
"06"	INTUS 300h, INTUS 340h Leser: ID/SN-Modus
"07"	INTUS 300l/m, INTUS 305l Leser
"08"	INTUS LBus Modus B (INTUS 1600, INTUS 600, INTUS 500, INTUS 400, INTUS 350H, INTUS FP, INTUS 600 FP)
"09"	INTUS 300ro
"0A"	INTUS LBus Modus C (INTUS 1600-II V2.5)
"0B"	INTUS 300h, INTUS 340h Leser: SN+ID-Modus (ab TCL Version 5.9)
"0C"	INTUS 300h, INTUS 340h Leser: SN-Modus (ab TCL Version 5.9)
"0D"	INTUS 300h, INTUS 340h Leser: ID-Modus (ab TCL Version 5.9)

Für den INTUS ACM4 Wiegand und INTUS ACM40 Wiegand siehe auch Bemerkung 4.

**Tabelle 2m****INTUS 3000 Layout-Version CV+77**

<b>Wert</b>	<b>Layout-Version</b>
'1', ..., '9'	1, ..., 9
'A', ..., 'Z'	10, ..., 35
'a', ..., 'z'	36, ..., 61

**Tabelle 2n****Terminalmodelle CV+228,3**

Eindeutiger 3-stelliger ASCII-Wert, der eine Analyse von Terminaltyp in CV+76,1 und digitaler I/O des Terminals in CV+27,1 erübrigt.

<b>Wert</b>	<b>Terminalmodell</b>	<b>CV+76,1</b>	<b>CV+27,1</b>	<b>Anmerkung</b>
'000'	INTUS 3300	'P'	"01"	
'001'	INTUS 3500	'P'	"01"	
'002'	INTUS 3300 ethertime	'T'	"00"	
'003'	INTUS 3000 ACM	'Z'	"05"	
(kein CV-Feld)	INTUS 3000 Server			
'004'	INTUS 3200 ethertime	'p', 't'	"01"	
'005'	INTUS 3100	'p'	"01"	
'006'	INTUS ACM4	'a'	"06"	
'007'	INTUS ACM4 Akku	'a'	"06"	
'008'	INTUS 3105	'p'	"03"	
'009'	INTUS 3400	'P'	"07"	
'010'	INTUS ACM4-II	'a'	"06"	CV+77 >= 'b'
'011'	INTUS ACM4-II Akku	'a'	"06"	CV+77 >= 'b' CV+24 Bit 4 gesetzt
'012'	INTUS ACM8	'A'	"08"	
'013'	INTUS ACM4-II Wiegand	'a'	"09"	
'014'	INTUS ACM4-II Akku Wiegand	'a'	"09"	CV+24 Bit 4 gesetzt
'015'	INTUS 3450	'P'	"07"	
'016'	INTUS 3450-time	'p'	"03"	
'017'	INTUS 3450-timeplus	'p'	"07"	
'018'	INTUS 3450-plus	'P'	"01"	
'019'	INTUS 3600	'B'	"01"	
'020'	INTUS ACM8e Rack	'A'	"08"	
'021'	INTUS ACM8e Wand	'A'	"08"	
'024'	INTUS 5300	'p'	"0B"	
'025'	INTUS 5300 PoE	'p'	"0C"	
'026'	INTUS ACM40	'a'	"06"	
'027'	INTUS ACM40 Akku	'a'	"06"	CV+24 Bit 4 gesetzt
'028'	INTUS ACM40 Wiegand	'a'	"09"	
'029'	INTUS ACM40 Akku Wiegand	'a'	"09"	CV+24 Bit 4 gesetzt
'030'	INTUS 3460-time	'p'	"03"	

'031'	INTUS 3460-timeplus	'p'	"07"	
'032'	INTUS 3460-plus	'P'	"01"	
'033'	INTUS 3660	'B'	"01"	
'034'	INTUS 5500 NT	'P' 'T'	"01" "0C"	mit IO-Option ohne IO-Option
'035'	INTUS 5500 PoE	'P' 'T'	"01" "0C"	mit IO-Option ohne IO-Option
'036'	INTUS 5500 24V	'P' 'T'	"01" "0C"	mit IO-Option ohne IO-Option
'037'	INTUS 5600 NT	'P' 'T'	"01" "0C"	mit IO-Option ohne IO-Option
'038'	INTUS 5600 PoE	'P' 'T'	"01" "0C"	mit IO-Option ohne IO-Option
'039'	INTUS 5600 24V	'P' 'T'	"01" "0C"	mit IO-Option ohne IO-Option
'040'	INTUS 5205 PoE	't'	"03"	
'041'	INTUS 5205 SNT (Steckernetzteil, 24V)	't'	"03"	
'042'	INTUS 5200 PoE	'p' 't'	"0B" "03"	mit IO-Option ohne IO-Option
'043'	INTUS 5200 24V	'p' 't'	"0B" "03"	mit IO-Option ohne IO-Option
'052'	INTUS 5540 NT	'P' 'T'	"01" "0C"	mit IO-Option ohne IO-Option
'053'	INTUS 5540 PoE	'P' 'T'	"01" "0C"	mit IO-Option ohne IO-Option
'054'	INTUS 5540 24V	'P' 'T'	"01" "0C"	mit IO-Option ohne IO-Option
'055'	INTUS ACM80e Rack	'A'	"08"	
'056'	INTUS ACM80e Wand	'A'	"08"	

**Tabelle 2o****Ethernet-Link Konfiguration CV+235,1 und Status CV+236,1**

<b>Wert</b>	<b>Konfiguration und Status</b>
'0'	Autonegotiation
'1'	Kein Link (nur als Status in CV+236,1)
'2'	10 Base T halbduplex
'3'	10 Base T vollduplex
'4'	100 Base Tx halbduplex
'5'	100 Base Tx vollduplex

**Bemerkung 1****zu CV+91,8 und CV+99,8:**

Bei den Zutrittskontrollmanagern INTUS 3000 ACM, INTUS ACM8, INTUS ACM8(0)e Rack und INTUS ACM8(0)e Wand kann die Verkabelung der externen Leser an LBus1 und LBus2 im Point-to-Point oder im Multipoint Verfahren ausgeführt werden. Siehe auch Abschnitt 11.3.2 und 11.5.2.

Beim Multipoint Verfahren muss im CV-Feld festgelegt werden, welcher Leser an welchem Steckplatz oder an welcher Klemme (INTUS ACM8(e)) bzw. an welchem Kanal (INTUS 3000 ACM) angeschlossen ist.

CV+91,8 für Leser 1 – 8 am LBus1  
CV+99,8 für Leser 9 – 16 am LBus2

Pro Leser ist ein Byte reserviert, in das die Nummer des Steckplatzes, des Kanals oder der Klemme eingetragen wird, an dem der Leser angeschlossen ist.

Für den INTUS 3000 ACM ist der Wert ab Werk und bei Eiskaltstart sowie Comstart für beide LBusse "0102030405060708". Das entspricht einer point-to-point Verbindung zwischen Kanal und Leser.

Für die INTUS Zutrittskontrollmanager, die mit einer TCL Version größer oder gleich Version 5.5 ausgeliefert werden, gilt: wird die Verkabelung Point-to-Point ausgeführt und ist so in CV+9,1 eingetragen, werden die Teilsteller CV+91,8 und CV+99,8 ignoriert.

Ist in CV+224,1 für einen LBus die einfache Adressierung ausgewählt (alle Leser haben die Hardwareadresse 1), so muss die Verkabelung Point-to-Point sein. Die Werte in CV+91,8 und CV+99,8 werden dann ebenfalls übergangen.

**Bemerkung 2****zu CV+5,1 mit CV+107,1 und CV+6,1 mit CV+108,1:**

Ab TCL Version 5.01 werden große Speichererweiterungen unterstützt, indem die Größen von TF-Feld und Notpuffer nicht mehr auf 765kB (maximal mögliche Darstellung in einem Byte) begrenzt ist, sondern nur noch durch den verfügbaren Speicher.

Die Größe des TF-Felds bzw. des Notpuffers wird wie vor TCL Version 5.01 in 3kB-Einheiten angegeben. Da im CV-Feld nur ein Byte für die Größenangabe vorgesehen war, ist eine Erweiterung erforderlich. Dazu wurden folgende Teilsteller festgelegt:

CV+107,1 ist das höherwertige Byte der Notpuffergröße in CV+5,1  
CV+108,1 ist das höherwertige Byte der TF-Größe in CV+6,1

**Beispiel:**

Auslesen der Notpuffergröße: Beachten Sie, dass bei Binärzahlen die interne Byteordnung "Little Endian" (wie bei Motorola-Prozessoren) ist.

DK,CV+5,ER+1,1:K,CV+107,ER,1:KW,258,ER,ER:  
DMU,(ER,5),3072,ER,10:SR,'NP-Groesse='&ER,10:

Umgekehrt sei die Notpuffergröße in Byte in ER,10 vorgegeben. Dann wird sie durch folgendes Programmstück in CV eingetragen:

DDI,(ER,10),3072,ER,5:KW,259,ER,ER:  
DK,ER+1,CV+5,1:K,ER,CV+107,1:

**Bemerkung 3****zu CV+110,5:**

Das Feld CV+110,5 gibt eine genaue Auskunft über den Speicherausbau. Es enthält die Größe des Speichers in Kilobyte (1024 Byte) als druckbare Dezimalzahl, z.B. '00256' für 256kB. Es handelt sich um die Bruttogröße, die auch die TCL Felder enthält. Detailinformationen in Kapitel 8.1.

**Bemerkung 4****Werte in CV+47,1,CV+49,1 sowie CV+83,1, CV+84,1 beim INTUS ACM4 Wiegand und INTUS ACM40 Wiegand**

Beim INTUS ACM4 Wiegand und INTUS ACM40 Wiegand sind die externen Leser mit den Adressen 1, 2, 9 und 10 immer freigegeben. Der Typ der externen Leser in den Feldern CV+47,1,CV+49,1, CV+83,1 und CV+84,1 ist "08" für INTUS LBus Modus B.

## 4.7 D Display

Das D-Feld hat die Syntax <Displayfeld> :

**D { [ \* <LBusadr> ] } { [ <Zeile> ] }**

Die Displays der abgesetzten Eingabestationen, wie z.B. der INTUS 1600 Terminals, werden mit **D[\*<LBusadr>]** angesprochen. Das interne Display wird mit **D** oder **D[\*0]** erreicht.

Bei mehrzeiligen Displays kann die Position mit einem Offset <Zahlenwert> ab dem Feldbeginn

**D+<Zahlenwert>**

oder zeilenorientiert mit

**D [<Zeile>]+<Zahlenwert>**

angegeben werden. Für ein externes Display muss dem **D** zunächst **[\*<LBusadr>]** folgen.

Das D-Feld für das interne Display

D
D+1
----
D+2019

hat eine Länge von 2020 Bytes. Die D-Felder für die Displays der abgesetzten Eingabestationen schließen sich an. Sie haben eine Länge von 80 Bytes (ab TCL Version 5.01, davor von 64 Bytes). Die Gesamtlänge ist 3044 Bytes für TCL Releases bis einschließlich 4.28. Ab TCL Version 5.01 beträgt die Gesamtlänge 3300 Bytes.

Der Inhalt des D-Feldes wird im Display angezeigt, wenn druckbare Zeichen hineingeschrieben werden. Die Wirkung von nichtdruckbaren Zeichen und ESC-Steuерsequenzen wird in Abschnitt 7 beschrieben.

Die für die Anzeige nutzbare Stellenzahl hängt von der jeweiligen Displayausführung ab (siehe CV+25,1 Displayvariante, CV+78,2 Zeilenanzahl und CV+80,3 Spaltenanzahl). Die restlichen Stellen können anderweitig, z.B. zur Ausgabe von ESC-Sequenzen, oder für verdeckte Passworteingabe, verwendet werden.

The Größe einer Displayzeile eines externen Lesers wird durch den Typ des externen Lesers in CV+47 bis CV+61 oder CV+83 bis CV+90 bestimmt. Sie beträgt 16 Zeichen für einen INTUS 1500 Leser und 20 Zeichen für INTUS 1600 Leser. Diese Leser können gemischt betrieben werden. Die Größe der Zeilen wird individuell bestimmt.

Bei Tastatureingaben werden die eingegebenen Zeichen im Display angezeigt (Echo). Nachdem die Eingabe mit der ENTER-Taste beendet wurde, stehen sie zur Verarbeitung im Feld D.

### Beispiel:

**DK, 'Hallo' ,D[\*1][1]+5:**

Schreibt 'Hallo' in die 2.Zeile und die 6.Spalte auf das Display der Eingabestation mit LBus Adresse 1.

## 4.8 Dx DNCIN-Steuerfeld

Das DNCIN-Steuerfeld Dx ist ein *<Indexfeld>*.

Es gibt die Indexfelder D2, D3 und D4. Sie kontrollieren das Verhalten der DNCIN Prozesse DNCIN0 bis DNCIN2 (DNCIN steht für Digital Numerical Control Input). Diese Prozesse sind jedoch nur dann im System vorhanden, wenn die entsprechende zusätzliche Schnittstelle zur freien Verfügung steht, also Kanal B, C oder D nicht z.B. für LBus-Kommunikation benötigt wird.

An den zusätzlichen seriellen Schnittstellen werden verschiedene Geräte, wie z.B. Waagen oder Leser, angeschlossen. Die zugehörigen Treiberprozesse empfangen die Daten der zu kontrollierenden Geräte und schreiben sie in die Ringpuffer \$5, \$8 oder \$A. Von dort liest der zugehörige DNCINx Prozess die Daten aus und leitet sie entsprechend den Angaben im Dx-Feld weiter.

Feldaufteilung	Bedeutung	Wertebereich
Dx,1	Konfiguration	siehe unten
Dx+1,2	Routing (wenn Dx='1', Dx='9')	'00' keine Routingbytes sonst 2 ASCII-Zeichen
Dx+3,4	Sprungziel, wenn \$4 voll ist (wenn Dx,1='1')	'0000' kein Sprung '0001'-'9999' Sprungziel
Dx+7,6	Offset für das Schreiben in TF (wenn Dx='6')	'000000'-'999999'

Das Dx-Feld ist 13 Bytes lang.

### Das Konfigurationsbyte

gibt das Ziel an, an das die Daten weitergegeben werden sollen.

Die Umlenkung muss so früh wie möglich im TCL-Programm erfolgen. Nach der Resetanweisung IR,S: kann es sonst passieren, dass vom angeschlossenen Gerät spontan gesendete Daten gemäß Voreinstellung an den Host übertragen werden.

### Dx,1= '0': Voreinstellung, Daten transparent in Sendepuffer Hostschnittstelle \$2

Die Daten aus \$5/\$8/\$A werden zeichenweise sofort, ohne Berücksichtigung vom CR-Zeichen, in den Sendepuffer der Hostschnittstelle \$2 geschrieben.

### Dx,1= '1': Daten in Notpuffer \$4

Die Daten aus \$5/\$8/\$A werden satzweise in den Notpuffer \$4 kopiert. Dies geschieht erst, wenn ein CR-Zeichen empfangen wurde. Es muss sichergestellt sein, dass nur Datensätze, die mit CR-Zeichen enden und nicht größer als 256 Byte sind, empfangen werden. Die im Notpuffer erfassten Daten sind durch das Datensicherungskonzept geschützt.

Wenn Dx+1,2 zwei Zeichen ungleich '00' enthält, werden sie als Routingbytes zusammen mit dem Online-Statusbyte 'T' dem Datensatz vorangestellt (siehe Abschnitt 6.3 MONOUT-).

Falls der Notpuffer \$4 voll ist, wird ein Sprung auf das in Dx+3,4 angegebene Sprungziel ausgeführt.

### Dx,1= '2': Transparente Dateneingabe

Die Daten werden dem Eingabeprozess übergeben, so dass die Daten mit einer Eingabeanweisung gelesen werden können (siehe Abschnitt 5.14.6 *Eingaben über die seriellen Zuschnittstellen*).

**Dx,1='3': Anschluss von RENA-Lesern****Dx,1= '5': keine Übertragung**

In diesem Fall ist der DNCINx Prozess deaktiviert, d.h. die Daten bleiben im Ringpuffer \$5/\$8/\$A. Sie können von dort mit der RD Anweisung ausgelesen werden.

**Dx,1= '6': Daten ins Tabellenfeld TF**

Die Daten werden zeichenweise in das Tabellenfeld TF eingetragen. Das Satzendezeichen CR wird mitkopiert.

In Dx+7,6 wird der Offset <Zahlenwert> angegeben, ab dem kopiert wird. Dieser Offset wird vom DNCINx Prozess nach jedem übertragenen Zeichen inkrementiert. Am Ende der Übertragung steht der Offset auf dem ersten freien Byte nach den übertragenen Daten.



Wenn das Ende des Feldes TF erreicht ist, wird der Offset nicht mehr verändert, und alle weiteren Zeichen gehen verloren. Dies kann z.B. dazu ausgenutzt werden, um Daten von den zusätzlichen seriellen Schnittstellen zu ignorieren.

**Dx,1= '7' (Dx,1= '8'): Daten in Empfangspuffer Hostschnittstelle \$1**

In diesem Fall werden die Daten satzweise in den Ringpuffer \$1 übertragen. Sie werden dann vom MONIN Prozess weiterverarbeitet, als wenn sie vom Host gesendet worden wären. Deshalb ist der vom MONIN-Prozess erwartete Datensatzaufbau einzuhalten (siehe Abschnitt 6.2 *MONIN*-)!

**Dx,1= '9': Daten geblockt in Sendepuffer Hostschnittstelle \$2**

Die Daten aus \$5/\$8/\$A werden satzweise (gegenüber zeichenweise bei Dx,1='0') in den Ringpuffer \$2 geschrieben, d.h. an den Host gesendet. Dadurch wird gewährleistet, dass bei Verwendung eines Protokolls, wie z.B. BSC, an der Hostschnittstelle der Datensatz nicht in mehrere Datenübertragungsblöcke gestückelt wird.

Wenn Dx+1,2 zwei Zeichen ungleich '00' enthält, werden sie als Routingbytes zusammen mit dem Online-Statusbyte '[' dem Datensatz vorangestellt. Diese Sätze unterliegen aber nicht dem Datensicherungskonzept und müssen auch nicht quittiert werden.

## 4.9 DL Programmreich

DL
DL+1
----
DL+...

Die Größe des Programmreichs DL (DownLoad) kann implizit in Abhängigkeit vom Feld TF und Notpuffer \$4 über den Setup oder über das CV-Feld (CV+5,1 mit CV+107,1 für Notpuffer \$4, CV+6,1 mit CV+108,1 für TF-Feld) eingestellt werden, da der nach Anlage des TF-Felds und des Notpuffers verbliebene Speicher im SRAM dem Programmreich zugeordnet wird. Detailinformationen in Kapitel 8.1.

Die TCL Anwenderprogramme werden beim Laden durch den MONIN Prozess in einen binären Zwischencode kompiliert und dieser im Programmreich DL gespeichert.

Da das DL-Feld binären TCL-Programmcode enthält, darf mit TCL-Anweisungen nicht darauf geschrieben werden.

Für /DL/ ist immer eine mindestens 8-stellige Operation zu verwenden.

### Beispiel:

AD,/DL/,0,D: ergibt die gesamte Größe des DL-Feldes

SB,/DL/,/@/,D: ergibt die Größe des geladenen Programms

AD,@/,0,D: ergibt die Größe des freien Speicherbereichs

## 4.10 Ex - Digitaler Eingang DI

Das Ex-Feld ist ein *<LBusindexfeld>*:

**E[<LBusadr>]<Zahlenkomp> | E<Index>**

Die digitalen Eingänge (DI) an den abgesetzten Eingabestationen werden mit **E[<LBusadr>]** angesprochen. Die DIs des Terminals sind mit **E** oder **E[0]** zu erreichen.

Zu jedem digitalen Eingang Ex gehört ein Zähler Zx. Zu den ersten 14 lokalen DIs gibt es außerdem die Taktüberwachungsfelder Tx, TAx und TOx.



Die Ereignissprünge für die digitalen Eingänge Ex, die zugehörigen Zähler Zx und die Taktüberwachung werden erst aktiviert, wenn das Maschinentaskfeld P2 auf '1' gesetzt ist, siehe Kap. 4.29.

Feldauftteilung	Bedeutung	Wertebereich
Ex,1	Wert des DIs	'0' low '1' high
Ex+1,1	Flankenstellung	'0' negativ, fallende Flanke (Änderung von Ex,1 nach '0') '1' positiv, steigende Flanke (Änderung von Ex,1 nach '1') '2' deaktiviert
Ex+2,2	Sprungziel bei Flanke wie in Ex+1,1 festgelegt	'00' kein Sprung '01'-'99' Sprungziel
Ex+4,1	Einheit für die Felder Tx, TOx und TAx der Taktüberwachung für lokale DIs	'1' bedeutet 100ms Einheiten '2' bedeutet 1s Einheiten
Ex+5,1	Steuerung der Taktüberwachung TAx für lokale DIs	'0' (Schreiben:) Start der Taktüberwachung mit der nächsten "positiven, steigender" (Ex+0,1 = Ex+1,1) Flanke, Start der Taktausfallüberwachung (TAx) sofort '0' (Lesen:) Taktausfallüberwachung noch nicht mit positiver Flanke aktiviert oder Taktausfallzeit (TAx) überschritten. '2' (Schreiben, nur INTUS 3000:) Start der Taktüberwachung und der Taktausfallüberwachung (TAx) mit der nächsten "positiven" (Ex+0,1 = Ex+1,1) Flanke '1' (Lesen:) Taktüberwachung aktiv, d.h. erste Flanke ist eingetroffen und die Taktausfallzeit (TAx) wurde (noch) nicht überschritten

**Tabelle 4.1 – Feldauftteilung beim digitalen Eingang**

Ex ist 6 Bytes lang.

Für abgesetzte Eingabestationen sind je 5 DIs vorgesehen, die aber nicht vorhanden sein müssen.

#### 4.10.1 Digitale Eingänge/Vandalenkontakte: Terminals/Zutrittskontrollmanager

Terminal	Digitale Eingänge	Vandalenkontakt
INTUS 3100	E0 und E1	E4
INTUS 3200 (ethertime)	E0 und E1	keiner
INTUS 3300 ethertime	keine	keiner
INTUS 3300	E0 und E1	E4
INTUS 3400	E0 und E1	E4
INTUS 3450-time INTUS 3460-time	keine	E4
INTUS 3450 INTUS 3450-plus INTUS 3450-timeplus INTUS 3460-plus INTUS 3460-timeplus	E0 und E1	E4
INTUS 3500 INTUS 3600 INTUS 3660	E0 und E1, sowie je nach Einbau bis zu 8 weitere digitale Eingänge: 4DI-Modul anstatt Kanal B: E5-E8 4DI-Modul anstatt Kanal C: E9-E12	E4
INTUS 5205 INTUS 5200 ohne IO-Option INTUS 5300 PoE INTUS 5500, 5540 ohne IO-Option INTUS 5600 ohne IO-Option	keine	E4
INTUS 5200 mit IO-Option INTUS 5300 INTUS 5500, 5540 mit IO-Option INTUS 5600 mit IO-Option	E0 und E1	E4
INTUS 3000 ACM	E0 und E1, sowie maximal 2 Leser-DI/DO-Module mit Adresse 0 und Adresse 1 oder 2: TCL Version 4, Version 5 und Version 6 bei voreingestellter Verkabelung MP/MP: Leser-DI/DO-Modul Adresse 0: E5-E12 Leser-DI/DO-Modul Adresse 1: E13-E20 Leser-DI/DO-Modul Adresse 2: E21-E28 TCL Version 6 bei Verkabelung PP/MP und PP/PP: siehe Kapitel 11.6	E4
INTUS ACM4	E0, E1, E5 – E8, je nach Verkabelung: E13 – E16, E21 – E24 Zuordnung siehe Kapitel 11.2	E4

INTUS ACM4 AKKU	E0, E1, E3 Quittierungstaster (ab TCL 6.10 optional, abhängig von CV+239,1) E5 – E8, je nach Verkabelung: E13 – E16, E21 – E24 Zuordnung siehe Kapitel 11.2	E4
INTUS ACM40 INTUS ACM40 AKKU	E0 – E3, E5 – E8, je nach Verkabelung: E13 – E16, E21 – E24 Zuordnung siehe Kapitel 11.1	E4
INTUS ACM8 INTUS ACM8(0)e Rack INTUS ACM8(0)e Wand	E0 – E3, je nach Verkabelung: E5 – E20, E5 – E12 und E21 – E28, E5 – E35 nur alle ungeraden Indizes Zuordnung siehe Kapitel 11.3 und 11.4	E4

**Tabelle 4.2 – Digitale Eingänge / Vandalenkontakt der Terminals**

Falls beim Terminal ein Vandalenkontakt vorhanden ist, trägt er den Index E4.

Vandalenkontakt	Wert in E4,1
geschlossen	'1'
offen	'0'

Alle Vandalenkontakte der Terminals werden auf "geschlossen", d.h. '1' in E4,1, initialisiert. Das gilt auch für Terminals, die keinen Vandalenkontakt besitzen.



Um einen offenen Vandalenkontakt bemerken zu können, wird ein Sprung in E4+2,2 einge-tragen und die Flanke E4+1,1 auf '0' (negativ, fallende Flanke) gesetzt.

P2 muss '1' sein. Beim Öffnen des Vandalenkontakte wird ein Sprung ausgelöst.

Beim INTUS ACM4 AKKU ist E3 der Quittierungstaster für L1 (LED Störung) und L2 (LED Sabotage). Ab TCL Version 6.10 kann der Quittierungstaster, abhängig von der Einstellung in CV+239,1, auch der Funktionstaste F21 zugeordnet werden.

#### 4.10.2 Digitale Eingänge/Vandalenkontakte der abgesetzten Leser

Abgesetzter Leser	Digitale Eingänge	Vandalenkontakt
INTUS 300l/m, INTUS 305l	keine	E[x]3
INTUS 300ro	keine	keiner
INTUS 300h, INTUS 340h	E[x]0 und E[x]1	keiner
INTUS 350H	E[x]0 und E[x]1	keiner
INTUS 400, INTUS 500, INTUS 600, INTUS 600 FP, ohne IO-Modul mit IO-Modul	keine E[x]0 und E[x]1	E[x]3 E[x]3
INTUS 1500, INTUS 1600	E[x]0 und E[x]1	E[x]3
INTUS FP	E[x]0 und E[x]1	E[x]3 (ab V1.21)

*Tabelle 4.3 - Digitale Eingänge / Vandalenkontakt der abgesetzten Leser*

Falls beim abgesetzten Leser ein Vandalenkontakt vorhanden ist, trägt er den Index E[<LBusadr>]3.

Vandalenkontakt des abgesetzten Lesers mit Adresse <LBusadr>	Wert in E[<LBusadr>]3,1
geschlossen	'1'
offen	'0'



Wird an ein INTUS 3000 mit einer TCL Version vor TCL 5.11 ein INTUS 1600 angeschlossen, so wird beim Öffnen des INTUS 1600 der Wert für den Vandalenkontakt in E[<LBusadr>]3,1 zu '1', umgekehrt zum Verhalten der anderen abgesetzten Eingabestationen.

Alle Vandalenkontakte der abgesetzten Eingabestationen werden auf "geschlossen", d.h. '1' in E[<LBusadr>]3,1, initialisiert. Das gilt auch für abgesetzte Eingabestationen, die nicht konfiguriert sind, und für abgesetzte Leser, die keinen Vandalenkontakt besitzen.

Um einen offenen Vandalenkontakt bemerkbar zu können, wird ein Sprung in E[<LBusadr>]3+2,2 eingetragen und die Flanke E[<LBusadr>]3+1,1 auf '0' (negativ, fallende Flanke) gesetzt. P2 muss '1' sein. Beim Öffnen des Vandalenkontakte wird ein Sprung ausgelöst.

#### Pseudo-Eingang für Online-/ Offline-Zustände der abgesetzten Leser

Der **Pseudo-Eingang** E[<LBusadr>]4 gibt Auskunft über Online-/ Offline-Zustände der abgesetzten Eingabestation, z.B. des INTUS 1600 Terminals.

Abgesetzter Leser mit Adresse <LBusadr>	Wert in E[<LBusadr>]4,1
online	'1'
offline	'0'

Um den Offline-Fall einer abgesetzten Eingabestation abfangen zu können, sollte ein Sprung in E[<LBusadr>]4+2,2 eingetragen werden und die Flanke E[<LBusadr>]4+1,1 auf '0' stehen. P2 muss '1' sein. So wird ein Sprung ausgelöst, wenn ein Offline-Zustand entdeckt wird.

### 4.10.3 Programmierhinweise

#### DI Flankeneinstellung und DI Sprung

Die Flankeneinstellung in Ex+1,1 beeinflusst neben den DI-Sprüngen auch die Zähler, Zx, und die Taktüberwachung. Die Zähler und die Taktüberwachung reagieren auf eine Flanke, die Ex+0,1 auf den Wert von Ex+1,1 setzt.

Deshalb sollte man die Flankeneinstellung beim Betrieb eines Zählers oder einer Taktüberwachung nach einer Initialisierungsphase (in der P2 '0' sein sollte) nicht mehr verändern. Wenn die Flankeneinstellung Ex+1,1 auf den (derzeit aktuellen) Wert von Ex+0,1 gesetzt wird, nimmt das TCL-System eine weitere Flanke zur Kenntnis.

Ein Wert von '2' in Ex+1,1 deaktiviert die DI-Sprünge, den dazugehörigen Zähler und die Taktüberwachung. Soll diese Deaktivierung rückgängig gemacht werden, dann sollte man bei Taktüberwachungsbetrieb mit Ex+1,1 auch das Teilstück Ex+5,1 (also mindestens Ex+1,5) überschreiben, da dort festgelegt ist, wie der Anlauf der Takt(ausfall)überwachung geschehen soll.

#### Wert des DIs

Im INTUS 3000 entspricht das Ex+0,1 Feld immer aktuell dem Pegel am digitalen Eingang.

Da sich der Wert auch schon sofort nach einem DI-Ereignissprung geändert haben kann, sollte im TCL-Programm für die weitere Verarbeitung die sprungauslösende Flankenstellung Ex+1,1 statt dem aktuellen Pegel des DI Ex+0,1 verwendet werden.

#### DI-Sprung, P2-Flag

Die DI-Sprünge werden nur ausgeführt, wenn das P2-Feld auf '1' gesetzt ist, das zugehörige Teilstück Ex+1,1 ungleich '2' ist, und ein Sprungziel in Ex+2,2 eingetragen ist.

Beim Warmstart werden die Sprungadressen im Ex+2,2 beibehalten, das P2-Flag jedoch auf '0' gesetzt. Daher ist das TCL-Programm in der Initialisierungsphase nach einem Warmstart vor ungewollten DI-Sprüngen geschützt bis es P2 auf '1' setzt.

Im INTUS 3000 wird die Nummer des DIs einer abgesetzten Eingabestation oder des Terminals nach einem DI-Sprung in P20+20,2 abgelegt. P20+24,2 enthält die Nummer der externen Eingabestation, bzw. '00' für das Terminal. Beide Felder sind nach Sprungauslösung bis zur nächsten Stopanweisung aktuell.

Wenn Ex+1,1 auf '2' gesetzt, d.h. der DI-Sprung deaktiviert wird, so wird ein schon ausgelöster DI-Sprung aus dem Interpreteranweisungspuffer \$3 gelöscht. Der zugehörige Zähler Zx wird angehalten. Weiterhin wird die in Tx und TAx spezifizierte Taktüberwachung angehalten.

#### 4.10.4 Taktüberwachung

Die Felder Ex+4,1 und Ex+5,1 haben eine Funktion, wenn die optionale Softwarefunktion "Taktüberwachung" mit der TCL Firmware installiert ist.

##### **Ex+4,1**

In Ex+4,1 wird die Zeiteinheit festgelegt, die den Zeitangaben in den Felder Tx, TAx und TOx zugrunde liegt. Es ist eine Einstellung von 100ms oder 1s möglich.

##### **Ex+5,1**

Schreibt man in Ex+5,1 eine '2' hinein, dann wird die Taktüberwachung, Tx, und die Taktausfallüberwachung, TAx, erst bei der nächsten "positiven" Flanke gestartet. Damit ist ein sauberer, problemloser Start einer unabhängigen Maschine möglich. Tut man das gleiche während einer laufenden Taktüberwachung, dann wird die Überwachung des aktuellen Taktzyklus ausgesetzt. Die Überwachung beginnt wieder mit dem nächsten Takt.

Schreibt man in Ex+5,1 eine '0' hinein, dann startet die Taktüberwachung, Tx, mit der nächsten "positiven" Flanke. Aber die Taktausfallüberwachung (TAx) wird sofort mit der entsprechenden TCL-Anweisung gestartet. Dies dürfte dann sinnvoll sein, wenn die Maschine ebenfalls von dem INTUS Terminal gestartet wird. Tut man das gleiche während einer laufenden Taktüberwachung, dann wird die Überwachung des aktuellen Taktzyklus mit der aktuellen TCL-Anweisung neu gestartet.

In dem Augenblick, in dem P2 auf '1' gesetzt wird, wobei die gesamte Taktüberwachung, die Zähler und die DI-Sprünge aktiviert werden, werden die Teilsteller Ex+5,1 auf '0' gesetzt, wenn sie nicht schon vorher zu '2' initialisiert war. Damit wird im Normalfall der sofortige Start der Taktausfallüberwachung (TAx) mit der P2 Aktivierung ausgelöst. Hat man vorher eine '2' in Ex+5,1 geschrieben, verzögert sich der Start der Taktausfallüberwachung auf die erste "positive" Flanke.

Das TCL-System setzt das Teilstellfeld Ex+5,1 automatisch bei Eintreffen der ersten Flanke auf '1'. Damit kann der Start der Maschine bei unabhängigem Anlauf durch das TCL Programm ermittelt werden.

Der Zähler Zx kann unabhängig von der Taktüberwachung verwendet werden. Die Taktüberwachung hat auf ihn keinen Einfluss.

## 4.11 ED Ergebnisfeld Divisionsrest

ED
ED+1
----
ED+9

Das ED-Feld ist 10 Bytes lang.

Es enthält den ganzzahligen Rest einer Division DI. Ansonsten kann das Feld frei verwendet werden.

## 4.12 ER Ergebnis

ER
ER+1
----
ER+9

Das ER-Feld ist 10 Bytes lang.

Wenn in einer Arithmetikanweisung AD, SB, MU oder DI kein Zielfeld (Ergebnisfeld) angegeben wird, steht das Ergebnis linksbündig im Feld ER. Die Länge des Ergebnisses findet sich immer in EZ. Daher kann das Ergebnis mit ER,(EZ) zu weiteren Berechnungen verwendet werden.

Ansonsten kann das Feld frei verwendet werden.

## 4.13 EV Ergebnisfeld Vorzeichen

EV
----

Das EV-Feld ist 1 Byte lang.

Es enthält das Vorzeichen der arithmetischen Operationen AD, SB, MU oder DI. Dies ist, da die Operanden positive ganze Zahlen sind, außer bei der Subtraktion immer '+'. Bei der Subtraktion ist es entweder '+' oder '-'.

Ansonsten kann das Feld frei verwendet werden.

## 4.14 EZ Ergebnislänge

EZ
EZ+1
EZ+2

Das EZ-Feld ist 3 Bytes lang.

Nach einer Arithmetikanweisung AD, SB, MU oder DI enthält das Feld EZ die Stellenanzahl des Ergebnisses als 3-stellige ASCII-Dezimalzahl.

Wird P20+1,1 auf '1' gesetzt, enthält das Feld EZ auch die Anzahl der durch die Anweisung RD eingelesenen Zeichen.

Ansonsten kann das Feld frei verwendet werden.

## 4.15 Gx Stillstandszeiten

Das Gx-Feld ist ein *<Indexfeld>*. Es gibt G0 bis G11.

Gx
Gx+1
----
Gx+7

Gx ist 8 Bytes lang.

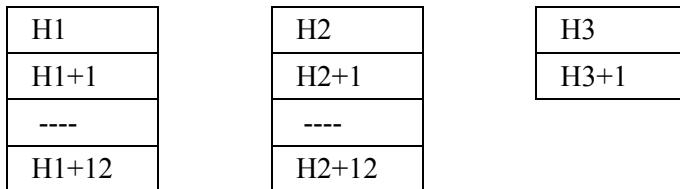
Die Anweisung C0 (Abschnitt 5.7) berechnet die Zeitdifferenz aus den in den Feldern H2 und H1 angegebenen Uhrzeiten in Sekunden und addiert sie rechtsbündig in einem der Felder Gx bzw. ND auf. Der Feldindex x=0,...,11 wird dabei aus H3 genommen. Jedem Stillstandsgrund kann ein Feld Gx zugeordnet werden, in dem die Zeiten akkumuliert werden. Die akkumulierten Zeiten dürfen 365 Tage nicht überschreiten.

Wenn H3 einen Wert größer als 11 enthält, werden die Zeiten in ND, nicht definierte Stillstandszeiten (Abschnitt 4.24), aufaddiert.

Ansonsten kann das Feld Gx frei verwendet werden.

## 4.16 Hx Hilfsfelder

Das Hx-Feld ist ein *<Indexfeld>*. Es gibt H1, H2 und H3 mit unterschiedlicher Länge.



H1 und H2 sind 13 Bytes lang. H3 ist 2 Bytes lang.

In die Felder H1 und H2 können Uhrzeit und Datum eingetragen und mit der Anweisung C0 (Abschnitt 5.7) die Zeitdifferenz berechnet werden.

Feldaufteilung		Bedeutung	Wert
H1,2	H2,2	Stunde	'00'-'23'
H1+2,1	H2+2,1	Trennzeichen	:
H1+3,2	H2+3,2	Minuten	'00'-'59'
H1+5,1	H2+5,1	Trennzeichen	:
H1+6,2	H2+6,2	Sekunden	'00'-'59'
H1+8,2	H2+8,2	Monat	'01'-'12'
H1+10,2	H2+10,2	Monatstag	'01'-'31'
H1+12,1	H2+12,1	Wochentag	'0' Sonntag '1' Montag ... '6' Samstag

H3 enthält bei der Anweisung C0 den Index für Gx.

Ansonsten können die Felder frei verwendet werden.

## 4.17 I Induktivkartenleser und Identifikation mit Biometrieleser

Wenn eine Eingabe über Biometrieleser mit der Anweisung E freigegeben wurde, wird die Templates-ID des identifizierten Fingers oder Handvenenscans nach einer Lesung in das I-Feld eingetragen. Weitere Informationen, wie z.B. beim Fingerprintleser die Finger-ID und der Fingerstatus, werden ebenfalls hinterlegt.

Hierbei handelt sich um die Identifikation mit einer 1:n Beziehung zu den Templates im Biometrieleser, im Gegensatz zu Verifikation mit einer 1:1 Beziehung zum „TemplateOnCard“ (siehe auch Abschnitt 4.22).

Des Weiteren wird das I-Feld für Eingaben über Induktivkartenleser (von RENA und Hengster) genutzt.

Feldaufteilung (88 Byte)	Feldaufteilung (115 Byte)	Bedeutung
I,80	I,107	Datenfeld
I+80,1	I+107,1	Konfiguration
I+81,1	I+108,1	Lesefehler
I+82,1	I+109,1	Codekennung
I+83,1	I+110,1	Lesernummer
I+84,3	I+111,3	Zeichenanzahl
I+87,1	I+114,1	Fingerprintleser: Finger-ID und Fingerstatus Handvenenscan-Leser: t.b.d. Induktivkartenleser: Kartenlage

Das I-Feld hat eine Länge von 88 oder 115 Bytes, je nach Einstellung im Setup bzw. im CV-Feld (CV+43,1).

### Datenfeld: I,80 oder I,107

Maximal 80 bzw. 107 Zeichen können als Daten der Lesung abgelegt werden.

Beim Biometrieleser wird die 8-stellige Templates-ID des identifizierten Fingers oder Handvenenscans eingetragen.

### Konfiguration: I+80 oder I+107

Bevor eine Lesereingabeanweisung ausgeführt wird, muss das Konfigurationsbyte belegt sein. Es gibt an, ob ein Eingabesprung auch bei Fehllesungen erfolgen soll oder nicht:

- '0'           ist die Voreinstellung. Bei korrekter Lesung wird der Sprung ausgeführt. Eine Fehllesung wird ignoriert, die Eingabefreigabe ist weiter aktiv.
- '1'           Der Sprung wird auch bei Fehllesung ausgeführt, die Lesefehler- und Codekennungsbytes werden entsprechend gesetzt. Sofern die Fehllesung Daten erzeugt hat, stehen diese zur weiteren Analyse zur Verfügung.

### Lesefehler: I+81 oder I+108

Wenn eine Eingabe erfolgt und das Konfigurationsbyte auf '1' gesetzt ist, kann über das Lesefehlerbyte festgestellt werden, ob die Lesung fehlerfrei war.

- '0'           Gutlesung
- '1'           Fehlerfall. Das Codekennungsbyte enthält den Fehlercode.

**Codekennung: I+82 oder I+109**

Bei einer gültigen Lesung enthält das Codekennungsbyte den jeweiligen Code einer Lesung.

Codekennung	Leser
'B'	Biometrieleser: Fingerprint, Handvenenscan
'T'	RENA Leser
'H'	Hengstler Leser

Im Fehlerfall enthält das Codekennungsbyte den Fehlercode (siehe Abschnitt 9.4 *Fehlercodes bei Lesereingaben*).

**Lesernummer: I+83 oder I+110**

Die Lesernummer gibt an, von welchem Leser die Eingabe erfolgte. Da das Lesernummerfeld nur eine Speicherzelle lang ist, werden für die externen Leser nicht nur Ziffern, sondern auch Buchstaben verwendet.

Lesernummer	Leser
'0'	interner Leser
'1'-'8'	externer Leser am LBus1
'9', 'A'-'G'	externer Leser am LBus2
'C'	Leser an Kanal B über DNCIN0 Prozess
'D'	Leser an Kanal C über DNCIN1 Prozess
'E'	Leser an Kanal D über DNCIN2 Prozess

Um die Nummer des Lesers, an der die Eingabe erfolgte, herauszufinden, kann auch P20+24,2 verwendet werden: Wenn eine Eingabe abgeschlossen ist, und der Eingabesprung in den Interpreteranweisungspuffer \$3 eingetragen wird, wird in P20+24,2 die Lesernummer als 2-stellige ASCII Zahl hinterlegt. Diese Information ist bis zur nächsten Stopanweisung aktuell.

**Zeichenanzahl: I+84,3 oder I+111,3**

Das Teilstück enthält rechtsbündig die Anzahl der gelesenen Zeichen als 3-stellige ASCII-Dezimalzahl.

**Fingerprintleser: Finger-ID und Fingerstatus: I+87 oder I+114**

Das Byte setzt sich aus der Bezeichnung des Fingers (Finger-ID) und dem Status des Fingers zusammen.

**Fingerstatus:** Es gibt den „normalen“ Finger und den Bedrohungsfinger, der eine Bedrohung der Person am Fingerprintleser anzeigt. Können am Fingerprintleser Finger eingelernt werden, so kann der Einlernvorgang mit dem Administratorfinger anstatt einer Passworteingabe gestartet werden.

Hand	Finger	Status: normal	Status: Bedrohung	Status: Administrator	Status: Bedrohung + Administrator
linke Hand	kleiner Finger	'0'	'@'	'P'	'' (Gravis)
	Ringfinger	'1'	'A'	'Q'	'a'
	Mittelfinger	'2'	'B'	'R'	'b'
	Zeigefinger	'3'	'C'	'S'	'c'
	Daumen	'4'	'D'	'T'	'd'
rechte Hand	Daumen	'5'	'E'	'U'	'e'
	Zeigefinger	'6'	'F'	'V'	'f'
	Mittelfinger	'7'	'G'	'W'	'g'
	Ringfinger	'8'	'H'	'X'	'h'
	kleiner Finger	'9'	'T'	'Y'	'i'

**Handvenenleser: I+87 oder I+114**

t.b.d.

**Induktivkartenleser: Kartenlage: I+87 oder I+114**

Das Kartenlagebyte gibt die Kartenlage an. Bei Lesern an den seriellen Zusatzschnittstellen kann die Kartenlage nicht erkannt werden.

**Inhalt des I-Feldes nach einem Terminal-Reset**

Wird der Anlaufmodus Warm- oder Kaltstart ausgeführt, ist das Datenfeld I,80 oder I,107 mit Leerzeichen '' gefüllt. Die Teilstufen ab I+80 oder I+107 werden mit Nullen '0' vorbesetzt.

## 4.18 KT Datum

Feldaufteilung	Bedeutung	Wert
KT,2	Monat	'01'-'12'
KT+2,2	Monatstag	'01'-'31'
KT+4,1	Wochentag	'0' Sonntag '1' Montag ... '6' Samstag
KT+5,4	Jahr	'0000'-'9999'

Das KT-Feld ist 9 Bytes lang.

### Datum setzen

Das KT-Feld muss als Ganzes mit einer Kopieranweisung gesetzt werden. Es darf keine Längenangabe und kein Offset verwendet werden. Die Zeichenfolge muss korrekt sein, es erfolgt nur eine eingeschränkte Plausibilitätsprüfung.

### Lesen des Datums

Das KT-Feld kann jederzeit gelesen werden. Dies geschieht etwa durch eine Kopieranweisung.

### Schaltjahr

Durch den im Terminal verwendeten Uhrenbaustein erfolgt die Umschaltung im Schaltjahr automatisch.

### Beispiel:

K, '011731998', KT:S: stellt das Datum auf 17. Jan 1998, Mittwoch

K, KT, D+15:S: zeigt das Datum im Display an.

## 4.19 Lx Lampe und Hupe

Das Lx-Feld ist ein *<LBusindexfeld>*:

**L[<LBusadr>]<Zahlenkomp> | L<Index>**

Die Lampen an den abgesetzten Eingabestationen werden mit **L[<LBusadr>]** angesprochen.  
Die Lampen des Terminals sind mit **L** oder **L[0]** zu erreichen.

Durch das Lampenfeld **L[<LBusadr>]0** bzw. **L0** wird die Hupe bedient.

Feldaufteilung	Bedeutung	Wert
Lx,1	Lampe	'A' Lampe aus 'E' Lampe ein 'B' Lampe blinken (Blinkfrequenz ca. 1Hz)

Lx ist 1 Byte lang.

Hupe und Lampen in den Terminals und Zutrittskontrollern:

Modell	Hupe	Lampe
INTUS 3100	vorhanden	L1 grün L2 rot
INTUS 3200 (ethertime), INTUS 3300 (ethertime)	vorhanden	keine
INTUS 3400 INTUS 3450 INTUS 3450-time INTUS 3450-timeplus INTUS 3450-plus INTUS 3460-time INTUS 3460-timeplus INTUS 3460-plus	vorhanden	L1 grün L2 rot L3 blau: ist immer an, wenn L1 und L2 aus sind. In diesem Zustand kann L3 auf 'B' und wieder auf 'E' eingestellt werden. 'A' wird wie 'E' behandelt.
INTUS 3500	vorhanden	keine
INTUS 3600 INTUS 3660	vorhanden	L1 grün L2 rot L3 blau: ist immer an, wenn L1 und L2 aus sind. In diesem Zustand kann L3 auf 'B' und wieder auf 'E' eingestellt werden. 'A' wird wie 'E' behandelt.
INTUS 5205 INTUS 5200	vorhanden	L3 blau: ist immer an. L3 kann auf 'B' und wieder auf 'E' eingestellt werden. 'A' wird wie 'E' behandelt.
INTUS 5300 INTUS 5300 PoE INTUS 5500 INTUS 5540 INTUS 5600	vorhanden	L1 – L5: L1 grün L2 rot L3 blau: ist immer an, wenn L1 und L2 aus sind. In diesem Zustand kann L3 auf 'B' und

		wieder auf 'E' eingestellt werden. 'A' wird wie 'E' behandelt. (Helligkeit bei INTUS 5300 in CV+233,1 einstellbar.) L4 grüne „Kommt-LED“ L5 rote „Geht-LED“
INTUS 3000 ACM	keine	keine
INTUS ACM4	keine	keine
INTUS ACM4 AKKU	vorhanden	L1 System-LED „Störung“ L2 System-LED „Sabotage“
INTUS ACM40	keine	keine
INTUS ACM40 AKKU	vorhanden	L1 System-LED „Störung“ L2 System-LED „Sabotage“
INTUS ACM8 INTUS ACM8(0)e Rack	vorhanden	L1 Status-LED rechts L2 Status-LED links, wenn die OK-LED an ist, sonst Steue- rung durch die Firmware
INTUS ACM8(0)e Wand	vorhanden	L1 Status-LED oben (S0) L2 Status-LED unten (S1), wenn die OK-LED an ist, sonst Steue- rung durch die Firmware

*Tabelle 4.4 – Hupe und Lampen der Terminals*

Hupe und Lampen in den abgesetzten Lesern:

Modell	Hupe	Lampe
INTUS 300h, INTUS 340h, INTUS 300l/m, INTUS 305l, INTUS 300ro	vorhanden	L[x]1 grün L[x]2 rot
INTUS 350H, INTUS 400, INTUS 500, INTUS 600, INTUS 1500, INTUS 1600, INTUS FP, INTUS 600 FP	vorhanden	L[x]1 grün L[x]2 rot

*Tabelle 4.5 – Hupe und Lampen der abgesetzten Leser*

Für abgesetzte Eingabestationen sind je 5 Lampen bzw. Hupe L[<LBusadr>]0 bis L[<LBusadr>]4 reserviert, die nicht alle vorhanden sein müssen.

Die Dauer der Aktivierung einer Hupe in einer externen Eingabestation kann mit P20+32,3 bestimmt werden, die Dauer der Aktivierung einer externen Lampe mit P20+26,3. Die Aktivierungszeit der internen Hupe und Lampen werden nicht über P20 gesteuert. Nach dem Einschalten einer Lampe oder Hupe kann das entsprechende Teilstück von P20 verändert werden, ohne dass sich die Einschaltzeit ändert.

 Die externen Lampen und Hupen, die über Zeitlimit in P20+26,3 und P20+32,3 ausgeschaltet werden, haben beim Zurücklesen der Felder Lx immer noch den aktiven Zustand 'B' oder 'E'.

## 4.20 LS Leser- und Terminalstatus

Das LS-Feld enthält Versionsnummern, die Seriennummer des Terminals, den (Fehler-)Status des Terminals und der angeschlossenen Leser.

Die Teilsteller LS+46,2 - LS+67,2 werden ab LS+142,2 wiederholt. Der Status aller 16 abgesetzten Leser am INTUS 3000 liegen so ab LS+142,2 - LS+187,2 hintereinander. Zugriffe auf diese Teilsteller führen zu einer Kommunikation auf dem LBus. Die Zugriffe sollten sparsam verwendet werden.

LS+92,4 - LS+99,1 gehören zu den Treiberfehlermeldungen.

Die Teilsteller LS+105,2 - LS+139,2 beschreiben den Status des Eingabeprozesses.

Zur besseren Lesbarkeit sind zwischen den Teilstellern Blanks ("20") eingefügt.

Feldaufteilung	Bedeutung	Wertebereich
LS+0,11	Seriennummer des CPU Boards	<ASCII-Konstante>
LS+12,4	Versionsnummer Treiber	<ASCII-Konstante>
LS+17,4	Versionsnummer RTK	<ASCII-Konstante>
LS+22,4	Versionsnummer TCL	<ASCII-Konstante>
LS+27,4	Versionsnummer Bitbus/Leser Controller (nur für INTUS 2000)	<ASCII-Konstante>
LS+32,4	Versionsnummer Displaycontroller (INTUS 2400)	<ASCII-Konstante>
LS+37,2	Terminal Status lokal (Batterie(n))	Tabelle 1
LS+39,1	Terminal Status Akku (nur INTUS ACM4 AKKU, INTUS ACM40 AKKU)	'1' : Netzspannung '0' : Akku Betrieb
LS+40,2	Status Leser	Tabelle 2
LS+43,2	Status Displaycontroller (nur INTUS 2000)	
LS+46,2	Status externer Leser 1	Tabelle 2
LS+49,2	Status externer Leser 2	Tabelle 2
LS+52,2	Status externer Leser 3	Tabelle 2
LS+55,2	Status externer Leser 4	Tabelle 2
LS+58,2	Status externer Leser 5	Tabelle 2
LS+61,2	Status externer Leser 6	Tabelle 2
LS+64,2	Status externer Leser 7	Tabelle 2
LS+67,2	Status externer Leser 8	Tabelle 2
LS+70,16	Seriennummer des Terminals	<ASCII-Konstante>
LS+86,5	Versionsnummer Ethernet-Controller	<ASCII-Konstante>
LS+92,4	Sprungziel bei Treiberfehler	'0000' kein Sprung '0001'-'9999' Sprungziel
LS+96,1	Fehlerzähler	'1'-'9'
LS+97,1	Kennzeichen des Treibers	Tabelle 4

Feldaufteilung	Bedeutung	Wertebereich
LS+98,1	Kennzeichen der Schnittstelle	Tabelle 5
LS+99,1	Fehlercode	Tabelle 6
LS+100,4	Versionsnummer TCL-Applikationsprogramm	<ASCII-Konstante>
LS+104,1	reserviert (Uhrstatus)	
LS+105,2	Anzahl der aktivierten Eingabesprünge	
LS+107,2	Eingabestatus der internen Leser	Tabelle 7
LS+109,2	Eingabestatus der externen Leser Adr.1	Tabelle 7
LS+111,2	Eingabestatus der externen Leser Adr.2	Tabelle 7
LS+113,2	Eingabestatus der externen Leser Adr.3	Tabelle 7
LS+115,2	Eingabestatus der externen Leser Adr.4	Tabelle 7
LS+117,2	Eingabestatus der externen Leser Adr.5	Tabelle 7
LS+119,2	Eingabestatus der externen Leser Adr.6	Tabelle 7
LS+121,2	Eingabestatus der externen Leser Adr.7	Tabelle 7
LS+123,2	Eingabestatus der externen Leser Adr.8	Tabelle 7
LS+125,2	Eingabestatus der externen Leser Adr.9	Tabelle 7
LS+127,2	Eingabestatus der externen Leser Adr.10	Tabelle 7
LS+129,2	Eingabestatus der externen Leser Adr.11	Tabelle 7
LS+131,2	Eingabestatus der externen Leser Adr.12	Tabelle 7
LS+133,2	Eingabestatus der externen Leser Adr.13	Tabelle 7
LS+135,2	Eingabestatus der externen Leser Adr.14	Tabelle 7
LS+137,2	Eingabestatus der externen Leser Adr.15	Tabelle 7
LS+139,2	Eingabestatus der externen Leser Adr.16	Tabelle 7
LS+142,2	Status externer Leser 1	Tabelle 2
LS+144,1	Lesertyp externer Leser 1 (ab TCL Version 5.5)	Tabelle 3
LS+145,2	Status externer Leser 2	Tabelle 2
LS+147,1	Lesertyp externer Leser 2 (ab TCL Version 5.5)	Tabelle 3
LS+148,2	Status externer Leser 3	Tabelle 2
LS+150,1	Lesertyp externer Leser 3 (ab TCL Version 5.5)	Tabelle 3
LS+151,2	Status externer Leser 4	Tabelle 2
LS+153,1	Lesertyp externer Leser 4 (ab TCL Version 5.5)	Tabelle 3
LS+154,2	Status externer Leser 5	Tabelle 2
LS+156,1	Lesertyp externer Leser 5 (ab TCL Version 5.5)	Tabelle 3
LS+157,2	Status externer Leser 6	Tabelle 2
LS+159,1	Lesertyp externer Leser 6 (ab TCL Version 5.5)	Tabelle 3
LS+160,2	Status externer Leser 7	Tabelle 2
LS+162,1	Lesertyp externer Leser 7 (ab TCL Version 5.5)	Tabelle 3
LS+163,2	Status externer Leser 8	Tabelle 2

Feldaufteilung	Bedeutung	Wertebereich
LS+165,1	Lesertyp externer Leser 8 (ab TCL Version 5.5)	Tabelle 3
LS+166,2	Status externer Leser 9	Tabelle 2
LS+168,1	Lesertyp externer Leser 9 (ab TCL Version 5.5)	Tabelle 3
LS+169,2	Status externer Leser 10	Tabelle 2
LS+171,1	Lesertyp externer Leser 10 (ab TCL Version 5.5)	Tabelle 3
LS+172,2	Status externer Leser 11	Tabelle 2
LS+174,1	Lesertyp externer Leser 11 (ab TCL Version 5.5)	Tabelle 3
LS+175,2	Status externer Leser 12	Tabelle 2
LS+177,1	Lesertyp externer Leser 12 (ab TCL Version 5.5)	Tabelle 3
LS+178,2	Status externer Leser 13	Tabelle 2
LS+180,1	Lesertyp externer Leser 13 (ab TCL Version 5.5)	Tabelle 3
LS+181,2	Status externer Leser 14	Tabelle 2
LS+183,1	Lesertyp externer Leser 14 (ab TCL Version 5.5)	Tabelle 3
LS+184,2	Status externer Leser 15	Tabelle 2
LS+186,1	Lesertyp externer Leser 15 (ab TCL Version 5.5)	Tabelle 3
LS+187,2	Status externer Leser 16	Tabelle 2
LS+189,1	Lesertyp externer Leser 16 (ab TCL Version 5.5)	Tabelle 3

**Ab TCL Version 5.5:**

Feldaufteilung	Bedeutung	Wertebereich
LS+190,4	Parametrierungskennung Leser 0	<ASCII-Konstante>
LS+194,4	Parametrierungskennung externer Leser 1	<ASCII-Konstante>
LS+198,4	Parametrierungskennung externer Leser 2	<ASCII-Konstante>
LS+202,4	Parametrierungskennung externer Leser 3	<ASCII-Konstante>
LS+206,4	Parametrierungskennung externer Leser 4	<ASCII-Konstante>
LS+210,4	Parametrierungskennung externer Leser 5	<ASCII-Konstante>
LS+214,4	Parametrierungskennung externer Leser 6	<ASCII-Konstante>
LS+218,4	Parametrierungskennung externer Leser 7	<ASCII-Konstante>
LS+222,4	Parametrierungskennung externer Leser 8	<ASCII-Konstante>
LS+226,4	Parametrierungskennung externer Leser 9	<ASCII-Konstante>
LS+230,4	Parametrierungskennung externer Leser 10	<ASCII-Konstante>
LS+234,4	Parametrierungskennung externer Leser 11	<ASCII-Konstante>
LS+238,4	Parametrierungskennung externer Leser 12	<ASCII-Konstante>
LS+242,4	Parametrierungskennung externer Leser 13	<ASCII-Konstante>
LS+246,4	Parametrierungskennung externer Leser 14	<ASCII-Konstante>
LS+250,4	Parametrierungskennung externer Leser 15	<ASCII-Konstante>
LS+254,4	Parametrierungskennung externer Leser 16	<ASCII-Konstante>

Feldaufteilung	Bedeutung	Wertebereich
LS+258,4	Firmwareversion Leser 0	<ASCII-Konstante>
LS+262,4	Firmwareversion externer Leser 1	<ASCII-Konstante>
LS+266,4	Firmwareversion externer Leser 2	<ASCII-Konstante>
LS+270,4	Firmwareversion externer Leser 3	<ASCII-Konstante>
LS+274,4	Firmwareversion externer Leser 4	<ASCII-Konstante>
LS+278,4	Firmwareversion externer Leser 5	<ASCII-Konstante>
LS+282,4	Firmwareversion externer Leser 6	<ASCII-Konstante>
LS+286,4	Firmwareversion externer Leser 7	<ASCII-Konstante>
LS+290,4	Firmwareversion externer Leser 8	<ASCII-Konstante>
LS+294,4	Firmwareversion externer Leser 9	<ASCII-Konstante>
LS+298,4	Firmwareversion externer Leser 10	<ASCII-Konstante>
LS+302,4	Firmwareversion externer Leser 11	<ASCII-Konstante>
LS+306,4	Firmwareversion externer Leser 12	<ASCII-Konstante>
LS+310,4	Firmwareversion externer Leser 13	<ASCII-Konstante>
LS+314,4	Firmwareversion externer Leser 14	<ASCII-Konstante>
LS+318,4	Firmwareversion externer Leser 15	<ASCII-Konstante>
LS+322,4	Firmwareversion externer Leser 16	<ASCII-Konstante>
LS+326,4	Eigenschaften Leser 0	“00000000“ – “FFFFFF“
LS+330,4	Eigenschaften externer Leser 1	“00000000“ – “FFFFFF“
LS+334,4	Eigenschaften externer Leser 2	“00000000“ – “FFFFFF“
LS+338,4	Eigenschaften externer Leser 3	“00000000“ – “FFFFFF“
LS+342,4	Eigenschaften externer Leser 4	“00000000“ – “FFFFFF“
LS+346,4	Eigenschaften externer Leser 5	“00000000“ – “FFFFFF“
LS+350,4	Eigenschaften externer Leser 6	“00000000“ – “FFFFFF“
LS+354,4	Eigenschaften externer Leser 7	“00000000“ – “FFFFFF“
LS+358,4	Eigenschaften externer Leser 8	“00000000“ – “FFFFFF“
LS+362,4	Eigenschaften externer Leser 9	“00000000“ – “FFFFFF“
LS+366,4	Eigenschaften externer Leser 10	“00000000“ – “FFFFFF“
LS+370,4	Eigenschaften externer Leser 11	“00000000“ – “FFFFFF“

Feldauftteilung	Bedeutung	Wertebereich
LS+374,4	Eigenschaften externer Leser 12	“00000000“– “FFFFFFFF“
LS+378,4	Eigenschaften externer Leser 13	“00000000“– “FFFFFFFF“
LS+382,4	Eigenschaften externer Leser 14	“00000000“– “FFFFFFFF“
LS+386,4	Eigenschaften externer Leser 15	“00000000“– “FFFFFFFF“
LS+390,4	Eigenschaften externer Leser 16	“00000000“– “FFFFFFFF“
LS+394,2	Spannung für Türöffner (INTUS ACM8, INTUS ACM8(0)e Rack, INTUS ACM8(0)e Wand)	’12’, ’24’ oder ’NV’
LS+396,2	Spannung für Leser (INTUS ACM8, INTUS ACM8(0)e Rack, INTUS ACM8(0)e Wand)	’12’, ’24’ oder ’NV’

**Tabelle 1****INTUS 3000 und INTUS 2000 Status LS+37,2**

Bei jedem Zugriff auf LS+37,2 wird ein interner Batterietest ausgeführt.

<b>Wert</b>	<b>Terminal Status</b>
'OK'	Batterie einwandfrei; Akku einwandfrei (nur INTUS ACM4(0) AKKU)
'01'	Batterie leer; Akku einwandfrei (nur INTUS ACM4(0) AKKU)
'02'	Batterie einwandfrei; Akku einwandfrei (nur INTUS ACM4(0) AKKU), Anlaufmodus (CV+8,1) ist Warmstart aber kein TCL-Applikationsprogramm geladen, wird nach Laden und Reset gelöscht
'03'	Batterie leer; Akku einwandfrei (nur INTUS ACM4(0) AKKU), Anlaufmodus (CV+8,1) ist Warmstart aber kein TCL-Applikationsprogramm geladen, wird nach Laden und Reset gelöscht
'04'	Batterie einwandfrei; Akku leer (nur INTUS ACM4(0))
'05'	Batterie leer; Akku leer (nur INTUS ACM4(0))
'06'	Batterie einwandfrei; Akku leer (nur INTUS ACM4(0)); Anlaufmodus (CV+8,1) ist Warmstart aber kein TCL-Applikationsprogramm geladen, wird nach Laden und Reset gelöscht
'07'	Batterie leer; Akku leer (nur INTUS ACM4(0)); Anlaufmodus (CV+8,1) ist Warmstart aber kein TCL-Applikationsprogramm geladen, wird nach Laden und Reset gelöscht
'10'	Batterie in der Speichererweiterung (PCMCIA-SRAM Karte) schwach (INTUS 3300/3500, INTUS 3400, INTUS 3000 ACM, INTUS ACM8)
'20'	Batterie in der Speichererweiterung (PCMCIA-SRAM Karte) leer (INTUS 3300/3500, INTUS 3400, INTUS 3000 ACM, INTUS ACM8)
'40'	Die Firmware CompactFlash Karte muss getauscht werden. (INTUS ACM8e)
'80'	Die Daten CompactFlash Karte muss getauscht werden. (INTUS ACM8e)

**Tabelle 2****Status externer Leser LS+142,2 - LS+187,2 und LS+40,2 - LS+67,2**

Bei jedem Zugriff auf eines dieser Teilstufen wird eine Abfrage nach dem Status des ausgewählten, externen Lesers gestartet.

<b>Wert</b>	<b>Status externer Leser</b>
'OK'	externer Leser in Ordnung
' '	externer Leser nicht vorhanden /konfiguriert Statusabfrage noch nicht beendet
'00'	Bitbusmodul ausgefallen (Leitrechner) (nur für INTUS 2000)
'01'	Batterie leer
'02'	Defektmeldung vom externen Leser (Status)
'03'	externer Leser offline
'04'	externer Leser online, Verschlüsselung ist aktiv, aber der Schlüssel ist falsch: deshalb keine Kommunikation möglich. (ab TCL Version 5.5)

**Tabelle 3****Lesertyp externer Leser LS+144,1 - LS+189,1**

Aus der Vielzahl der Werte von '0' - '9' und 'A' - 'Z' für Lesertypen sind hier nur die Werte für externe Leser mit Lese-/Schreibfunktionalität aufgeführt. Weitere Werte sind auf Anfrage bei PCS erhältlich.

<b>Wert</b>	<b>Lesertyp externer Leser</b>
'A'	Lese-/Schreibfunktionalität für Legic prime und advant Medien, z.B. INTUS 600
'D'	Lese-/Schreibfunktionalität für Mifare classic und DESFire EV1 Medien, z.B. INTUS 600
'H'	Lese-/Schreibfunktionalität für Hitag 1 und 2 Medien, z.B. INTUS 600

**Tabelle 4-6 für Treiberfehler:**

Bei Fehlern in Treibern gibt es die Möglichkeit einen Sprung auszulösen.

Das Sprungziel kann in LS+92,4 eingetragen werden.

**Tabelle 4****Kennzeichen des Treibers LS+97**

<b>Wert</b>	<b>Bedeutung</b>
'1'	V.24-Treiber, BSC-Slave-Treiber
'5'	Fifo-Treiber
'.'	Partyline/Bitbus-Treiber
'?'	Eingabe-Treiber
'@'	Consol-Treiber
'A'	User-Treiber
'G'	Dido-Treiber
'S'	System (INTUS ACM4 AKKU, INTUS ACM40 AKKU und INTUS ACM8, INTUS ACM8(0)e Rack, INTUS ACM8(0)e Wand)

**Tabelle 4.1****INTUS ACM4 AKKU und INTUS ACM40 AKKU:**

<b>LS+97</b>	<b>LS+98</b>	<b>LS+99</b>	
'S'	'V'	'0'	Ausfall der Netzspannung
		'1'	Rückkehr zur Netzspannung
	'A'	'0'	Akku nicht vollständig geladen
		'1'	Akku geladen

**Tabelle 4.2****INTUS ACM8, INTUS ACM8e Rack und Wand:**

<b>LS+97</b>	<b>LS+98</b>	<b>LS+99</b>	
'S'	'L'	'0'	Leserspannung, siehe LS+396,2
	'T'	'0'	Türöffnerspannung, siehe LS+394,2
	'F'	'0'	Nur INTUS ACM8e Rack und Wand: Firmware-Flash hat Fehler -> tauschen
		'1'	Speicher-Flash hat Fehler -> tauschen

**Tabelle 5****Kennzeichen der Schnittstelle LS+98**

<b>Wert</b>	<b>Bedeutung</b>
'1'	Kanal 1/A
'2'	Kanal 2/B
'3'	Kanal 3/C
'4'	Kanal 4/D
'5'	Kanal 5 (nicht INTUS 3000)
'6'	Kanal 6 (nicht INTUS 3000)
'V'	Meldungen zur Versorgungsspannung (INTUS ACM4(0) AKKU)
'A'	Meldungen zum Akku (INTUS ACM4(0) AKKU)
'L'	Leserspannung (INTUS ACM8, INTUS ACM8(0)e Rack, INTUS ACM8(0)e Wand)
'T'	Türöffnerspannung (INTUS ACM8, INTUS ACM8(0)e Rack, INTUS ACM8(0)e Wand)

**Tabelle 6****Treiberfehlercode LS+99**

<b>Wert</b>	<b>Fehlercode</b>
'0'	Akku nicht vollständig geladen (INTUS ACM4(0) AKKU)
	Ausfall der Netzspannung (INTUS ACM4(0) AKKU)
'1'	Puffer Überlauf (Empfangspuffer voll, Zeichen/Datensatz verworfen) (V.24-Treiber, Partyline-Treiber)
	Akku geladen (INTUS ACM4(0) AKKU)
	Rückkehr zur Netzspannung (INTUS ACM4(0) AKKU)
'3'	overrun error (V.24-Treiber)
'4'	Ungültige Uid, maximale Satzlänge überschritten (Partyline-Treiber)
'7'	BSC-Slave online (BSC-Slave-Treiber) (INTUS 5600, 5540, 5500, 5200 ab TCL 6.70)
'8'	parity error, framing error, break detected (V.24-Treiber)
'9'	BSC-Slave offline (BSC-Slave-Treiber) (INTUS 5600, 5540, 5500, 5200 ab TCL 6.70)

**Tabelle 7****Status des Eingabeprozesses LS+107,2 - LS+139,2**

Das 1.Byte ist das höherwertige Byte. Mehrere Werte können mit logischem ODER verknüpft werden. Darüber hinaus werden beide Bytes vom TCL-System mit "30" geodert, um schließlich lesbare ASCII-Zahlen zu erhalten.

1.Byte	Status der Eingaben
'1'	Tastatur Eingabe aktiviert
'2'	Eingabe in B-Feld aktiviert
'4'	Eingabe in M-Feld oder I-Feld aktiviert
2.Byte	
'1'	Eingabefreigabe, wartet auf Lesung
'2'	Freigabe oder R,E konnte noch nicht gesendet werden (nicht für INTUS 3000)
'4'	Freigabe wurde noch nicht gesendet (nicht für INTUS 3000)
'8'	Externer Leser ging offline.

**Beispiel:**

Im normalen Betrieb enthält das Feld LS+109,2 den Wert '71', wenn für den externen Leser an der LBus Adresse 1 die Eingabe für Tastatur, Barcode und Magnetkarte freigegeben wurde.

Nach einem Offline-Gehen des externen Lesers wird solange '79' angezeigt, bis eine Lesung erfolgte.

## 4.21 LZ Gesamtaufzeit

LZ
LZ+1
----
LZ+7

Das LZ-Feld ist 8 Bytes lang.

Die Anweisung C1 (Abschnitt 5.8) berechnet die Zeitdifferenz der Felder AE und AB in Sekunden und trägt sie rechtsbündig in das Feld LZ ein.

Ansonsten kann das Feld beliebig verwendet werden.

## 4.22 M Magnetkartenleser und weitere Kartenleser

Wenn eine Eingabe über

- Magnetkartenleser
- Proximityleser
- Biometrieleser in Kombination mit Proximityleser  
für die Verifikation mit „TemplateOnCard“  
(siehe auch Identifikation mit Biometrieleser in Abschnitt 4.17)
- Speicherchipkartenleser
- Leser mit OMRON-Emulation

mit der Anweisung E freigegeben wurde, werden die Kartendaten und weitere Informationen nach einer Lesung in das M-Feld eingetragen.

**Hinweis zum Biometrieleser in Kombination mit Proximityleser für die Verifikation mit „TemplateOnCard“:**

Ob Biometriedaten geprüft wurden, kann über Codekennung (siehe unten) und gegebenenfalls Fehlercode (siehe Abschnitt 9.4) überprüft werden.

Feldaufteilung (88 Byte)	Feldaufteilung (115 Byte)	Bedeutung
M,80	M,107	Datenfeld
M+80,1	M+107,1	Konfiguration
M+81,1	M+108,1	Lesefehler
M+82,1	M+109,1	Codekennung
M+83,1	M+110,1	Lesernummer
M+84,3	M+111,3	Zeichenanzahl
M+87,1	M+114,1	Leserichtung

Das Magnetkartenleserfeld M hat eine Länge von 88 oder 115 Bytes, je nach Einstellung im Setup bzw. im CV-Feld (CV+43,1). Die Einstellung auf 115 Byte Länge ist nur bei Verwendung von einem Magnetkarten Spur3 Leser notwendig, da diese Spur maximal 107 Zeichen zulässt.

**Datenfeld: M,80 oder M,107**

Maximal 80 bzw. 107 Zeichen können als Daten der Lesung abgelegt werden.

**Konfiguration: M+80 oder M+107**

Bevor eine Lesereingabeanweisung ausgeführt wird, muss das Konfigurationsbyte belegt sein. Es gibt an, ob ein Eingabesprung auch bei Fehllesungen erfolgen soll oder nicht:

- '0' ist die Voreinstellung. Bei korrekter Lesung wird der Sprung ausgeführt. Eine Fehllesung wird ignoriert, die Eingabefreigabe ist weiter aktiv.
- '1' Der Sprung wird auch bei Fehllesung ausgeführt, die Lesefehler- und Codekennungsbytes werden entsprechend gesetzt. Sofern die Fehllesung Daten erzeugt hat, stehen diese zur weiteren Analyse zur Verfügung.

**Lesefehler: M+81 oder M+108**

Wenn eine Eingabe erfolgt und das Konfigurationsbyte auf '1' gesetzt ist, kann über das Lesefehlerbyte festgestellt werden, ob die Lesung fehlerfrei war.

- |     |  |
|-----|--|
| '0' | Gutlesung  |
| '1' | Fehlerfall. Das Codekennungsbyte enthält den Fehlercode. |

**Codekennung: M+82 oder M+109**

Bei einer gültigen Lesung enthält das Codekennungsbyte den jeweiligen Code einer Lesung.

Codekennung	Magnetkarte
'V'	Proximitykarte mit Biometriesegment und Biometriedaten
'W'	Proximitykarte mit Biometriesegment aber ohne Biometriedaten
'X'	Magnetkarte Spur 1
'Y'	Proximitykarte, Speicherchipkarte, Magnetkarte Spur 2
'Z'	Magnetkarte Spur 3

Im Fehlerfall enthält das Codekennungsbyte den Fehlercode (siehe Abschnitt 9.4 *Fehlercodes bei Lesereingaben*).

**Lesernummer: M+83 oder M+110**

Die Lesernummer gibt an, von welchem Leser die Eingabe erfolgte. Da das Lesernummerfeld nur eine Speicherzelle lang ist, werden für die externen Leser nicht nur Ziffern, sondern auch Buchstaben verwendet.

Lesernummer	Leser
'0'	interner Leser
'1'-'8'	externer Leser am LBus1
'9', 'A'-'G'	externer Leser am LBus2
'C'	Leser an Kanal B über DNCIN0 Prozess
'D'	Leser an Kanal C über DNCIN1 Prozess
'E'	Leser an Kanal D über DNCIN2 Prozess

Um die Nummer des Lesers, an der die Eingabe erfolgte, herauszufinden, kann auch P20+24,2 verwendet werden: Wenn eine Eingabe abgeschlossen ist, und der Eingabesprung in den Interpretieranweisungspuffer \$3 eingetragen wird, wird in P20+24,2 die Lesernummer als 2-stellige ASCII Zahl hinterlegt. Diese Information ist bis zur nächsten Stopanweisung aktuell.

**Zeichenanzahl: M+84,3 oder M+111,3**

Das Teilstück enthält rechtsbündig die Anzahl der gelesenen Zeichen als 3-stellige ASCII-Dezimalzahl.

**Leserichtung: M+87 oder M+114**

Das Leserichtungsbyte gibt die Leserichtung an. Bei Magnetkarten ist keine Rückwärtslesung möglich.

Leserichtung	Bedeutung
'0'	nicht erkannt
'1'	vorwärts (von links nach rechts)

Bei Lesern an den seriellen Zusatzschnittstellen kann die Leserichtung nicht erkannt werden.

### Inhalt des M-Feldes nach einem Terminal-Reset

Wird der Anlaufmodus Warm- oder Kaltstart ausgeführt, ist das Datenfeld M,80 oder M,107 mit Leerzeichen ' ' gefüllt. Die Teilstücke ab M+80 oder M+107 werden mit Nullen '0' vorbesetzt.

### Wiegandleser am INTUS ACM4 Wiegand und INTUS ACM40 Wiegand oder am Wiegand-Modul/intern (4-polige RIA-Klemme/B3100-010):

Anhand der Schalterstellung auf dem Wiegand-Modul und der Anzahl der Datenbits werden die Formate erkannt.

Aufbau der Lesung im Datenfeld des M-Felds:

Feldaufteilung	Bedeutung
M,1	Formatkennung
M+1,3	Anzahl der empfangenen Impulse
ab M+4	Nutzdaten in dezimaler oder hexadezimaler Schreibweise

Die Inhalte sind in einem separaten Dokument beschrieben und auf Anfrage bei PCS erhältlich.

### Wiegandleser im INTUS 3100, 3200, 3300 und 3500 mit Adapterplatine (6-polige Klemme/B3100-006):

Unterstützung ab TCL Version 5.05 und 5.07.

Es werden 5 Leser-Formate für Wiegand-Leser erkannt, bei denen die Nutzinformation bereits extrahiert im M-Feld auszulesen ist:

- Wiegand 2801-Format (28-Bit; Wiegand, Casi-Rusco etc.); Typcode '0'
- Wiegand 4001-Format (40-Bit; Casi-Rusco); Typcode '1'
- 33-Bit Format (Simons&Voss); Typcode '2'
- Wiegand 26 Bit Format (HID 10301); Typcode '3'
- Wiegand 4401-Format (44-Bit; Casi-Rusco); Typcode '4'

Wenn keines dieser Formate erkannt wird (Typcode '9'), dann werden die übertragenen Rohdaten in ASCII-Hex Format ('0'-'9', 'A'-'F') im M-Feld abgelegt. Parity-Auswertung und die Extraktion der Nutzdaten muss dann vom TCL-Programm vorgenommen werden.

Generell werden der Lesung der Typcode und die übertragene Anzahl der Bits auf der Wiegand-Schnittstelle vorangesetzt. Der Typcode ist ein Zeichen in M+0,1 und die Anzahl der Bits wird als dreistellige ASCII-Dezimalzahl in M+1,3 abgelegt. Ab M+5 beginnen die Daten der Lesung. Wenn die Daten der Lesung Bitfolgen enthält, die in ASCII-Hex dargestellt werden, dann werden diese Bitfolgen von rechts (LSB) her in ASCII-Hex Nibbles umgewandelt und wenn notwendig links (MSB) mit Nullen aufgefüllt.

Zum Beispiel wird eine sechsstellige Binärzahl 101011 so in eine zweistellige ASCII-Hex Darstellung '2B' umgewandelt.

Eine Lesung im Wiegand 2801-Format wird wie folgt im M-Feld abgelegt:

M+0,1	M+1,3	M+4,3	M+7,5
'0'	'028'	3 Byte ASCII-Dez Firmencode	5 Byte ASCII-Dez Kartennummer

Eine Lesung im Wiegand 4001-Format hat die Aufteilung:

<b>M+0,1</b>	<b>M+1,3</b>	<b>M+4,6</b>	<b>M+10,6</b>
'1'	'040'	6 Byte ASCII-Dez Firmencode	6 Byte ASCII-Dez Kartennummer

Eine Lesung im 33-Bit Format (Simons&Voss) erzeugt folgende Struktur:

<b>M+0,1</b>	<b>M+1,3</b>	<b>M+4,5</b>	<b>M+9,5</b>
'2'	'033'	5 Byte ASCII-Dez Firmencode	5 Byte ASCII-Dez Kartennummer

Eine Lesung im Wiegand 26-Bit-Format wird wie folgt im M-Feld abgelegt:

<b>M+0,1</b>	<b>M+1,3</b>	<b>M+4,3</b>	<b>M+7,5</b>
'3'	'026'	3 Byte ASCII-Dez Firmencode	5 Byte ASCII-Dez Kartennummer

Eine Lesung im Wiegand 4401-Format hat die Aufteilung:

<b>M+0,1</b>	<b>M+1,3</b>	<b>M+4,7</b>	<b>M+11,7</b>
'4'	'044'	7 Byte ASCII-Dez Firmencode	7 Byte ASCII-Dez Kartennummer

Das P1-Parity-Flag wird derzeit nicht ausgewertet!

Wenn das Format einer Lesung nicht bestimmt werden kann, dann ergibt sich folgende Aufteilung:

<b>M+0,1</b>	<b>M+1,3</b>	<b>M+4,yy</b>
'9'	'xxx'	yy Byte ASCII-Hex Lesserdaten (roh)

Dabei ist yy das ganzzahlige Ergebnis der Formel  $(xxx+3)/4$ . Man beachte, dass Störungen eventuell eine Lesung des Typcodes '9' durch Hinzufügen oder Übergehen von Bits aus den anderen Formaten erzeugen können. Bei den Formaten '0' bis '2' und '4' werden die darin definierten Parity-Bits ausgewertet (außer P1 in Format '4') und ergeben im Fehlerfall (Wert '1' in M+81 bzw. M+108) den Wert 'A' in M+82 bzw. M+109. Eine Lesung des Typs '9' kann keinen Fehler erzeugen.

Bei einer Wiegand-Lesung sollte deshalb neben dem Fehlerbyte auch die korrekte Typkennung und Bitanzahl in M+0,4 überprüft werden, ehe die Lesung als korrekt weiterverarbeitet wird.

## 4.23 MI MONIN-Steuerfeld

Feldauflistung	Bedeutung
MI,1	Datensatzformat für \$6
MI+1,1	Timeout \$6 und \$1
MI+2,1	Datensatzformat für \$9
MI+3,1	Timeout \$9 und \$1
MI+4,1	Datensatzformat für \$B
MI+5,1	Timeout \$B und \$1
MI+6,1	Umleitung
MI+7,1	Display Ausgabe

Das MI-Feld ist 8 Bytes lang.

Zuordnung für die zusätzlichen seriellen Schnittstellen:

serielle Schnittstelle	Sendepuffer	MONIN-Adressbyte
Kanal B	\$6	'N'
Kanal C	\$9	'O'
Kanal D	\$B	'P'

### Datensatzformat: MI,1 und MI+2,1 und MI+4,1

Diese Bytes legen das Datensatzformat der Datensätze fest, die zu den seriellen Schnittstellen in die Senderingpuffer übertragen werden sollen. Die maximale Datensatzlänge beträgt inklusive Adressbyte, gegebenenfalls Satzlänge und CR-Zeichen 256 Byte.

Wert	Datensatzformat
'0'	Format 1: <Adressbyte> <Daten> <CR>
'1'	Format 2: <Adressbyte> <Satzlänge> <Daten> <CR>

#### Format 1:

Im Format 1 müssen Daten an Kanal B, C oder D ein CR-Zeichen enthalten, da hier das CR-Zeichen als Satztrennzeichen benötigt wird.

#### Format 2:

Im Format 2 wird die Satztrennung anhand einer anzugebenden Satzlänge (dreistellige ASCII-Zahl) durchgeführt. Der MONIN-Prozess kopiert genau die angegebene Anzahl von Zeichen in den Senderingpuffer \$6, \$9 oder \$B, alle weiteren Zeichen bis zum nächsten CR-Zeichen werden verworfen. Innerhalb der Satzlänge enthaltene CR-Zeichen werden nicht als Satzende interpretiert, sondern mit übertragen.

#### Timeout: MI+1,1 und MI+3,1 und MI+5,1

Wenn Datensätze von der Hostschnittstelle zu den seriellen Schnittstellen übertragen werden, wird sowohl der Empfangspuffer Hostschnittstelle \$1 wie auch der jeweilige Sendepuffer mit einem Timeout von ca. 20 Sekunden überwacht.

Die Bytes MI+1, MI+3 und MI+5 zeigen das Ergebnis der Timeout-Überwachung an. Sie müssen mit einer TCL-Anweisung wieder auf '0' gesetzt werden.

- |     |   |
|-----|---|
| '0' | kein Timeout  |
| '1' | Timeout Sendepuffer \$6, \$9 oder \$B: Wenn der jeweilige Sendepuffer für mehr als ca. 20 Sekunden blockiert ist, wird der nächste Datensatz verworfen, damit keine Blockade des Empfangspuffers \$1 auftritt. Zusätzlich wird eine Fehlermeldung ausgegeben. |
| '2' | Timeout Empfangspuffer \$1: Es kommen für mehr als ca. 20 Sekunden keine Daten mehr vom Leitrechner. Zusätzlich wird eine Fehlermeldung ausgegeben.   |

#### **Umleitung: MI+6,1**

Manche Leitrechner sind nicht in der Lage die notwendigen Adressbytes für den MONIN-Prozess den Datensätzen voranzustellen. Deshalb werden abhängig vom Wert des Umleitungsbytes alle vom Host kommenden Daten an eine bestimmte Adresse weitergegeben.

Wenn diese Umleitung erfolgt, dann muss das TCL-Programm an Stelle des MONIN-Prozesses die Kommunikation mit dem Host übernehmen. Insbesondere werden vom MONIN Prozess keine Datensatzquittungen und keine Interpreteranweisungen mehr weitergeleitet!

Wert	Umleitung
'0'	reguläre Verarbeitung durch MONIN
'1'	\$1 --> \$7
'2'	TTY0 --> \$7
'3'	TTY0 --> \$6 (Kanal B)
'4'	TTY0 --> \$9 (Kanal C)
'5'	TTY0 --> \$B (Kanal D)
'6'	TTY0 --> Display
'7'	TTY0 --> Virtuelles Display (VI-Feld) (nicht für INTUS 3000)

TTY0 steht für den Empfangspuffer des Treiberprozesses Hostschnittstelle (TTYIN0), der Software unter dem MONIN Prozess.

#### **Display-Ausgabe: MI+7,1**

Das Byte steuert die Ausgabe auf das Terminal-Display.

Bit	Wert	Display-Ausgabe
0	0	Unterdrückung der Displayausgabe: Datensätze mit ungültigem Adressbyte nicht ins Display ausgeben.
	1	Voreinstellung: Datensätze mit ungültigem Adressbyte ins Display ausgeben.
1	0	Voreinstellung: Die Verarbeitung erfolgt wie bisher durch MONIN.
	1	\$1 transparent ins Display: Alle im Ringpuffer \$1 empfangenen Zeichen ohne Interpretation direkt an den Displaytreiber ausgeben. Dadurch können VT100 Escapesequenzen direkt durch den Displaytreiber verarbeitet werden.

## 4.24 ND Nicht definierte Stillstandszeiten

ND
ND+1
----
ND+7

Das ND-Feld ist 8 Bytes lang.

Die Anweisung C0 (Abschnitt 5.7) berechnet die Zeitdifferenz aus den in den Feldern H2 und H1 angegebenen Uhrzeiten in Sekunden und addiert sie rechtsbündig in einem der Felder Gx (Abschnitt 4.15) bzw. ND auf. Der Feldindex  $x=0,..,11$  wird dabei aus H3 genommen. Wenn H3 einen Wert größer als 11 enthält, werden die Zeiten in ND aufaddiert. Sie dürfen 365 Tage nicht überschreiten.

Ansonsten ist das Feld ND beliebig zu verwenden.

## 4.25 Ox Digitaler Ausgang DO

Das Ox-Feld ist ein *<LBusindexfeld>*:

**O[<LBusadr>]<Zahlenkomp> | O<Index>**

Die digitalen Ausgänge DO an den abgesetzten Eingabestationen werden mit **O[<LBusadr>]** angesprochen. Die DOs des Terminals sind mit **O** oder **O[0]** zu erreichen.

Feldaufteilung	Bedeutung	Wert
Ox,1	digitaler Ausgang DO	'A' DO aus 'E' DO ein 'B' DO blinken (Blinkfrequenz ca. 1Hz)

Ox ist 1 Byte lang.

Der Zustand des digitalen Ausgangs ist nur bei einem bistabilen Relais auslesbar.

Digitale Ausgänge in den Terminals und Zutrittskontrollmanagern:

Modell	Digitale Ausgänge
INTUS 3100	O0
INTUS 3200 (ethertime)	O0
INTUS 3300 ethertime	keine
INTUS 3300	O0 und O1
INTUS 3400	O0
INTUS 3450-time INTUS 3460-time	keine
INTUS 3450 INTUS 3450-timeplus INTUS 3460-timeplus	O0
INTUS 3450-plus INTUS 3460-plus	O0 und O1
INTUS 3500 INTUS 3600 INTUS 3660	O0 und O1
INTUS 5205 INTUS 5200 ohne IO-Option INTUS 5300 PoE INTUS 5500, 5540 ohne IO-Option INTUS 5600 ohne IO-Option	keine
INTUS 5300	O0
INTUS 5200 mit IO-Option INTUS 5500, 5540 mit IO-Option	O0 und O1

INTUS 5600 mit IO-Option	
INTUS 3000 ACM	O0 und O1, sowie maximal 2 Leser-DI/DO-Module mit Adresse 0 und Adresse 1 oder 2: TCL Version 4, Version 5 und Version 6 bei voreingestellter Verkabelung MP/MP: Leser-DI/DO-Modul Adresse 0: O5-O12 Leser-DI/DO-Modul Adresse 1: O13-O20 Leser-DI/DO-Modul Adresse 2: O21-O28 TCL Version 6 bei Verkabelung PP/MP und PP/PP: siehe Kapitel 11.6
INTUS ACM4 INTUS ACM4 AKKU	O0 – O3, O5, O7, O21, O23 bei Verkabelung PP/PP O13, O15 bei Verkabelung PP/MP Zuordnung siehe Kapitel 11.2
INTUS ACM40 INTUS ACM40 AKKU	O0, O1, O2 bistabiles Relais. Behält den Zustand auch während eines Resets und bei Strom- ausfall bei. Der Zustand ist auslesbar. O5, O7, O21, O23 bei Verkabelung PP/PP O13, O15 bei Verkabelung PP/MP Zuordnung siehe Kapitel 11.1
INTUS ACM8 INTUS ACM8e Rack INTUS ACM8e Wand	O0 – O3, je nach Verkabelung: O5 – O20, O5 – O12 und O21 – O28, O5 – O35 nur alle ungeraden Indizes Zuordnung siehe Kapitel 11.3 und 11.4
INTUS ACM80e Rack INTUS ACM80e Wand	O0 – O2, O3 ist abhängig von der Schalterstellung im Zutrittskontrollmanager ein monostabiles oder ein bistabiles Relais. Das bistabile Re- lais behält den Zustand auch während eines Resets und bei Stromausfall bei. Der Zu- stand ist auslesbar.  je nach Verkabelung: O5 – O20, O5 – O12 und O21 – O28, O5 – O35 nur alle ungeraden Indizes Zuordnung siehe Kapitel 11.3 und 11.4

**Tabelle 4.6 – Digitale Ausgänge der Terminals**

Digitale Ausgänge in den abgesetzten Lesern:

Modell	Digitale Ausgänge
INTUS 300l/m, INTUS 305l ohne Zusatzmodul	keine

mit Relaismodul mit codiertem Relais	O[x]0 O[x]0 und O[x]1
INTUS 300ro	keine
INTUS 300h, INTUS 340h,	O[x]0
INTUS 350H	O[x]0
INTUS 400, INTUS 500, INTUS 600, INTUS 600 FP ohne IO-Modul mit IO-Modul	keine O[x]0 und O[x]1
INTUS 1500, INTUS 1600, INTUS FP	O[x]0

*Tabelle 4.7 – Digitale Ausgänge der abgesetzten Leser*

**Für abgesetzte Eingabestationen** sind die Ausgänge O[<LBusadr>]0 und O[<LBusadr>]1 reserviert, die jedoch nicht vorhanden sein müssen.

Die Aktivierungszeit der Ausgänge O5 bis O36 des ACM, wie auch der abgesetzten Leser lassen sich mit dem Feld P20+29,3 kontrollieren. Die Aktivierungszeit der internen, digitalen Ausgänge O0 - O4 werden nicht über P20+29,3 gesteuert. Nach Schalten eines DOs kann P20+29,3 verändert werden, ohne dass sich die Einschaltzeit ändert.



Achtung: Digitale Ausgänge, die über das Zeitlimit in P20+29,3 ausgeschaltet werden, haben beim Zurücklesen der Felder Ox immer noch den aktiven Zustand 'E' oder 'B'.

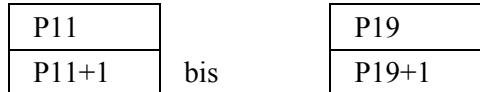
## 4.26 Px Parameterfelder

Das Px-Feld ist ein *<Indexfeld>*.

P0, P1, P2, P3, P10, P20, P21 und P22 haben besondere Bedeutungen und werden in den folgenden Abschnitten getrennt beschrieben.



P4 bis P9 sind jeweils 1 Byte lang und stehen zur freien Verfügung.



P11 bis P19 sind jeweils 2 Bytes lang und stehen zur freien Verfügung.

## 4.27 P0 Schlüsselschalterflag

Das Feld Px ist ein *<Indexfeld>*. Das P0-Feld hat eine spezielle Bedeutung:

Feldauftteilung	Bedeutung	Wert
P0,1	Schlüsselschalterstellung	'0' Schlüssel nicht betätigt '1' Schlüssel betätigt

P0 ("Peh-Null" nicht "Peh-Oh") ist 1 Byte lang.

Da das INTUS 3000 Terminal keinen Schlüsselschalter besitzt, wird auch P0 nicht verwendet.

Das TCL-Programm kann auch mit einer Eingabebeanweisung für die Funktionstaste F17 auf die Betätigung des Schlüssels warten (nur INTUS 2000).

## 4.28 P1 Notpuffer-Statusflag

Das Feld Px ist ein <Indexfeld>. Das P1-Feld hat eine spezielle Bedeutung:

Feldaufteilung	Bedeutung	Wert
P1,1	Notpufferstatus	'0' Notpuffer frei '1' Notpuffer voll
P1+1,4	Sprungziel	'0000' kein Sprung '0001'-'9999' Sprungziel

Das Notpuffer-Statusflag P1 ist 5 Bytes lang.

Die Sendeanweisung SE übergibt Datensätze in den Notpuffer \$4. Ist der Notpuffer voll, dann wird P1 auf '1' gesetzt. Sobald im Notpuffer wieder Platz für einen Datensatz ist (256 Bytes frei sind), wird P1 auf '0' gesetzt.

Wenn sich der Status des Feldes P1,1 ändert, wird das Sprungziel im Teilstück P1+1,4 angesprungen, sofern es von '0000' verschieden ist.

## 4.29 P2 Maschinentaskflag

Das Feld Px ist ein <Indexfeld>. Das P2-Feld hat eine spezielle Bedeutung:

Feldaufteilung	Bedeutung	Wert
P2,1	INTUS 2000: Maschinentask aktivieren	'0' deaktivieren '1' aktivieren
	INTUS 3000: DI-Ereignissprünge und zugehörige Zähler aktivieren; seit TCL Version 5.02 wird mit P2 auch die Taktüberwachung aktiviert (Tx und TAx)	'0' deaktivieren '1' aktivieren

P2 ist 1 Byte lang.

Solange im INTUS 3000 P2 auf '0' gesetzt ist, ist das TCL-Programm, z.B. in der Initialisierungsphase, vor ungewollten DI-Sprüngen, Zählerereignissen und Taktüberwachungereignissen geschützt. Erst wenn P2 auf '1' gesetzt wird, werden die Ereignissprünge für die digitalen Eingänge Ex, die zugehörigen Zähler Zx und Taktüberwachungen Tx, TAx aktiviert. Siehe auch die beiden Abschnitte 4.10 *Ex-Digitaler Eingang DI* und 4.49 *Zx Zähler zum digitalen Eingang*.

Bei Warm- und Kaltstart wird P2 auf '0' gesetzt.

Wenn P2 auf '0' gesetzt wird, werden schon ausgelöste DI-Sprünge aus dem Interpreteranweisungspuffer \$3 entfernt.

## 4.30 P3 Betriebsstatusflag

Das Feld Px ist ein <Indexfeld>. Das P3-Feld hat eine spezielle Bedeutung:

Feldaufteilung	Bedeutung	Wert
P3,1	Betriebsstatus	Tabelle 1
P3+1,4	Sprungziel	'0000' kein Sprung '0001'-'9999' Sprungziel

P3 ist 5 Bytes lang.

**Tabelle 1**

**Betriebsstatusflag P3,1**

Betriebsstatus	Wert	Zustand
Automatisches Senden:	'0'	Online-Betrieb, Normalbetrieb, automatisches Senden von Datensätzen
	'1'	Offline-Betrieb, Notbetrieb, Datensätze im Notpuffer zwischenspeichern
Senden auf Anforderung:	'2'	Einzelnen Datensatz senden und Warten auf positive Quittung
	'3'	Einzelübertragung eingestellt; einzelnen Datensatz positiv quittiert.
	'4'	Quittungstimeout bei Einzelübertragung

### Automatisches Senden:

#### P3,1 = '0': Online-Betrieb

Im Normalbetrieb, wenn das Terminal online am Leitrechner betrieben wird, wird ein neuer Datensatz, der mit der Anweisung SE oder WR,\$4 in den Notpuffer geschrieben wurde, sofort gesendet und auch vom Leitrechner quittiert.

#### P3,1 = '1': Offline-Betrieb

Wenn ein gesendeter Datensatz nicht innerhalb der eingestellten logischen Quittungszeit vom Leitrechner quittiert wird, geht das Terminal in den Offline-Betrieb, da vermutlich keine Verbindung zum Leitrechner besteht. Die logische Quittungszeit wird im Setup oder in CV+7,1 eingestellt.

Wenn der Leitrechner vorübergehend außer Betrieb ist, können im Terminal trotzdem Daten erfasst und im Notpuffer gespeichert werden, bis der Leitrechner wieder betriebsbereit ist. Die Notpuffergröße ist auf den erwarteten Bedarf im Setup bzw. CV+5,1 einstellbar. Das Ereignis, dass der Notpuffer voll ist, kann mit dem Notpuffer-Statusflag P1 abgefangen werden.

Der erste nicht quittierte Satz wird im Abstand der logischen Quittungszeit solange wiederholt, bis er quittiert wird. Daraufhin geht das Terminal wieder in den Normalbetrieb, d.h. P3,1 wird auf '0' gesetzt.

Wenn der Notpuffer voll war, d.h. das Notpuffer-Statusflag P1,1 = '1' gesetzt wurde, wird nun, sobald in den Notpuffer ein Datensatz maximaler Länge (256 Byte) passt, P1 wieder auf '0' gesetzt.

**Senden auf Anforderung:****Zustandsübergänge zwischen '2', '3' und '4'**

In manchen Fällen, z.B. wenn das INTUS 3000 Terminal für die Kommunikation mit dem Host an ein Modem angeschlossen ist, ist es unerwünscht, dass die Datensätze automatisch vom MONOUT-Prozess gesendet und wiederholt werden, bis sie quittiert sind. In solchen Fällen kann das Senden von Datensätzen aus dem Notpuffer mit TCL-Anweisungen gesteuert werden.

**P3,1 = '3': Einzelübertragung; Datensatz positiv quittiert.**

Wenn das Flag P3,1 auf '3' gesetzt wird, dann wird das automatische Senden unterbunden. Für das Senden auf Anforderung ist P3,1 = '3' der Grundzustand. Er wird durch eine positive Quittung des einzelnen Datensatzes wieder erreicht.

Nach P3,1 = '3' darf der Betriebsstatus nicht auf '0', sondern er muss auf '1' zurückgesetzt werden.

**P3,1 = '2': Einzelnen Datensatz senden**

Wenn ein Datensatz gesendet werden soll, muss der TCL-Programmierer P3,1 auf '2' setzen.

Wenn der gesendete Datensatz negativ quittiert wurde, wird er sofort wiederholt. P3,1 bleibt auf '2'.

Wenn der Satz positiv quittiert wurde, wird das Flag P3,1 vom MONOUT-Prozess wieder auf '3' gesetzt.

**P3,1 = '4': Quittungtimeout bei Einzelübertragung**

Wenn die Satzquittung nicht innerhalb der logischen Quittungszeit empfangen wurde, wird P3,1 vom MONOUT-Prozess auf '4' gesetzt.

Um den nächsten Satz zu senden, muss P3,1 wieder auf '2' gesetzt werden.

**Ereignissprung beim Betriebsstatuswechsel:**

Wenn ein Sprungziel im Teilstellfeld P3+1,4 eingetragen ist, wird bei jeder Betriebsstatusänderung in P3,1 ein Sprung ausgelöst.

**Hinweise:**

Das automatische Senden sollte nicht wechselweise mit dem Senden auf Anforderung verwendet werden. Die Sendeunterbrechung mit P3,1 = '3' funktioniert in beiden Fällen.

Das P3-Feld wird durch einen Warmstart und die Resetanweisung R,S nicht beeinflusst.

## 4.31 P10 MONOUT-Steuerfeld, Routinginformation

Das Feld Px ist ein <Indexfeld>. Das P10-Feld hat eine spezielle Bedeutung:

Das P10-Feld enthielt früher nur die Routinginformation für den Leersatz und ist jetzt treffender als MONOUT-Steuerfeld zu bezeichnen.

Der MONOUT-Prozess, und damit der Einsatz des P10-Feldes, wird im Abschnitt 6.3 beschrieben.

Das Teilstück P10+14,1, das Datensatz-Trennzeichen, wird immer ausgewertet.

Der Rest des MONOUT-Steuerfeldes P10 wird nur ausgewertet, wenn der MONOUT-Prozess die Sendedatensätze formatiert, d.h. P20+22,1 = '0' gesetzt ist.

Feldaufteilung	Bedeutung	Wert
P10,8	Routing Bytes des Leersatzes	'0000 0000' Voreinstellung (linksbündig zu lesen)
P10+8,1	Anzahl der Routing Bytes des Leersatzes (für alle Datensätze zu verwenden)	'0'-'8' '2' Voreinstellung
P10+9,1	Position der logischen Satznummer im Datensatz	Tabelle 1
P10+10,1	nach dem Notbetrieb: Leersatz als letzten Satz, wenn der Notpuffer leer ist, senden	'1' Leersatz senden, Voreinstellung sonst keinen L. senden
P10+11,1	Sätze mit Statusflag '*'	Tabelle 2
P10+12,2	Anzahl Sätze je Übertragungsblock	'01'-'25' '01' Voreinstellung
P10+14,1	Datensatz Trennzeichen im Übertragungsblock	CR-Zeichen ("0D") ist Voreinstellung

P10 ist 15 Bytes lang.

**Tabelle 1**

**Position der logischen Satznummer im Datensatz**

Wert	Bedeutung
'0'	Voreinstellung: Satznummer ab Position 0 vor der Routinginformation
'1' - '9': Wert, der größer oder gleich der Anzahl Routingbytes (P10+8,1) ist.	Satznummer ab Position <Wert>, üblicherweise direkt nach der Routinginformation.

**Tabelle 2****Sätze mit Statusflag '\*'**

Ein Datensatz, der als Statusflag ein '\*' Zeichen enthält, ist nur bei bestehender Online Verbindung zum Host ( $P3,1 = '0'$ ) relevant. Sonst, im Notbetrieb ( $P3,1 = '1'$ ), werden diese Datensätze aus dem Notpuffer gelöscht.

<b>Wert</b>	<b>Bedeutung</b>
'0'	Voreinstellung, Das ASCII-Zeichen '*' wird nicht als Statusflag interpretiert.
'1'	Das ASCII-Zeichen '*' wird als Statusflag erkannt.

**Datensatz Trennzeichen im Übertragungsblock:**

Ist das Trennzeichen nicht das CR-Zeichen ("0D"), so wird am Ende des Übertragungsblocks vom MONOUT-Prozess ein CR-Zeichen angefügt.

**Hinweis:**

Durch die Resetanweisung R,S wird das P10-Feld nicht verändert.

## 4.32 P20 Universelles Konfigurationsfeld

Das Feld Px ist ein *<Indexfeld>*. Das P20-Feld hat eine spezielle Bedeutung:

Mit dem Parameterfeld P20

- kann die Ausführung verschiedener TCL-Anweisungen beeinflusst werden.
- kann die Maschinentask und der MONOUT Prozess gesteuert werden.
- können Programmteile leichter gemeinsam verwendet werden.

Feldaufteilung	Bedeutung	Wertebereich
P20,1	<i>A</i> Anweisung: Anzahl Aktualisierungen	'0': Akt. 1 Feld gleichzeitig '1': Akt. 10 Felder gleichzeitig (für abgesetzte Eingabestationen wirkungslos)
P20+1,1	<i>RD</i> Anweisung: Anzahl gelesener Zeichen	'0': Anzahl unbekannt '1': Anzahl im EZ Feld
P20+2,2	<i>RD</i> Anweisung: Datentransparenz	Tabelle 1
P20+4,1	<i>M1</i> Anweisung: Prüfverfahren	'0': Modulo 10 Prüfung '1': Modulo 11 Prüfung
P20+5,1	<i>M1</i> Anweisung: Sonderzeichen	'*', 'x' oder '0' als Sonderzeichen bei Modulo 11
P20+6,1	<i>T</i> Anweisung: Zeiteinheit	'0': Zeiteinheit 100ms '1': Zeiteinheit 10 ms
P20+10,8	INTUS 2000: Modus der Bitbus DI/DO und Analog Module	
P20+18,1	INTUS 2000: Modus der Bitbus DI/DO und Analog Module	
P20+19,1	Maschinentask, DI-Sprünge	'0': Voreinstellung wegen Kompatibilität zu alten Versionen '1': DI-Sprünge aus \$3 werden verworfen, wenn die Maschinentask (P2) oder die DI (Ex+1,1) bereits deaktiviert ist.
P20+20,2	Nummer des DI, der den DI-Sprung oder den Zählersprung ausgelöst hat.	'00'-'99', aktuell bis zur nächsten Stopanweisung
P20+22,1	MONOUT: Formatierung der Sendedatensätze	'0': Voreinstellung: Formatierung der Sendedatensätze durch MONOUT '1': keine Formatierung der Sendedatensätze: MONOUT sendet transparent
P20+23,1	Display-Umschaltung	Tabelle 2 für INTUS 2000 Tabelle 3 für INTUS 3000
P20+24,2	Nummer des zum aktuellen Eingabesprung, DI-Sprung	'00' lokales Terminal '01'-'16' abgesetzte Eingabestation aktuell bis zur nächsten Stopanweisung

Feldauftteilung	Bedeutung	Wertebereich
	und Zähler-Sprung gehörenden abgesetzten Lesers.	
	Nummer des zum aktuellen Timeoutsprung gehörenden Timers.	'00'-'99' Timernummer aktuell bis zur nächsten Stopanweisung
P20+26,3	Einschaltdauer LEDs (nicht für lokale LEDs)	'001'-'254' * 400ms ' '(3 Leerz) ohne Begrenzung oder wie '000' maximal mögliche Einschaltdauer, abhängig vom Lesertyp
P20+29,3	Einschaltdauer DOs (nicht für lokale DOs, Ausnahme: O5-O36 des ACM)	'001'-'254' * 400ms ' '(3 Leerz) ohne Begrenzung oder wie '000' maximal mögliche Einschaltdauer, abhängig vom Lesertyp
P20+32,3	Einschaltdauer Hupe (nicht für lokale Hupe)	'001'-'254' * 400ms ' '(3 Leerz) ohne Begrenzung oder wie '000' maximal mögliche Einschaltdauer, abhängig vom Lesertyp
P20+35,3	Display Kontrast	000-064 2-zeiliges Display 000-064 240x64 Display 000-255 320x240 Display

**Ab TCL Version 5.01:**

P20+38,1	Größe SP-Feld	'0' SP ist 6 Byte groß '1' SP ist 8 Byte groß
P20+39,8	unkorrigierte Uhrzeit	Format wie UR-Feld: HH:MM:SS muss ebenso als Ganzes mit einer Kopieranweisung gesetzt werden.
P20+47,9	unkorrigiertes Datum	Format wie KT-Feld: mmddjjjj muss ebenso als Ganzes mit einer Kopieranweisung gesetzt werden.
P20+56,1	Sommerzeit-Flag (Status)	'0' Winterzeit '1' Sommerzeit
P20+57,5	GMT/UTC Abweichung	'-0720' – '+0720'
P20+62,1	Kontrolle der Sommer/Winterzeitumschaltung	'0' Winterzeit (fest) '1' Sommerzeit (fest) '2' automatische Umschaltung
P20+63,11	Start der Sommerzeit	Sommerzeitumschaltung mit Datum und Stunde: mmddjjjjhh Datumsformat mmddjjjj wie KT-Feld, Stundenangabe hh mit '00' – '23' Ab TCL Version 6.10: regelbasierte Sommerzeitumschaltung mit dem POSIX TZ-Format (Beschreibung des Formats siehe unten)

P20+74,11	Ende der Sommerzeit, Start der Winterzeit	Winterzeitumschaltung mit Datum und Stunde: mmmttwjjjjhh Datumsformat mmmttwjjjj wie KT-Feld, Stundenangabe hh mit '00' – '23' Ab TCL Version 6.10: regelbasierte Winterzeitumschaltung mit dem POSIX TZ-Format (Beschreibung des Formats siehe unten)
P20+85,1	Kontrolle des Netzwerk- Zeitabgleichs	'0' kein automatischer Abgleich Andere Werte sind reserviert.
P20+86,1	Status des Netzwerk-Zeitab- gleichs	'0' kein Netzabgleich erfolgt '1' Netzabgleich erfolgt

### POSIX TZ-Format

Ab TCL Version 6.10 kann die Sommer-/ Winterzeitumschaltung im P20-Feld regelbasiert angegeben werden. Die Zeitpunkte müssen dann nicht mehr jährlich angepasst werden.

Es wird das POSIX TZ-Format verwendet:

Mm.w.d/h Dies spezifiziert den Umschaltzeitpunkt als Tag *d* der Woche *w* im Monat *m* zur Stunde *h*.

Der Tag *d* muss zwischen 0 (Sonntag) und 6 (Samstag) liegen. Die Woche *w* muss zwischen 1 und 5 liegen; Woche 1 ist die erste Woche in der der Wochentag *d* auftaucht und Woche 5 ist die letzte Woche in der der Wochentag *d* vorkommt. Der Monat *m* darf Werte zwischen 1 und 12 annehmen, die Stunde *h* Werte zwischen 0 und 23.

Der gesamte Start- bzw. Endzeitpunkt muss in P20+63,11 bzw. P20+74,11 passen und ist mit Leerzeichen aufzufüllen.

Derzeit gilt für Zentraleuropa:

- Die Sommerzeit beginnt am letzten Sonntag im März um 2 Uhr:  
IK,'M3.5.0/02 ', P20+63:
- Die Sommerzeit endet am letzten Sonntag im Oktober um 3 Uhr:  
IK,'M10.5.0/03 ', P20+74:

Die Felder in dem Bereich P20+39,48 sind in Abschnitt 6.6 Sommer-/ Winterzeitumschaltung genauer beschrieben.

### Tabelle 1

#### RD Anweisung: Datentransparenz P20+2 und P20+3

Das Lesen aus einem Ringpuffer ist transparent, wenn das CR-Zeichen nicht als Datensatzbegrenzer gewertet wird.

Der Ringpuffer wird transparent gelesen, wenn das entsprechende Bit gesetzt ist.

In der Tabelle wird das Bitmuster als Hexadezimalwert angegeben. Die Werte können mit logischem Oder verknüpft werden.

**P20+2:**

Ringpuffer	Bitmuster für Transparenz	Voreinstellung
\$1	"02"	"00"
\$2	"04"	"00"
\$5	"20"	"00"
\$6	"40"	"00"
\$7	"80"	"00"

**P20+3:**

Ringpuffer	Bitmuster für Transparenz	Voreinstellung
\$8	"01"	"00"
\$9	"02"	"00"
\$A	"04"	"00"
\$B	"08"	"00"
\$C	"10"	"10"
\$D	"20"	"20"

**Tabelle 2****INTUS 2000 Display-Umschaltung P20+23,1**

Wert	Spaltenzahl	Zeilenzahl
'1' (default)	40	6
'2'	40	8
'3'	40	16
'4'	30	16
'5'	30	8

**Tabelle 3****INTUS 3000 Display-Umschaltung für 240x64 Pixel Display P20+23,1**

Wert	Font	Spaltenzahl	Zeilenzahl
'1' (default)	6x8	40	8
'2'	6x16	40	4
'3' (wie '1')	6x8	40	8
'4'	8x8	30	8
'5'	8x16	30	4

Nach der Display-Umschaltung ist das Display gelöscht, die Ausgaben müssen wiederholt werden.

Änderungen an Zeichensätzen und Zeichenmapping sind zurückgesetzt. Siehe Abschnitt 7.3.

### 4.33 P21 Setupsteuerung

Das Feld Px ist ein *<Indexfeld>*. Das P21-Feld hat eine spezielle Bedeutung:

Feldaufteilung	Bedeutung	Wertebereich
P21,1	Setup Bedienung	'0' nicht möglich '1' möglich (Default, nach IR,S oder Kaltstart) <u>Ab TCL Version 5.5:</u> '0' nicht möglich '1' Setup Ebene 1 möglich '2' Setup Ebene 2 möglich '3' Setup Ebene 1+2 möglich '4' Setup Ebene 3 möglich '5' Setup Ebene 1+3 möglich '6' Setup Ebene 2+3 möglich '7' Setup Ebene 1+2+3 möglich (Default)
P21+1,1	Setup aktiv	'0' Setup inaktiv '1' Setup aktiv <u>Ab TCL Version 5.5:</u> '0' Setup inaktiv '1' Setup Ebene 1 '2' Setup Ebene 2 '3' Setup Ebene 3
P21+2,4	Sprungziel	'0000' kein Sprung '0001'-'9999' Sprungziel

P21 ist 6 Bytes lang.

Wenn ein Sprungziel im Teilstfeld P21+2,4 eingetragen ist, wird es bei jeder Änderung von P21+1,1 angesprungen.

## 4.34 P22 Erweiterte Benutzerschnittstelle

Das Feld Px ist ein <Indexfeld>. Das P22-Feld hat eine spezielle Bedeutung:

Einführung in TCL Version 6.50, erweitert in TCL Version 6.60.

Feldaufteilung	Modul	Bedeutung	Wertebereich
P22,8	Grafikprozess	Maskenumschaltung (ab TCL Version 6.50)	<ASCII-Konstante> Dateinamen der Maske ohne die Dateinamenerweiterung '.qml', mit Leerzeichen aufgefüllt. Voreinstellung: 'default'
P22+8,20		ID aus Maskenarchiv	<ASCII-Konstante>
P22+28,4		Revision aus Maskenarchiv	<ASCII-Konstante>
P22+32,1		Validierung des Maskenarchivs	'1' ok, '0' nicht validiert
P22+33,20	Grafikprozess oder TCL Firmware	ID aus Keyboarddatei	<ASCII-Konstante>
P22+53,4		Revision aus Keyboarddatei	<ASCII-Konstante>
P22+57,1		Validierung der Keyboarddatei	'1' ok, '0' nicht validiert
P22+58,8	INTUS Sound	reserviert	'STOP' führt zum Abbruch der Audio-Ausgabe
P22+66,20		ID aus Audioarchiv	<ASCII-Konstante>
P22+86,4		Revision aus Audioarchiv	<ASCII-Konstante>
P22+90,1		Validierung des Audioarchivs	'1' ok, '0' nicht validiert
P22+91,3		Lautstärke Einstellung in Prozent	'000'-'100' Voreinstellung: '070'

P22 ist 94 Bytes lang.

### Grafikprozess

Der Grafikprozess verwendet Masken zur

- Ausgabe ins Display
- Eingabe über Funktionstasten
- Eingabe über Tastatur
- Display und Touch (oberer Bereich über dem Display) können über die TCL-Funktionalität hinaus verwendet werden.

Eine Maske wird in einer QML-Datei definiert.

Das Handbuch *INTUS 5600 Grafik Handbuch – QML und Qt Creator (Bestellnummer D5600-003)* führt in die Erstellung der Masken und die Anbindung an TCL ein.

**Defaultmaske 'default' :**

Im INTUS 5600 steht immer die Defaultmaske 'default' zur Verfügung. Sie bildet ein 240x64 Pixel Display nach und stellt je nach Wert in P20+23,1 die entsprechende Anzahl Zeilen und Spalten dar. Der voreingestellte Wert '1' in P20+23,1 führt zu 8 Zeilen und 40 Spalten auf dem Display. Diese Defaultmaske wird verwendet, wenn eine TCL-Applikation geladen ist, und keine anderen Masken im Terminal zur Verfügung stehen.

Das Defaultprogramm des INTUS 5600 verwendet die eigene Maske 'defprog'.

Verwendet ein TCL-Programm eigene Masken, müssen sie in einem Maskenarchiv '\*.igma' vorab mit INTUS RemoteConf ins Terminal geladen werden.

**Maskenumschaltung:**

Das Teilstück P22,8 steuert die Maskenumschaltung des Grafikprozesses: wenn der Dateinamen der Maske ohne die Erweiterung '.qml' in P22,8 kopiert wird, schaltet der Grafikprozess sofort auf diese Maske um und stellt sie im Display dar. Falls der Dateinamen kürzer als 8 Zeichen ist, muss P22,8 mit Leerzeichen aufgefüllt werden. Ist die Datei mit dem angegebenen Namen nicht im Terminal vorhanden, generiert der Interpreter den Fehler 287.

**Beispiel:**

`IK, 'shcolors', P22:`

Der Grafikprozess schaltet auf die Maske, die in der Datei shcolors.qml definiert ist, um und zeigt sie an.

**INTUS Sound**

INTUS Sound steuert das optionale INTUS Sound Modul im INTUS 5200, 5500, 5540 und INTUS 5600.

Verwendet ein TCL-Programm Soundausgaben, müssen die zugehörigen Dateien in einem INTUS Sound Audioarchiv '\*.iaa' vorab mit INTUS RemoteConf ins Terminal geladen werden.

## 4.35 PO Prozedurstatusflag

Feldauftteilung	Bedeutung	Wert
PO,1	Prozedurstatus (Protokollstatus)	'0' Online zum Host '1' Offline zum Host '2' INTUS 2000: Ethernet-Controller ausgefallen  <u>Ab TCL Version 5.5:</u> '0' Online zum Host, Datentransfer möglich '1' Offline zum Host '3' Online zum Host, Datentransfer nicht möglich, da noch kein Login auf der Hostschnittstelle erfolgt ist.
PO+1,4	Sprungziel	'0000' kein Sprung '0001'-'9999' Sprungziel
PO+5,1	Anmeldestatus	<u>Ab TCL Version 5.5:</u> '0' nicht angemeldet '1' einfacher Zugriff '2' administrativer Zugriff

Das PO-Feld ("Peh-Oh" nicht "Peh-Null") ist 6 Bytes lang.

Das PO-Flag zeigt an, ob auf Protokollebene eine Verbindung zum Leitrechner besteht (Online) oder nicht (Offline).

Ab TCL Version 5.5 kann über das PO-Feld auch abgefragt werden, ob ein Login auf der Hostschnittstelle erfolgt ist, und wenn ja mit welcher Berechtigungsstufe. Die Konfiguration des Login-Dialogs erfolgt im CV-Feld (ab CV+132, siehe Abschnitt 4.6). Im Abschnitt 6.4.1 ist dieser Schutz der Hostschnittstelle beschrieben.

Beim BSC-Protokoll wird nach Ablauf des Poll Timeout (einstellbar im Setup oder in CV+15,1) das PO-Flag auf '1' (Offline) gesetzt. Wenn das Terminal wieder angepollt wird, dann wird das Flag auf '0' zurückgesetzt.

Beim TTY-Protokoll wird das PO-Flag durch die SR Anweisung nach Ablauf des dort spezifizierbaren Timeouts gesetzt und durch den MONIN Prozess bei Empfang eines Zeichens zurückgesetzt.

Bei einer Verbindung über TCP/IP wird das PO-Flag nach Verbindungsaufbau auf '0' gesetzt, bei Verbindungsabbau auf '1'.

Mit dem PO-Flag lässt sich der Terminalbetriebsstatus (Online, Offline) in der Regel schneller feststellen als mit dem P3-Flag.

Auf die Veränderung des PO-Flags kann mit einem Sprung reagiert werden, indem in PO+1,4 ein Sprungziel ungleich '0000' eingetragen wird.

#### **4.36 PN Personalnummer**

PN
PN+1
----
PN+9

Das PN-Feld hat eine Länge von 10 Bytes.

Dieses Feld kann beliebig verwendet werden. PN kann eine Personalnummer aufnehmen.

#### **4.37 S S-Feld**

S
S+1
----
S+254

Das S-Feld hat eine Länge von 255 Byte.

Es steht dem Anwender frei zur Verfügung.

## 4.38 Sx Schnittstellenparameter

Das Sx-Feld ist ein *<Indexfeld>*. S1, S2, S3 und S4 enthalten die Parameter der Schnittstellphysik und der Schnittstellentreiber für alle Protokolle (Prozeduren).

Die Zuordnung der Schnittstellen des Terminals zu den Sx-Feldern:

Terminalschnittstelle	Teilfeld für das Protokoll in CV	Schnittstellen-Parameterfeld
Kanal A, Hostschnittstelle	CV+31,1	S1
Kanal B	CV+32,1	S2
Kanal C	CV+33,1	S3
Kanal D	CV+34,1	S4

Achtung: Beim INTUS ACM8 und INTUS ACM8(0)e Rack gilt:

- S2 für die oben eingebauten Leserschnittstellen
- S3 für die unten eingebauten Leserschnittstellen.

Die Schnittstellenparameterfelder Sx sind eine Kopie der Setup-Parameter. Jedes der hier beschriebenen Teilstücke kann im Setup eingestellt werden. Eine detailliertere Erklärung der Felder, die möglichen Werte und die Voreinstellungen sind in den Betriebshandbüchern zu finden. In den folgenden Tabellen wird auf die Namen der zugehörigen Menüpunkte Bezug genommen.

Die Bedeutung der Teilstücke ist abhängig von dem an der jeweiligen Schnittstelle eingestellten Protokoll. Das kann zurzeit TTY, BSC oder TCP/IP sein (siehe CV+31,1 bis CV+34,1). Wenn das Protokoll (Prozedur) im Setup geändert wird, ändert sich auch die Bedeutung der Teilstücke. Für die einzelnen Protokolle ist immer nur ein Teil des Sx-Feldes von Bedeutung.

Undokumentierte Teilstücke sind reserviert. Sowohl sie, als auch reservierte Bits in dokumentierten Teilstücken, dürfen nicht verändert werden.

Die Werte der anderen Teilstücke können mit TCL-Anweisungen geändert werden. Bei einem Terminal-Reset werden sie dann in das EEPROM übernommen. Dies ist identisch zum Vorgehen bei Änderungen von Setup-Parametern im CV-Feld (Abschnitt 4.6).

In den Tabellen werden Werte für einzelne Bits als Hexadezimalzahl angegeben. Die Werte können mit logischem Oder in die Teilstücke von Sx eingetragen werden.

Für das TTY-Protokoll gilt folgende Aufteilung des Sx-Felds:

**TTY-Protokoll (-Prozedur):**

Feldauf- teilung	Bits	Bedeutung	Wertebereich oder Setupmenü
Sx+30,1		Empfangspuffergröße	Puffergröße Empf.
Sx+31,1		Sendepuffergröße	Puffergröße Senden
Sx+32,1	0..6	reserviert	
	7	Empfang: EOL ignorieren	Empf. Ignor. EOL
Sx+33,1	0	Empfang: EOL→CR	Empf. EOL→CR
	1	reserviert	
	2	Empfang: XON/XOFF ein/aus	Empf. XON/XOFF
	3	reserviert	
	4	Senden: XON/XOFF ein/aus	Senden: XON/XOFF
	5	Empfang: Zeichen ignorieren	Empf. Zeichen Unterdrük- cken
	6,7	reserviert	
Sx+34,1	0	Sendeverarbeitung ein/aus	Senden Verarb.
	1	reserviert	
	2	Senden: CR → EOL	Senden CR→EOL
	3	Modemsteuerung 1	Senden RTS/CTS
	4	Modemsteuerung 2	Senden RTS/CTS/DTR/DCD
	5..7	reserviert	
Sx+35,1		reserviert	
Sx+36,1	0..3	Baudrate	Baudrate (Tabelle 1)
	4..5	Anzahl Datenbits	Datenformat (Tabelle 2)
	6	Stop Bits	Datenformat "00" 1 Stop Bit, "40" 2 Stop Bits
	7	reserviert	muss "80" sein
Sx+37,1	0	Parity en-/disable	Datenformat "00" Disable, "01" Enable
	1	Parity Typ	Datenformat "00" Even, "02" Odd
	2..6	reserviert	
	7	reserviert	muss für das Fingerprint- modul in INTUS 3100 und INTUS 5300 gesetzt sein (ab TCL V5.31, V6.00)
Sx+38,1	0	reserviert	

Feldaufteilung	Bits	Bedeutung	Wertebereich oder Setupmenü
	1	Empfangsverarbeitung ein/aus	Empf. Verarb.
	2..7	reserviert	
Sx+39,1	0..2	reserviert	
	3	Zweidrahtsteuerung	"08" (siehe Zu Sx+39,1:)
	4..7	reserviert	
Sx+42,1		Empfang: Lösch-Zeichen	Empf. Löschzeichen
Sx+44,1		Empfang: EOF/ Counter	Empf. EOF/ Counter
Sx+45,1		Empfang: EOL1/ Timer	Empf. EOL1
Sx+46,1		Empfang: EOL2-Zeichen	Empf. EOL2
Sx+47,1		Senden: EOL1-Zeichen	Senden EOL
Sx+48,1		Senden: EOL2-Zeichen	Senden EOL
Sx+49,1		Empfang: Ignoriere-Zeichen	Empf. Ignorezeichen
Sx+50,1	0	DTR	"00" low, "01" high
	1	RTS	"00" low, "02" high
	2, 3	reserviert	
	4	Setzen von DTR ist möglich (abhängig von einer evtl. eingestellten Modemsteuerung, vgl. Sx+34,1)	
	5	Setzen von RTS ist möglich (abhängig von einer evtl. eingestellten Modemsteuerung, vgl. Sx+34,1)	
	6, 7	reserviert	
Sx+51,1		Nur INTUS 5300 FP, Kanal C (S3+51,1): Auswahl Kommunikation über COM A oder COM B	'1' COM A für Leser '2' COM B für Sensor
		Nur INTUS ACM8, INTUS ACM8(0)e Rack und Wand, Kanal B und C (S2+51,1 und S3+51,1): Auswahl der Leserschnittstelle (nur Endtestunterstützung)	'1', '2', '3', '4'

**Zu Sx+39,1:**

In der TCL Version 4.22 wurde die Möglichkeit eingebaut, eine serielle Zusatzschnittstelle aus TCL über den TTY-Treiber in einer RS485-Zweidrahtkonfiguration zu betreiben. Bei dieser Konfiguration wird der RS485-Sendebaustein vor jedem Senden eingeschaltet und danach wieder ausgeschaltet. Weiterhin werden die während des Sendens

empfangenen Daten bereits im Treiber verworfen. Die Sendedaten werden also nicht ge“echo“t.

Dazu konfiguriert man zunächst im Setup die Zusatzschnittstelle auf den TTY-Modus und erlaubt weder Software- (XON/XOFF) noch Hardware-Handshake (RTS/CTS bzw. RTS/CTS/DTR/DCD).

Weiterhin muss das RS485 Schnittstellenmodul, das in INTUS 3300, 3500 und im INTUS 3000 ACM verbaut ist, für den Zweidrahtbetrieb konfiguriert sein: die beiden mit „2-Draht“ bezeichneten Steckbrücken (Jumper) müssen gesteckt sein, und einer der beiden aus drei Steckbrücken bestehenden Konfigurationsblöcke RX bzw. TX sollte auf allen drei Positionen, „BIAS“ (2x) und „Term.“ mit Steckern kurzgeschlossen sein. Keinesfalls dürfen alle beide Konfigurationsblöcke RX und TX mit (insgesamt sechs) Steckbrücken kurzgeschlossen werden.

Zur Aktivierung der Zweidrahtsteuerung muss man das Bit 4 in Sx+39 setzen, etwa mit dem TCL-Kommando

OR, "08", S2+39, S2+39:

welches dieses Bit für die erste Zusatzschnittstelle setzt (S2=Kanal B, S3=Kanal C). Diese Änderung sollte sofort (nach dem ersten Senden) wirken.

### **Warnung:**

Da die Änderung in Sx auch in den Setup-Parametern eingetragen wird, jedoch weder mit Hilfe von INTUS RemoteSetup noch im Setup-Menü angezeigt wird, kann die Zweidrahtsteuerung eigentlich nur sicher nach einem Eiskalt- oder Comstart (oder Beschreiben von Sx) wieder ausgeschaltet werden. Daher: bei merkwürdigen Effekten im Handshake-Bereich einen Eiskalt- oder Comstart ausführen.

### **Zu Sx+50,1:**

Wird eine serielle (Zusatz-)Schnittstelle im TTY-Modus mit XON/XOFF betrieben, so können die Signale DTR und RTS der jeweiligen Schnittstelle manuell angesteuert werden. Dies geschieht über das Sx+50,1-Feld. Sx+50,1 wird bei einem Warmstart zurückgesetzt.

Der Programmierer muss berücksichtigen, dass ein gesetztes Bit den Pegel des TTL-Signals auf high setzt, dieser aber durch nachgeschaltete Treiberbausteine möglicherweise invertiert werden kann.

Für das BSC-Protokoll gilt folgende Aufteilung des Sx-Felds:

**BSC-Protokoll (-Prozedur):**

Feldauft- teilung	Bits	Bedeutung	Wertebereich oder Setupmenü
Sx+30,1		Empfangspuffergröße	Puffergröße Empf.
Sx+31,1		Sendepuffergröße	Puffergröße Senden
Sx+32,1		reserviert	
Sx+33,1		reserviert	
Sx+34,1		reserviert	
Sx+35,1		reserviert	
Sx+36,1	0..3	Baudrate	Baudrate (Tabelle 1)
	4, 5	Anzahl Datenbits	Datenformat (Tabelle 2)
	6	Stop Bits	Datenformat "00" 1 Stop Bit, "40" 2 Stop Bits
	7	reserviert	muss "80" sein
Sx+37,1	0	Parity en-/disable	Datenformat "00" Disable, "01" Enable
	1	Parity Typ	Datenformat "00" Even, "02" Odd
	2..7	reserviert	
Sx+38,1		reserviert	
Sx+39,1		reserviert	
Sx+40,1		Group-ID	Group id (Tabelle 3)
Sx+41,1		Device-ID	Device id (Tabelle 3)
Sx+42,1		PAD-Anzahl	PAD Anzahl
Sx+43,1		Poll-Timeout	Poll Timeout
Sx+44,1		Daten-Timeout	Daten Timeout
Sx+45,1		Verzögerungs-Timeout	Sendeverz.
Sx+46,1		Antwort-Timeout	Quit.-Timeout
Sx+47,1		EOL	"0d" für Carriage Return (für INTUS FP)
Sx+48,12		reserviert	
Sx+60,1		Online-Status des BSC-Slaves (INTUS 5600, 5540, 5500, 5200 ab TCL 6.70)	'1' Slave online '0' Slave offline

**Tabelle 1**

für TTY und BSC: Sx+36,1 Baudrate in den Bits 0..3

<b>Wert</b>	<b>Baudrate</b>	<b>Wert</b>	<b>Baudrate</b>
"00"	Reserviert	"01"	50
"02"	75	"03"	110
"04"	134.5	"05"	150
"06"	200	"07"	300
"08"	600	"09"	1200
"0A"	1800	"0B"	2400
"0C"	4800	"0D"	9600
"0E"	19.2 Kbaud	"0F"	38.4 Kbaud

In der Tabelle wird das Bitmuster für die Bits 0 - 3 als Hexadezimalwert angegeben. Die Werte können mit logischem Oder in Sx+36,1 eingetragen werden.

**Tabelle 2**

für TTY und BSC: Sx+36,1 Anzahl Datenbits in den Bits 4..5

<b>Wert</b>	<b>Anzahl Datenbits</b>
"00"	5
"10"	6
"20"	7
"30"	8 (Voreinstellung)

In der Tabelle wird das Bitmuster für die Bits 4 und 5 als Hexadezimalwert angegeben. Die Werte können mit logischem Oder in Sx+36,1 eingetragen werden.

**Tabelle 3**

für BSC: Group-ID in Sx+40,1 und Device-ID in Sx+41,1:

<b>Wert</b>	<b>Bedeutung</b>
"00"	@
"01"	A
"02"	B
:	:
"1A"	Z

Für TCP/IP gilt folgende Aufteilung des Sx-Felds:

**TCP/IP:**

Feldaufteilung	Bits	Bedeutung	Wertebereich oder Setupmenü
Sx,1	0,1	IP-Stack	IP-Stack "00" IPv4 "01" IPv4+IPv6 "02" IPv6
	2..6	reserviert	
	7	Auflösung der Ziel-MAC-Adresse	"00" aus eingehendem Paket "80" über ARP-Request
Sx+1,16		IPv6 Terminal-IP-Adresse	IPv6 Adresse
Sx+17,1		IPv6 Präfixlänge	IPv6 Prefix
Sx+18,16		IPv6 Router-IP-Adresse	IPv6 Router
Sx+40,4		TCP/IP Stack Software-version	
Sx+44,6		MAC-Adresse (Ethernet, TokenRing)	
Sx+50,4		IPv4 Terminal-IP-Adresse	IPv4 Adresse, Terminal (IPA)
Sx+54,4		IPv4 Host-IP-Adresse	Host Adresse, Host (IPA)
Sx+58,2		Port-Nummer	Port-Nr.
Sx+61,4		IPv4 Subnetzmaske	IPv4 Netzmaske, IP-Maske
Sx+65,4		IPv4 Gateway-IP-Adresse	IPv4 Router, Gateway (IPA)
Sx+69,1		Verbindungsaubau	Verb.-Aufbau "00" passiv "01" aktiv "02" passiv/RAS
Sx+70,16		IPv6 Host-IP-Adresse	Host Adresse

Die Byte-Ordnung bei den IPv4-Adressen und der Port-Nummer ist nicht die "Motorola-Byte-Ordnung", die ansonsten im TCL-System gilt: Sonder das 1. Byte ist das niedwertigste Byte (LSB) und das letzte Byte ist das höchswertigste Byte (MSB).

#### 4.39 SP Offset (für die PT Anweisung)

SP
SP+1
----
SP+5
----
SP+7

Bis zur TCL Version 4.28 war das SP-Feld 6 Bytes lang. Ab TCL Version 5.01 ist SP abhängig von P20+38,1 6 Bytes lang ( $P20+38,1 = '0'$ ) oder 8 Bytes lang ( $P20+38,1 = '1'$ ). Entsprechend ergibt /SP/ den Wert 6 oder 8. Da P20+38,1 mit '0' initialisiert wird, sind neuere TCL Versionen voll kompatibel mit älteren Versionen. Jedoch kann eine Suche in einem TF-Feld, dass größer als 1MB ist, nur mit dem Parameter  $P20+38,1 = '1'$  korrekt durchgeführt werden.

In das SP-Feld trägt die Anweisung PT nach erfolgreicher Suche den Offset des nächsten Datensatzes als 6- oder 8-stellige ASCII-Zahl ein.

Ansonsten kann das Feld beliebig verwendet werden.

#### 4.40 ST Gesamtstillstandszeit

ST
ST+1
----
ST+7

Das ST-Feld ist 8 Bytes lang.

Die Anweisung C0 (Abschnitt 5.7) berechnet die Zeitdifferenz der Felder H2 und H1 in Sekunden und addiert sie rechtsbündig zu dem Wert im ST-Feld.

Ansonsten kann das Feld beliebig verwendet werden.

#### 4.41 T T-Feld

T
T+1
----
T+254

Das T-Feld hat eine Länge von 255 Byte.

Es steht dem Anwender frei zur Verfügung.

## 4.42 Tx Taktüberwachung

Das Tx-Feld ist ein *<Indexfeld>*. Es gibt T0 bis T13.

Feldaufteilung	Bedeutung	Wertebereich
Tx,3	Untergrenze Taktzeit	'000' – '999' in Einheiten von 100ms oder 1s, abhängig von Ex+4,1
Tx+3,2	Sprung wenn "positive" Flanke früher als Untergrenze (Tx,3) ist	'00' Funktion deaktiviert '01' – '99' Sprungziel
Tx+5,3	Obergrenze Taktzeit	'000' – '999' in Einheiten von 100ms oder 1s, abhängig von Ex+4,1
Tx+8,2	Sprung wenn "positive" Flanke später als Obergrenze (Tx+5,3) ist	'00' Funktion deaktiviert '01' – '99' Sprungziel

Tx ist 10 Bytes lang. Das Tx-Feld legt die Taktüberwachung für den lokalen digitalen Eingang des Ex-Felds mit demselben *<Index>* fest.

Ab TCL Version 5.02 werden die Tx-Felder für die Taktüberwachung verwendet, wenn die optionale Softwarefunktion "Taktüberwachung" mit der TCL Firmware installiert ist.

In Tx können die Taktunter- und obergrenzen samt den dazugehörigen Sprüngen getrennt angegeben werden. Eine oder beide Grenzüberwachungen kann deaktiviert werden, wenn das Sprungziel oder der Grenzwert Null ist.

Die in Tx angegebenen Sprünge werden nur ausgeführt, wenn tatsächlich "positive" (siehe Beschreibung des Ex-Felds in Abschnitt 4.10) Flanken eintreffen. Der Sprung für die Untergrenze wird ausgeführt, wenn eine "positive" Flanke in kürzerer Zeit als in Tx,3 angegeben nach der letzten "positiven" Flanke eintrifft.

Der Sprung für die Obergrenze wird ausgeführt, wenn eine "positive" Flanke in längerer Zeit als in Tx+5,3 angegeben nach der letzten "positiven" Flanke eintrifft.

Die im Tx-Feld angegebenen Sprünge sind nicht aktiv, wenn P2 den Wert '0' hat oder das dazugehörige Ex-Feld in Ex+1,1 den Wert '2'.

Nach dem Ober- oder Untergrenzensprung enthält das TOx-Feld (Abschnitt 4.46) die (absolute) Differenz zwischen Soll- und Ist-Zeit. Diese kann auch Null sein, da die interne Auflösung der Systemzeit eine geringe Spanne möglich macht, die durch Rundungen verschwindet.

Wenn eine "positive" Flanke ausbleibt, dann erfolgt kein Sprung für die Obergrenze. Dieser Fall kann mit Hilfe der im TAx-Feld (Abschnitt 4.43) angebbaren Zeitüberwachung abgefangen werden.

Wichtig für das Verständnis der Taktüberwachung ist auch die in Ex+5,1 festgelegte Startbedingung der Überwachung.

Für die ausgelösten Taktüberwachungssprünge gelten die gleichen Aussagen, wie für die DI-Sprünge (siehe Abschnitt 4.10). Insbesondere werden die Teilsteller P20+24,2 zu '00' (lokale DI) und P20+20,2 auf die Nummer des auslösenden DIs gesetzt.

Eine Beschreibung der Zusammenhänge der Taktüberwachungsfunktion findet sich in Abschnitt 6.5.

## 4.43 TAx Taktausfallüberwachung

Das TAx-Feld ist ein *<Indexfeld>*. Es gibt TA0 bis TA13.

Feldaufteilung	Bedeutung	Wertebereich
TAx,3	Taktausfallzeit	'000' – '999' in Einheiten von 100ms oder 1s, abhängig von Ex+4,1
TAx+3,2	Sprung wenn "positive" Flanke später als die Taktausfallzeit (TAx,3) ist	'00' Funktion deaktiviert '01' – '99' Sprungziel

TAx ist 5 Bytes lang. Das TAx-Feld legt Taktausfallüberwachung für den lokalen digitalen Eingang des Ex-Feld mit demselben *<Index>* fest.

Ab TCL Version 5.02 werden die TAx-Felder für die Taktüberwachung verwendet, wenn die optionale Softwarefunktion "Taktüberwachung" mit der TCL Firmware installiert ist.

In TAx kann die maximale Zeit angegeben werden, nach der auch ein Obergrenzensprung nicht mehr erwartet wird. Wenn die in TAx,3 angegebene Zeitspanne verstrichen ist, ohne dass eine "positive" Flanke eingetroffen ist, dann wird der in TAx+3,2 angegebene Sprung ausgelöst. Hat eines der beiden Teilstücke den Wert Null, dann ist die Taktausfallfunktion deaktiviert.

Wenn der Sprung ausgelöst wird, dann wird Ex+5,1 auf '0' gesetzt, was bedeutet, dass eine folgende "positive" Flanke die gesamte Taktüberwachung (Tx) neu starten kann. In Kompatibilität zum INTUS 2000 wird die Taktausfallüberwachung sofort neu gestartet, so dass weitere Taktausfallsprünge möglich sind. Um das zu verhindern, sollte entweder TAx+3,2 auf '00' gesetzt oder die gesamte DI-Funktion durch das Setzen von Ex+1,1 auf '2' deaktiviert werden (Achtung: Zähler Zx bleibt auch stehen).

Im INTUS 3000 ist das TAx-Feld unabhängig von dem Tx-Feld. Es kann schon alleine für eine Taktüberwachung einer maximalen Obergrenze verwendet werden. Beim INTUS 2000 musste zur Aktivierung der Vorgaben im TAx-Feld auch das Tx-Feld beschrieben werden, d.h. das TAx-Feld sollte vor dem Tx-Feld beschrieben werden. Dies ist im INTUS 3000 nicht mehr nötig.

Der im TAx-Feld angegebene Sprung ist nicht aktiv, wenn P2 den Wert '0' hat oder das dazugehörige Ex-Feld in Ex+1,1 den Wert '2'.

Wichtig für das Verständnis der Taktausfallüberwachung ist auch die in Ex+5,1 festgelegte Startbedingung der Überwachung.

Für die ausgelösten Taktausfallsprünge gelten die gleichen Aussagen, wie für die DI-Sprünge (siehe Abschnitt 4.10). Insbesondere werden die Teilstücke P20+24,2 zu '00' (lokale DI) und P20+20,2 auf die Nummer des auslösenden DIs gesetzt.

Eine Beschreibung der Zusammenhänge der Taktüberwachungsfunktion findet sich in Abschnitt 6.5.

## 4.44 TF Tabellenfeld

TF
TF+1
----
TF+...

Die Größe des Tabellenfeldes TF kann im Setup oder im CV-Feld CV+6,1 zusammen mit CV+108,1 eingestellt werden. Die Voreinstellung ist 48kB. Das TF-Feld kann aber durchaus erheblich erweitert werden. Bis TCL Version 4.28 war die Größe des TF-Felds auf 765kB beschränkt. Mit Hilfe der neuen CV-Teilfelder, P20+38,1 und dem vergrößerten SP-Feld sind Größen jenseits von 765kB seit TCL Version 5.01 möglich.

TF steht dem Anwender frei zur Verfügung. Die Daten im Feld TF sind batteriegepuffert. Sie gehen nach einem Netzausfall nicht verloren, wenn im Setup der Anlaufmodus auf Warmstart gestellt ist (CV+8,1) und ein Warmstart möglich ist.

Für /TF/ ist immer eine mindestens 8-stellige Operation zu verwenden.

## 4.45 TM Time - Uhrzeit im 12-Stunden-Format

Ab TCL Version 5.50.

Feldaufteilung	Bedeutung	Wert
TM,2	Stunde	'01'-'12'
TM+2,1	Trennzeichen	':' oder jedes andere darstellbare ASCII-Zeichen
TM+3,2	Minuten	'00'-'59'
TM+5,1	Trennzeichen	':' oder jedes andere darstellbare ASCII-Zeichen
TM+6,2	Sekunden	'00'-'59'
TM+8,2	Zusatz	'am', 'pm'

Das TM-Feld ist 10 Bytes lang.

Im TM-Feld kann die Uhrzeit im 12-Stunden Format ausgelesen werden. Das Feld hat immer die gleiche Zeit wie das UR-Feld.

UR- und TM-Feld verwenden das gleiche Uhrzeit-Trennzeichen.

### Stellen der Uhrzeit

Das TM-Feld muss als Ganzes mit einer Kopieranweisung gesetzt werden. Es darf keine Längenangabe und kein Offset verwendet werden. Die Zeichenfolge muss korrekt sein, es erfolgt nur eine eingeschränkte Plausibilitätsprüfung.

### Trennzeichen

Der Uhrenbaustein trägt jede Sekunde die aktuelle Uhrzeit in das Feld UR und TM ein. Das Trennzeichen wird im Sekudentakt durch ein Leerzeichen ersetzt. Dadurch entsteht bei Verwendung der Aktualisierungsanweisung A ein Blinkeffekt in der Displayanzeige.

### Lesen der Uhrzeit

Das Feld kann jederzeit gelesen werden. Dies geschieht etwa durch eine Kopieranweisung. Beim Lesen der Uhrzeit kann durch das blinkende Trennzeichen entweder das Trennzeichen oder ein Leerzeichen gelesen werden. Dies ist zu beachten, wenn die Uhrzeit an den Leitrechner gesendet und dort weiterverarbeitet wird.

Das TM-Feld besitzt Intelligenz, so dass für Kopieranweisungen mit unterschiedlicher Längenangabe eine möglichst vernünftige Uhrzeit geliefert wird.

### Beispiel:

DK, TM, D:	12:12:12am
DK, TM, D, 5:	12:12
DK, TM, D, 7:	12:12am
DK, TM, D, 8:	12:12:12
DK, TM, D, 10:	12:12:12am

## 4.46 TOx Taktabweichung

Das TOx-Feld ist ein *<Indexfeld>*. Es gibt **TO0** bis **TO13**.

Feldauftteilung	Bedeutung	Wertebereich
TOx,3	Taktabweichung	'000' – '999' in Einheiten von 100ms oder 1s, abhängig von Ex+4,1

TOx ist 3 Bytes lang. Das TOx-Feld zeigt die Taktabweichung für den lokalen digitalen Eingang des Ex-Felds mit demselben *<Index>* an. Es hat diese Funktion seit der TCL Version 5.02, wenn die optionale Softwarefunktion "Taktüberwachung" mit der TCL Firmware installiert ist.

Wenn einer der im Tx-Feld angebbare Unter- oder Obergrenzensprünge ausgelöst wird, dann findet sich im dazugehörigen TOx-Feld der absolute Betrag der Abweichung der Zeitspanne zwischen der "positiven" Flanke (bzw. des Beschreibens von Ex+5,1 mit '1') und des jeweiligen Grenzwerts. Aufgrund der höheren Auflösung der internen Systemzeitgeber kann der gerundete Betrag auch den Wert Null annehmen.

Der Inhalt des Felds ist undefiniert, wenn der im TAx-Feld (Abschnitt 4.43) angebbare Taktausfallsprung ausgelöst wird. Die Abweichung findet sich in diesem Fall in TAx,3.

Eine Beschreibung der Zusammenhänge der Taktüberwachungsfunktion findet sich in Abschnitt 6.5.

## 4.47 TR Trace

Es gibt eine Reihe von Testmöglichkeiten für TCL-Programme, die über das TR-Feld gesteuert werden können. Eine Beschreibung des Testkonzepts findet sich im Abschnitt 8.5 *TCL-Programm testen*.

Feldaufteilung	Bedeutung	Wert
TR,1	zuletzt durchlaufenes Sprungziel ins Display	falls '1', sonst '0'
TR+1,1	Tracepuffer (*) ins Display	falls '1', sonst '0'
TR+2,1	Tracepuffer (*) nur im Fehlerfall ins Display	falls '1', sonst '0'
TR+3,1	zuletzt durchlaufenes Sprungziel in den Sendepuffer für Kanal B \$6	falls '1', sonst '0'
TR+4,1	zuletzt durchlaufenes Sprungziel in den Hostsendepuffer \$2	falls '1', sonst '0'
TR+5,1	Verzögerung nach jeder TCL-Anweisung, um die Sprungziele im Display lesen zu können	<n> * ½ s, <n>= 0,...,9
TR+6,1	Ausgabesteuerung der Fehlermeldungen	Tabelle 1
TR+7,2	Routing bei Ausgabe in den Hostsendepuffer \$2	'00' keine Routingbytes sonst 2 ASCII-Zeichen
TR+9,1	Prozesskennung (**)	Tabelle 2
TR+10,3	letzter aufgetretener Fehler (**)	Siehe in den Abschnitten 9.1 bis 9.3.2
TR+13,4	Zusatzinformation (**)	
TR+17,4	letztes Sprungziel bevor der Fehler aufgetreten ist (wenn MONIN oder INTERP) (**)	
TR+21,3	Anweisungsnummer nach Sprungziel (wenn MONIN oder INTERP) (**)	
TR+24,3	Systemfehlernummer (Fehler bei einem System Call) (**)	
TR+27,4	Unterprogrammsprung, wenn ein Fehler aufgetreten ist	'0000' kein Unterprogrammsprung '0001'-'9999' Unterprogrammsprung
TR+31,1	erweiterte Fehlerprüfung	falls '1' werden semantische Fehlermeldungen, die mit '*' im Abschnitt 9.2 gekennzeichnet sind, ausgegeben, sonst '0'

TR ist 32 Bytes lang.

(\*) Der **Tracepuffer** enthält die letzten fünf durchlaufenen Sprungziele.

(\*\*) TR+9 bis TR+26 enthält die Bestandteile einer **TCL-Fehlermeldung** (siehe Abschnitt 8.5.1).

**Tabelle 1****Ausgabesteuerung der Fehlermeldungen TR+6**

Eine Addition der Werte ist zulässig, so dass eine Fehlermeldung auf mehreren Kanälen ausgegeben werden kann.

Wert	Ausgabe nach...
'0'	Fehlermeldung unterdrücken
'1'	Fehlermeldung auf Display (default)
'2'	Fehlermeldung nach \$2
'4'	Fehlermeldung nach \$6

**Tabelle 2****Prozesskennung in TR+9**

Wert	Prozess	Wert	Prozess
'A'	MONIN	'L'	TTYOUT2
'B'	INTERP	'M'	TTYOUT3
'C'	TIMEOUT	'N'	MASCH
'D'	EINGABE	'O'	MONOUT
'E'	AKTUELL	'P'	DEFAULT
'F'	TTYIN0	'Q'	DNCIN0
'G'	TTYIN1	'R'	DNCIN1
'H'	TTYIN2	'S'	DNCIN2
T	TTYIN3	'T'	BSCM-RX
'J'	TTYOUT0	'U'	BSCM-TX
'K'	TTYOUT1		

## 4.48 UR Uhrzeit

Feldaufteilung	Bedeutung	Wert
UR,2	Stunde	'00'-'23'
UR+2,1	Trennzeichen	':' oder jedes andere darstellbare ASCII-Zeichen
UR+3,2	Minuten	'00'-'59'
UR+5,1	Trennzeichen	':' oder jedes andere darstellbare ASCII-Zeichen
UR+6,2	Sekunden	'00'-'59'

Das UR-Feld ist 8 Bytes lang.

### Stellen der Uhrzeit

Das UR-Feld muss als Ganzes mit einer Kopieranweisung gesetzt werden. Es darf keine Längenangabe und kein Offset verwendet werden. Die Zeichenfolge muss korrekt sein, es erfolgt nur eine eingeschränkte Plausibilitätsprüfung.

### Trennzeichen

Der Uhrenbaustein trägt jede Sekunde die aktuelle Uhrzeit in das Feld UR ein. Das Trennzeichen wird im Sekudentakt durch ein Leerzeichen ersetzt. Dadurch entsteht bei Verwendung der Aktualisierungsanweisung A ein Blinkeffekt in der Displayanzeige.

### Lesen der Uhrzeit

Das Feld kann jederzeit gelesen werden. Dies geschieht etwa durch eine Kopieranweisung. Beim Lesen der Uhrzeit kann durch das blinkende Trennzeichen entweder das Trennzeichen oder ein Leerzeichen gelesen werden. Dies ist zu beachten, wenn die Uhrzeit an den Leitrechner gesendet und dort weiterverarbeitet wird.

### Beispiel:

DK, '10-15-00',UR:      stellt die Uhr auf 10 Uhr 15.

DK,UR,S:

## 4.49 Zx Zähler zum digitalen Eingang

Das Zx-Feld ist ein *<LBusindexfeld>*.

**Z [*<LBusadr>*] <Zahlenkomp> | Z<*Index*>**

Beim INTUS 3000 steht für jeden lokalen digitalen Eingang DI und für die DIs der externen Eingabestationen ein Zählerfeld Zx zur Verfügung.

Die Zähler für die DIs an den abgesetzten Eingabestationen werden mit **Z[<LBusadr>]** ange- sprochen. Die Zähler für die DIs des Terminals sind mit **Z** oder **Z[0]** zu erreichen.

Feldaufteilung	Bedeutung	Wert
Zx,7	Zähler	'0000000'-'9999999'
Zx+7,7	Obergrenze Zähler	'0000000'-'9999999'
Zx+14,2	Sprung bei Erreichen der Obergrenze	'00' kein Sprung '01'-'99' Sprungziel

Zx ist 16 Bytes lang.

Für die Anzahl der DIs und deren Funktion ist die Beschreibung des Feldes *Ex - Digitaler Eingang DI* (Abschnitt 4.10) hinzu zu ziehen.

Wenn P2 auf '1' gesetzt ist, werden die Zählerfelder Zx bei jedem Ereignis am entsprechenden digitalen Eingang automatisch inkrementiert.

Der Zähler eines DIs zählt die in Ex+1,1 eingestellte Flanke. Eine '2' in der Flankeneinstellung hält den Zähler an.

Wenn P2 auf '0' gesetzt wird, werden die DI-Sprünge deaktiviert und die Zähler angehalten. Die Maximalwerte aller Zähler Zx+7,7 werden auf '0000000' zurückgesetzt.

Wenn ein Ereignissprung durch das Erreichen der Obergrenze des Zählerfelds ausgelöst wird, dann wird das Sprungziel in Zx+14,2 auf '00' gesetzt. Die auslösende Zählernummer, die mit der DI-Nummer identisch ist, wird in P20+20,2 eingetragen. Die LBus Adresse der zugehörigen Eingabestation bzw. des Terminals steht in P20+24,2. Die Angaben sind bis zur nächsten Stopanweisung aktuell. Wenn als Obergrenze '0000000' eingetragen ist, dann wird ein Sprung ausgelöst, wenn eine positive Flanke beim Zählerstand von '9999999' eintrifft.

Um den auslösenden Sprung wieder zu aktivieren, sollte in der Ereignisbehandlungsroutine der Zähler und eventuell der Maximalwert neu gesetzt werden, ehe das Sprungziel wieder in Zx+14,2 eingetragen wird.

## 5 TCL-Anweisungen

### 5.1 @ Sprung

Syntax:

@<Label>:

Erklärung:

Die Bearbeitung des TCL-Programms wird an der Stelle der Sprunganweisung beendet und am Sprungziel <Label> fortgesetzt. Die Werte, die <Label> annehmen kann, werden per Setup oder in CV+63,1 festgelegt. Die Voreinstellung ist 0-1023.

Ein Sprung an den Anfang der aktuellen Zeile wird durch die Anweisung @@: spezifiziert.

**Beispiel 1: (absoluter Sprung)**

D#1: I,P12,0-99: @1: Sprung nach Sprungziel 1 (Endlosschleife).

**Beispiel 2: (indirekter Sprung)**

```
D#0: K,'11',P12:  
D#10: @ (P12,2):  
D#11: P,UR,<>'02',@14: SR,UR: K,'12',P12: @14:  
D#12: P,UR,<>'06',@14: SR,UR: K,'13',P12: @14:  
D#13: P,UR,<>'23',@14: SR,UR: K,'11',P12: @14:  
D#14: T10,@10: S:
```

In diesem Beispiel wird jede Sekunde die Uhrzeit überprüft und um 2.00, 6.00 und 23.00 Uhr ein Datensatz an den Host gesendet. Wenn ein Satz gesendet wurde, wird jeweils ein neues Sprungziel in P12 eingetragen und über den indirekten Sprung in #10 angesprungen. (Dadurch entfällt die Prüfung der Minute und Sekunde.)

### 5.2 ! Kommentar einfügen

Syntax:

!<Text>:

Erklärung:

Der <Text>, der auf das '!' Zeichen folgt, wird bis zum nächsten '!' vom Compiler als Kommentar gewertet.

Im Kommentar dürfen keine unbalancierten '-Zeichen und keine "-Zeichenpaare, die keine Hexadezimalzahlen enthalten, vorkommen.

Ein Nachteil bei der Verwendung dieser Kommentaranweisung ist, dass die Zeichen mit dem Programm ins Terminal übertragen werden, was zusätzliche Ladezeit kostet.

## 5.3 # Sprungziel

**Syntax:**

#<Dezimalzahl>:

**Erklärung:**

Sprungziele (Labels) sind keine ausführbaren Anweisungen. Sie dienen nur der Kennzeichnung einer Stelle in einem Programm. Der Zahlenbereich für <Dezimalzahl> wird per Setup oder in CV+63,1 festgelegt. Die Voreinstellung ist 0-1023.

Sprungziele sind immer eindeutig. Wird dasselbe Sprungziel öfters verwendet, gilt jeweils die letzte Verwendung dieses Sprungziels. Jede neue Verwendung eines Sprungziels belegt auch neuen Speicher im Programmbereich. Es entsteht eventuell "toter Code", der nicht mehr angesprungen werden kann.

**Hinweis für große Programme**

Bei großen Anwendungen kann es zum Engpass bei der Anzahl der Sprungziele kommen. In diesem Fall gibt es folgende Möglichkeiten, das Problem zu umgehen:

- Ein Sprungziel sollte nur vergeben werden, wenn es auch angesprungen wird. (Ein Sprungziel ist keine Zeilennummer!)
- Durch Verwendung von Unterprogrammen können Sprungziele eingespart werden.
- Mit der R,D Anweisung können Teile des TCL Programms überladen werden.
- Am Anfang von jedem Programmdatensatz steht ein internes Sprungziel. Durch die Sprunganweisung @@: kann daher an den Anfang der Programmzeile gesprungen werden.

**Beispiel:**

D ... : T1,@@: S:

ist äquivalent zu

D#10: ... : T1,@10: S:

## 5.4 % Unterprogrammsprung

**Syntax:**

```
%<Label>: Unterprogramm-Sprung  
%: Unterprogramm-Rücksprung
```

**Erklärung:**

Durch die Unterprogramm-Sprunganweisung wird das TCL Programm an dem Sprungziel <Label> fortgesetzt. Die Rücksprungadresse und die aktuelle Zeilennummer werden gerettet. Es ist eine Schachtelungstiefe von 16 Unterprogrammebenen möglich.

Die Werte, die <Label> annehmen kann, werden per Setup oder in CV+63,1 festgelegt. Die Voreinstellung ist 0-1023.

Durch den Unterprogramm-Rücksprung wird das Programm an der Stelle fortgesetzt, an der die zugeordnete Unterprogramm-Sprunganweisung stand.

Verwenden Sie keine Stopanweisung in einem Unterprogramm. Die Fortsetzung des Programmablaufs wird durch einen Ereignissprung bestimmt. Führt dieser Ereignissprung wieder in ein anderes Unterprogramm, kann ein unerwarteter Programmablauf erfolgen oder ein Programmstacküber- oder -unterlauf auftreten.

**Beispiel:**

```
D#0: F,P4,2,0: %100: S:  
D#100: K,E(P4,1),D+(P5,1),1:  
D#102: I,P4,0-7: I,P5,0-7:  
D#104: P,P5,='0',@106: @100:  
D#106: %:
```

Die Zustände der digitalen Eingänge E0-E7 werden ins Display geschrieben.

## 5.5 A Display aktualisieren

**Syntax:**

A:

**Erklärung:**

Die Aktualisierungsanweisung bewirkt, dass die Anzeige des **zuletzt** mit einer Kopieranweisung ins Display ausgegebenen Feldes aktualisiert wird, wenn sich der Feldinhalt ändert. Die A Anweisung muss unmittelbar nach der Kopieranweisung stehen. Der Quelloperand der Kopieranweisung muss ein (Teil-)Feld <Feldadr> sein. Die Angabe einer <Zeichenverknüpfung> als Quelloperand kann nicht aktualisiert werden.

Wieviel Aktualisierungen gleichzeitig möglich sind, bestimmt P20,1:

**P20,1 = '0' (Voreinstellung)**

Es ist nur eine Aktualisierung im Display möglich. Sie wirkt solange, bis eine neue Ausgabe in das Display erfolgt. (Dieser Modus ist kompatibel zu den PCT88x Terminals.)

**P20,1 = '1'**

Es ist möglich, durch mehrere Kopieranweisungen mit anschließenden Aktualisierungsanweisungen bis zu 10 Felder im Display des Terminals gleichzeitig zu aktualisieren. Die Aktualisierung wirkt solange, bis eine neue Ausgabe in das Teilstück des Displays erfolgt, in dem die Aktualisierung wirkt.

Ein Löschen des Displays oder Datensätze ohne gültiges Adressbyte, die von MONIN empfangen und ins Display ausgegeben werden, halten alle Aktualisierungen an.

Durch eine Tastatureingabe wird die laufende Aktualisierung nicht angehalten, wenn sie nicht ein Teilstück der Aktualisierung überschreibt.

**Abgesetzte Eingabestationen:**

Der Wert P20,1 = '1' ist für die Displays der abgesetzten Eingabestationen wirkungslos. Es ist immer nur eine Aktualisierung im Display möglich. Sie wirkt solange, bis eine neue Ausgabe in das Display erfolgt.

**Beispiel 1:**

**DK,UR,D: A: S:**

Die Anzeige der Uhrzeit auf dem Display wird aktualisiert.

**Beispiel 2:**

**DK,'1',P20,1:**

**DK,E0,D+2,1: A: K,E1,D+3,1: A: S:**

Die Werte der digitalen Eingänge 0 und 1 werden aktualisiert.

## 5.6 AD Addition

**Syntax:**

```
AD,<Zahlenwert>,<Zahlenwert>{,<Feldadr>{,<Anz>} } :
```

**Erklärung:**

Die erste Zahl <Zahlenwert> und die zweite Zahl <Zahlenwert> werden addiert. Die Operanden und das Ergebnis sind positive ganze Zahlen. Es können maximal 10-stellige Operanden ein ebenfalls 10-stelliges Ergebnis liefern.

Die Abspeicherung des Ergebnisses hängt von den weiteren optionalen Parametern ab:

- Es sind keine weiteren Parameter angegeben: Sind <Feldadr> und damit auch <Anz> nicht angegeben, steht das Ergebnis linksbündig im ER Feld. (Kompatibilität zu den PCT88x-Terminals)
- <Feldadr> ist ohne eine Längenangabe <Anz> angegeben: Als Länge wird das Minimum von der Restlänge des (Teil-)Feldes <Feldadr> und '10' genommen. Das Ergebnis wird rechtsbündig mit führenden Nullen nach <Feldadr> geschrieben.
- <Feldadr> und <Anz> sind angegeben: Das Ergebnis wird rechtsbündig mit führenden Nullen, maximal 10-stellig eingetragen.

Die Ergebnislänge steht rechtsbündig im EZ Feld.

Das Vorzeichen des Ergebnisses steht im EV Feld (hier immer '+').

Wenn das Ergebnis in ER rechtsbündig benötigt wird, kann die Anweisung C3 aufgerufen, oder ER als Ergebnisfeld explizit angegeben werden.

**Beispiel 1:**

```
D#1: K,'56',H1: AD,(H1,2),'100',D+1,3: K,EV,D: S:
```

Es wird '56' und '100' addiert.

Das Ergebnis wird mit Vorzeichen im Display dargestellt ('+156').

**Beispiel 2:**

D#1: AD,'56','100':	Das ER Feld enthält '1560000000'.
---------------------	-----------------------------------

D#1: AD,'56','100',ER:	Das ER Feld enthält '0000000156'.
------------------------	-----------------------------------

## 5.7 C0 Stillstandsdauer berechnen

**Syntax:**

C0 :

**Erklärung:**

Die Anweisung C0 berechnet die Zeitdifferenz der Zeiten in den Feldern H1 und H2 in Sekunden. Der Wert in H1 wird von H2 subtrahiert und das Ergebnis im Feld Gx und im Feld ST rechtsbündig aufaddiert. Das Feld H3 gibt dabei den Index x= 0,..,11 des Feldes Gx an. Entsprechend der folgenden symbolischen Gleichung:

$$G(H3) = G(H3) + (H2 - H1).$$

Ist die Zahl in H3 größer als 11, wird die Zeitdifferenz in ND (nicht definierter Stillstandsgrund) aufaddiert.

Das Format für die Anfangs- und Endezeit ist im Abschnitt 4.16 *Hx Hilfsfelder* erklärt.

Die Anweisung C0 kann dazu verwendet werden, Stillstandzeiten von Maschinen zu erfassen. Jedem Stillstandsgrund kann ein Feld Gx, x= 0,..,11 zugeordnet werden, in dem die Zeiten akkumuliert werden. Diese Zeiten dürfen 365 Tage nicht überschreiten.

## 5.8 C1 Auftragsdauer berechnen

**Syntax:**

C1 :

**Erklärung:**

Zur Auftragsbearbeitung sind die Felder AB (Auftragsbeginn), AE (Auftragsende) und LZ (Laufzeit) vorgesehen. Die Anweisung C1 berechnet die Auftragsdauer in Sekunden aus der Zeitdifferenz von AB und AE und trägt sie rechtsbündig in das Feld LZ ein. Entsprechend der folgenden symbolischen Gleichung: LZ = AE - AB

Die Felder Gx, ND und ST werden gelöscht!

Das Format für die Zeiten ist im Abschnitt 4.1 *AB* Auftragsbeginn und 4.2 *AE* Auftragsende erklärt.

Für die Jahreszahl wird das aktuelle Jahr aus dem KT-Feld verwendet. Ist der Startzeitpunkt in AB größer als der Endzeitpunkt in AE, so wird angenommen, dass ein Jahreswechsel stattgefunden hat. Für AB wird die Jahreszahl dann auf das vorangegangene Jahr korrigiert.

## 5.9 C2 Felder Gx initialisieren

**Syntax:**

C2 :

**Erklärung:**

Diese Anweisung ist nur aus Kompatibilität zur Programmiersprache TCL84 der Terminalserie PCT88x vorhanden. Sie hat keine Wirkung.

## 5.10 C3 Ergebnis im ER Feld rechtsbündig ausgeben

**Syntax:**

C3 :

**Erklärung:**

Die Anweisung C3 verschiebt das Ergebnis im ER Feld rechtsbündig.

Diese Anweisung existiert aus Kompatibilitätsgründen zu den PCT88x-Terminals, bei denen das Ergebnis einer Arithmetikanweisung (AD, SB, MU, DI) im ER Feld linksbündig steht.

**Beispiel:**

```
DF,ER,10,0:K,'12',H1:K,'13',H2:AD,(H1,2),(H2,2):C3:
```

Nach der Anweisung AD steht '2500000000' im Feld ER, nach der Anweisung C3 '000000025'.

## 5.11 C4 Trace ausgeben

**Syntax:**

C4 :

**Erklärung:**

Diese Anweisung gibt die 5 zuletzt durchlaufenen Sprungziele, den Tracepuffer, ins Display (aber nicht ins Feld D!) aus.

Ein Unterprogramm-Rücksprung wird mit RET. angegeben. Die Anzahl der ausgeführten Anweisungen nach dem letzten Sprungziel wird in Klammern angezeigt.

Dadurch kann festgestellt werden, an welcher Stelle ein TCL-Programm steht.

**Beispiel:**

```
IR,S:  
D#0:#1:#2:  
D#3:%4:@5:  
D#4:%:  
D#5:  
D#6:K,'Hallo Welt',D:C4:S:  
I@0:
```

führt zu folgender Ausgabe auf dem Display:

0003>0004>RET.>0005>0006>(0002)

## 5.12 D Dekrementieren

Syntax:

```
D, <Feldadr>{, <Anz>}, <Ugrenze>-<Ogrenze>{, @<Label>}:  
<Ugrenze> ::= <Zahlenwert>  
<Ogrenze> ::= <Zahlenwert>
```

Erklärung:

Mit der Dekrementier-Anweisung wird der Zahlenwert in <Feldadr> im Bereich zwischen <Ogrenze> und <Ugrenze> um eins heruntergezählt.

Untergrenze <Ugrenze> und Obergrenze <Ogrenze> können bis TCL Version 4.28 maximal 7-stellig sein. Ab TCL Version 5.01 wurde diese Beschränkung auf 10 Stellen erweitert. Es wird immer rechtsbündig in dem angegebenen (Teil-)Feld <Feldadr> dekrementiert. Die Stellenanzahl der <Ogrenze> bestimmt dabei die Zählerlänge.

Wenn <Feldadr> keinen Zahlenwert enthält oder einen Zahlenwert außerhalb des angegebenen Bereichs, dann wird er auf <Ogrenze> (mit führenden Nullen) gesetzt.

Wird die Länge <Anz> angegeben, muss sie immer größer oder gleich der Stellenanzahl von <Ogrenze> sein und sollte 7 bzw. 10 nicht überschreiten. Ist <Anz> größer als die Restlänge des (Teil-)Feldes, wird eine Fehlermeldung generiert.

Wenn die Längenangabe <Anz> fehlt oder größer als 7 bzw. 10 ist, dann werden **die letzten sieben Speicherzellen** des Feldes zur Abspeicherung verwendet, unabhängig davon, auf welche Speicherzelle im Feld <Feldadr> zeigt.

Wenn kein Sprungziel <Label> angegeben ist, wird bei Erreichen der Untergrenze wieder mit <Ogrenze> als Ergebnis fortgesetzt.

Ist eine Sprungadresse <Label> angegeben, wird bei Unterschreiten der Untergrenze das Ergebnis auf <Ogrenze> gesetzt und ein Sprung an die angegebene Adresse ausgeführt.

Diese Anweisung kann in Verbindung mit der Prüfanweisung verwendet werden, um Schleifen zu implementieren.

**Beispiel 1:**

DD,P4,0-6:

6,5,4,3,2,1,0,6,5 usw.

**Beispiel 2:**

Das Beispiel 2 zur Inkrementieranweisung I in Abschnitt 5.18 geht auf die Längenangabe <Anz> und die Stellenanzahl der <Ogrenze> ein. Es ist sinngemäß auf die Dekrementieranweisung D übertragbar.

## 5.13 DI Division

**Syntax:**

**DI, <Zahlenwert>, <Zahlenwert>{, <Feldadr>{, <Anz>} } :**

**Erklärung:**

Die erste Zahl <Zahlenwert> wird durch die zweite Zahl <Zahlenwert> dividiert. Die Operanden und das Ergebnis sind positive ganze Zahlen. Es können maximal 10-stellige Operanden ein ebenfalls 10-stelliges Ergebnis liefern.

Die Abspeicherung des Ergebnisses hängt von den weiteren optionalen Parametern ab:

- Es sind keine weiteren Parameter angegeben: Sind <Feldadr> und damit auch <Anz> nicht angegeben, steht das Ergebnis linksbündig im ER Feld. (Kompatibilität zu den PCT88x-Terminals)
- <Feldadr> ist ohne eine Längenangabe <Anz> angegeben: Als Länge wird das Minimum von der Restlänge des (Teil-)Feldes <Feldadr> und '10' genommen. Das Ergebnis wird rechtsbündig mit führenden Nullen nach <Feldadr> geschrieben.
- <Feldadr> und <Anz> sind angegeben: Das Ergebnis wird rechtsbündig mit führenden Nullen, maximal 10-stellig eingetragen.

Der ganzzahlige Rest steht rechtsbündig im Feld ED.

Die Ergebnislänge steht rechtsbündig im EZ Feld.

Das Vorzeichen des Ergebnisses steht im EV Feld (hier immer '+').

Wenn das Ergebnis in ER rechtsbündig benötigt wird, kann die Anweisung C3 aufgerufen, oder ER als Ergebnisfeld explizit angegeben werden.

**Beispiel:**

**D#1:K, '56', H1: DI, '100', (H1,2), D+1, 3: K, EV, D: S:**

Es wird '100' durch '56' (in H1) dividiert. Das Ergebnis wird mit Vorzeichen im Display dargestellt ('+001').

## 5.14 E Eingabe über Tastatur und Kartenleser

Alle Syntaxregeln für die Eingabeanweisung sind hier zusammengefasst. Die Erklärungen zur Funktionsweise der Anweisung E und den Syntaxregeln folgen in Unterkapiteln.

Syntax:

```
E, <Eingabecom>:  
E, (<Eingabecom>{/<Eingabecom>}*) :  
<Eingabecom>      ::=      <Fkttaste>, @<Label> |  
                           <Eingabeaddr>, <Format>, @<Label>
```

Definitionen für die Funktionstasteneingabe:

```
<Fkttaste>      ::=      F<Dezimalzahl> |  
                           F(<Zahlenzeiger>) |  
                           F [<LBusadr>]<Zahlenkomp>  
<LBusadr>       ::=      <Zahlenwert>
```

Definitionen für andere Eingaben:

```
<Eingabeaddr>   ::=      <Alphataste> | <Leser> |  
                           <Leserfeld>
```

Definitionen für die Tastatureingabe:

```
<Alphataste>    ::=      <Displayfeld>{+<Zahlenwert>}  
  
<Displayfeld>   ::=      D{[*<LBusadr>]}{[<Zeile>]}  
<LBusadr>        ::=      <Zahlenwert>  
<Zeile>          ::=      <Zahlenwert>
```

Definitionen für die Lesereingabe:

```
<Leser>          ::=      <Leserfeld>[*<Codetyp>] |  
                           <Leserfeld>[<Lnr>{,<Lnr>}*] |  
                           <Leserfeld>[*<Codetyp>,<Lnr>{,<Lnr>}*]  
<Leserfeld>      ::=      <Leserfieldname>{+<Zahlenwert>}  
<Leserfieldname> ::=      B | M | I  
  
<Codetyp>         ::=      <Zahlenwert>  
  
<Lnr>            ::=      <LBusadr>{-<LBusadr>}  
<LBusadr>        ::=      <Zahlenwert>
```

### Definitionen für das Eingabeformat:

```
<Format>      ::=  {<Ugrenze>- }<Ogrenze><Zeichentyp>
<Ugrenze>    ::=  <Zahlenwert>
<Ogrenze>    ::=  <Zahlenwert>
<Zeichentyp> ::=  A | D | N | S | Z
```

### Einführung in die Eingabeanweisung

Eine Eingabe über

- Alphanumerische Tasten
- Funktionstasten und Schlüsselschalter
- Barcodeleser
- Biometrieleser
- Induktivkartenleser
- Magnetkartenleser
- Proximityleser
- Speicherchipkartenleser
- Leser mit OMRON-Emulation

ist nur möglich, wenn die entsprechende Eingabeeinheit in einer Eingabeanweisung freigegeben wurde. Es können sowohl die internen Eingabeeinheiten im Terminal als auch die externen, abgesetzten Eingabeeinheiten am LBus freigegeben werden. Darüber hinaus ist es möglich, Leser an die seriellen Zusatzschnittstellen, Kanal B, C oder D, des INTUS 3000 anzuschließen.

Für jede freigegebene Eingabeeinheit ist ein Sprungziel <Label> anzugeben. Die Sprunganweisung nach <Label> wird als Ereignissprung in den Interpreteranweisungspuffer \$3 geschrieben, wenn eine Eingabe abgeschlossen ist. Der Interpreter führt den Sprung jedoch erst dann aus, wenn er die nächste Stopanweisung bearbeitet hat.

Soll das Programm erst nach erfolgter Eingabe fortgesetzt werden, muss eine Stopanweisung unmittelbar auf die Eingabeanweisung folgen.

Wenn die Eingabe abgeschlossen ist, muss für eine weitere Eingabe zunächst wieder eine Freigabe durch die entsprechende E Anweisung erfolgen.

### 5.14.1 Abgesetzte Eingabestationen - LBus Adressen

Mit der Eingabeanweisung können Eingabeeinheiten vom INTUS 3000 und auch von abgesetzten Eingabestationen am LBus freigegeben werden. Diese abgesetzten Eingabestationen können beispielsweise INTUS 1600 Terminals mit Leser und eventuell Tastatur oder auch nur abgesetzte Leser sein.

Für jede abgesetzte Eingabestation muss der Typ im Setup oder im CV Feld (CV+47 - CV+61 und CV+83 - CV+90) eingestellt werden. Dabei wird eine LBus Adresse verteilt. Ohne vorangegangene Konfiguration im Setup oder im CV Feld kann eine externe Eingabestation vom TCL-Laufzeitsystem nicht erkannt werden.

TCL ist für den Betrieb eines INTUS 3000 mit 2 LBussen ausgelegt. Dabei können über zwei serielle Schnittstellen jeweils 8, also insgesamt 16 abgesetzte Leser am INTUS 3000 betrieben werden.

Zuordnung der Eingabestation mit - je nach Konfiguration - Leser, alphanumerischer Tastatur und Funktionstasten zu einer LBus Adresse:

Eingabestation	<LBusadr>
INTUS 3000	keine Angabe oder '0'
Eingabestation am 1. LBus	'1'-'8'
Eingabestation am 2. LBus	'9'-'16'

Wenn eine Eingabe abgeschlossen wurde und der zugehörige Ereignissprung in den Ringpuffer \$3 eingetragen wird, wird im Parameterfeld P20+24,2 die LBus Adresse des auslösenden Geräts eingetragen. Sie bleibt bis zur nächsten Stopanweisung aktuell.

#### Beispiel 1:

```
D#0:E,(B[3],1-10N,@10): S:  
D#10:K,B,D,10: S:
```

Es wird der externe Barcodeleser mit der LBus Adresse 3 freigegeben.

#### Beispiel 2:

```
D#0:E,(F[1]0,@17/F[1]1,@18):
```

Im INTUS 1600 Terminal mit LBus Adresse 1 werden die Funktionstasten 0 und 1 freigegeben (Aufschrift F1 und F2).

### **5.14.2 Gleichzeitige Eingabe von mehreren Eingabeeinheiten**

In einer Eingabeanweisung können gleichzeitig mehrere Eingabeeinheiten freigegeben werden (Alternativeingaben). Alle Alternativen werden durch '(' und ')' geklammert und die einzelnen Alternativen <Eingabecom> werden durch '/' getrennt.

Da der Interpreter nach einer Eingabeanweisung weitere Anweisungen ausführt, ist es möglich, mehrere Eingabeanweisungen hintereinander ausführen zu lassen. Jede dieser Eingabeanweisungen wird sofort verarbeitet.

Wenn ein Eingabemedium (z.B. die Tastatur) bereits in einer früheren Anweisung freigegeben wurde, wird die neue Freigabe ignoriert.

Wenn ein Eingabemedium noch nicht in einer früheren Anweisung freigegeben wurde, wird es zusätzlich freigegeben. Die vorherigen Freigaben bleiben davon unberührt.

Ist in einer Eingabeanweisung die Eingabe sowohl vom INTUS 3000 als auch von abgesetzten Eingabestationen, wie z.B. mehreren INTUS 1600 Terminals, freigegeben und erfolgt an einem INTUS 1600 eine Eingabe, so ist sie nur für dieses Terminal nicht mehr aktiv. Über das INTUS 3000 oder andere INTUS 1600 kann weiterhin eine Eingabe erfolgen.

Aus diesem Grund kann eine Eingabeanweisung durchaus mehrere Sprünge durch Eingaben an unterschiedlichen Eingabeeinheiten zur Folge haben.

Bei einer gleichzeitigen Lesung über zwei gleichartige Leser, z. B. Magnetkartenleser, muss verhindert werden, dass die zweite Lesung die erste im Leserfeld, hier also Feld M, überschreibt, bevor sie verarbeitet wurde. Dies geschieht dadurch, dass die zweite Lesung erst dann in das Feld M kopiert wird, wenn nach dem Eingabesprung der ersten Lesung eine Stopanweisung vom Interpreter ausgeführt wurde.

#### **Beispiel 1:**

Die Eingabeanweisung

**DE, (D,1-10N,@10/M[1],1-80N,@20):**

gibt zwei Eingaben frei, eine interne Tastatureingabe und eine externe Magnetkarteneingabe vom Leser mit der LBus Adresse 1. Wenn jetzt eine Magnetkarteneingabe erfolgt, so wird der Sprung nach #20 ausgelöst. Es ist aber weiter möglich, Daten über die Tastatur einzugeben. Wenn jetzt die Eingabe mit ENTER abgeschlossen wird, wird der Sprung nach #10 ausgelöst.

#### **Beispiel 2:**

Die Eingabeanweisung

**DE, (M[0],1-80N,@10/M[1],1-80N,@20):**

gibt statt der internen Tastatur, wie im Beispiel 1, den internen Magnetkartenleser frei. Bei einer Magnetkarteneingabe am Leser mit LBus Adresse 1 wird der Sprung nach #20 ausgelöst. Wird gleichzeitig eine Magnetkarte am internen Leser eingelesen, wird der zugehörige Ereignissprung nach #10 erst dann ausgeführt, wenn nach der Abarbeitung des Ereignisses Magnetkarteneingabe am Leser Nummer 1, der Interpreter eine Stopanweisung ausführt. Zu diesem Zeitpunkt wird dann auch die zweite Lesung in das M Feld übernommen.

**Beispiel 3:**

Die beiden Anweisungen

**DE, (M[1],1-80N,@10): E, (M[2],1-80N,@20): S:**

sind äquivalent zur Anweisung

**DE, (M[1],1-80N,@10/M[2],1-80N,@20): S:**

Es sind jeweils beide Leser freigegeben.

Eine neue Anweisung **E,(M[1],1-80N,@30):** wird ignoriert, wenn inzwischen keine Lesereingabe erfolgt ist, da der Leser in dem Fall bereits freigegeben ist. Bei einer Eingabe erfolgt der Sprung nach #10 und nicht nach #30.

### 5.14.3 Funktionstasteneingabe

Der Teil der Syntax für die Eingabe über Funktionstasten wird hier für die Übersichtlichkeit nochmals wiederholt.

#### Syntax:

```
E, (<Eingabecom>{/<Eingabecom>}*) :
<Eingabecom> ::= <Fkttaste>, @<Label> |
                  <Eingabeaddr>, <Format>, @<Label>
<Fkttaste>      ::= F<Dezimalzahl> | F(<Zahlenzeiger>) |
                  F [<LBusadr>]<Zahlenkomp>
<LBusadr>       ::= <Zahlenwert>
```

Wenn eine durch die Eingabeanweisung freigegebene Funktionstaste gedrückt wird, wird der entsprechende Ereignissprung ausgelöst. Die Eingabe ist damit abgeschlossen.

Die Funktionstasten der abgesetzten Eingabestationen werden mit  $F[<LBusadr>]<Zahlenkomp>$  angesprochen.

Die INTUS 3000 Terminal Modelle haben bis zu 20 Funktionstasten. Es beginnt mit der **Funktionstaste F0**. Die Beschriftung der Tasten auf den unterschiedlichen Tastaturen ist der folgenden Tabelle zu entnehmen. Bei Touch-Tastaturen, die ein Passepartout hinterlegt haben, ist die Beschriftung frei wählbar.

Bezeichnung der Funktionstaste im TCL	Beschriftung der Funktionstasten (wie PC-Tastatur)	Zuordnung von Symboltasten und ACM-Tasten
F0	F1	?
F1	F2	*
F2	F3	→
F3	F4	→ →
F4	F5	← →
F5	F6	↑
F6	F7	↓
F7	F8	
...	...	
F15	F16	
Achtung: F16 und F17 haben eine besondere Verwendung, siehe Abschnitte unten		

F18	F19	
F19	F20	
F20	F21	
F21	F22	Lesertesttaste, Quittierungstaste

**Tabelle 5.1 – Zuordnung TCL Funktionstaste zu Beschriftung****ENTER-Taste, F16**

Der ENTER-Taste ist die Funktionstaste F16 zugeordnet. Wird als Alternativeingabe die Tastatur und F16 (ENTER) zugelassen, wird der Funktionstastensprung nur dann ausgeführt, wenn vorher keine Zeichentaste gedrückt oder alle eingegebenen Zeichen mit der C-Taste gelöscht wurden. Im anderen Fall wird der Tastatureingabesprung ausgeführt. Nach der Eingabe der Funktionstaste F16 ist auch die Tastatureingabe wieder gesperrt. Ebenso ist nach einer Eingabe über die Tastatur auch die Funktionstasteneingabe F16 wieder gesperrt.

**CLEAR-Taste, F17 (im INTUS 3000)**

Der CLEAR-Taste ist die Funktionstaste F17 zugeordnet. Wenn F17 freigegeben und die Tastatur-Eingabe leer ist, wird die C-Taste als F17 weitergegeben.

**Schlüsselschalter, F17 (nicht im INTUS 3000)**

Dem Schlüsselschalter ist die Funktionstaste F17 zugeordnet. Er muss ca. 1 Sekunde betätigt werden, bis der Ereignissprung ausgelöst wird, da das System zuerst prüft, ob in den Setup verzweigt werden soll.

**Tastaturvarianten mit mehr als 16 Funktionstasten**

Tastaturvarianten mit 16 bis 20 Funktionstasten weisen eine Lücke in der Zuordnung der Funktionstasten bei den letzten vier Tasten auf, da F16 und F17 für ENTER-Taste und Schlüsselschalter reserviert sind.

Die 17. bis 20. Funktionstaste sind deshalb mit F18 - F21 anzusprechen.

**Funktionstaste F21 der Zutrittskontrollmanager**

Die Taste auf der Frontseite der Modelle INTUS ACM8, INTUS ACM8(0)e Rack und INTUS ACM8(0)e Wand ist Funktionstaste F21 zugeordnet.

INTUS ACM4 AKKU und INTUS ACM40 AKKU besitzen einen Quittierungstaster für L1 (LED Störung) und L2 (LED Sabotage):

- Beim INTUS ACM40 AKKU ist der Quittierungstaster Funktionstaste F21 zugeordnet.
- Beim INTUS ACM4 AKKU war bis zur TCL Version 6.10 der digitale Eingang E3 der Quittierungstaster. Ab TCL Version 6.10 ist die Zuordnung über CV+239,1 wählbar. Der Quittierungstaster des INTUS ACM4 AKKU kann jetzt ebenfalls der Funktionstaste F21 zugeordnet werden, um alle Zutrittskontrollmanager kompatibel zu machen.

### **Funktionstasten der Subterminals**

Die abgesetzten Eingabestationen INTUS 1600 und INTUS 1500 haben je nach Modell unterschiedlich viele Funktionstasten. Maximal stehen die Funktionstasten F0 bis F5 zur Verfügung.

Hat das INTUS 1600-II einen Firmwarestand V2.5 oder höher und wird es im INTUS LBus Modus C betrieben, kann wie im INTUS 3000 die ENTER-Taste als Funktionstaste F16 angesprochen werden.

#### **Beispiel 1:**

```
D#0: K, 'Taste',D: E,(F0,@1): S:  
D#1: K, 'Ok',D+6: S:
```

Im Display wird der Text 'Taste' angezeigt und die Funktionstaste F0 freigegeben. Nach korrekter Eingabe wird der Sprung nach Sprungziel #1 ausgeführt.

#### **Beispiel 2:**

```
D#0: E,(F[8]0,@17/F[8]1,@18):
```

Funktionstastenfreigabe von Taste 0 und Taste 1 der abgesetzten Eingabestation mit LBus Adresse 8.

### 5.14.4 Tastatureingabe

Der Teil der Syntax für die Eingabe über die alphanumerische Tastatur wird hier für die Übersichtlichkeit nochmals wiederholt.

#### Syntax:

```
E, (<Eingabecom>{/<Eingabecom>}*) :  
  <Eingabecom>      ::=  <Fkttaste>, @<Label> |  
                        <Eingabeaddr>, <Format>, @<Label>  
  <Eingabeaddr>     ::=  <Alphataste> | <Leser> |  
                        <Leserfeld>  
  
  <Alphataste>      ::=  <Displayfeld>{+<Zahlenwert>}  
  <Displayfeld>      ::=  D{[*<LBusadr>]}{[<Zeile>]}  
  <LBusadr>          ::=  <Zahlenwert>  
  <Zeile>             ::=  <Zahlenwert>  
  
  <Format>            ::=  {<Ugrenze>-}<Ogrenze><Zeichentyp>  
  <Ugrenze>          ::=  <Zahlenwert>  
  <Ogrenze>          ::=  <Zahlenwert>  
  <Zeichentyp>        ::=  A | D | N | Z | S
```

Wird die Tastatureingabe freigegeben, dann sind davon die Tasten des numerischen Zehnerblocks und die Tasten des Alphablocks, aber nicht die Funktionstasten betroffen. Die eingegebenen Zeichen werden sofort im Display angezeigt, wenn sie für die freigegebene Tastatureingabe zulässig sind.

Alle Terminals, die eine alphanumerische Tastatur besitzen, haben auch ein Display. Die Displays haben unterschiedlich viele Zeilen und Spalten. Subterminals können ebenfalls einen Zehnerblock, aber eventuell kein Display besitzen.

#### Subterminals

Diese Subterminals können in unterschiedlichen INTUS LBus Modi (Typ der externen Leser in den Feldern CV+47 – CV+61, CV+83 – CV+90) betrieben werden.

INTUS LBus Modus	Tastatureingabe
A	2x16 Zeichen Display Tastatureingabe vereinfacht implementiert
B	2x20 Zeichen Display Tastatureingabe vereinfacht implementiert
C INTUS 1600-II, ab Firmware V2.5	2x20 Zeichen Display Tastatureingabe vollständig implementiert

**Tabelle 5.2 –INTUS LBus Modi der Subterminals**

Die Implementierungsunterschiede werden im Laufe des Abschnitts erklärt.

Subterminals optional mit Tastatur	Display	INTUS LBus Modi für die Eingabe
INTUS 1500	optional	A
INTUS 1600	optional	A, B
INTUS 1600-II	optional	A, B, C (ab Firmware V2.5)
INTUS 500	kein	B
INTUS 600	kein	B

**Tabelle 5.3 – Subterminals mit Tastatur und Display, INTUS LBus Modi**

#### Formatierung der Eingabe im Display <Displayfeld>

Die Tastaturen, bzw. Displays, der Subterminals werden mit D[\*<LBusadr>] ... angesprochen. Die Schreibweise mit \* im Unterschied zu der Angabe bei den Funktionstasten und den Lesern ergibt sich aus der Historie von TCL. D[<Zeile>] ... wird verwendet, um zeilenorientiert im Display und im D Feld arbeiten zu können.

Die Cursorposition im Display, an der die Eingabe erfolgen soll, und die Adresse im zugehörigen D Feld wird durch <Displayfeld>+<Zahlenwert> in der Eingabeanweisung angegeben.

Alle zulässigen Zeichen, die über die Tastatur eingegeben werden, werden sofort im Display angezeigt (Echo). Die Cursorposition wird automatisch nach jedem Tastendruck inkrementiert. Gleichzeitig werden die Zeichen in das Feld D ab dem angegebenen Offset eingetragen.

##### Beispiel 1:

D#0: E,(D[2]+5,1-10N,@10): S:

ist bei dem 2-zeiligen Display mit 40 Spalten äquivalent zu

D#0: E,(D+85,1-10N,@10): S:

Die Eingabe erfolgt in der dritten Zeile ab Spalte 5, im Feld D werden die Zeichen ab  $2*40+5 = 85$  eingetragen.

##### Beispiel 2:

D#0: E,(D[\*1][1]+3,1-5N,@18):

Tastatureingabe in Zeile 2, Spalte 4 von maximal 5 Zeichen am Subterminal mit LBus-Adresse 1.

#### Eingabeformat <Format> ::= {<Ugrenze>-}<Ogrenze><Zeichentyp>

Die über die alphanumerische Tastatur eingegebenen Daten können mit der Angabe eines Eingabeformats <Format> hinsichtlich der Länge und des Zeichentyps überprüft werden.

Zeichen, die nicht dem angegebenen <Zeichentyp> entsprechen, werden ignoriert.

**Eingabeformat für Terminals und Subterminals in INTUS LBus Modus C**

<Zeichentyp>		Zugelassene Zeichen	Abschluss der Eingabe	Formatierung im Display
N	Numerisch	0-9 und mehrere ';' oder ','	mit ENTER-Taste	rechtsbündig mit führenden Nullen
D	Dezimalzahlen	0-9 und einmal ';' oder ','	mit ENTER-Taste	rechtsbündig mit führenden Nullen
A	Alpha-numerisch	0-9, ';', A-Z, a-z	mit ENTER-Taste	rechtsbündig mit führenden Nullen
Z	Darstellbare Zeichen, mit Zeichenprüfung	"20"- "7F"	mit ENTER-Taste	keine Formatierung
S	Darstellbare Zeichen, ohne Zeichenprüfung	"20"- "7F"	Die Eingabe ist automatisch beendet, wenn <Ogrenze> Zeichen eingegeben wurden.	keine Formatierung

**Tabelle 5.4 – Zeichentypen für Terminals und Subterminals mit INTUS LBus Modus C****Ab TCL 6.50:****Eingabeformat für verdeckte Eingabe am INTUS 5600, 5540, 5200, 5205**

<Zeichentyp>		Zugelassene Zeichen	Abschluss der Eingabe	Formatierung im Display
P	Darstellbare Zeichen, mit Zeichenprüfung	"20"- "7F"	mit ENTER-Taste	Verdeckte Eingabe, Darstellung mit Zeichen aus CE+1,1 keine Formatierung
Q	Darstellbare Zeichen, ohne Zeichenprüfung	"20"- "7F"	Die Eingabe ist automatisch beendet, wenn <Ogrenze> Zeichen eingegeben wurden.	Verdeckte Eingabe, Darstellung mit Zeichen aus CE+1,1 keine Formatierung

**Tabelle 5.5 – Zeichentypen für verdeckte Eingabe am Terminal**

Die ENTER-Taste wird solange ignoriert, bis die Mindestanzahl <Ugrenze> der Zeichen eingegeben wurde. Wenn die Maximalanzahl <Ogrenze> der Zeichen erreicht ist, werden alle Tasten bis auf ENTER ignoriert.

Achtung: Je nach Anzahl Subterminals am LBus kann mit dem <Zeichentyp> S die Performance sinken!

### Eingabeformat für Subterminals in INTUS LBus Modus A und B

<Zeichentyp>		Zugelassene Zeichen	Abschluss der Eingabe	Formatierung im Display
N	Numerisch	0-9 und mehrere !'	mit ENTER-Taste	rechtsbündig mit führenden Nullen
D	Dezimalzahlen	0-9 und einmal !'	mit ENTER-Taste	rechtsbündig mit führenden Nullen
A	Alpha-numerisch	0-9, !', A-Z, a-z	mit ENTER-Taste	rechtsbündig mit führenden Nullen
Z	Darstellbare Zeichen, mit Zeichenprüfung	"20"- "7F"	mit ENTER-Taste	rechtsbündig mit führenden Nullen

**Tabelle 5.6 – Zeichentyp für Subterminals mit INTUS LBus Modus A und B**

Bei Subterminals, die in Modus A oder B betrieben werden, wird die Mindestanzahl <U-grenze> der Zeichen nicht ausgewertet. Wenn die Maximalanzahl <Ogrenze> der Zeichen erreicht ist, dann werden alle Tasten bis auf die ENTER-Taste ignoriert.

Das Eingabeformat 'S' steht nicht zur Verfügung. Die Formatierung im Display unterscheidet sich für das Eingabeformat 'Z':

#### Formatierung der Eingabe im Display abhängig vom Eingabeformat

Die zeichenweise Eingabe erfolgt von links nach rechts. Bei Zahlen ist aber eine rechtsbündige Darstellung erwünscht. Deshalb wird bei numerischer ('N', 'D') und alphanumerischer ('A') Eingabe nach Ende der Eingabe die Zeichenfolge rechtsbündig mit führenden Nullen formatiert:

aus 111\_\_ wird 000111,

aus 123456 wird 123456.

#### Terminal und Subterminal INTUS LBus Modus C:

Bei 'Z' und 'S' wird keine Formatierung durchgeführt.

#### Subterminal in INTUS LBus Modus A und B:

Bei einem Subterminal, das in Modus A oder B betrieben wird, wird auch beim Eingabeformat 'Z' die Zeichenfolge rechtsbündig mit führenden Nullen ausgegeben.

#### Korrektur der Eingabe

Solange die ENTER-Taste nicht gedrückt wurde, können die eingegebenen Zeichen noch korrigiert werden. Die Eingabe kann aber auch über eine alternative Lesereingabe oder eine Funktionstaste beendet werden. Die Tastatureingabe ist dann hinfällig.

#### Terminal und Subterminal INTUS LBus Modus C:

Mit der C-Taste kann das zuletzt eingegebene Zeichen gelöscht werden. An diese Stelle wird das im Feld CE definierte Zeichen eingesetzt.

Wird die C-Taste innerhalb kurzer Zeit zweimal betätigt, wird die gesamte Eingabe gelöscht.

Bei Subterminals ohne Display wird immer die komplette Eingabe gelöscht.

#### Verdeckte Eingabe

**Display des Terminals:** Das Feld D ist größer als die Stellenanzahl im Display. Wird als Offset ein Wert eingegeben, der außerhalb des Darstellungsbereichs des Displays liegt, erfolgt die Eingabe ohne Zeichenecho. Die Zeichen werden aber trotzdem im Feld D eingetragen.

Ab TCL Version 6.50 stehen die neuen Zeichentypen 'P' und 'Q' für INTUS 5600/5540/5200/5205 zur Verfügung. Sie entsprechen in der Funktionalität den Zeichentypen 'Z' und 'S', beim Zeichenecho werden aber die eingegebenen Zeichen durch das Zeichen aus CE+1,1 ersetzt.

**Display des Subterminals:** Das Feld D ist doppelt so groß wie die sichtbaren Zeilen im Display. Eingaben in die nicht sichtbaren Zeilen werden als '\*' in die sichtbaren Zeilen gespiegelt.

Dies ist z.B. zur Eingabe eines Passwortes praktisch.

**Beispiel 1:**

```
D#0: K, 'Auftragsnummer ____ ',D: E, (D+15,3-5N,@2): S:
```

Eingabe von 3-5 numerischen Zeichen über die Tastatur. Darstellung der Zeichen auf dem Display ab Stelle 15. Das Eingabefeld ist mit '\_\_\_\_' gekennzeichnet. Nachdem die Eingabe mit ENTER abgeschlossen wurde, stehen die Ziffern rechtsbündig mit führenden Nullen im Feld D und die Anweisung nach Sprungziel '2' wird bearbeitet.

**Beispiel 2:**

```
D#0:K, 'Auftragsnummer ',D:K,AN,D+15,4:E, (D+15,1-4Z,@2):S:
```

```
D#2: ...
```

In diesem Fall ist das Eingabefeld mit einem Vorgabewert aus AN bereits vorbelegt, der nun übernommen werden kann.

### 5.14.5 Lesereingabe

Der Teil der Syntax für die Lesereingabe wird hier für die Übersichtlichkeit nochmals wiederholt.

#### Syntax:

```

E, (<Eingabecom>{/<Eingabecom>}*) :

<Eingabecom>      ::=  <Fkttaste>, @<Label> |  
                      <Eingabeaddr>, <Format>, @<Label>

<Eingabeaddr>     ::=  <Alphataste> | <Leser> |  
                      <Leserfeld>

<Leser>            ::=  <Leserfeld>[*<Codetyp>] |  
                      <Leserfeld>[<Lnr>{, <Lnr>}*] |  
                      <Leserfeld>[*<Codetyp>, <Lnr>{, <Lnr>}*]

<Leserfeld>        ::=  <Leserfieldname>{+<Zahlenwert>}

<Leserfieldname>   ::=  B | M | I

<Codetyp>          ::=  <Zahlenwert>

<Lnr>              ::=  <LBusadr>{-<LBusadr>}

<LBusadr>          ::=  <Zahlenwert>

<Format>            ::=  {<Ugrenze>- }<Ogrenze><Zeichentyp>
<Ugrenze>          ::=  <Zahlenwert>
<Ogrenze>          ::=  <Zahlenwert>
<Zeichentyp>        ::=  A | D | N | Z

```

An das Terminal können

- Barcodeleser
- Biometrieler
- Induktivkartenleser
- Magnetkartenleser
- Proximityleser
- Speicherchipkartenleser
- Leser mit OMRON-Emulation

angeschlossen werden. Der Lesertyp muss im Setup oder im CV-Feld eingestellt werden (siehe Betriebshandbuch oder Abschnitt 4.6 *CV Setup* Tabelle 2j für interne Leser und Tabelle 2l für externe Leser).

**<Leserfeld>**

Jedem Lesertyp ist ein Feld <Leserfeld>

- B Barcodeleser
- I Induktivkartenleser und Biometrieler
- M Magnetkartenleser und Proximityleser, Speicherchipkartenleser sowie Leser mit OMRON-Emulation

zugeordnet. Die drei Leserfelder B, M und I sind gleich aufgebaut. Sie sind in den Abschnitten 4.4, 4.22 und 4.17 beschrieben.

Die Daten einer Lesung werden in dem jeweiligen Feld ab dem angegebenen <Zahlenwert> abgespeichert. Am Ende der Leserfelder stehen Konfigurations- und Statusbytes.

Wenn eine Eingabe über einen Leser erfolgte, werden die Daten entsprechend dem angegebenen <Format> geprüft. Ist das Format und der <Codetyp> in Ordnung, ist die Eingabe abgeschlossen.

Das Konfigurationsbyte in den Leserfeldern bestimmt, ob eine Fehllesung ignoriert wird, oder ob sie anhand der Eintragungen im Leserfeld analysiert werden soll.

Die Synchronisation von mehreren Lesungen gleicher Art in die Leserfelder B, M, I erfolgt über die Stopanweisung. Nach einem Eingabesprung wird das Leserfeld solange nicht überschrieben, bis eine Stopanweisung ausgeführt wird. Es können also ohne möglichen Datenverlust erst neue Eingaben aktiviert und anschließend die Lesereingabe ausgewertet werden.

**<Format>**

Die über Leser eingegebenen Daten können mit der Angabe eines Eingabeformats <Format> hinsichtlich der Länge und der zugelassenen Zeichen überprüft werden.

Durch <Ugrenze> und <Ogrenze> wird die Länge der Eingabe festgelegt. Eine Lesereingabe mit zu vielen oder zu wenigen Zeichen wird als Fehler behandelt.

Durch <Zeichentyp> wird der Zeichentyp überprüft. Eine Lesereingabe, die nicht zugelassene Zeichen enthält, wird ebenfalls als Fehler behandelt.

<Zeichentyp>		Zugelassene Zeichen
N	Numerisch	0-9 und mehrere ','
D	Dezimalzahlen	0-9 und einmal ','
A	Alphanumerisch	0-9, ',', A-Z, a-z
Z	Darstellbare Zeichen	"20"- "7F"

### <Codetyp>

Für jeden der Lesertypen gibt es bestimmte Codetypen, die selektiv freigegeben werden können:

#### Zum B-Feld:

Codetyp	Barcode
0	Code 2/5 Matrix
1	Code 2/5 Interleave
2	Code 39
3	Code 2/5 Industrial
4	Code 93

Bei den USB Barcode Lesern, die am INTUS 5200/5500/5540/5600 angeschlossen werden, wird der Codetyp ignoriert. Eine Lesung kommt immer ins B-Feld.

#### Zum I-Feld:

Codetyp	Biometrieleser
Keine Angabe von <Codetyp>	<b>Identifikation mit Biometrieleser:</b> Das I-Feld enthält nach erfolgreicher Identifizierung des Fingers oder des Handvenenscans die 8-stellige Templates-ID. Beim Fingerprintleser sind zusätzlich die Bezeichnung und der Status des Fingers enthalten.
8-stellige Templates-ID	<b>Verifikation mit Biometrieleser:</b> Die Lesung ist erfolgreich, wenn die Templates-ID des identifizierten Fingers oder Handvenenscans mit der Templates-ID, die als <Codetyp> angegeben wurde, übereinstimmt.

Codetyp	Induktivkartenleser
0	RENA
1	Hengstler

#### Zum M-Feld:

Codetyp	Magnetkarten-, Proximity-, Speicherchipkartenleser Leser mit OMRON-Emulation
1	Magnetkarte Spur 1
2	Magnetkarte Spur 2, Proximitykarte, Speicherchipkarte
3	Magnetkarte Spur 3

Die Angabe des Codetyps ist optional. Wenn sie weggelassen wird, werden alle möglichen, bzw. im Setup eingestellten, Codes freigegeben.

**Achtung:** Beim Auslesen der Codekennung aus den Leserfeldern wird nur beim B-Feld dieselbe Bezeichnung für den Codetyp zurückgegeben, wie bei der E Anweisung angegeben wurde! Z.B. steht für Magnetkartenleser Spur 2 ein 'Y' in M+82,1!

### Beispiel 1:

D#0: E,(B[\*1],1-10N,@10): S:

D#10: K,B,D,10: S:

Es wird eine Barcodelesung nur für Code 2/5 Interleaved freigegeben.

### Beispiel 2:

D#0: E,(M[\*2],1-80N,@1): S:

D#1: K,M,D,80: S:

Es wird nur Spur2 freigegeben.

### Beispiel 3:

D#0: E,(B[\*2,1],1-10N,@10): S:

D#1: K,B,D,10: S:

Es wird der Barcodeleser des abgesetzten Lesers mit der LBus Adresse 1 nur für Code 39 freigegeben.

### Beispiel 4:

D#0: E,(B[\*2,0-8],1-32Z,@18):

Strichcode \*2 (Code 39) für den INTUS 3000 Leser und alle externen Leser am 1. LBus freigeben.

## Checksummenprüfungen

Soweit möglich werden die eingelesenen Kartendaten durch eine Checksumme überprüft.

- **Barcode:**  
Normalerweise ist auf den Karten keine Checksumme enthalten, sodass die Daten nicht überprüft werden können. Es ist jedoch empfehlenswert, auf neuen Karten eine Checksumme nach dem Modulo 10 Verfahren (siehe Abschnitt 5.21 *M1 Modulo 10/11 Fehlerprüfung*) anzufügen. Sie können dann mit der Anweisung M1 geprüft werden.
- **Magnetkarte:**  
Auf der Magnetkarte ist in der Regel (nach DIN 9785) eine VRC- und LRC-C checksumme aufgebracht. Diese wird vom Lesercontroller geprüft. Im Fehlerfall und bei Karten, die keine Checksummen enthalten, werden die Karten trotzdem gelesen und das Lesefehlerbyte M+81,1 (bzw. M+108,1) entsprechend gesetzt.

### 5.14.6 Eingaben über die seriellen Zusatzschnittstellen

An die Zusatzschnittstellen kann ein beliebiges Gerät angeschlossen werden, dessen Daten mit der Eingabeanweisung verarbeitet werden können. So lassen sich beispielsweise auch Proximity- oder Magnetkartenleser an eine Zusatzschnittstelle anschließen. Siehe auch Abschnitt 4.8 *Dx DNCIN-Steuerfeld* und Abschnitt 2.1.2 *Datenfluss zwischen Kanal B und Terminal*.

Folgende Punkte müssen erfüllt sein, damit die Datenübertragung funktioniert:

- Die Schnittstellenparameter Baudrate und Datenformat der seriellen Zusatzschnittstelle, Kanal B, C oder D, müssen im Setup bzw. im CV-Feld (Abschnitt 4.6) richtig eingestellt sein.
- Die empfangenen Datensätze müssen mit <CR> abgeschlossen sein.
- Im DNCIN-Steuerfeld Dx muss das Konfigurationsbyte Dx,1='2' gesetzt sein.

Angesprochen werden diese Leser wie die abgesetzten Eingabestationen am LBus mit einer <LBusadr>:

Adresse	Serielle Zusatzschnittstelle	Lesernummer im Leserfeld
'12'	Kanal B	'C'
'13'	Kanal C	'D'
'14'	Kanal D	'E'

Es wird immer das Leserfeld M verwendet. Maximal werden 80 Zeichen eingetragen, die Restlichen werden verworfen. Beim Fremdleser ist das Lesefehlerbyte (z.B. M+81,1) immer '0' (Gutlesung) und die Codekennung (z.B. M+82,1) ist 'Y' (Spur 2).

#### Beispiel:

**D#0: E,(M[12],1-10N,@10):**

An Kanal B ist ein Magnetkartenleser angeschlossen. (Die Adresse könnte natürlich auch die LBus Adresse 12 sein.)

### 5.14.7 Eingabefreigabe zurücknehmen

Alle Eingabefreigaben, sowohl intern als auch extern, können einzeln oder gemeinsam mit der Resetanweisung

**R,E{,<Eingabeaddr> | <Fkttaste>}\*:**

jederzeit gelöscht werden (siehe Abschnitt 5.27 R Reset).

Es wird auch der Ereignissprung im Interpreteranweisungspuffer \$3 gelöscht, wenn schon eine Eingabe erfolgte, aber der Sprung noch nicht vom Interpreter ausgeführt wurde.

Achtung: Bei abgesetzten Lesern gibt es eine Sonderbehandlung: beim Zurücknehmen einer Freigabe werden alle Freigaben dieses Lesers zurückgenommen. Dies hat historische Gründe und wird aus Kompatibilitätsgründen fortgeführt.

#### Beispiel:

**R,E,B[2],F5:**

setzt alle Freigaben für den abgesetzten Barcodeleser mit LBus-Adresse 2 und die Funktionstaste 5 am Terminal zurück. Beachten sie hierzu den oben stehenden Hinweis.

## 5.15 EO Exklusives Oder

**Syntax:**

`EO, <kQop>, <eQop>, <Feldadr>{, <Anz>} :`

**Erklärung:**

Die durch `<kQop>` und `<eQop>` angegebenen Speicherzellen werden bitweise durch die logische Operation "Exklusives Oder" (EO) verknüpft und das Ergebnis in `<Feldadr>` gespeichert.

Bei Operanden, die mehrere Bytes lang sind, gilt die "Motorola-Byte-Ordnung". Das heißt, das erste Byte ist das höchstwertigste Byte.

Der optionale Parameter `<Anz>` gibt die (Teil-)Feldlänge an, in der das Ergebnis abgespeichert wird. Die Länge ergibt sich aus dem Minimum der Feldlängen von `<kQop>`, `<eQop>`, `<Feldadr>` und `<Anz>` (falls vorhanden).

**Beispiel:**

`DEO,E1+1,"01",E1+1,1:`

invertiert die Flankenstellung des digitalen Eingangs E1.

## 5.16 F Felder füllen

**Syntax:**

`F, <Feldadr>, <Anz>, <Char> :`

**Erklärung:**

Die Anweisung F belegt das Feld ab der Adresse `<Feldadr>` für die Länge `<Anz>` (1-65535) mit dem druckbaren ASCII-Zeichen `<Char>`. Achtung: `<Char>` wird ohne einfache oder doppelte Hochkommas angegeben!

Die Anweisung F schreibt über Feldgrenzen hinweg bis die Anzahl `<Anz>` erreicht ist. Dies gilt nicht für die Felder TF und DL. Das Ende des Speicherbereichs für alle TCL Felder wird auch nicht überschritten.

Mit der Fillanweisung können also Felder überschrieben werden, die dem angegebenen Feld `<Feldadr>` im Speicherbereich folgen.

**Beispiel 1:**

`DF,D,/D/, :`

Das gesamte Feld D und damit das Display wird mit Leerzeichen ("20") beschrieben und dadurch gelöscht.

**Beispiel 2:**

`DF,D+12,10,0:`

Das Display wird ab Stelle 12 mit zehn Nullen beschrieben.

**Beispiel 3:**

`DF,L1,5,E:`

Die Lampen L1 bis L5 werden eingeschaltet (Lx mit 'E' füllen). Hierbei wird ausgenutzt, dass die Felder Lx im Speicher hintereinander stehen und keine Prüfung der Feldgrenzen vorgenommen wird!

## 5.17 GR Grafikfunktionen

Syntax:

```
GR, <Kommando>, <Parameterliste>:  
<Kommando> ::= 0 | 1 | 2 | 10 | 11  
<Parameterliste> siehe unten.
```

Erklärung:

Mit der Anweisung GR werden folgende Grafikfunktionen ausgeführt:

<Kommando>	Grafikfunktion
0	Rechteckbereich löschen
1	Rechteck zeichnen
2	Linie zeichnen
10	Bitmap ausgeben
11	Analoguhr zeichnen

Auf den 2-zeiligen Textdisplays wird die Anweisung ignoriert.

Die Anzahl der Parameter in der Parameterliste ist abhängig vom Kommando.

Teil von <Parameterliste>	Bedeutung	Wertebereich
<x>, <x1>, <x2>	Wert auf der x-Achse des Displays (unterer Rand) in Pixel	ab 0 aufwärts bis zur Breite des Displays
<y>, <y1>, <y2>	Wert auf der y-Achse des Displays (linker Rand) in Pixel	ab 0 aufwärts bis zur Höhe des Displays
<m>	Zeichenmodus	0 setzen oder löschen 1 XOR 2 OR
<d>	Linienstärke in Pixel	ab 1 aufwärts
<f>	Füllmodus	0 ungefüllt 1 gefüllt
<bmpdata>	Bitmapdaten	Bitmapdaten , wie sie von BMP2TCL erzeugt werden. ,S' ESC-Sequenzen wie in Abschnitt 7.6.2 beschrieben.
<r>	Radius in Pixel	ab 1 aufwärts
<z>	Zeigerbreite in Pixel	ab 1 aufwärts

**Grafikfunktionen:**

- **GR,0,<x1>,<y1>,<x2>,<y2>[,<m>]:**  
Rechteckbereich löschen.  
Die Punkte (x1, y1) und (x2, y2) spannen das Rechteck auf, das gelöscht werden soll.
- **GR,1,<x1>,<y1>,<x2>,<y2>[,<m>[,<d>[,<f>]]]:**  
Rechteck zeichnen.  
Die Punkte (x1, y1) und (x2, y2) spannen das Rechteck auf, das gezeichnet werden soll.
- **GR,2,<x1>,<y1>,<x2>,<y2>[,<m>[,<d>]]:**  
Linie zeichnen.  
Die Punkte (x1, y1) und (x2, y2) geben die beiden Endpunkte der Linie an.
- **GR,10,<x>,<y>[,<m>],<bmpdata>:**  
Bitmap ausgeben.  
Der Punkt (x, y) gibt die obere, linke Ecke des Bitmaps an.
- **GR,11,<x>,<y>,<r>[,0[,<z>]]:**  
Analoguhr zeichnen.  
Der Punkt (x, y) ist der Mittelpunkt der Uhr mit einem Radius <r>.

**Beispiel:**

DGR,11,120,32,20:A:

Es wird eine Uhr angezeigt und automatisch aktualisiert.

## 5.18 I Inkrementieren

**Syntax:**

```
I, <Feldadr>{, <Anz>}, <Ugrenze>-<Ogrenze>{, @<Label>}:
  <Ugrenze> ::= <Zahlenwert>
  <Ogrenze> ::= <Zahlenwert>
```

**Erklärung:**

Mit der Inkrementier-Anweisung wird der Zahlenwert in <Feldadr> im Bereich zwischen <Ugrenze> und <Ogrenze> um eins heraufgezählt.

Untergrenze <Ugrenze> und Obergrenze <Ogrenze> können bis einschließlich TCL Version 4.28 maximal 7-stellig, ab TCL Version 5.01 10-stellig sein. Es wird immer rechtsbündig in dem angegebenen (Teil-)Feld <Feldadr> inkrementiert. Die Stellenanzahl der <Ogrenze> bestimmt dabei die Zählerlänge.

Wenn <Feldadr> keinen Zahlenwert enthält oder einen Zahlenwert außerhalb des angegebenen Bereichs, dann wird er auf <Ugrenze> (mit führenden Nullen) gesetzt.

Wird die Länge <Anz> angegeben, muss sie immer größer oder gleich der Stellenanzahl von <Ogrenze> sein und sollte 7 bzw. 10 nicht überschreiten. Ist <Anz> größer als die Restlänge des (Teil-)Feldes, wird eine Fehlermeldung generiert.

Wenn die Längenangabe <Anz> fehlt oder größer als 7 bzw. 10 ist, dann werden **die letzten sieben Speicherzellen** des Feldes zur Abspeicherung verwendet, unabhängig davon, auf welche Speicherzelle im Feld <Feldadr> zeigt.

Wenn kein Sprungziel <Label> angegeben ist, wird bei Erreichen der Obergrenze wieder mit <Ugrenze> als Ergebnis fortgesetzt.

Ist eine Sprungadresse <Label> angegeben, wird bei Überschreiten der Obergrenze das Ergebnis auf <Ugrenze> gesetzt und ein Sprung an die angegebene Adresse ausgeführt.

Diese Anweisung kann in Verbindung mit der Prüfanweisung verwendet werden, um Schleifen zu implementieren.

**Beispiel 1:**

DI,P4,0-6:

0,1,2,3,4,5,6,0,1,2 usw.

**Beispiel 2:**

DI,H1,4,0-999:

Das Feld ist 13 Bytes lang. Die ersten vier Stellen H1+0,..,H1+3 sind das Teifeld (<Anz>= '4'). Inkrementiert wird ab Adresse H1+1, d.h. die Einerstelle ist H1+3, die Zehnerstelle ist H1+2 und die Hunderterstelle H1+1 (0000,0001,..,0999,0000,...). H1+0 wird nicht verändert, da die Obergrenze '999' nur dreistellig ist.

DI,H1,0-999:

Inkrementiert werden die letzten drei Stellen am Ende des Feldes, d.h. die Einerstelle ist H1+12, die Zehnerstelle ist H1+11 und die Hunderterstelle H1+10, das Teifeld ist aber siebenstellig, da eine Angabe fehlt und beginnt ab H1+6. Die führenden Stellen werden auf '0' gesetzt (0000000, 0000001, .., 0000999, 0000000). H1+6,..,H1+9 werden nicht verändert, da die Obergrenze '999' nur dreistellig ist.

## 5.19 K Kopieren

**Syntax:**

K, <kQop>, <Feldadr>{, <Anz>} :

**Erklärung:**

Mit der Anweisung K wird die Zeichenkette <kQop> in ein (Teil-)Feld ab <Feldadr> kopiert. Dabei wird die Anzahl der zu kopierenden Zeichen durch das Minimum der folgenden Längen bestimmt:

- Länge von <kQop>
- Restlänge des (Teil-)Feldes ab <Feldadr>
- <Anz>, wenn angegeben.

**Beispiel 1:**

K,D,S,10:

Es werden die ersten 10 Zeichen aus dem Display in das Feld S kopiert.

**Beispiel 2:**

K,AN,D+10:

Es wird das ganze AN Feld mit einer Länge von 10 Zeichen in das Display ab der 10. Stelle kopiert.

**Beispiel 3:**

K,T,H1:

Es werden 13 Zeichen (Länge des Feldes H1) aus dem Feld T in das Feld H1 kopiert.

**Beispiel 4:**

K,<'Die Uhrzeit ist '&UR>,D:

Auf dem Display erscheint z.B.: Die Uhrzeit ist 10:30:05.

## 5.20 KW Kopieren mit Wandeln

Syntax:

```
KW, <Modus>, <kQop>, <Feldadr>{, <Wdh>} :
<Modus>      ::= <Zahlenwert>
<Wdh>        ::= <Zahlenwert>
```

Erklärung:

Mit der Anweisung KW wird eine Zeichenkette <kQop> in ein (Teil-)Feld ab <Feldadr> kopiert und entsprechend dem angegebenen <Modus> konvertiert.

**Konvertierungsmodus <Modus>**

Der Wert des Konvertierungsmodus enthält 2 Komponenten.

Die untere Speicherzelle von <Modus> kann folgende Werte und Bedeutungen annehmen:

<Modus> & "00FF"	Quellobjekt (mit Beispiel)	Zielobjekt (mit Beispiel)
0	binäre Zahl ("0B")	Hexzahl als ASCII-Zeichenkette ('0B')
1	Hexzahl als ASCII-Zeichenkette ('0B')	binäre Zahl ("0B")
2	binäre Zahl ("0B")	Dezimalzahl als ASCII-Zeichenkette ('011')
3	Dezimalzahl als ASCII-Zeichenkette ('011')	binäre Zahl ("0B")
4	binäre Zahl ("0B")	Oktalzahl als ASCII-Zeichenkette ('13')
5	Oktalzahl als ASCII-Zeichenkette ('013')	binäre Zahl ("0B")
6	binäre Zahl ("0B")	Binärzahl als ASCII-Zeichenkette ('00001011')
7	Binärzahl als ASCII-Zeichenkette ('00001011')	binäre Zahl ("0B")

Die obere Speicherzelle von <Modus> bestimmt die Länge der Konvertierungsobjekte:

<Modus> & "FF00"	Länge der binären Zahl	Länge der Hexzahl als ASCII-Zeichenkette	Länge der Dezimalzahl als ASCII-Zeichenkette
0*256	1 Byte	2 Bytes	3 Bytes
1*256	2 Bytes	4 Bytes	5 Bytes
2*256	3 Bytes	6 Bytes	8 Bytes
3*256	4 Bytes	8 Bytes	10 Bytes

<b>&lt;Modus&gt; &amp; "FF00"</b>	<b>Länge der bi-nären Zahl</b>	<b>Länge der Oktalzahl als ASCII-Zeichenkette</b>	<b>Länge der Binärzahl als ASCII-Zeichenkette</b>
0*256	1 Byte	3 Bytes	8 Bytes
1*256	2 Bytes	6 Bytes	16 Bytes
2*256	3 Bytes	8 Bytes	24 Bytes
3*256	4 Bytes	11 Bytes	32 Bytes

**Wiederholungsfaktor <Wdh>**

Die Anzahl der zu kopierenden und konvertierenden Objekte wird aus dem Minimum der folgenden Werte bestimmt:

- Länge von <kQop>, geteilt durch die Quellobjektlänge
- Restlänge des (Teil-)Feldes ab <Feldadr>, geteilt durch die Zielobjektlänge
- Wiederholungsfaktor <Wdh>, wenn angegeben.

Zusammengefasst sind also folgende Konvertierungen möglich:

<b>&lt;Modus&gt;</b>	<b>Quellobjekt</b>	<b>Zielobjekt</b>
0	1 Byte Binärwert	2-stellige ASCII-Hexzahl
1	2-stellige ASCII-Hexzahl	1 Byte Binärwert
2	1 Byte Binärwert	3-stellige ASCII-Dez.zahl
3	3-stellige ASCII-Dez.zahl	1 Byte Binärwert
4	1 Byte Binärwert	3-stellige ASCII-Oktalzahl
5	3-stellige ASCII-Oktalzahl	1 Byte Binärwert
6	1 Byte Binärwert	8-stellige ASCII-Binärzahl
7	8-stellige ASCII-Binärzahl	1 Byte Binärwert
256	2 Byte Binärwert	4-stellige ASCII-Hexzahl
257	4-stellige ASCII-Hexzahl	2 Byte Binärwert
258	2 Byte Binärwert	5-stellige ASCII-Dez.zahl
259	5-stellige ASCII-Dez.zahl	2 Byte Binärwert
260	2 Byte Binärwert	6-stellige ASCII-Oktalzahl
261	6-stellige ASCII-Oktalzahl	2 Byte Binärwert
262	2 Byte Binärwert	16-stellige ASCII-Binärzahl
263	16-stellige ASCII-Binärzahl	2 Byte Binärwert
512	3 Byte Binärwert	6-stellige ASCII-Hexzahl
513	6-stellige ASCII-Hexzahl	3 Byte Binärwert
514	3 Byte Binärwert	8-stellige ASCII-Dez.zahl
515	8-stellige ASCII-Dez.zahl	3 Byte Binärwert
516	3 Byte Binärwert	8-stellige ASCII-Oktalzahl
517	8-stellige ASCII-Oktalzahl	3 Byte Binärwert
518	3 Byte Binärwert	24-stellige ASCII-Binärzahl
519	24-stellige ASCII-Binärzahl	3 Byte Binärwert
768	4 Byte Binärwert	8-stellige ASCII-Hexzahl

769	8-stellige ASCII-Hexzahl	4 Byte Binärwert
770	4 Byte Binärwert	10-stellige ASCII-Dez.zahl
771	10-stellige ASCII-Dez.zahl	4 Byte Binärwert
772	4 Byte Binärwert	11-stellige ASCII-Oktalzahl
773	11-stellige ASCII-Oktalzahl	4 Byte Binärwert
774	4 Byte Binärwert	32-stellige ASCII-Binärzahl
775	32-stellige ASCII-Binärzahl	4 Byte Binärwert

### **Beispiel 1:**

**DKW,0,CV+8,D,1:**

Im Setupparameterfeld CV+8,1 steht der Anlaufmodus als binäre Zahl. Die KW Anweisung zeigt im Display lesbar an, welcher Anlaufmodus eingestellt ist. Enthält CV+8,1 "01" für Kaltstart, so wird im Display die 2-stellige ASCII-Hexzahl '01' ausgegeben.

### **Beispiel 2:**

**DKW,259,'00267',T:**

Die fünfstellige Dezimalzahl als ASCII-Konstante wird in eine zweistellige binäre Zahl umgewandelt und in T abgelegt. Die beiden ersten Speicherzellen von T enthalten "010B".

### **Beispiel 3:**

**IK,"494B572C302C542C54462C34373A53522C27494B2C22272654462C393426  
27222C543A273A0D4953522C542C34373A",T:**

**IKW,0,T,TF,47:SR,'IK,"&TF,94&"',T:' :**

**ISR,T,47:**

Dieses TCL-Programm ist ein *Quine*, also ein Programm, dass eine Kopie des eigenen Quellcodes ausgibt.

## 5.21 M1 Modulo 10/11 Fehlerprüfung

Syntax:

```
M1, <kQop>, <Länge>, @<Label>:  
<Länge>      ::= <Zahlenwert> | *
```

Erklärung:

Diese Anweisung überprüft eine Zeichenkette <kQop> mit der Länge <Länge> nach dem Modulo 10 oder Modulo 11 Prüfsummenverfahren, wobei die letzte Speicherzelle die Prüfziffer enthält. Wenn für <Länge> anstatt einem <Zahlenwert> (3 - 255) ein '\*' angegeben ist, wird die Länge der letzten Barcodelesung verwendet.

In P20+4,1 wird das Prüfsummenverfahren festgelegt:

P20+4,1	Prüfsummenverfahren
'0'	Modulo 10 Prüfung
'1'	Modulo 11 Prüfung

In P20+5,1 kann ein Sonderzeichen ('\*', 'x' oder '0') definiert werden, das verwendet wird, wenn bei der Modulo 11 Prüfung die Prüfsumme 10 ist, da für die Prüfziffer nur 1 Speicherzelle zur Verfügung steht.

Wenn die Prüfsumme stimmt, wird der Sprung nach <Label> ausgeführt.

Dieses Verfahren wird normalerweise als zusätzliche Prüfung von Barcodekarten verwendet. Bei Lesereingaben sollten zunächst die Fehlerprüfungen aus dem Feld B und danach die Modulo 10 oder Modulo 11 Prüfung vorgenommen werden.

### Erklärung der Prüfsummenverfahren

#### Modulo-10-Verfahren

Beispiel: 4 7 2 0 5 6 0 x      x= unbekannte Prüfziffer

1. Addition der ungeraden Positionen:  $4+2+5+0 = 11$
2. Gewichtung mit Faktor 3:  $11*3 = 33$
3. Addition der geraden Positionen:  $7+0+6 = 13$
4. Addition aus Schritt 2 und 3:  $33+13 = 46$
5. Bilden der Differenz zum nächsten Vielfachen von 10. Die Differenz ist die Prüfziffer. In diesem Fall hat die Prüfziffer den Wert 4.

### **Modulo-11-Verfahren**

Beispiel: 1 2 3 4 5 x    x= unbekannte Prüfziffer

1. Gewichtungsfaktor vergeben: Jeder Ziffer wird ein Gewichtungsfaktor 2,..,7 zugeordnet. Die Zuordnung beginnt bei der letzten Ziffer mit Faktor 2 und schreitet von rechts nach links fort. Die Faktoren werden zyklisch wiederholt, wenn die Ziffernfolge mehr als 6 Ziffern enthält. 6 5 4 3 2
2. Multiplikation der Ziffer mit dem Faktor ergibt: 6 10 12 12 10
3. Addition der Produkte:  $6 + 10 + 12 + 12 + 10 = 50$
4. Division der Summe durch 11:  $50 / 11 = 4$  Rest 6
5. Den Rest von 11 subtrahieren. Wenn die Differenz kleiner als 10 ist, dann ist die Differenz die Prüfziffer. Im anderen Fall, muss nochmals 10 subtrahiert werden, um die Prüfziffer zu erhalten. In diesem Fall hat die Prüfziffer den Wert  $11 - 6 = 5$ .

**Beispiel:**

**DM1,D+10,10,@50:**

**DM1,B,\*,@50:**

## 5.22 MU Multiplikation

### Syntax:

MU, <Zahlenwert>, <Zahlenwert>{, <Feldadr>{, <Anz>} } :

### Erklärung:

Die erste Zahl <Zahlenwert> wird mit der zweiten Zahl <Zahlenwert> multipliziert. Die Operanden und das Ergebnis sind positive ganze Zahlen. Es können maximal 10-stellige Operanden ein ebenfalls 10-stelliges Ergebnis liefern.

Die Abspeicherung des Ergebnisses hängt von den weiteren optionalen Parametern ab:

- Es sind keine weiteren Parameter angegeben: Sind <Feldadr> und damit auch <Anz> nicht angegeben, steht das Ergebnis linksbündig im ER Feld. (Kompatibilität zu den PCT88x-Terminals)
- <Feldadr> ist ohne eine Längenangabe <Anz> angegeben: Als Länge wird das Minimum von der Restlänge des (Teil-)Feldes <Feldadr> und '10' genommen. Das Ergebnis wird rechtsbündig mit führenden Nullen nach <Feldadr> geschrieben.
- <Feldadr> und <Anz> sind angegeben: Das Ergebnis wird rechtsbündig mit führenden Nullen, maximal 10-stellig eingetragen.

Die Ergebnislänge steht rechtsbündig im EZ Feld.

Das Vorzeichen des Ergebnisses steht im EV Feld (hier immer '+').

Wenn das Ergebnis in ER rechtsbündig benötigt wird, kann die Anweisung C3 aufgerufen, oder ER als Ergebnisfeld explizit angegeben werden.

### Beispiel:

D#1: K, '56', H1: MU, (H1,2), '10', D+1, 3: K, EV, D: S:

Es wird '56' mit '10' multipliziert. Das Ergebnis wird mit Vorzeichen im Display dargestellt ('+560').

## 5.23 OR Oder

### Syntax:

OR, <kQop>, <eQop>, <Feldadr>{, <Anz>} :

### Erklärung:

Die durch <kQop> und <eQop> angegebenen Speicherzellen werden bitweise durch die logische Operation "Oder" (OR) verknüpft und das Ergebnis in <Feldadr> gespeichert.

Bei Operanden, die mehrere Bytes lang sind, gilt die "Motorola-Byte-Ordnung". Das heißt, das erste Byte ist das höchstwertigste Byte.

Der optionale Parameter <Anz> gibt die (Teil-)Feldlänge an, in der das Ergebnis abgespeichert wird. Die Länge ergibt sich aus dem Minimum der Feldlängen von <kQop>, <eQop>, <Feldadr> und <Anz> (falls vorhanden).

### Beispiel :

DOR,H1,"02",H1:

Bit 2 in H1+0 setzen.

## 5.24 P Vergleich

### 5.24.1 Standardoperationen

**Syntax:**

```
P, <kQop>, <Op><eQop>{, <Anz>}, @<Label>:  
<Op> ::= > | < | = | >= | <= | <>
```

**Erklärung:**

Die Zeichenkette `<kQop>` wird mit der Zeichenkette `<eQop>` verglichen. Der Vergleich erfolgt byteweise binär. Wieviel Zeichen verglichen werden, wird durch das Minimum der Längen der beiden Operanden und den optionalen Parameter `<Anz>` bestimmt. Die Zeichenanzahl muss größer als 0 und kleiner als 32768 sein. Wenn der Vergleich zutrifft, wird der Sprung nach `<Label>` ausgeführt.

**Beispiel 1:**

```
D#0: P,H1,='22234',@1: K,'FALSCH',D+10: S:  
D#1: K,'RICHTIG',D+10: S:
```

**Beispiel 2:**

```
D#0: P,H1,=H2,5,@1: K,'FALSCH',D+10: S:  
D#1: K,'RICHTIG',D+10: S:
```

**Beispiel 3:**

```
D#0: I,H1,2,0-8:  
D#1: P,H1,<'08',2,@20:
```

### 5.24.2 PIN-Code Verifikation ab TCL V6.20 und V6.70

Die Pin-Code Prüfung steht ab TCL Version 6.20 und 6.70 nur zur Verfügung, wenn CV+307,1 den Wert '1' enthält. Aufgrund der Ressourcenanforderung ist die Implementierung nicht auf allen Terminals möglich.

**Syntax:**

```
P, <kQop>, <Op><eQop>, @<Label>:  
<kQop> ::= 'eingegebene PIN'  
          zu prüfender PIN-Code im Klartext  
<Op>    ::= PIN[0]  
<eQop> ::= 'verschlüsselte PIN'  
          zu prüfender mit öffentlichem Schlüssel verschlüsselter  
          PIN-Code im Base64-Format
```

**Erklärung:**

Für eine sichere PIN-Code Prüfung ist Unterstützung durch die TCL-Firmware erforderlich. Ab TCL Version 6.20 und 6.70 ist ein Algorithmus basierend auf Elliptischen Kurven (ECIES, Curve 25519) verfügbar. In CV+262,1 ist für diesen Algorithmus der Wert '0' eingetragen.

Der Operator `<Op>` für diese PIN-Code Verifikation ist `PIN[0]`.

Durch den Einsatz eines privaten und, daraus abgeleitet, öffentlichen Schlüssels ergibt sich die Möglichkeit PIN-Codes sicher abzulegen. Hierfür wird der PIN-Code mit Hilfe des öffentlichen Schlüssels verschlüsselt, eine Prüfung (Entschlüsselung) benötigt hingegen den privaten

Schlüssel.

## Zu <kQop>: Länge des PIN-Codes 'eingegebene PIN'

Aus Sicherheitsgründen muss der PIN-Code eine Mindestlänge von 4 Zeichen aufweisen und ist auf maximal 16 Zeichen begrenzt.

## Konfiguration des privaten Schlüssels:

Die Konfiguration des privaten Schlüssels kann im CV-Feld CV+263,44 im Base64 Format vorgenommen werden.

Alternativ kann je nach Terminalmodell der private Schlüssel auch über

- INTUS RemoteConf ab V1.02.00
  - INTUS RemoteSetup ab V3.15.00
- im ASCII-Hex oder Base64 Format konfiguriert werden.

## Zu <eQop>: Erzeugen des verschlüsselten PIN-Codes 'verschlüsselte PIN'

- Der öffentliche Schlüssel QA (passend zum privaten Schlüssel im Terminal) muss bekannt sein.
- Wähle eine sichere Zufallszahl r (Anforderungen beachten)
- Generiere R = r \* G (G = Generator der Curve25519)
- Generiere S = r \* QA
- Bilde Hash H = SHA256(S)
- Verschlüssele PIN = zeichenweise(PIN exor H)
- Lösche r, S, H
- Daten für Verifikation = base64(R + verschlüsselte PIN)

Setzen Sie sich bei Fragen zur Implementierung mit PCS in Verbindung.

## Beispiel:

I<sub>K</sub>, 'AAIDBAUGBwgJEBESEXQVFhcYGSAhIiMkJSYnKCKwMWI=' ,CV+263:

Konfiguration des privaten Schlüssels über das CV-Feld im Base64-Format:

```
IR,S:  
IR,S:  
D#0:  
DP,'data',PIN[0]'yVUYUA9F2fqEwnT71s16Pah4LDv5L8T/wxbi7Ekc8hIZMx46',@10:  
DSR,'verify failed':S:  
D#10:SR,'success':S:  
I@0:
```

Das TCL-Programm prüft den PIN-Code 'data' und gibt 'success' aus.

## 5.25 PS Prüfsumme anhängen

Syntax:

```
PS, <Feldadr>, <Anz>{, <Code>{, <Transparenzzeichen>} } :  

  <Code>           ::= <Dezimalzahl>  

  <Transparenzzeichen> ::= <Dezimalzahl>
```

Erklärung:

Die Anweisung PS berechnet eine Prüfsumme nach dem in <Code> angegebenen Verfahren über die Zeichenkette ab <Feldadr> mit der Länge <Anz> und hängt das Ergebnis von 1 oder 2 Bytes an die Zeichenkette an.

Deshalb müssen am Ende des angegebenen Feldes mindestens 2 Speicherzellen frei sein. Kann die Prüfsumme im Feld nicht mehr angehängt werden, wird eine Fehlermeldung erzeugt.

Mit <Code> können folgende Verfahren zur Berechnung einer Prüfsumme eingestellt werden:

<Code>	Verfahren	Ergebnis
0	LRC (XOR)	1 Byte binär
1	CRC16	2 Bytes binär (BSC)
2 (default)	modulo 256 Summe	2 Bytes ASCII-Hex
3	modulo 256 Summe	1 Byte binär
4	Mazak	2 Bytes binär
5	CRC (HDLC)	2 Bytes binär (HDLC)
6	CRC8 (SNI)	1 Byte binär

Wenn <Code> nicht angegeben ist, wird 2 angenommen.

Bei transparenter Datenübertragung wird <Transparenzzeichen> im Datensatz nicht in der Prüfsummenberechnung berücksichtigt. Bei doppelt vorkommenden Transparenzzeichen wird nur das zweite Zeichen in der Berechnung der Prüfsumme verwendet.

Das Transparenzzeichen <Transparenzzeichen> kann definiert werden, und muss eine Zahl zwischen 0 und 255 sein. Die Voreinstellung ist 0 und bedeutet, dass alle Zeichen in der Berechnung berücksichtigt werden.

Ist beispielsweise das Transparenzzeichen 16 (DLE,"10"), so wird in der Zeichenkette bei jedem Vorkommen von "1010" nur eine "10" in der Berechnung berücksichtigt.

**Beispiel 1:**

```
IK, '12345', TF:PS,TF,5,0:
```

TF+6 enthält als Ergebnis "31".

**Beispiel 2:**

```
D#0: K, '[' ,S: K,PN,S+1,10: PS,S,11: SR,S,13: S:
```

Führt im Normalbetrieb zu demselben Datensatzaufbau wie SE,PN: S:

## 5.26 PT Tabellenvergleich

Syntax:

```
PT, <kQop>, <Z1>, <Z2>, <Feldadr>, <Modus>, @<Label>:  
<Z1>      ::=  <Zahlenwert>  
<Z2>      ::=  <Zahlenwert>  
<Modus>    ::=  B<Zahlenwert> | S<Zahlenwert> | S* |  
                  <Zahlenwert> | *
```

Erklärung:

Die Tabellenprüfanweisung PT sucht in einem (Teil-)Feld ab <Feldadr> satzweise oder zeichenweise (<Modus> ist S<Zahlenwert> oder S\*) nach einer Suchzeichenkette <kQop> der Länge <Z1>.

Satzstruktur

Wenn in <Modus> eine satzweise Suche angegeben wurde wird für die Anweisung PT das Feld, in dem gesucht wird, nicht, wie sonst in der TCL Sprache üblich, als lineares Feld betrachtet, sondern als eine Tabelle, die aus Sätzen fester Länge besteht.

Jeder Satz besteht aus zwei Teilfeldern, dem Suchfeld der Länge <Z1> und dem Ergänzungsfeld der Länge <Z2>, wobei das Ergänzungsfeld die Länge 0 haben darf. Die Länge eines Tabelleneintrags (Satzlänge) ist <Z1> + <Z2>.

Satz 1:	Suchfeld 1	Ergänzungsfeld 1
Satz 2:	Suchfeld 2	Ergänzungsfeld 2
....	...	...
Satz n-1:	Suchfeld n-1	Ergänzungsfeld n-1
Satz n:	Suchfeld n	Ergänzungsfeld n

Wenn die Zeichenkette gefunden wurde...

- wird das Ergänzungsfeld an die Suchzeichenkette angehängt. Die Ergänzungszeichen werden verworfen, wenn das Feldende nach der Suchzeichenkette bereits erreicht ist, oder die Suchzeichenkette eine Konstante ist.
- In das SP Feld wird der Offset des **nächsten** Datensatzes 6- bzw. 8-stellig abgespeichert. Wenn die Suche nicht erfolgreich war, wird der Wert nicht verändert. Bei der Suche in einem Teifeld ist zu beachten, dass der Offset relativ zum gesamten Feld berechnet wird.
- Es wird der Sprung nach <Label> ausgeführt.

**Suchmodus <Modus>**

Die PT Anweisung vergleicht die Suchzeichenkette <kQop> der Länge <Z1> mit den Suchfeldern. Das Suchverfahren und die Anzahl der zu prüfenden Tabelleneinträge müssen angegeben werden.

- **Binäres Suchen eines Datensatzes: B<Zahlenwert>**

In einer binär sortierten Tabelle ist es möglich, den Suchalgorithmus „binäres Suchen“ anzuwenden. Es wird bei jedem Suchschritt die Tabelle halbiert und in der Hälfte weiter gesucht, in der die gesuchte Zeichenkette sein muss. Die Anzahl der Sätze, die durchsucht werden soll, muss mit <Zahlenwert> angegeben werden. Im Erfolgsfall enthält das SP

Feld den Offset des Suchfelds **nach** dem gefundenen Satz. Die Satzlänge muss davon subtrahiert werden, um den Offset des gefundenen Satzes zu erhalten.

- ***Lineare Satzsuche in einer Tabelle mit bekannter Länge: <num\_val>***

Wenn nur ein <Zahlenwert> im <Modus> Parameter angegeben ist, dann wird die Tabelle linear nach einem passenden Suchfeld (Schlüssel) durchsucht. Diese Methode wird bei unsortierten Tabellen angewendet. Die Anzahl der Datensätze in der Tabelle muss mit <num\_val> im <Modus> Parameter angegeben werden. Im Erfolgsfall enthält das SP Feld den Offset des Suchfelds **nach** dem gefundenen Satz. Die Satzlänge muss davon subtrahiert werden, um den Offset des gefundenen Satzes zu erhalten.

- ***Lineare Satzsuche in einer Tabelle mit unbekannter Länge: \****

Wenn nur ein '\*' im <Modus> Parameter angegeben ist, dann wird die Tabelle linear nach einem passenden Suchfeld (Schlüssel) durchsucht bis das Ende des Felds <Feldadr> oder bis ein Suchfeld gefunden ist, dass mit einem '\*' beginnt. Dort wird die Suche erfolglos abgebrochen. Diese Methode wird bei unsortierten Tabellen angewendet deren Länge nicht bekannt ist. Im Erfolgsfall enthält das SP Feld den Offset des Suchfelds **nach** dem gefundenen Satz. Die Satzlänge muss davon subtrahiert werden, um den Offset des gefundenen Satzes zu erhalten.

- ***Zeichenweises Suchen: S<Zahlenwert>***

Es wird zeichenweise nach einer Zeichenkette gesucht. Die Suche wird nach der angegebenen Zahl von Zeichen <Zahlenwert> oder am Ende des Felds <Feldadr> abgebrochen. Im Erfolgsfall enthält das Feld SP den Offset des ersten Zeichens der gefundenen Zeichenkette.

- ***Zeichenweises Suchen: S\****

Es wird zeichenweise nach einer Zeichenkette gesucht. Die Suche wird entweder bis zum Tabellenende oder bis zu dem Zeichen '\*' durchgeführt. Im Erfolgsfall enthält das Feld SP den Offset des ersten Zeichens der gefundenen Zeichenkette.

### Beispiel 1:

D#0: PT, '12345', 5, 0, T, B20, @8:

Prüft, ob die 5-stellige Nummer '12345' im sortierten Feld T vorhanden ist. Die Satzlänge ist 5 Bytes (5+0). Es wird binär gesucht. 20 Sätze werden überprüft. Ist die Nummer vorhanden, wird der Sprung ausgeführt.

### Beispiel 2

D#0: K, '12345', PN: PT, PN, 5, 2, T, 20, @8:

Prüft, ob eine 5-stellige Personalnummer '12345' im Feld T vorhanden ist. Die Satzlänge ist 7 Bytes (5+2). 20 Sätze werden überprüft. Ist die Nummer im Textfeld gefunden, werden die zwei hinter der Nummer im Textfeld stehenden Zeichen an die fünf im Feld PN stehenden Zeichen angefügt.

### Beispiel 3:

D#0: PT, '12345', 5, 0, TF, S\*, @8:

Es wird zeichenweise solange in der Tabelle gesucht, bis entweder '12345' gefunden wird; in diesem Fall wird der Sprung ausgeführt, oder ein '\*' bzw. das Ende des Suchfeldes erreicht wurde; in diesem Fall wird die folgende Anweisung ausgeführt.

**Beispiel 4:**

```
IR,S:  
D#0:K,'0123456789',TF:  
DPT,'5',1,0,TF+5,10,@5:  
DK,'NICHT GEFUNDEN',D:S:  
D#5:K,<'GEFUNDEN INDEX '&SP>,D:S:  
I@0:
```

Im Display wird „GEFUNDEN INDEX 000006“ ausgegeben, obwohl ab TF+5 gesucht wird.

## 5.27 R Reset

**Syntax:**

```
R, <Modul>:  
<Modul> siehe 1. und 2. Liste weiter unten
```

**Erklärung:**

Die Resetanweisung vereinigt die verschiedensten Initialisierungen und Rücksetzungen im TCL-Programmiersystem.

<Modul> kann eine Resetfunktion spezifizieren und auslösen, oder mit einem Kennbuchstaben mehrere Resetfunktionen starten. Die 1. Liste enthält die einzelnen Resetfunktionen. Die Kennbuchstaben für die zusammengesetzten Resetfunktionen folgen in der 2. Liste.

**Einzelne Resetfunktionen:**

- **R,D{ ,<Label>} :**  
setzt den Programmladezeiger auf den Anfang des Programmreichs, bzw. auf eine Sprungadresse <Label> im Programm. Ab dieser Stelle kann dann das Programm überladen werden (Overlay-Technik). Siehe auch Abschnitt 8.3.1 *Laden in den Programmreich*.

Alle nach dem angegebenen Sprungziel geladenen Sprungziele einschließlich dem Angegebenen werden gelöscht und können nicht mehr angesprungen werden.

- **R,E{ ,<Eingabeaddr> | <Fkttaste>}\* :**  
(Die Syntax von <Eingabeaddr> und <Fkttaste> sind bei der E-Anweisung definiert.)  
sperrt alle oder die angegebenen Funktionstasten-, Tastatur- und Karteneingaben. Die Sperrung kann sich auf die lokal an dem Terminal angeschlossenen Eingabeeinheiten und auf die abgesetzten Eingabeeinheiten am LBus beziehen. Auch ein schon in \$3 eingetragener, aber noch nicht ausgeführter, Eingabe-Ereignissprung wird gelöscht.

Achtung: Das Sperren der Tasten an einer abgesetzten Eingabestation sperrt gleichzeitig auch den Leser – und umgekehrt. (Das ist beabsichtigt und hat historische Gründe.)

Siehe auch Abschnitt 5.14.7 *Eingabefreigabe zurücknehmen*.

- **R,F{ ,\\$<Rp>}\* :**  
löscht die angegebenen Ringpuffer. Die Angabe von \$3 und \$4 ist nicht zulässig. Wenn keine Ringpuffer angegeben werden, dann werden alle außer \$3 und \$4 gelöscht. Da diese Funktion auch implizit ein **R,R,f ,\\$<Rp>}\*** ausführt, werden alle entsprechenden Ringpuffer Sprünge gelöscht.
- **R,L{ ,<Zahlenwert>} :**  
setzt die logische Satznummer für den MONOUT-Prozess auf den <Zahlenwert> ('1' - '9999'). Ohne die Angabe von <Zahlenwert> wird '1' eingesetzt.  
L bezieht sich auf die gesicherte Datenübertragung vom Terminal zum Host durch den MONOUT-Prozess. Die Satznummer im Terminal wird mit der im Leitrechner synchronisiert. R,L wird vom Leitrechner gesendet, wenn der Notpuffer leer ist und keine weiteren Datensätze zu quittieren sind. Siehe auch Abschnitt 6.3.4 *Betriebsart 3*.
- **R,M:**  
Diese Anweisung ist nur aus Kompatibilität zur Terminalserie PCT 88x vorhanden. Die Anweisung hat keine Funktion.
-

**R,P:**

setzt den Stackpointer für die Unterprogramm-Sprünge zurück. Wird diese Anweisung benötigt, weist das in der Regel auf eine fehlerhafte Programmierung hin.

- **R,R{,\$<Rp>}\*:**

setzt alle angegebenen aktiven Ringpuffersprünge zurück. Die Angabe von \$3 und \$4 ist nicht zulässig. Wenn kein Ringpuffer angegeben ist, werden alle aktiven Sprünge zurückgesetzt. Siehe auch Abschnitt 5.31 *RS Ringpuffer-Sprung*.

- **R,T{,<Zahlenwert>{-<Zahlenwert>}}\*:**

setzt die mit <Zahlenwert> (0-99) bezeichneten Zeitüberwachungen zurück. Werden keine weiteren Angaben gemacht, dann werden alle Timeouts zurückgesetzt. Ein Timeoutsprung wird nicht ausgeführt. Auch ein schon in \$3 eingetragener, aber noch nicht ausgeführter Timeoutsprung, wird gelöscht. Siehe auch Abschnitt 5.36.

**Zusammengesetzte Resetfunktionen:**

- **R,V:**

löscht die TCL-Felder genauso, wie es beim Kaltstart (siehe Abschnitt 8.4.3.2) geschieht, bis auf folgende Ausnahmen: P0, P1, P3, P10 und die Ex-Felder werden gerettet. Die Maximalwerte der lokalen Zähler werden zurückgesetzt.

Außerdem wird R,P: ausgeführt.

- **R,I:**

wirkt wie die Anweisungsfolge R,T: R,R: R,E: R,V:

Zusätzlich wird P2= '0' gesetzt.

- **R,S:**

wirkt wie die Anweisungsfolge R,I: R,D:

## 5.28 RD Ringpuffer lesen

**Syntax:**

```
RD, $<Rp>, <Feldadr>{, <Anz>} {, T<Zeit>, @<Label>} :  
<Zeit> ::= <Zahlenwert>
```

**Erklärung:**

Die Anweisung RD veranlasst, dass der Inhalt des Ringpuffers \$<Rp> zeichenweise ausgelesen und in das angegebene Zielfeld ab <Feldadr> kopiert wird. Es kann ohne Interpretation der Daten, also transparent, gelesen werden, oder es wird das CR-Zeichen ("0D") als Endekriterium betrachtet.

Werden die Daten transparent gelesen, muss <Anz> angegeben werden.

Werden die Daten nicht transparent gelesen, ist die Angabe von <Anz> optional. Fehlt <Anz>, wird die Zeichenanzahl auf die Ringpuffergröße (siehe Abschnitt 6.1) begrenzt. Ansonsten wird ausgelesen bis entweder die angegebene Zeichenanzahl <Anz>, oder das Feldende erreicht ist, oder bis ein CR-Zeichen ("0D"), erkannt wird.

Vom **Notpuffer \$4** können die Zeichen nicht transparent, sondern immer nur bis zum nächsten CR-Zeichen ausgelesen werden. Eine angegebene Zeichenanzahl <Anz> wird nicht berücksichtigt.

Konnten genügend Zeichen aus dem Ringpuffer gelesen werden, dann wird die auf RD folgende Anweisung ausgeführt.

### Anzahl der gelesenen Zeichen

Wenn P20+1,1 eine '1' enthält, wird die Zeichenanzahl der letzten RD Anweisung in das EZ Feld eingetragen.

### Transparenz

In P20+2,2 kann für alle Ringpuffer einzeln die Art der Übertragung eingestellt werden.

### Zeitüberwachung

Wenn nicht genügend Zeichen im Puffer vorhanden sind, wartet der Interpreter, bis die Zeit <Zeit> (1-21474836 \* 0.1 s) abgelaufen ist. Der Timeoutsprung wird direkt vom Interpreter ausgeführt und nicht als Ereignissprung in den Interpreteranweisungspuffer \$3 geschrieben.

Die Timeout Zeiteinheit ist unabhängig von P20+6,1 immer 100ms.

Während die Zeitüberwachung der RD Anweisung läuft, werden keine weiteren Anweisungen vom Interpreter ausgeführt. Der Timeout sollte deshalb immer möglichst klein gewählt werden. Wenn länger auf Zeichenempfang gewartet werden soll, dann verwenden Sie die Anweisung **RS Ringpuffer-Sprung** (Abschnitt 5.31).

Aus Kompatibilitätsgründen kann der Timeout weggelassen werden. Das sollte aber nicht benutzt werden, denn dann wartet der Interpreter 25 Sekunden. Nach Ablauf der Zeitüberwachung wird eine TCL-Systemfehlermeldung ins Display ausgegeben (siehe Abschnitt 9.3 Systemfehlermeldungen).

Ein nicht konfigurierter Ringpuffer führt immer sofort zum Timeout.

### Hinweis:

Die Resetanweisung R,F{\$<Rp>}\* leert die angegebenen Ringpuffer.

**Beispiel 1:**

```
D#0:R,F,$7:WR,$2,'77':RD,$7,AN,5,T10,@10:K,AN,D:S:  
D#10: K,'Timeout',D: S:
```

Der \$7 Puffer wird geleert, dann sendet das Terminal eine Ladeanforderung '77' an den Leitrechner und wartet anschließend mit der Anweisung RD mit Timeout auf dessen Antwort. Während der Timeout läuft, können keine weiteren TCL-Anweisungen verarbeitet werden.

**Beispiel 2:**

Das harte Warten in der Anweisung RD im vorigen Beispiel kann durch die Anweisung RS umgangen werden:

```
D#0: R,F,$7: WR,$2,'77': RS,$7,5,@1: T20,@10: S:  
D#1: RD,$7,AN,5,T0,@10: K,AN,D: S:  
D#10: R,R: K,'Timeout',D: S:
```

Die Anweisung RD wird nur ausgeführt, wenn mindestens 5 Zeichen als Quittung empfangen wurden.

## 5.29 RL Linksshift

**Syntax:**

```
RL,<kQop>,<Zahlenwert>,<Feldadr>{,<Byteanz>}:  
<Byteanz> ::= '1' | '2' | '4'
```

**Erklärung:**

Das durch <kQop> angegebene Byte/Wort/Doppelwort wird bitweise <Zahlenwert> (maximal 32) mal links geshiftet und das Ergebnis in <Feldadr> gespeichert. Die links führenden Bits gehen verloren, es werden rechts Nullen nachgezogen.

<Byteanz> bestimmt über wieviel Speicherzellen geshiftet wird:

<Byteanz>	Speichereinheit
'1'	Byte (8Bit)
'2'	Wort (16 Bit)
'4'	Doppelwort (32 Bit)

Als Voreinstellung für <Byteanz> wird die Länge von <kQop>, höchstens aber '4', genommen.

<Byteanz> bestimmt sich aus dem Minimum der Länge von <kQop>, der Restlänge von <Feldadr> und der angegebenen <Byteanz>.

Bei Operanden, die mehrere Bytes lang sind, gilt die "Motorola-Byte-Ordnung". Das heißt, das erste Byte ist das höchstwertigste Byte.

**Beispiel:**

```
D#0: RL,"1234",4,T,2: KW,0,T,D,2: S:
```

ergibt im Display '2340'. In den zwei Bytes T+0,2 wird das Wort "1234" binär um 4 Bits nach links geshiftet.

## 5.30 RR Rechtsshift

**Syntax:**

```
RR, <kQop>, <Zahlenwert>, <Feldadr>{, <Byteanz>}:
<Byteanz> ::= '1' | '2' | '4'
```

**Erklärung:**

Das durch <kQop> angegebene Byte/Wort/Doppelwort wird bitweise <Zahlenwert> (maximal 32) mal rechts geshiftet und das Ergebnis in <Feldadr> gespeichert. Die rechts führenden Bits gehen verloren, es werden links Nullen nachgezogen.

<Byteanz> bestimmt über wieviel Speicherzellen geshiftet wird:

<Byteanz>	Speichereinheit
'1'	Byte (8Bit)
'2'	Wort (16 Bit)
'4'	Doppelwort (32 Bit)

Als Voreinstellung für <Byteanz> wird die Länge von <kQop>, höchstens aber '4', genommen.

<Byteanz> bestimmt sich aus dem Minimum der Länge von <kQop>, der Restlänge von <Feldadr> und der angegebenen <Byteanz>.

Bei Operanden, die mehrere Bytes lang sind, gilt die "Motorola-Byte-Ordnung". Das heißt, das erste Byte ist das höchstwertigste Byte.

**Beispiel:**

```
D#0: RR, "1234", 4, T, 2: KW, 0, T, D, 2: S:
```

ergibt im Display '0123'. In den zwei Bytes T+0,2 wird das Wort "1234" binär um 4 Bits nach rechts geshiftet.

## 5.31 RS Ringpuffer-Sprung

### Syntax:

RS, \$<Rp>, <Anz>, @<Label>:

### Erklärung:

Mit der Anweisung RS wird eine Überwachung des Ringpuffers \$<Rp> aktiviert.

Die Ringpuffer \$3 (Interpreter–Anweisungspuffer) und \$4 (Notpuffer) bilden eine Ausnahme. Sie können nicht überwacht werden.

Für einen Senderingpuffer \$<Rp> (z.B. \$6) gilt, dass ein Ereignissprung mit Sprungziel <Label> ausgeführt wird, sobald eine bestimmte Zeichenanzahl <Anz> im Ringpuffer \$<Rp> erreicht ist. So ist <Anz> = '0' sinnvoll, um festzustellen, ob ein Datensatz bereits gesendet wurde.

Für einen Empfangsringpuffer \$<Rp> (z.B. \$7) gilt dagegen, dass bei Erreichen - oder Überschreiten - einer bestimmten Zeichenanzahl <Anz> im Ringpuffer \$<Rp> ein Ereignissprung mit Sprungziel <Label> ausgeführt wird. Hier ist <Anz> mit einem Wert größer als '0' sinnvoll, um auf den Empfang von einem Datensatz zu reagieren.

Die Anweisung RS wartet, im Gegensatz zur Anweisung RD, nicht bis Zeichen im Puffer stehen. Wenn die Zeichenanzahl im Ringpuffer \$<Rp> erreicht wird, und der Interpreter auf einer Stopanweisung steht, wird der Sprung nach <Label> ausgeführt.

Nach jedem Sprung muss die Überwachung erneut durch eine RS Anweisung aktiviert werden.

Die Überwachung eines Ringpuffers kann durch die Resetanweisung R,R,\$<Rp> zurückgesetzt werden. Die Resetanweisung R,F{\$<Rp>}\* löscht die angegebenen Ringpuffer.

### Beispiel:

```
D#0: RS,$7,5,@2: T20,@1: %100: S:  
D#1: R,R,$7: F,D,80, : K,'Timeout',D: @0:  
D#2: K,'Quittung ok',D: RD,$7,D+40: @0:  
D#100:....: %:
```

Durch die Anweisung RS wird auf eine Quittung im Interpreter-Datenpuffer \$7 gewartet. Wenn sie nicht innerhalb von 2 Sekunden kommt, wird 'Timeout' ausgegeben, im anderen Fall wird die Quittung mit der Anweisung RD gelesen.

Die Anweisung RS (im Gegensatz zur Anweisung RD mit Timeout) wartet nicht, bis Zeichen im Puffer stehen. Dadurch können in Sprungziel 0 weitere Anweisungen ausgeführt werden (angedeutet durch den Unterprogrammsprung).

## 5.32 S Stop

**Syntax:**

S :

**Erklärung:**

Der Interpreter führt eine Folge von TCL-Anweisungen bis zu einer Stopanweisung aus. Danach wartet er, bis Anweisungen im Interpreteranweisungspuffer \$3 stehen. Dies können Anweisungen aus Datensätzen vom Host mit MONIN-Adressbyte 'T' sein, oder Ereignissprünge. Eine Anweisung, die hinter einer Stopanweisung in einer Programmzeile steht, wird nicht (mehr) ausgeführt.

Welche Ereignisse in TCL dazu führen, dass Sprunganweisungen in \$3 eingetragen werden, kann im Abschnitt 2.2 nachgelesen werden.

Wenn eine Sprunganweisung im Puffer \$3 steht, wird das TCL Programm an der entsprechenden Stelle fortgesetzt.

Stopanweisungen müssen im TCL-Programm vorkommen, um auf Ereignisse reagieren zu können. Zum Beispiel stehen sinnvollerweise nach Eingabe- und Timeoutanweisungen Stopanweisungen.

**Beispiel:**

D#0: SR,'77': F,D,/D/, : K,'Nicht bereit',D: S:

Es wird '77' an den Leitrechner gesendet, das Display gelöscht und 'Nicht bereit' angezeigt, danach wird das Programm beendet. Der Interpreter wartet, bis Anweisungen vom Leitrechner empfangen werden.

## 5.33 SB Subtraktion

### Syntax:

SB, <Zahlenwert>, <Zahlenwert>{, <Feldadr>{, <Anz>} } :

### Erklärung:

Die zweite Zahl <Zahlenwert> wird von der ersten Zahl <Zahlenwert> abgezogen. Die Operanden sind positive ganze Zahlen. Es können maximal 10-stellige Operanden ein ebenfalls 10-stelliges Ergebnis liefern.

Die Abspeicherung des Ergebnisses hängt von den weiteren optionalen Parametern ab:

- Es sind keine weiteren Parameter angegeben: Sind <Feldadr> und damit auch <Anz> nicht angegeben, steht das Ergebnis linksbündig im ER Feld. (Kompatibilität zu den PCT88x-Terminals)
- <Feldadr> ist ohne eine Längenangabe <Anz> angegeben: Als Länge wird das Minimum von der Restlänge des (Teil-)Feldes <Feldadr> und '10' genommen. Das Ergebnis wird rechtsbündig mit führenden Nullen nach <Feldadr> geschrieben.
- <Feldadr> und <Anz> sind angegeben: Das Ergebnis wird rechtsbündig mit führenden Nullen, maximal 10-stellig eingetragen.

Die Ergebnislänge steht rechtsbündig im EZ Feld.

Die Subtraktion kann ein negatives Ergebnis liefern. Das Vorzeichen des Ergebnisses steht in EV ('+' oder '-' ).

Wenn das Ergebnis in ER rechtsbündig benötigt wird, kann die Anweisung C3 aufgerufen, oder ER als Ergebnisfeld explizit angegeben werden.

### Beispiel:

D#1: K, '56', H1: SB, (H1,2), '100', D+1, 3: K, EV, D: S:

Es wird '100' von '56' (in H1,2) subtrahiert. Das Ergebnis wird mit Vorzeichen im Display dargestellt ('-044').

## 5.34 SE Senden an Host (über Notpuffer \$4)

**Syntax:**

```
SE, <Zeichenkette>{, @<Label1>} {, T<Zeit>, @<Label2>} :
<Zeichenkette>    ::= <Zfolge>{&<Zfolge>}*
<Zeit>            ::= <Zahlenwert>
<Label1>          ::= <Zahlenwert>
<Label2>          ::= <Zahlenwert>
```

**Erklärung:**

Die Sendeanweisung SE kopiert die angegebene Zeichenkette in den Notpuffer \$4. Anschließend wird ein CR-Zeichen angehängt. Die maximal zulässige Sendesatzlänge inklusive den Speicherzellen zum Datensatzaufbau, aber ohne CR-Zeichen, beträgt 255 Bytes. Siehe auch Abschnitt 6.3.1.

**Achtung:** Die Syntax für die Verknüpfung von mehreren Zeichenfolgen ist hier anders als bei den anderen Anweisungen: '<' und '>' fehlen!

Im Gegensatz zur Anweisung SR werden die Datensätze im Notpuffer \$4 gespeichert, wo sie vor Verlust geschützt sind. Jeder im Notpuffer erfasste Datensatz wird erst dann gelöscht, wenn er vom Leitrechner quittiert wurde. In diesem Puffer tritt auch bei Stromausfall kein Datenverlust auf, da er batteriegepuffert ist. Das Datensicherungskonzept und der Datensatzaufbau ist im Abschnitt 6.3 *MONOUT-* beschrieben.

**Sendefehler-Sprung @<Label1>**

Der Interpreter wartet bis die Daten in den Notpuffer \$4 übertragen werden konnten. Falls der Notpuffer durch die Flusskontrolle des Protokolls oder durch Offline gehen des Hosts voll ist, wartet der Interpreter eine Zeit lang. Wenn das Sprungziel <Label1> angegeben ist, wartet der Interpreter maximal 1 Sekunde und springt nach Ablauf der Zeitüberwachung nach <Label1>. Wenn <Label1> fehlt, wartet er maximal 25 Sekunden und gibt nach Ablauf der Zeitüberwachung eine TCL-Systemfehlermeldung ins Display aus (Abschnitt 9.3 *Systemfehlermeldungen*). Wird die Anweisung durch einen solchen Timeout beendet, wird der Datensatz verworfen.

**Notpuffer-Statusflag P1**

Wenn die Datensätze vom Leitrechner nicht mehr quittiert werden (Offline-Betrieb), kann es passieren, dass der Notpuffer \$4 bereits voll ist, wenn ein neuer Datensatz erfasst werden soll. In diesem Fall wird das Notpuffer-Statusflag P1 auf „Notpuffer voll“ gesetzt (P1,1='1'). (Wenn ein Sprungziel in P1+1,4 eingetragen ist, führt das zu einem Ereignissprung nach der nächsten Stopanweisung.) Auch in diesem Fall wird ein Sprung nach <Label1> ausgeführt, wenn er angegeben ist. Wenn nicht, unterbleibt eine Fehlermeldung, da durch das P1 Flag die Fehler situation bereits bekannt gemacht wurde.

**Hostreaktionstimeout-Sprung T<Zeit>, @<Label2>**

Am Ende der Anweisung kann eine Zeitüberwachung mit Sprung nach <Label2> angegeben werden, mit der der Empfang von Daten vom Host überwacht wird. Die Zeiteinheit für die Zeitangabe <Zeit> beträgt 100ms, unabhängig von P20+6,1.

Nachdem der Satz in den Notpuffer \$4 geschrieben wurde, wird eine Zeitüberwachung angefordert. Während dieser Zeit wird der Empfangspuffer \$1 auf ankommende Zeichen geprüft. Nach Empfang eines beliebigen Zeichens wird die Zeitüberwachung abgebrochen und die nächste TCL Anweisung bearbeitet.

Wird innerhalb der Überwachungszeit kein Zeichen empfangen, so führt der Interpreter den Timeoutsprung nach <Label2> aus.

**Beispiel 1:**

D#1: E,(M,1-20N,@2): S:  
D#2: SE,'22['&UR&M,20: @1:

Der Datensatz, der über den Notpuffer an den Host gesendet wird, besteht aus:

Satzkennung (Routinginformation) '22'

Statusflag '[' für "Online", d.h. der Datensatz ist aktuell.

Daten: Uhrzeit, Magnetkarteneingabe von bis zu 20 Zeichen.

MONOUT fügt zwischen Daten und CR-Zeichen eine Checksumme ein.

**Beispiel 2:**

DSE,P10,4&'\*Keep alive'&UR:

Der Datensatz, der über den Notpuffer an den Host gesendet wird, besteht aus:

Satzkennung (Routinginformation): 4 Bytes aus dem Teifeld P10,8

Statusflag '\*' für "Satz nur relevant, wenn eine Verbindung zum Host besteht."

Daten: 'Keep alive', Uhrzeit

MONOUT fügt zwischen Daten und CR-Zeichen eine Checksumme ein.

## 5.35 SR Senden an Host (direkt über Ringpuffer \$2)

Syntax:

```
SR,<Zeichenkette>{,@<Label1>}{},T<Zeit>,@<Label2>}:  
<Zeichenkette> ::= <Zfolge>{&<Zfolge>}*  
<Zeit> ::= <Zahlenwert>  
<Label1> ::= <Zahlenwert>  
<Label2> ::= <Zahlenwert>
```

Erklärung:

Die Sendeanweisung SR kopiert die angegebene Zeichenkette in den Sendepuffer der Hostschnittstelle \$2. Anschließend wird ein CR-Zeichen angehängt. Die maximal zulässige Sendesatzlänge beträgt 255 Bytes (ohne CR-Zeichen).

**Achtung:** Die Syntax für die Verknüpfung von mehreren Zeichenfolgen ist hier anders als bei den anderen Anweisungen: '<' und '>' fehlen!

Die Anweisung SR wird nur ausgeführt, wenn das Prozedurstatusflag PO,1= '0' (online) ist. Im anderen Fall PO,1='1' wird der Datensatz verworfen.

Da die Datensätze nicht gepuffert werden, müssen sie auch nicht wie bei der Anweisung SE quittiert werden. Wenn Sie eine gesicherte Datenerfassung und Übertragung benötigen, dann sollten Sie die Anweisung SE verwendet.

### Sendefehler-Sprung @<Label1>

Der Interpreter wartet bis die Daten in den Sendepuffer \$2 übertragen werden konnten. Dies kann durch die Flusskontrolle des Protokolls oder durch Offline gehen des Hosts länger dauern. Wenn das Sprungziel <Label1> angegeben ist, wartet der Interpreter maximal 1 Sekunde und springt nach Ablauf der Zeitüberwachung nach <Label1>. Wenn <Label1> fehlt, wartet er maximal 25 Sekunden und gibt nach Ablauf der Zeitüberwachung eine TCL-Systemfehlermeldung ins Display aus (Abschnitt 9.3 *Systemfehlermeldungen*). Wird die Anweisung durch einen solchen Timeout beendet, wird der Datensatz verworfen.

### Hostreaktiontimeout-Sprung <Zeit>, @<Label2>

Am Ende der Anweisung kann eine Zeitüberwachung mit Sprung nach <Label2> angegeben werden, mit dem der Empfang von Daten vom Host überwacht wird. Die Zeiteinheit für die Zeitangabe <Zeit> beträgt 100ms, unabhängig von P20+6,1.

Nachdem der Satz in den \$2 Sendepuffer geschrieben wurde, wird eine Zeitüberwachung angefordert. Während dieser Zeit wird der Empfangspuffer \$1 auf ankommende Zeichen geprüft. Nach Empfang eines beliebigen Zeichens wird die Zeitüberwachung abgebrochen und die nächste TCL Anweisung bearbeitet.

Ist das Prozedurstatusflag PO,1 = '1' (offline zum Host), oder wird innerhalb der Überwachungszeit kein Zeichen empfangen, so führt der Interpreter den Timeoutsprung nach <Label2> aus.

Der Sprung wird sofort ausgeführt, wenn der Host offline ist.

### **Prozedurstatusflag PO**

Bei einem Verbindungsprotokoll, das eine Feststellung des Verbindungsstatus nicht erlaubt, z.B. TTY-Protokoll mit XON/XOFF Flusskontrolle, wird bei einem durch eine blockierte Flusskontrolle ausgelöstem Timeout das Prozedurstatusflag PO gesetzt (PO,1='1').

Das PO-Flag wird durch den MONIN Prozess bei Empfang eines Zeichens zurückgesetzt. Dadurch kann ähnlich wie bei BSC und TCP/IP eine Offline-Überwachung durchgeführt werden.

BSC und TCP/IP-Protokolle erkennen den Offline Zustand auf Protokollebene und setzen das Prozedurstatusflag PO entsprechend.

#### **Hinweis:**

Mit der Resetanweisung R,F{\$<Rp>}\* werden die angegebenen Ringpuffer gelöscht.

## 5.36 T Timeout

**Syntax:**

```
T{[<Nr>]}<Zeit>, @<Label>:  
<Nr> ::= <Zahlenwert>  
<Zeit> ::= <Zahlenwert>
```

Die Anweisung T bewirkt, dass nach Ablauf der angegebenen Zeit *<Zeit>* eine Sprunganweisung mit dem Sprungziel *<Label>* in den Interpreteranweisungspuffer \$3 eingetragen wird. Es können bis zu 100 voneinander unabhängige Zeitüberwachungen bzw. Wecker in Auftrag geben werden. Die Timeouts werden mit *<Nr>* (0-99) gekennzeichnet. Ist *<Nr>* nicht angegeben, wird der Wert 0 eingesetzt.

Beim Zeitüberwachungssprung wird die Nummer des auslösenden Timers als 2-stellige ASCII-Zahl ('00'-'99') in P20+24,2 geschrieben. Der Inhalt ist nur bis zur nächsten Stopanweisung aktuell.

Vor TCL Version 6.0 standen 63 verschiedene Timer ('00'-'62') zur Verfügung.

Der Interpreter hält nach Abarbeitung der Timeoutanweisung nicht an, sondern verarbeitet sofort alle weiteren Anweisungen bis zu einer Stopanweisung. Nach Ablauf der Zeit *<Zeit>* wird die laufende Verarbeitung nicht unterbrochen, sondern erst nach einer Stopanweisung wird der Timeoutsprung dem \$3 Puffer entnommen.

**Zeitangabe *<Zeit>*:**

Die Zeiteinheit für die Zeitangabe wird in P20+6,1 festgelegt.

P20+6,1	Zeiteinheit
'0' (default)	0,1 s (100ms)
'1'	0,01s (10ms)

Der Wertebereich für die Zeitangabe ist 1-21474836.

Ein oder mehrere laufende Zeitüberwachungen können mit der Resetanweisung R,T jederzeit beendet werden. Dabei wird auch ein schon in \$3 eingetragener, aber noch nicht ausgeführter, Sprung gelöscht.

Eine neue Timeoutanweisung für Timer *<Nr>* setzt Zeit und Sprungziel neu. Die alte Anweisung ist wie bei der Anweisung R,T zurückgesetzt.

**Beispiel 1:**

```
D#0: K,'Bitte warten',D: T10,@1: S:  
D#1: F,D,24, : T10,@0: S:
```

Auf dem Display blinkt der Text "Bitte warten" in Intervallen von 1 Sekunde.

**Beispiel 2:**

```
DI,P11,0-99: K,P11,D: T10,@@: S:
```

Nach Ablauf der Zeitüberwachung wird hier auf die Inkrementier-Anweisung I,P11,.. gesprungen.

## 5.37 UN Und

**Syntax:**

```
UN, <kQop>, <eQop>, <Feldadr>{, <Anz>} :
```

**Erklärung:**

Die durch <kQop> und <eQop> angegebenen Speicherzellen werden bitweise durch die logische Operation "Und" (UN) verknüpft und das Ergebnis in <Feldadr> gespeichert.

Bei Operanden, die mehrere Bytes lang sind, gilt die "Motorola-Byte-Ordnung". Das heißt, das erste Byte ist das höchstwertigste Byte.

Der optionale Parameter <Anz> gibt die (Teil-)Feldlänge an, in der das Ergebnis abgespeichert wird. Die Länge ergibt sich aus dem Minimum der Feldlängen von <kQop>, <eQop>, <Feldadr> und <Anz> (falls vorhanden).

**Beispiel :**

```
D#0: UN, "FD", H1, H1:
```

Bit 2 in H1+0 löschen.

**Beispiel:**

```
D#0: UN,H1,"02",T: P,T,="00",@"1: K,'Bit gesetzt',D: S:
```

```
D#1: K,'Bit nicht gesetzt',D: S:
```

Prüfen, ob Bit 2 in H1+0 gesetzt ist.

## 5.38 WO Schreiben in Ringpuffer ohne CR

**Syntax:**

```
WO, $<Rp>, <kQop>{, <Anz>} :
```

**Erklärung:**

Mit der Anweisung WO (Schreiben ohne CR-Zeichen) wird eine Zeichenkette <kQop> in den Ringpuffer \$<Rp> kopiert, ohne dass ein CR-Zeichen angefügt wird. Die maximal zulässige Zeichenanzahl beträgt 255 Bytes.

Der Interpreter wartet bis zu 25 Sekunden auf das fehlerfreie Schreiben. Im Fehlerfall erscheint eine TCL-Systemfehlermeldung im Display (Abschnitt 9.3 *Systemfehlermeldungen*).

Die Anweisung WO dient nur zur zeichenweisen Übertragung über die Senderingpuffer (\$2,\$6,...), z.B. zum Senden von Steuerzeichen an einen angeschlossenen Drucker.

**Hinweis:**

Mit der Resetanweisung R,F{\$<Rp>}\* werden die angegebenen Ringpuffer gelöscht.

**Beispiel :**

```
WO,$6,"0D0A":
```

CR und LF werden auf die Zusatzschnittstelle ausgegeben.

## 5.39 WR Schreiben in Ringpuffer mit CR

**Syntax:**

WR, \$<Rp>, <kQop>{, <Anz>} :

**Erklärung:**

Mit der Anweisung WR wird eine Zeichenkette <kQop> in den Ringpuffer \$<Rp> kopiert, anschließend wird ein CR-Zeichen angefügt. Die maximal zulässige Zeichenanzahl beträgt 255 Bytes.

Der Interpreter wartet bis zu 25 Sekunden auf das fehlerfreie Schreiben. Im Fehlerfall erscheint eine TCL-Systemfehlermeldung (Abschnitt 9.3 *Systemfehlermeldungen*).

**Hinweise:**

- WR,\$3 ist nicht zulässig. In den Interpreteranweisungspuffer \$3 darf nicht geschrieben werden.
- Verwenden Sie die SE Anweisung statt WR,\$4 (Notpuffer).
- Verwenden Sie die SR Anweisung statt WR,\$2 (Sendepuffer Hostschnittstelle).
- Die WR,\$2 Anweisung schreibt nur auf \$2, falls das Prozedurstatusflag PO,1='0' (online) ist. Im anderen Fall wird der Datensatz verworfen.
- Mit der Resetanweisung R,F{\$<Rp>}\* werden die angegebenen Ringpuffer gelöscht.
- Das Beschreiben interner Ringpuffer (z.B. \$1) kann, je nach Auslastung des Terminals und Design der Host-Kommunikation, zu kurzzeitigen Verklemmungen führen!

**Beispiel:**

WR, \$2,AN,5:

Die 5-stellige Auftragsnummer wird mit CR-Zeichen in den Rechnerausgangspuffer geschrieben.

## 5.40 XA Binäre Addition

Syntax:

```
XA, <kQop>, <eQop>, <Feldadr>{, <Byteanz>}:  
<Byteanz> ::= '1' | '2' | '4'
```

Erklärung:

Die binäre Addition mit XA entspricht einer Vektoraddition mit Elementen der Länge <Byteanz>. Die durch <kQop> und <eQop> angegebenen Speicherzellen werden binär elementweise addiert, dabei werden Überträge ignoriert. Das Ergebnis wird in <Feldadr> gespeichert. Die Länge des Ergebnisses ist das Minimum der Längen von <kQop>, <eQop> und der Restlänge des (Teil-)Feldes ab <Feldadr>.

<Byteanz> gibt an, wie lang die Elemente der Addition sind:

<Byteanz>	binäre Addition über...	Ergebnis modulo...
'1'	Byte (8 Bit)	modulo $2^{**}8$ (256)
'2'	Wort (16 Bit)	modulo $2^{**}16$ (65536)
'4'	Doppelwort (32 Bit)	modulo $2^{**}32$ (16777216)

Als Voreinstellung für <Byteanz> wird das Minimum der Längen von <kQop> und <eQop>, höchstens aber '4', genommen.

Bei Operanden, die mehrere Bytes lang sind, gilt die "Motorola-Byte-Ordnung". Das heißt, das erste Byte ist das höchstwertigste Byte.

Beispiel:

```
D#0: XA,"8D","8A",T: KW,0,T,D,1: S:
```

die Addition ergibt "0117", modulo 256 ergibt "17" und die Umwandlung in ASCII-Zeichen (KW) ergibt '17' im Display.

## 5.41 XS Binäre Subtraktion

**Syntax:**

```
XS, <kQop>, <eQop>, <Feldadr>{, <Byteanz>}:
<Byteanz> ::= '1' | '2' | '4'
```

**Erklärung:**

Die binäre Subtraktion mit XS entspricht einer Vektorsubtraktion mit Elementen der Länge <Byteanz>. Die durch <kQop> und <eQop> angegebenen Speicherzellen werden binär elementweise subtrahiert, dabei werden Überträge ignoriert. Das Ergebnis wird in <Feldadr> gespeichert. Die Länge des Ergebnisses ist das Minimum der Längen von <kQop>, <eQop> und der Restlänge des (Teil-)Feldes ab <Feldadr>.

<Byteanz> gibt an, wie lang die Elemente der Subtraktion sind:

<Byteanz>	binäre Addition über...	Ergebnis modulo...
'1'	Byte (8 Bit)	modulo $2^{**}8$ (256)
'2'	Wort (16 Bit)	modulo $2^{**}16$ (65536)
'4'	Doppelwort (32 Bit)	modulo $2^{**}32$ (16777216)

Als Voreinstellung für <Byteanz> wird das Minimum der Längen von <kQop> und <eQop>, höchstens aber '4', genommen.

Bei Operanden, die mehrere Bytes lang sind, gilt die "Motorola-Byte-Ordnung". Das heißt, das erste Byte ist das höchstwertigste Byte.

**Beispiel:**

```
D#0: XS,"8D","8A",T: KW,0,T,D,1: S:
```

die Subtraktion ergibt "03", und die Umwandlung in ASCII-Zeichen (KW) ergibt '03' im Display.



## 6 Laufzeitsystem

Die INTUS 3000 Terminals arbeiten auf der Basis eines Realzeit-Betriebssystems. Mehrere Prozesse nehmen die Aufgaben des Terminals wahr:

Prozess	Funktion
MONIN	TCL-Programme und Daten vom Leitrechner laden, TCL-Programme übersetzen
MONOUT	Daten gegen Verlust schützen und an den Leitrechner übertragen
INTERP	Interpreter: TCL-Programme ausführen
TIMEOUT	Timeout-Default: Zeit überwachen
AKTUELL	Aktualisierungen durchführen
EINGABE	Tastatur- und Lesereingaben verarbeiten, je nach Konfiguration Daten von den Zusatzschnittstellen, Kanal B bis D, verarbeiten.
MASCH	Maschinentask: Digitale Eingangssignale verarbeiten
BSCM-RX, BSCM-TX	Daten des Empfangsringpuffers \$C und des Senderingpuffers \$D verarbeiten, siehe Abschnitt 6.1.8
DNCIN0-2	(Digital Numerical Control INput) Daten von Kanal B, C oder D verarbeiten
TTYIN0-3, TTYOUT0-3	Treiber: Kommunikation mit Leitrechner und Geräten an Kanal B bis D durchführen

**Tabelle 6.1 – Prozesse im INTUS TCL**

Der Datenaustausch zwischen den Prozessen erfolgt über Ringpuffer und Message Passing. Das Message Passing ist für den Benutzer nicht sichtbar und nicht zugänglich. Die Ringpuffer dagegen sind mit TCL-Anweisungen zugänglich.

In diesem Abschnitt werden die Ringpuffer zusammenfassend behandelt, sowie MONIN und MONOUT näher besprochen. Die anderen Prozesse sind im Rahmen der Abschnitte Felder und Anweisungen soweit beschrieben, wie es für den TCL-Programmierer von Interesse ist. Zum Beispiel werden im Abschnitt *Dx DNCIN-Steuerfeld* die DNCIN Prozesse, und bei der Eingabeanweisung E der EINGABE Prozess behandelt. Die Abschnitte *Ex - Digitaler Eingang DI, Zx Zähler zum digitalen Eingang, Tx Taktüberwachung, TAx Taktausfall, TOx Taktabweichung, P2 Maschinentaskflag* und der Abschnitt 6.4 enthalten die Beschreibung des Maschinentask MASCH.

## 6.1 Ringpuffer

Es gibt folgende Ringpuffer:

Nummer	Ringpuffer	Größe
\$1	Empfangspuffer Hostschnittstelle	256 Byte
\$2	Sendepuffer Hostschnittstelle	273 Byte
\$3	Interpreter-Anweisungspuffer	512 Byte
\$4	Notpuffer	einstellbar (3kB – 12 MB)
\$5	Empfangspuffer Kanal B	256 Byte
\$6	Sendepuffer Kanal B	256 Byte
\$7	Interpreter-Datenpuffer	256 Byte
\$8	Empfangspuffer Kanal C	256 Byte
\$9	Sendepuffer Kanal C	256 Byte
\$A	Empfangspuffer Kanal D	256 Byte
\$B	Sendepuffer Kanal D	256 Byte
\$C	Empfangspuffer für abgesetzte Eingabestationen am LBus	256 Byte
\$D	Sendepuffer für abgesetzte Eingabestationen am LBus	256 Byte

*Tabelle 6.2 – Ringpuffer in INTUS TCL*

Prinzipiell kann auf alle Ringpuffer lesend und schreibend zugegriffen werden. Dabei sind jedoch die jeweilige Funktion des Ringpuffers und die normale Verarbeitung zu beachten. Es ist z.B. weder sinnvoll, den Sendepuffer der 2. Zusatzschnittstelle (\$6) zu lesen, da diese Daten automatisch gesendet werden, noch ist es sinnvoll in den Interpreteranweisungspuffer (\$3) zu schreiben.

Die Datensätze im Ringpuffer sind höchstens so groß wie der Ringpuffer selbst und werden durch das CR-Zeichen ("0D") beendet. Die Daten können aber auch transparent aus den Ringpuffern gelesen werden (siehe RD Anweisung und die Teilsteller P20+2,1 sowie P20+3,1).

### 6.1.1 Empfangspuffer Hostschnittstelle (\$1)

Alle Daten, die vom Leitrechner an das Terminal gesendet werden, kommen in den Ringpuffer \$1. Der MONIN-Prozess sorgt für die Weiterverarbeitung entsprechend dem Adressbyte am Anfang eines Datensatzes.

Ein DNCIN-Prozess kann Daten in \$1 schreiben, wenn sein Konfigurationsbyte Dx,1='7' gesetzt ist.

### **6.1.2 Sendepuffer Hostschnittstelle (\$2)**

Alle Daten, die vom Terminal an den Leitrechner gesendet werden, laufen über den Ringpuffer \$2. Es gibt folgende Quellen, die in den Ringpuffer schreiben:

- TCL-Anweisungen SR und WR,\$2
- Datensätze aus dem Notpuffer \$4 ( TCL-Anweisung SE und Bearbeitung durch den MONOUT-Prozess)
- Datensätze von Kanal B, C oder D (gesendet vom zugehörigen DNCIN-Prozess, wenn sein Konfigurationsbyte Dx,1='0' oder Dx,1='9' ist.)
- Fehlermeldungen des TCL-Systems, wenn TR+6,1 auf '2' gesetzt ist.
- Letztes Sprungziel, wenn TR+4,1 auf '1' gesetzt ist.

### **6.1.3 Interpreter-Anweisungspuffer (\$3)**

Wenn der Interpreter eine Stopanweisung S ausgeführt hat, holt er sich die nächste Anweisung aus dem Ringpuffer \$3. Hier werden einerseits die Ereignissprünge eingetragen und andererseits vom MONIN-Prozess die übersetzten Interpreteranweisungen mit Adressbyte 'T' abgelegt.

Das Konzept der Ereignissprünge ist im Abschnitt 2.2 beschrieben.

### **6.1.4 Notpuffer (\$4)**

Dieser Ringpuffer, dessen Größe im Setup oder in CV+5,1 eingestellt werden kann, ist ein batteriegepufferter Datenbereich, der auch bei Stromausfall vor Datenverlust geschützt ist. Die enthaltenen Datensätze werden vom MONOUT-Prozess automatisch gesendet und erst dann gelöscht, wenn der korrekte Empfang vom Leitrechner quittiert wurde. In ihm können auch im Offline-Betrieb alle erfassten Datensätze gespeichert werden.

Mit der TCL-Anweisung SE können Datensätze in den Notpuffer geschrieben werden. Ein DNCIN-Prozess schreibt Daten in den Notpuffer, wenn sein Konfigurationsbyte Dx,1='1' ist.

### **6.1.5 Empfangspuffer Kanal B, C oder D (\$5, \$8 oder \$A)**

Alle Daten, die an der 2., 3. oder 4. seriellen Zusatzschnittstelle (Kanal B, C oder D) empfangen werden, kommen in den Ringpuffer \$5, \$8 oder \$A. Der zugehörige DNCIN-Prozess sorgt für die Weiterleitung der Daten, je nachdem, was im Konfigurationsbyte Dx,1 eingestellt ist.

Ist allerdings Dx,1='5', so ist der DNCIN-Prozess deaktiviert, und die Daten bleiben im Empfangspuffer bis sie durch eine RD-Anweisung ausgelesen werden.

### **6.1.6 Sendepuffer Kanal B, C oder D (\$6, \$9 oder \$B)**

Alle Daten, die vom Terminal über die 2., 3. oder 4. serielle Zusatzschnittstelle gesendet werden, laufen über den Ringpuffer \$6, \$9 oder \$B.

Es gibt folgende Quellen, die in die Senderingpuffer schreiben:

- TCL-Anweisungen WR,\$6 und WR,\$9 und WR,\$B
- Der MONIN-Prozess schreibt Datensätze mit Adressbyte 'N', 'O' oder 'P' in \$6, \$9 oder \$B.
- Nur für \$6: Fehlermeldungen des TCL-Systems, wenn TR+6,1 auf '4' gesetzt ist.
- Nur für \$6: Letztes Sprungziel, wenn TR+3,1 auf '1' gesetzt ist.

### 6.1.7 Interpreter-Datenpuffer (\$7)

Datensätze vom Leitrechner, die mit dem Adressbyte 'J' beginnen, werden vom MONIN-Prozess in den Ringpuffer \$7 geschrieben. Die Daten können dann mit der TCL-Anweisung RD,\$7 ausgelesen und im TCL-Programm weiterverarbeitet werden.

### 6.1.8 Sendepuffer (\$D) und Empfangspuffer (\$C) für abgesetzte Eingabestationen am LBus

Für spezielle Funktionen werden über den Sendepuffer \$D und den Empfangspuffer \$C Escapesequenzen als Messages an die INTUS 1600 Terminals gesendet und von ihnen wiederum am INTUS 3000 Terminal empfangen. Die einzelnen Steuersequenzen sind im INTUS 1600 Programmierhandbuch nachzulesen.

Daten, die vom INTUS 3000 Terminal über den LBus, z.B. an die INTUS 1600 Terminals, gesendet werden, laufen über den Ringpuffer \$D. Die Datensätze sind als LBus-Message mit folgendem Format zu senden:

Bedeutung	Wert	Länge
Länge der Message	binärer Wert ohne die 4 Byte Header, Low-Byte zuerst	2 Byte
Group-Id	derzeit immer '@'	1 Byte
User-Id	<LBusadr> + '@', z. B. 'D' für Adresse 4	1 Byte
Message	z. B. Steuersequenzen	max. 252 Byte

*Tabelle 6.3 – Format der LBus-Message*

Alle Daten, die vom LBus empfangen werden, kommen in den Ringpuffer \$C. Die Datensätze haben den gleichen Aufbau wie für \$D beschrieben.



Achtung: Bis zur TCL Version 6.10 galt: wenn eine Antwort erwartet wird, dann darf keine Message an eine andere abgesetzte Eingabestation gesendet werden, weil nur dann die Antwort des zuletzt über \$D adressierten Gerätes an \$C geschickt wird.

## 6.2 MONIN-Prozess

Der MONIN-Prozess liest die vom Leitrechner kommenden Datensätze aus dem Empfangspuffer \$1 und reicht sie an den richtigen Adressaten weiter. Den Adressaten erkennt er am Adressbyte, dem jeweils ersten Byte jedes Datensatzes. Beginnt der Datensatz mit einem gültigen Adressbyte, dann leitet der MONIN-Prozess die Daten in den entsprechenden Ringpuffer oder in das entsprechende Feld weiter, wobei das Adressbyte selbst unterdrückt wird.

Je nach Konfiguration schreibt auch ein DNCIN-Prozess seine empfangenen Daten in den Empfangspuffer \$1.

### 6.2.1 Aufbau der Datensätze

Folgende Adressbytes sind möglich:

Adressbyte	Datensatztyp
'D'	TCL Programmzeile in den Downloadbereich
'E'	TCL Programmzeile ins EEPROM
'T'	Interpreteranweisung (\$3)
'J'	Datensatz an Interpreter Datenpuffer (\$7)
'N'	Datensatz an Kanal B (2. serielle Schnittstelle) (\$6)
'O'	Datensatz an Kanal C (3. Serielle Schnittstelle) (\$9)
'P'	Datensatz an Kanal D (4. serielle Schnittstelle) (\$B)
'Q'	positive Empfangsquittung für Sendesatz
'R'	Terminal-Reset durch den Leitrechner
'S'	negative Empfangsquittung für Sendesatz

*Tabelle 6.4 – Adressbytes für Datensätze an den MONIN*

**Adressbytes D, E, I, oder J**

Datensätze, die mit einem der Adressbytes D, E, I, oder J beginnen, müssen wie folgt aufgebaut sein:

**<Adressbyte><Daten><CR>**

Die maximale Satzlänge beträgt mit Adressbyte und CR-Zeichen 256 Byte.

- Adressbyte D: Der Datensatz soll in den Programmreich DL geladen werden.
- Adressbyte E: Der Datensatz soll in den EEPROM-Programmbereich geladen werden.
- Adressbyte I: Der Datensatz soll nach seiner Übersetzung in den Interpreter-Anweisungspuffer (\$3) kopiert werden, d.h. der Datensatz muss TCL-Anweisungen enthalten, die dann sofort vom Interpreter ausgeführt werden. Nicht ausführbare Anweisungen (Kommentar, Sprungziel) sind nicht zulässig und werden ignoriert.
- Adressbyte J: Der Datensatz soll in den Interpreter-Datenpuffer (\$7) kopiert werden. Die Daten können dann mit der Anweisung RD gelesen werden.

**Adressbytes N, O oder P**

- Adressbyte N: Der Datensatz soll in den Sendepuffer von Kanal B (\$6) kopiert, d.h. an der 2. seriellen Zusatzschnittstelle ausgegeben werden.
- Adressbyte O: Der Datensatz soll in den Sendepuffer von Kanal C (\$9) kopiert, d.h. an der 3. seriellen Zusatzschnittstelle ausgegeben werden.
- Adressbyte P: Der Datensatz soll in den Sendepuffer von Kanal D (\$B) kopiert, d.h. an der 4. seriellen Zusatzschnittstelle ausgegeben werden.

Die Datensätze können eines der folgenden beiden Formate haben:

**<Adressbyte><Daten><CR>**

oder

**<Adressbyte><Satzlänge><Daten><CR> .**

Das verwendete Format wird im MONIN-Steuerfeld MI (Abschnitt 4.23) eingetragen:

- MI+0,1: Datensatzformat für Kanal B, Sendepuffer \$6
- MI+2,1: Datensatzformat für Kanal C, Sendepuffer \$9
- MI+4,1: Datensatzformat für Kanal D, Sendepuffer \$B

Die maximale Satzlänge beträgt inkl. Adressbyte, Satzlänge und CR-Zeichen 256 Byte.

### **Adressbyte Q**

Der Datensatz ist eine positive Empfangsquittung. Er hat entweder das Format

**<Adressbyte><CR>**

oder, wenn im Setup (bzw. in CV+44,1) die Generierung einer logischen Satznummer durch den MONOUT-Prozess eingestellt wurde, das Format

**<Adressbyte><Satznummer><CR>**

Die logische Satznummer ist eine 4-stellige ASCII-Zahl mit dem Wertebereich '0001' bis '9999'.

Der Datensatz mit Adressbyte Q quittiert den einwandfreien Empfang des zuletzt aus dem Notpuffer (\$4) gesendeten Datensatzes bzw. des Datensatzes mit der logischen Satznummer <Satznummer> und aller vorangegangenen Datensätze. Die positiv quittierten Datensätze werden aus dem Notpuffer gelöscht.

### **Adressbyte S**

Der Datensatz ist eine negative Empfangsquittung. Er hat entweder das Format

**<Adressbyte><CR>**

oder, wenn im Setup (bzw. in CV+44,1) die Generierung einer logischen Satznummer durch den MONOUT-Prozess eingestellt wurde, das Format

**<Adressbyte><Satznummer><CR>**

Die logische Satznummer ist eine 4-stellige ASCII-Zahl mit dem Wertebereich '0001' bis '9999'.

Der Datensatz mit dem Adressbyte S besagt, dass der zuletzt aus dem Notpuffer (\$4) gesendete Datensatz bzw. der Datensatz mit der logischen Satznummer <Satznummer> nicht einwandfrei empfangen worden ist und noch einmal gesendet werden soll. Die Datensätze vor dem negativ quittierten Datensatz gelten als positiv quittiert und werden aus dem Notpuffer gelöscht.

Auch wenn die Generierung einer logischen Satznummer eingestellt ist, werden negative Quittungen ohne Satznummer akzeptiert (siehe auch Abschnitt 6.3.4).

### **Adressbyte R**

Der Datensatz hat das Format

**<Adressbyte><CR>**

und löst ein Terminal-Reset aus. Das Terminal läuft daraufhin mit dem im Setup (CV+8,1) eingestellten Anlaufmodus an. Siehe auch Abschnitt 8.4 *TCL-Programm starten*.

Nur mit diesem Terminal-Reset werden die geänderten Parameter des CV-Feldes auch eingestellt und im Terminal wirksam (Abschnitt 4.6 *CV Setup*).

Dieses Adressbyte ist nicht mit der Anweisung R (Reset) zu verwechseln!

## **6.2.2 Steuerung des MONIN-Prozesses**

Der MONIN-Prozess wird mit Hilfe des MI Feldes (Abschnitt 4.23) gesteuert. Der Programmierer hat folgende Möglichkeiten:

- Das Datensatzformat für Datensätze einstellen, die an Kanal B, C oder D über Sendepuffer \$6, \$9 oder \$B gesendet werden sollen.
- Die Übertragung der Datensätze in die Sendepuffer und den Empfang von Daten vom Host mit Timeout überwachen.
- Die Umleitung für Datensätze festlegen, falls der Leitrechner keine Adressbytes liefern kann.
- Die standardmäßige Ausgabe von Datensätzen mit ungültigem Adressbyte auf dem Display unterdrücken.
- Alle Datensätze, die in \$1 empfangen werden, ohne Interpretation auf das Display ausgeben. Dadurch können VT100-Escapesequenzen direkt durch den Displaytreiber verarbeitet werden.

## 6.3 MONOUT-Prozess

Der MONOUT-Prozess sorgt dafür, dass die vom Terminal erfassten Daten ohne Datenverlust den Leitrechner erreichen. Bei vielen im Terminal erfassten Daten, z.B. den Anwesenheitszeiten von Mitarbeitern, muss gewährleistet sein, dass kein Datenverlust auftritt, auch wenn das Stromversorgungsnetz ausfällt, der Leitrechner nicht betriebsbereit (offline) ist oder Übertragungsfehler auftreten.

### Das Datensicherungskonzept

- Mit der Anweisung SE oder WR,\$4 werden die Datensätze im Notpuffer \$4 gespeichert. Der Notpuffer ist ein batteriegepufferter Speicherbereich, der auch bei Netzausfall die Daten erhält.
- Die Größe des Notpuffers kann im Setup oder in CV+5,1 auf den erwarteten Bedarf an zu schützenden Daten eingestellt werden.
- Wenn der Notpuffer voll ist, zeigt es das Notpuffer-Statusflag P1,1 an. Das TCL-Programm kann darauf mit einem Ereignissprung reagieren. Das Sprungziel wird in P1+1,4 eingetragen.
- Damit die Daten im Notpuffer beim Anlauf nicht verlorengehen, muss im Setup als Anlaufmodus (CV+8,1) Warmstart eingestellt werden.

### 3 Betriebsarten des MONOUT-Prozesses

Der MONOUT-Prozess sendet die Daten an den Leitrechner, die mit der Anweisung SE oder WR,\$4 in den Notpuffer geschrieben wurden. Seine Arbeitsweise hängt von 2 Teifeldern ab:

P20+22,1	bestimmt, ob MONOUT die Datensätze aus dem Notpuffer vor dem Senden formatiert, oder ob er sie transparent sendet.
CV+44,1	bestimmt, ob MONOUT die Datensätze vor dem Senden mit einer logischen Satznummer versieht. (CV+44,1 kann über den Setup "TCL:Logische Satznummer"=J/N eingestellt werden.)

Daraus ergeben sich 3 Betriebsarten:

1. Transparentes Senden: P20+22,1 = '1' und CV+44,1 wird ignoriert.
2. Datensätze ohne logische Satznummer formatieren und senden, Quittungen ohne logische Satznummer empfangen: P20+22,1 = '0' und CV+44,1 = "00"
3. Datensätze mit logischer Satznummer formatieren und senden, Quittungen mit logischer Satznummer empfangen: P20+22,1 = '0' und CV+44,1 = "01"

Die Datensätze werden erst dann aus dem Notpuffer gelöscht, wenn sie vom Leitrechner positiv quittiert wurden. Die positiven und negativen Quittungen (Adressbyte 'Q' und 'S') empfängt der MONIN-Prozess und reicht sie dem MONOUT-Prozess weiter. Der Quittungsmechanismus ist für jede der 3 Betriebsarten verschieden.

### Betriebsstatus des Terminals

Das Betriebsstatusflag P3,1 gibt Auskunft über den Betriebszustand des Terminals. Auf einen Wechsel des Betriebszustands kann mit einem Ereignissprung reagiert werden, wenn in P3+1,4 ein Sprungziel eingetragen ist.

- **Automatisches Senden (online/offline):** Normalerweise sendet der MONOUT-Prozess die Datensätze aus dem Notpuffer sofort und automatisch, d.h. unabhängig vom Interpreter. Dieser Normalbetrieb wird mit P3,1='0' (online) angezeigt. Wenn ein gesendeter Datensatz nicht innerhalb der logischen Quittungszeit (einzustellen im Setup oder in CV+7,1) quittiert wurde, wird auf den Notbetrieb umgestellt. Der Betriebsstatus ist dann P3,1='1' (offline). Der erste nicht quittierte Datensatz wird im Abstand der logischen Quittungszeit solange wiederholt, bis er quittiert wird. Danach geht das Terminal wieder in den Normalbetrieb über.
- **Senden auf Anforderung:** In manchen Fällen, z.B. wenn das INTUS 3000 Terminal für die Kommunikation mit dem Host an ein Modem (über den TTY-Treiber) angeschlossen ist, ist es unerwünscht, dass die Datensätze automatisch gesendet und wiederholt werden, bis sie quittiert sind. Mit den drei Zuständen '2', '3' und '4' des Betriebsstatusflags P3,1 kann ein Senden auf Anforderung, das vom TCL-Programm gesteuert wird, durchgeführt werden.

Siehe auch Abschnitt 4.30 *P3 Betriebsstatusflag*.

Wenn das Prozedurstatusflag PO auf '1' (offline zum Host) gesetzt ist, sendet MONOUT ebenfalls nichts. Das PO Prozedurstatusflag zeigt an, ob auf Protokollebene eine Verbindung zum Leitrechner besteht. Bei den meisten Kommunikationsprotokollen, so z.B. BSC oder TCP/IP, lässt sich so bereits zu einem früheren Zeitpunkt als über einen Quittungstimeout feststellen, ob das Terminal offline ist. Siehe Abschnitt 4.35 *PO Prozedurstatusflag*.

#### 6.3.1 Aufbau des Sendedatensatzes

Der Feldaufbau und die Feldreihenfolge des Sendesatzes hängen zum einen von der Betriebsart des MONOUT-Prozesses zum anderen vom P10-Feld und der SE-Anweisung ab. Einige Felder sind optional.

Es ist bei der SE-Anweisung darauf zu achten, dass die maximale Satzlänge, inklusive aller vom MONOUT-Prozess hinzugefügten Bytes, 255 Bytes nicht überschreitet.

Die folgende Tabelle enthält alle möglichen Felder des Sendedatensatzes.

Teilfeld	Wert	Länge
<b>Logische Satznummer</b>	'0001'-9999'	4 Byte
<b>Statusflag</b>	[' online, aktuell '] offline, gepuffert '*' nur relevant, wenn Verbindung zum Host besteht	1 Byte
<b>Routingbytes</b>	ASCII-Zeichen	0-8 Byte
<b>Daten</b>		max.255 Byte
<b>Checksumme</b>	2-stellige ASCII-Zahl	2 Byte
<b>Datensatz-Endezeichen</b>	ASCII-Zeichen ungleich CR-Zeichen ("0D")	1 Byte
	CR-Zeichen ("0D")	1 Byte

*Tabelle 6.5 – Sendedatensatz mit optionalen Bestandteilen*

### **6.3.1.1 Logische Satznummer**

Mit Hilfe der logischen Satznummer kann der Leitrechner eine Satzverdoppelung oder einen Satzverlust erkennen. Die logische Satznummer ist optional und wird vom MONOUT-Prozess in Betriebsart 3 hinzugefügt.

Die logische Satznummer ist eine vierstellige Zahl mit Werten von '0001' bis '9999'. Beim Kaltstart des Terminals wird der Wert der logischen Satznummer auf '0001' gesetzt und bei jedem neuen gesendeten Satz inkrementiert. Beim Warmstart des Terminals bleibt der Wert der logischen Satznummer unverändert.

#### **Synchronisation der logischen Satznummer**

Wenn der Leitrechner die TCL-Anweisung `R,L{,<Zahlenwert>}` sendet, synchronisiert er seine logische Satznummer mit der im Terminal. Ein sinnvoller Zeitpunkt dafür ist, wenn der Notpuffer leer ist und keine weiteren Datensätze vom Leitrechner zu quittieren sind.

### **6.3.1.2 Statusflag**

#### **Statusflag '[': online**

Mit dem Statusflag '[' wird dem Leitrechner mitgeteilt, dass der empfangene Datensatz aktuell (online) ist.

Das Statusflag '[' wird vom MONOUT-Prozess automatisch an den Anfang des Datensatzes, bzw. hinter die logische Satznummer eingetragen, wenn in der SE-Anweisung keine Routingbytes und kein Statusflag eingetragen ist.

#### **Statusflag ']': offline**

MONOUT ändert das Statusflag '[' (online) in das Statusflag ']' im Datensatz, wenn der Betriebsstatus des Terminals offline ist ( $P3,1=1$ ). Damit teilt MONOUT dem Leitrechner mit, dass der empfangene Datensatz zwischengepuffert werden musste und bei der Ankunft im Host nicht mehr aktuell (sondern offline) ist. Das kann bei manchen Anwendungen von Interesse sein. Zum Beispiel ist es nicht sinnvoll, auf eine gepufferte Zeitsaldoanfrage einen Antwortsatz zu senden.

#### **Statusflag '\*'**

Datensätze, die nur relevant sind, wenn eine Verbindung zum Host besteht, können mit dem Statusflag '\*' gekennzeichnet werden. Wenn der Betriebsstatus des Terminals offline ( $P3,1=1$ ) ist, werden diese Sätze aus dem Notpuffer gelöscht.

Der MONOUT-Prozess verarbeitet '\*' als Statusflag nur, wenn  $P10+11,1=1$  gesetzt ist. Bei der Voreinstellung ( $P10+11,1=0$ ) wird '\*' als ein normales Zeichen behandelt.

Bis zur TCL Version 2.6 mussten Datensätze, die nur beim Betriebsstatus online des Terminals ( $P3,1=0$ ) relevant sind, entweder mit der Anweisung SR über den Ringpuffer \$2 (und damit sofort und ohne Datensicherungskonzept) gesendet werden, oder gegebenenfalls erst später im Host verworfen werden, falls sie nicht mehr aktuell waren.

Auch das Statusflag '\*' muss in der SE-Anweisung an der korrekten Position hinter den Routingbytes ( $P10+8,1$ ) stehen.

### 6.3.1.3 Routingbytes

Datensätze können durch Einfügen von Routingbytes für die Weiterverarbeitung im Leitrechner gekennzeichnet werden.

Routingbytes sind optional und müssen in der SE-Anweisung oder auch in der WR,\$4-Anweisung vom TCL-Programmierer angegeben werden. Wenn Routingbytes den Daten vorangestellt werden, dann muss auch das Statusflag '[' oder '\*' vom TCL-Programmierer im Datensatz eingetragen werden.

Sie stehen im Sendedatensatz als erstes Feld oder nach der logischen Satznummer als zweites Feld.

Die Anzahl der Routingbytes kann zwischen 0 und maximal 8 Bytes betragen und muss in P10+8,1 eingetragen werden (Voreinstellung: '0'). Diese Länge ist in allen Datensätzen, in denen Routinginformation eingefügt wird, zu verwenden.

### 6.3.1.4 Checksumme

Der MONOUT-Prozess erzeugt in den Betriebsarten 2 und 3 über jeden Datensatz eine Checksumme. Sie wird nach den Daten und vor dem Datensatz-Endezeichen vom MONOUT-Prozess eingefügt und besteht aus 2 Bytes.

Die Checksumme schließt, wenn vorhanden, die logische Satznummer, die Routingbytes und das Statusflag mit ein.

#### Bildungsalgorithmus

Es werden alle Bytes eines Datensatzes modulo 256 aufsummiert. Die ASCII-Darstellung der beiden Ziffern der entstehenden Hexzahl wird als Checksumme verwendet. Die Hexziffern A,..,F werden als Großbuchstaben übertragen.

#### Beispiel

DSE, 'RB[abc':

erzeugt, wenn MONOUT in der Betriebsart 2 arbeitet, den Sendedatensatz:

RB	[	abc	15	CR Zeichen
hexadezimal:				
"52"	"42"	"5B"	"61" "62" "63"	"31" "35" "0D"

Die 2-stellige ASCII-Zahl '15' ist die Checksumme.

Checksummenberechnung:

Addiert man die Bytes: "52" + "42" + "5B" + "61" + "62" + "63" erhält man "215". Nach einer Modulo 256 Operation erhält man die beiden hexadezimalen Ziffern "15", die danach in die druckbaren Zeichen '15' umgewandelt werden.

### 6.3.1.5 Satzendezeichen

Der MONOUT-Prozess hängt als letztes Zeichen an jeden Datensatz ein CR.

Wenn das MONOUT-Steuerfeld P10+14,1 ein Zeichen ungleich dem CR-Zeichen enthält, wird als letztes Zeichen des Datenübertragungsblocks, der abhängig von P10+12,2 einen oder mehrere Datensätze enthält, vom MONOUT ein CR-Zeichen angefügt.

### **6.3.1.6 Leersatz**

Ein Leersatz ist ein normaler Sendedatensatz und wird optional bei der Betriebsart 2 und 3 automatisch durch den MONOUT-Prozess gesendet (Voreinstellung, P10+10,1='1'). Das Senden des Leersatzes kann durch P10+10,1='0' unterdrückt werden.

Der Leersatz ist aufgebaut wie ein normaler Datensatz enthält aber keine Daten. Er hat folgende Funktion:

Wenn das Terminal im Betriebszustand Offline war und nun wieder eine Verbindung zum Leitrechner besteht, werden alle gepufferten Datensätze nacheinander aus dem Notpuffer an den Leitrechner gesendet. Wenn alle Datensätze vom Host quittiert sind, sendet der MONOUT-Prozess einen Leersatz als Zeichen dafür, dass keine weiteren gepufferten Datensätze mehr im Notpuffer sind.

#### **Routingbytes im Leersatz**

Wenn der letzte normale Datensatz mit Routinginformation versehen war, wird auch der Leersatz mit der entsprechenden Anzahl an Routingbytes gesendet.

In P10+0,8 wird die Routinginformation des Leersatzes linksbündig eingetragen. Die Voreinstellung ist '00000000'. P10+8,1 enthält die zu verwendende Anzahl Bytes der Routinginformation in P10,8. Der Wert darf zwischen '0' und '8' liegen, die Voreinstellung ist '2'. Das Teilfeld, das die Routingbytes enthält, kann mit P10,(P10+8,1) ausgelesen werden.

#### **Quittierung des Leersatzes**

Der Leersatz muss wie alle anderen Datensätze quittiert werden. Wird er nicht innerhalb der logischen Quittungszeit quittiert, wird er ebenso wie die normalen Datensätze mit der Kennung ']' versehen und im Abstand der logischen Quittungszeit wiederholt gesendet, bis er quittiert ist. Damit ergibt sich ein Offline-Leersatz, dem dann ein Online-Leersatz folgt.

#### **Hinweis**

Wenn das Terminal im Offline-Betrieb ist, und der Notpuffer mit der Anweisung "RD,\$4" ausgelesen wird, dann wird der Leersatz trotzdem an den Rechner gesendet, er kann nicht vom Interpreter gelöscht werden.

### **6.3.2 Betriebsart 1 (transparentes Senden)**

Die einfachste Betriebsart des MONOUT-Prozesses ist das transparente Senden. Dazu muss P20+22,1 auf '1' (keine Verarbeitung durch MONOUT) gesetzt werden.

In dieser Betriebsart werden eventuell enthaltene Routingbytes im Datensatz vom MONOUT-Prozess nicht verarbeitet. Es wird auch kein Leersatz gesendet, d.h. das Konfigurationsfeld CV+44,1 für die logische Satznummer und alle Teilfelder des MONOUT-Steuerfeldes P10 bis auf P10+14,1 werden vom MONOUT-Prozess ignoriert.

#### **Senden**

Der Datensatzaufbau wird vollständig vom TCL-Programmierer festgelegt. So wie er die Datensätze mit der Anweisung SE oder WR,\$4 in den Notpuffer schreibt, werden sie an den Host gesendet.

An diese Datensätze in der SE-Anweisung wird durch den MONOUT-Prozess lediglich das Datensatz-Endezeichen aus P10+14,1 angehängt.

#### **Datensatzaufbau**

**<Daten aus SE-Anweisung><Satzendezeichen>**

Der Datensatz enthält also weder log. Satznummer, Statusflag und Checksumme.

#### **Datensatzquittierung**

Für jeden Datensatz, der aus dem Notpuffer gesendet wird, muss der Leitrechner den korrekten Empfang bestätigen.

#### **Positive Quittung**

**Q<CR>**

#### **Negative Quittung**

**S<CR>**

Wenn der Datensatz positiv quittiert ist, wird er aus dem Notpuffer gelöscht und der nächste Datensatz gesendet. Erhält MONOUT eine negative Satzquittung, wird der letzte Datensatz sofort wiederholt.

Wenn ein gesendeter Datensatz nicht innerhalb der logischen Quittungszeit, die im Setup oder in CV+7,1 eingestellt wird, quittiert ist, geht das Terminal in den Betriebsstatus offline (P3,1='1') über, da vermutlich keine Verbindung mehr zum Leitrechner besteht (siehe auch Abschnitt 4.30 *P3 Betriebsstatusflag*).

#### **Beispiel**

**DSE, 'Hallo':**

erzeugt den Sendedatensatz:

Haloo

### 6.3.3 Betriebsart 2 (Voreinstellung)

In dieser Betriebsart werden die Datensätze aus der SE-Anweisung durch den MONOUT-Prozess durch Statusflag und Checksumme ergänzt (Voreinstellung). In dieser Betriebsart ist auch das Senden von mehreren Datensätzen in Blöcken möglich.

Ein Leersatz kann gesendet oder unterdrückt werden.

Um diese Betriebsart einzustellen, müssen  $P20+22,1 = '0'$  (Verarbeitung durch MONOUT) und  $CV+44,1 = "00"$  (Datensatz ohne logische Satznummer) gesetzt werden. In dieser Betriebsart werden alle Teifelder des MONOUT-Steuerfeldes P10 ausgewertet.

#### Sendedatensatz

Die Sendedatensätze unterscheiden sich in Betriebsart 2 von Betriebsart 3 nur durch das fehlende Feld "Log. Satznummer".

#### Datensatzaufbau (Voreinstellung):

[<RoutingBytes>]<Statusflag><Daten aus SE-Anweisung><Checksumme> <Satzendezeichen>

Die Datensätze werden einzeln gesendet, wenn  $P10+12,2='01'$  eingestellt ist (Voreinstellung).

Es ist auch möglich, mehrere Datensätze in Blöcken zu senden und zu quittieren. Dazu wird die Anzahl der Datensätzen, die gemeinsam gesendet und quittiert werden sollen, in  $P10+12,2$  festgelegt ('01' bis '25'). Das Trennzeichen zwischen den Datensätzen im Datenübertragungsblock wird in  $P10+14,1$  eingestellt. Das CR-Zeichen ist die Voreinstellung. Wird ein anderes Zeichen als Trennzeichen definiert, dann fügt der MONOUT-Prozess am Ende des Datenübertragungsblocks ein CR-Zeichen an.

#### Datensatzquittierung

Für jeden Datensatz/Datenblock, der aus dem Notpuffer gesendet wird, muss der Leitrechner den korrekten Empfang bestätigen.

#### Positive Quittung

Q<CR>

#### Negative Quittung

S<CR>

Beim Empfang einer negativen Quittung wird der letzte Datensatz/Datenblock wiederholt. Beim Empfang einer positiven Quittung wird der nächste Datensatz/Datenblock quittiert.

#### Beispiele

##### Beispiel 1:

DSE, 'Hallo':

erzeugt, wenn P10 die Voreinstellungen enthält, den Sendedatensatz:

[	Haloo	4B
---	-------	----

MONOUT hat das Statusflag '[' am Anfang des Datensatzes eingefügt, da er hinter der Anzahl Routingbytes ( $P10+8,1='2'$ ) kein '[' gefunden hat.

Ohne Quittierung durch den Host folgt nach der logischen Quittungszeit:

]	Haloo	4D
---	-------	----

nach dem Empfang der positiven Quittung schickt MONOUT den Leersatz, denn der Notpuffer ist leer. Der Leersatz enthält keine Routingbytes, da der letzte Datensatz auch keine Routingbytes hatte.

[	5B
---	----

**Beispiel 2:**

DSE, '00[Datensatz':

erzeugt, wenn P10 die Voreinstellungen enthält, den Sendedatensatz:

00	[	Datensatz	69
----	---	-----------	----

ohne Quittierung durch den Host folgt nach der logischen Quittungszeit:

00	]	Datensatz	6B
----	---	-----------	----

nach dem Empfang der positiven Quittung schickt MONOUT den Leersatz, denn der Notpuffer ist leer. Die Routingbytes werden aus P10,8 und die zugehörige Länge aus P10+8,1 entnommen.

00	[	BB
----	---	----

**Beispiel 3:**

DSE, '[[[soso':

erzeugt, wenn P10 die Voreinstellungen enthält, den Sendedatensatz:

[[	[	soso	D5
----	---	------	----

MONOUT hat das 3. '[' Zeichen als Statusflag erkannt, denn es steht hinter der Anzahl Routingbytes (P10+8,1='2').

Ohne Quittierung durch den Host folgt nach der logischen Quittungszeit:

[[	]	soso	D7
----	---	------	----

nach dem Empfang der positiven Quittung schickt MONOUT den Leersatz, denn der Notpuffer ist leer. Die Routingbytes werden aus P10,8 und die zugehörige Länge aus P10+8,1 entnommen.

00	[	BB
----	---	----

**Beispiel 4:**

DSE, '[Hallo':

erzeugt, wenn P10 die Voreinstellungen enthält, den Sendedatensatz:

[	Haloo	A6
---	-------	----

MONOUT hat das Statusflag '[' am Anfang des Datensatzes eingefügt, da er hinter der Anzahl Routingbytes (P10+8,1='2') kein '[' gefunden hat.

Ohne Quittierung durch den Host folgt nach der logischen Quittungszeit:

]	Haloo	A8
---	-------	----

nach dem Empfang der positiven Quittung schickt MONOUT den Leersatz, denn der Notpuffer ist leer. Der Leersatz enthält keine Routingbytes, da der letzte Datensatz auch keine Routingbytes hatte.

[	5B
---	----

ohne Quittierung durch den Host folgt nach der logischen Quittungszeit:

]	5D
---	----

### **6.3.4 Betriebsart 3**

In dieser Betriebsart werden die Datensätze aus der SE-Anweisung durch den MONOUT-Prozess durch Logische Satznummer, Statusflag und Checksumme ergänzt. In dieser Betriebsart ist auch das Senden von mehreren Datensätzen in Blöcken möglich.

Ein Leersatz kann gesendet oder unterdrückt werden.

Um diese Betriebsart einzustellen, müssen P20+22,1 = '0' (Verarbeitung durch MONOUT) und CV+44,1="01" (Datensatz mit logische Satznummer) gesetzt werden. In dieser Betriebsart werden alle Teifelder des MONOUT-Steuerfeldes P10 ausgewertet.

#### **Sendedatensatz**

Die Sendedatensätze unterscheiden sich Betriebsart 3 von Betriebsart 2 nur durch das zusätzliche Feld "Log. Satznummer".

#### **Datensatzaufbau (Voreinstellung)**

**<Log. Satznummer>[<RoutingBytes>]<Statusflag><Daten aus SE-Anweisung><Checksumme> <Satzendezeichen>**

Die Position der logischen Satznummer kann in P10+9,1 (von 0-9) festgelegt werden. Voreinstellung ist '0', d.h. die Logische Satznummer steht am Satzanfang.

Ist die Position in P10+9,1 gleich der Länge der Routinginformation (P10+8,1), dann steht die 4-stellige Satznummer hinter der Routinginformation.

Es wird sogar eine Position, die größer als die Länge der Routinginformation (P10+8,1), aber kleiner oder gleich '9' ist, akzeptiert. Der Datensatz wird an der entsprechenden Position für die 4-stellige Satznummer aufgetrennt.

#### **Datensatzquittierung**

Die Datensatzquittungen haben bei Betriebsart 3 ein anderes Format, da im Sendedatensatz logische Satznummern vom MONOUT-Prozess eingefügt werden.

##### **Positive Quittung:**

**Q<Satznummer><CR>**

##### **Negative Quittung:**

**S<Satznummer><CR>**

oder auch nur

**S<CR>**

Enthalten die zu übertragenden Datensätze eine logische Satznummer, dann kann jeder Datensatz einzeln oder nur der letzte Datensatz im Block durch Q<Satznummer><CR> quittiert werden. Dies führt zu einer Beschleunigung des Datentransfers, da asynchron Quittungen empfangen werden können und seltener auf Quittungen vom Leitrechner gewartet werden muss.

Wenn mit der falschen Satznummer quittiert wird, dann wird der letzte Datensatz wiederholt.

Wurde ein Datensatz im Block nicht einwandfrei empfangen und dieser mit S<Satznummer><CR> negativ quittiert, dann wird der Block ab diesem Datensatz wiederholt. Die Datensätze bis zu diesem negativ quittierten Datensatz sind damit positiv quittiert.

Wird eine negative Quittung ohne Satznummer empfangen, wird der ganze Block wiederholt.

#### **Beispiele**

##### **Beispiel 1:**

DSE, 'RO[Hallo]':

erzeugt, wenn P10 neben den Voreinstellungen im Teifeld "Position der logischen Satznummer" P10+9,1='2' enthält, den Sendedatensatz:

RO	0005	[	Hallo	B1
----	------	---	-------	----

ohne Quittierung durch den Host folgt nach der logischen Quittungszeit:

RO	0005	]	Hallo	B3
----	------	---	-------	----

nach dem Empfang der positiven Quittung schickt MONOUT den Leersatz, denn der Notpuffer ist leer. Die Routingbytes werden aus P10,8 und die zugehörige Länge aus P10+8,1 entnommen.

00	0006	[	81	
----	------	---	----	--

### **Beispiel 2:**

DSE, 'Hallo':

erzeugt, wenn P10 neben den Voreinstellungen im Teifeld "Position der logischen Satznummer" P10+9,1='2' enthält, den Sendedatensatz:

[	H	0007	allo	12
---	---	------	------	----

MONOUT hat das Statusflag '[' am Anfang des Datensatzes eingefügt, da er hinter der Anzahl Routingbytes (P10+8,1='2') kein '[' gefunden hat. Die logische Satznummer fügt er anschließend gemäß P10+9,1 nach der 2. Position ein.

Ohne Quittierung durch den Host folgt nach der logischen Quittungszeit:

]	H	0007	allo	14
---	---	------	------	----

nach dem Empfang der positiven Quittung schickt MONOUT den Leersatz, denn der Notpuffer ist leer. Die Positionierung der logischen Satznummer führt dazu, dass der Leersatz an der 2. Position ein Leerzeichen enthält.

[		0008	43	
---	--	------	----	--

## 6.4 Sicherheitskonzept ab TCL Version 5.5

### 6.4.1 Login auf der Hostschnittstelle

Ein Login auf der Hostschnittstelle verhindert unberechtigte Datenverbindungen zum Terminal. Diese Option ist aus Kompatibilitätsgründen standardmäßig deaktiviert.

Die Konfiguration dieses Logins findet im CV-Feld im Teilbereich CV+132,26 statt (siehe Abschnitt 4.6).

Das Terminal sendet nach einem erfolgreichen Verbindungsaufbau eine Passwort-Abfrage. Die Applikation auf dem Host-System muss sich am Terminal anmelden. Eine erfolgreiche Anmeldung wird dem Host mitgeteilt.

Die TCL-Firmware unterscheidet zwei Berechtigungsstufen: „einfacher Zugriff“ und „administrativer Zugriff“.

Beim „einfachen Zugriff“ kann das Terminal bereits Daten versenden (aus Notpuffer bzw. mittels SR-, WR- oder WO-Anweisung, verarbeitet jedoch nur Interpreterdatensätze (‘J’) sowie Quittungssätze. (‘Q’, ‘S’)). Nur beim „administrativen Zugriff“ werden alle anderen Datensätze akzeptiert, insbesondere der Download des TCL Programms. Werden Daten an das Terminal geschickt, die aufgrund der aktuellen Berechtigung nicht verarbeitet werden können, so wird dies dem Host-System mitgeteilt.

Ein Abmelden auf der Hostschnittstelle erfolgt automatisch beim Trennen der Hostverbindung, soweit dies erkannt werden kann. Es gibt jedoch auch die Möglichkeit zu jedem Zeitpunkt einen mit ‚L‘ beginnenden Satz (z.B. “Logout“) an das Terminal zu senden um eine Abmeldung zu erreichen. Dies kann insbesondere dann notwendig sein, wenn die Berechtigungsstufe geändert werden soll.

Werden aufeinanderfolgend 5 falsche Passwörter an das Terminal geschickt, so wird der Login für 5 Minuten gesperrt. Datensätze, die in dieser Zeit eintreffen, werden als Anmeldeversuch interpretiert und entsprechend zurückgewiesen, dabei wird auch die Sperrzeit verlängert!

Im PO-Feld (siehe Abschnitt 4.35) kann abgefragt werden, ob ein Login bereits erfolgt ist.

Das Terminal schickt folgende Meldungen, die vom Host-System erkannt werden müssen:

- Anmeldung notwendig: “PWD:\r”
- Anmeldung einfacher Zugriff erfolgreich: “AUTH OK [data]!\r”
- Anmeldung administrativer Zugriff erfolgreich: “AUTH OK [admin]!\r”
- Datensatz verworfen: “DROPPED!\r”
- Login vorübergehend gesperrt: “LOCKED!\r”

Die Meldungen des Terminals werden bei Aktivierung (CV+132,1) mit dem Wert ‚1‘ im jeweiligen MONOUT-Format gesendet. Dies kann auch transparent sein wie oben beschrieben. Wird die Meldung im Format des MONOUT-Prozesses gesendet (siehe Abschnitt 4.31 *P10 MONOUT-Steuerfeld, Routinginformation*) und ist dabei eine Satznummer konfiguriert, so wird das festgelegte Ersetzungszeichen anstelle der Satznummer gesendet. Das Statusflag ist immer ‚!‘.

Wird der Login (CV+132,1) mit dem Wert ‚2‘ aktiviert, so werden die Login-Meldungen immer transparent gesendet.

### **6.4.2 Verschlüsselung auf der Hostschnittstelle**

Das Abhören der Kommunikation auf der Hostschnittstelle zwischen Host-System und Terminal kann durch eine Verschlüsselung der übertragenen Daten verhindert werden. So wird das Protokollieren des Stammsatz-Downloads oder des Upload von Buchungsdaten verhindert. Diese Option ist aus Kompatibilitätsgründen standardmäßig deaktiviert (siehe CV+159,1).

Zur Verschlüsselung wird der vom NIST (National Institute of Standards and Technology) als AES (Advanced Encryption Standard) ausgewählte Rijndael-Algorithmus eingesetzt.

Da es sich um einen Block-Algorithmus handelt wird er im 128bit OFB-Modus (OFB=Output-Feedback-Mode) betrieben, um einen Datenstrom beliebiger Länge verschlüsseln zu können. Kommt es bei der Übertragung zu einem Zeichenverlust, so können die nachfolgenden Daten nicht mehr entschlüsselt werden! Aus diesem Grund wird bei einem Verbindungsauftakt der Algorithmus neu initialisiert.

Das Terminal benötigt einen sicheren Schlüssel! Diese werden üblicherweise mittels MD5-Algorithmus aus einer sogenannten Passphrase erzeugt (Eine Passphrase ist im Wesentlichen nichts anderes als ein sehr langes Passwort).

Der Algorithmus erzeugt 16 Bytes, die dann als Schlüssel verwendet werden können. Für eine sichere Passphrase werden mindestens 20 Zeichen (wirklich zufällig gewählte Zeichen aus dem gesamten darstellbaren ASCII-Vorrat) benötigt. Einfache Sätze als Passphrase sollten mindestens 107 Zeichen haben. (Siehe „The passphrase FAQ“, <http://www.stack.nl/~gallactus/remailers/passphrase-faq.html>).



**Die Eingabe einer Passphrase sowie die Erzeugung des MD5-Hash-Wertes wird von INTUS RemoteSetup V2 übernommen, um die Sicherheit der Verschlüsselung zu gewährleisten.**

Mehr Informationen auf

<http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf>

<http://csrc.nist.gov/CryptoToolkit/aes/rijndael/>

<http://www.iaik.tu-graz.ac.at/research/krypto/AES/>

<http://www.iaik.tu-graz.ac.at/research/krypto/AES/old/%7Erijmen/rijndael/>

### **6.4.3 Verschlüsselung des LBus**

Das Abhören der Kommunikation auf dem LBus zwischen Terminal und abgesetzten Lesern kann durch eine Verschlüsselung der übertragenen Daten verhindert werden. Die Verschlüsselung kann für einzelne Leser am LBus gestartet werden.

Folgende externe Leser können mit der Verschlüsselung auf dem LBus umgehen:

- INTUS 1600-II
- INTUS 400, INTUS 500, INTUS 600 und INTUS 600FP
- INTUS 350H

Der Verschlüsselungsalgorithmus und die Schlüsselgenerierung wird wie bei der Verschlüsselung der Hostschnittstelle gehandhabt (siehe Abschnitt 6.4.2).

Diese Option ist aus Kompatibilitätsgründen standardmäßig deaktiviert.

#### **6.4.4 Setup mit Berechtigungsstufen**

Das Setup erhält 3 Berechtigungsstufen. Abhängig vom eingegebenen Passwort wird die jeweilige Ebene aktiviert. Haben zwei Ebenen das gleiche Passwort, so wird die Ebene mit der höheren Berechtigungsstufe gewählt. Das bisherige Passwort (CV+70,6) mit 6 Stellen wird für Ebene 1 verwendet, die beiden neuen Passworte (CV+116,8 und CV+124,8) haben eine Länge von 8 Stellen.

- **Berechtigungsstufe 1:** der Haustechniker kann das Kommunikationsprotokoll für die Verbindung zum Host konfigurieren und die TCP/IP-Parameter einstellen.
- **Berechtigungsstufe 2:** wie Ebene 1, zusätzlich kann der Betreuer/Partner LBus1 und LBus2 mit den externen Lesern einrichten, die TCL-Parameter und die Sommer-/ Winterzeitumschaltung konfigurieren und das Terminal einer Wartungsgruppe zuordnen. Ein Login auf der Hostschnittstelle kann eingerichtet werden.
- **Berechtigungsstufe 3:** wie Ebene 2, zusätzlich kann der Systemverwalter die Kommunikation auf der Hostschnittstelle und auf dem LBus verschlüsseln.

Im P21-Feld (siehe Abschnitt 4.33) wird konfiguriert, ob und mit welchen Berechtigungsstufen das Setup gestartet werden darf. Es kann abgefragt werden, ob und in welcher Berechtigungsstufe das Setup aktiv ist. Damit ist es z.B. möglich, den Zugriff auf Berechtigungsstufe 3 des Setups am Terminal zu verhindern, da die relevanten Parameter ausschließlich durch die Host-Applikation konfiguriert werden.

## 6.5 Taktüberwachung (Maschinentask)

Mit TCL Version 5.02 wird die aus der INTUS 2000 Serie bekannte Taktüberwachung für die digitalen Eingänge (dort auch Maschinentask genannt) in die INTUS 3000 Serie übernommen. Die Taktüberwachung ist eine optionale Softwarefunktion, die bei der Bestellung eines Terminals zusätzlich erworben werden muss.

Damit haben die Felder und Teifelder Ex+4,2, Tx, TAx und TOx, die schon immer in den auf dem INTUS 3000 verfügbaren TCL Versionen vorhanden waren, die gleiche Funktion wie im INTUS 2000. Eine eventuell abweichende Funktionalität wird im Abschnitt 6.5.1 weiter unten beschrieben.

Im INTUS 3000 existieren für jeden digitalen Eingang, egal ob lokal oder auf einem abgesetzten Leser, ein

- Ex- (genauer E[<LBusadr>]x-) Feld für den Status und die Kontrolle eines digitalen Eingangs und ein
- Zx- (genauer Z[<LBusadr>]x-) Feld, das am digitalen Eingang vorkommende Flanken zählt.

Für die ersten 14 (0, ..., 13) lokalen, digitalen Eingänge gibt es zusätzlich die Felder

- Tx (T0 ... T13),
- TAx (TA0 ... TA13) und
- TOx, (TO0 ... TO13),

mit deren Hilfe eine Taktüberwachung realisiert werden kann. Die Taktüberwachung wird ebenso wie die Eingänge und die Zähler erst aktiv, wenn das P2-Flag auf '1' gesetzt wird.

Im INTUS 3300/3500 sind derzeit standardmäßig die lokalen, digitalen Eingänge E0 und E1 (oder E[0]0 und E[0]1) vorhanden. Dazu kommt der lokale Vandalenkontakt E4 (oder E[0]4). Mit einem optionalen 4-DI Steckmodul auf dem Steckplatz der ersten zusätzlichen seriellen Schnittstelle, Kanal B, sind weitere vier digitale Eingänge, E5 bis E8 verfügbar. Ein optionales 4-DI Steckmodul auf dem Steckplatz der zweiten seriellen Schnittstelle, Kanal C, stellt die vier digitalen Eingänge E9 bis E12 zur Verfügung. Für alle hier genannten Eingänge kann die Taktüberwachung konfiguriert werden.

In den Abschnitten, die die Felder

- Ex (Abschnitt 4.10)
- Tx (Abschnitt 4.42)
- TAx (Abschnitt 4.43)
- TOx (Abschnitt 4.46)
- P2 (Abschnitt 4.29)

beschreiben, finden sich detaillierte Hinweise, wie diese Felder für eine Taktüberwachung zu initialisieren sind. Die untenstehende Skizze verdeutlicht das Zusammenspiel der verschiedenen über die Felder Tx und TAx einstellbaren Teilfunktionen der Taktüberwachung.

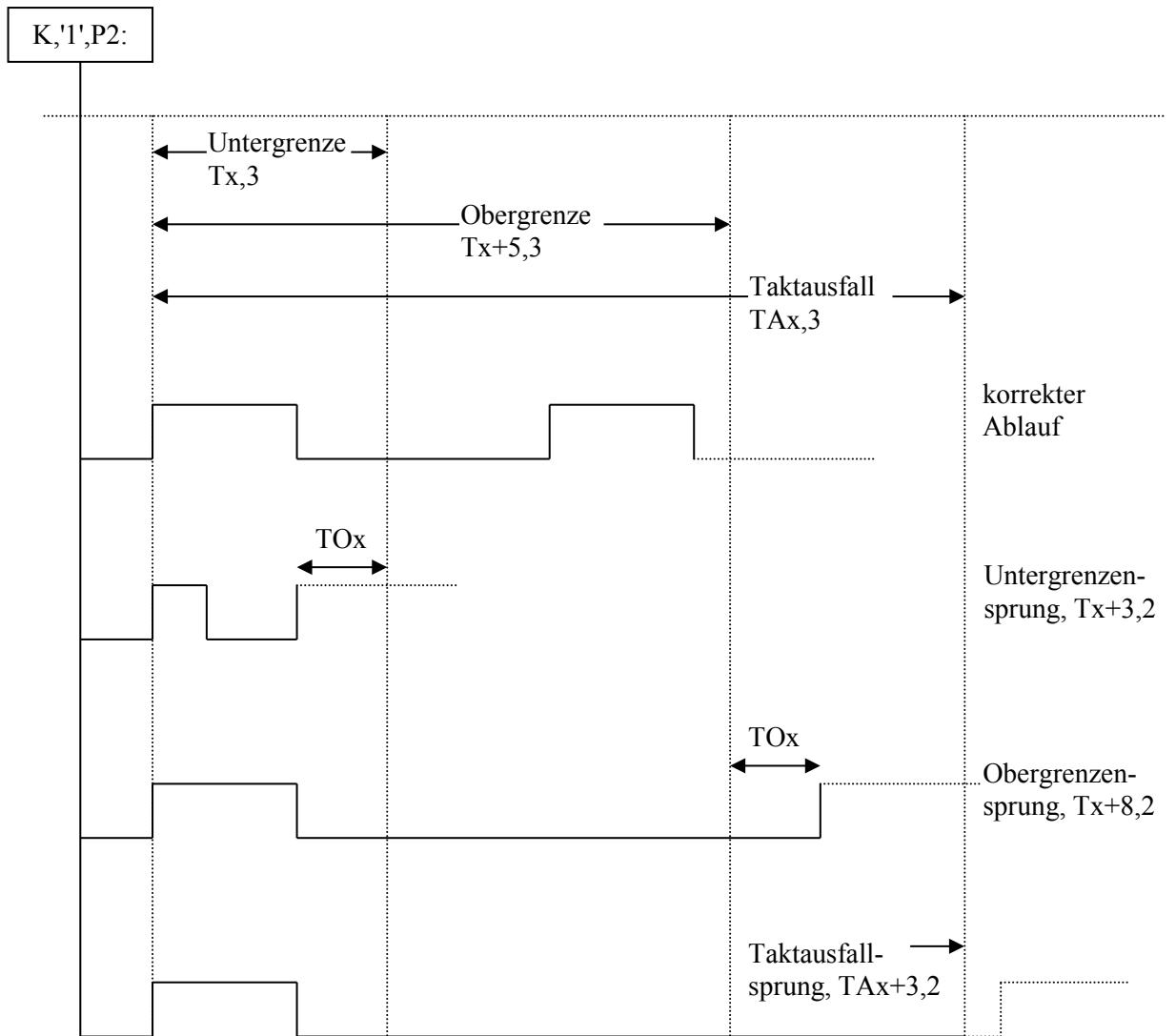


Abbildung 6.1 - Übersicht über die Funktionen der Taktüberwachung

Alle ausgelösten Sprünge der Taktüberwachung werden wie die DI-Sprünge in den \$3 Interpreteranweisungspuffer geschrieben. Wenn das (Kompatibilitäts-) Feld P20+19,1 ungleich '0' ist, werden die Sprünge aus dem \$3-Puffer gelöscht, wenn in der Zwischenzeit entweder das P2-Flag auf '0' oder das Teilstfeld Ex+1,1 auf '2' gesetzt wurde.

Die Startbedingung der Taktausfallüberwachung (TAx) wird mit Hilfe des Teilstells Ex+5,1 festgelegt.

Es wird empfohlen, die Felder Tx und TAx bei laufender Taktüberwachung nicht mehr zu ändern. Beim INTUS 3000 werden die Änderungen sofort übernommen und führen zum Neustart der Taktüberwachung; die Reaktion des INTUS 2000 ist undefined.

### 6.5.1 Kompatibilität mit der INTUS 2000 Serie

- Ex+5,1: kann beim INTUS 2000 nur mit '0' beschrieben werden. Es gibt keinen verzögerten Start der Taktausfallüberwachung TAx mit Hilfe des Werts '2'.
- P20+20,2 und P20+24,2: beim INTUS 2000 wird nur P20+20,2 mit der Nummer des DIs belegt, P20+20,2 bleibt unberührt. Dies sollte jedoch keine Rolle spielen, da in P20+24,2 immer nur '00' zu erwarten ist.
- Beim INTUS 2000 muss TAx vor Tx bzw. vor der Aktivierung von P2 beschrieben werden, beim INTUS 3000 sind die Felder unabhängig.
- Eine Änderung der Werte einer laufenden Taktüberwachung ist nur beim INTUS 3000 definiert; dabei wird die Taktüberwachung mit den neuen Werten sofort wieder gestartet. Beim INTUS 2000 kann das zu undefinierten Resultaten führen.
- Die Taktüberwachung ist beim INTUS 3000 aufgrund eines internen Systemtakts mit hoher Auflösung präziser. Es kann eine Genauigkeit von 3ms erwartet werden. Damit kann es passieren, dass die INTUS 3000 Taktüberwachung Taktfehler meldet, die aufgrund der geringeren Auflösung (100ms) im INTUS 2000 nicht gemeldet werden.
- E4: Der zum E4 (bzw. E[0]4) Feld gehörende DI ist beim INTUS 3000 der Vandalenkontakt. Daher sind die Felder T4, TA4 und TO4 nicht sinnvoll für eine Taktüberwachung nutzbar. Sie sollten auch nicht für andere Zwecke benutzt werden, weil sie sehr wohl aktiv sein können.
- Wegen der unterschiedlichen Hardware sind die Nummern der tatsächlich vorhandenen DIs nicht kompatibel zum INTUS 2000. Während die DIs 0 und 1 (E0 und E1) beim INTUS 3000 immer vorhanden sind, gibt es derzeit keine physikalisch nach außen geführten DIs mit den Nummern 2 und 3. Das Aufsteckmodul auf Kanal B hat DIs mit den Nummern 5 bis 8 (E5 bis E8), auf Kanal C sind es die DIs 9 bis 12 (E9 bis E12). Deshalb muss ein TCL-Programm, das eine Taktüberwachung auf mehr als den beiden ersten DIs verwendet, angepasst werden. Es ist dabei darauf zu achten, dass die Indizes x für Ex, Tx, TAx und TOx konsistent geändert werden. Eine indizierte Schreibweise wie z.B. E(P20+20,2) sollte dabei ebenfalls berücksichtigt werden.

## 6.6 Sommer-/ Winterzeitumschaltung

Mit TCL Version 5.01 ist eine umfassende Zeit- und Datumskontrolle in das TCL System integriert. Dabei wurden folgende Ziele verfolgt:

- Automatisierte Sommer- und Winterzeitumschaltung, aber auch
- durch den Leitrechner initiierte Sommer-/ Winterzeitumschaltung
- Zeitsynchronisation über GMT (Greenwich Mean Time) bzw. UTC (Coordinated Universal Time)

Die Erweiterung ist voll auf- und abwärtskompatibel zu bestehenden TCL Versionen, d.h. die Felder UR und KT enthalten wie bisher die lokale Uhrzeit und das lokale Datum. Bestehende TCL Programme laufen ohne Modifikationen. Für die neuen Funktionen wurde das P20-Feld erweitert.



Wegen den gesetzgeberischen Unsicherheiten wurde keine algorithmische Lösung implementiert. Stattdessen müssen die Parameter im Feld P20 explizit vom Leitrechner per TCL-Anweisungen initialisiert werden.

Eine Einstellung über den Terminal-Setup ist nicht vorgesehen.

Der folgende Abschnitt beschreibt die neu eingeführten Felder und deren Bedeutung. Der übernächste Abschnitt gibt Anwendungshinweise.

### 6.6.1 Felder zur Kontrolle der Uhrzeit und des Datums

Die Teilsteder zur Uhrzeitkontrolle beginnen bei P20+39 und bestehen aus zwei Bereichen, der Zeitkontrolle P20+57,30 und dem aktuellen Zeitstatus P20+39,18.

<b>P20 Teilsteder</b>		<b>Bedeutung</b>
aktueller Zeitstatus	P20+39,8	unkorrigierte Uhrzeit im Format (hh:mm:ss)
	P20+47,9	unkorrigiertes Datum im Format (mmttwjjjj)
	P20+56,1	Sommerzeitflag ('0' Winterzeit, '1' Sommerzeit) (nicht beschreibbar, wird nach Modifikation von P20+62,1 automatisch gesetzt)
Zeitkontrolle	P20+57,5	GMT/UTC-Abweichung (+/- xxxx min)
	P20+62,1	Kontrolle der Sommer-/ Winterzeitumschaltung
	P20+63,11	Sommerzeitanfang jährlich über den Kalender oder (ab TCL Version 6.10) immerwährend mit POSIX TZ-Format
	P20+74,11	Sommerzeitende bzw. Winterzeitanfang jährlich über den Kalender oder (ab TCL Version 6.10) immerwährend mit POSIX TZ-Format
	P20+85,2	reserviert für NTP Uhrzeitsynchronisation

*Tabelle 6.6 – Zusammenfassung der P20-Teilsteder zur Uhrzeitkontrolle*

### 6.6.1.1 Teifelder aktueller Zeitstatus

Den beiden ersten Teifeldern des aktuellen Zeitstatus, P20+39,17, kommt in den folgenden Abschnitten eine besondere Bedeutung zu. Sie werden auch mit unkorrigiertes Zeit/Datum-Paar bezeichnet.

Im Gegensatz zu den Möglichkeiten der beiden Felder UR und KT kann mit dem aktuellen Zeitstatus, P20+39,18, auch die Stunde 2A und 2B bei der Umschaltung zur Winterzeit eindeutig dargestellt werden: während der Stunde 2A ist der Sommerzeitstatus P20+56,1 = '1' (Sommerzeit). Ab 2B:00 Uhr ist der Sommerzeitstatus P20+56,1 = '0' (Winterzeit).

Die Reihenfolge der Teifelder ist so angelegt, dass mit den ersten vier Teifeldern, P20+39,23, die lokale Zeit und das lokale Datum eindeutig beschrieben sind, gleichgültig wo auf der Welt die Zeit mit dem INTUS 3000 Terminals erfasst wird. Dies kann in internationalen Unternehmen, bei denen Buchungen aus verschiedenen Nationen zusammengeführt werden sollen, nützlich sein.

#### Wichtiger Hinweis, um Zeit und Datum konsistent zu halten:



Eine Änderung im Bereich des aktuellen Zeitstatus muss auf einmal (d.h. in einer **TCL** Anweisung) erfolgen. Das komplette unkorrigierte Zeit/Datum-Paar, **P20+39,17**, muss neu gesetzt werden, um Datum und Uhrzeit in konsistentem Zustand zu erhalten.

Ein Beispiel erläutert den Zusammenhang: wird das Datum kurz vor Mitternacht und die Uhrzeit kurz nach Mitternacht gesetzt, dann wird ein Tag übersprungen.

Aus ähnlichen Gründen muss der Bereich der Zeitkontrolle, zumindest aber Sommerzeitanfang und Winterzeitanfang, P20+63,22, auf einmal (atomar) geschrieben werden.

Setzen Sie immer zuerst den Bereich der Zeitkontrolle, P20+57,30, und danach das unkorrigierte Zeit/Datum-Paar, P20+39,17. (Wenn die Zeitkontrolle zusammen mit dem unkorrigierten Zeit/Datum-Paar in P20 geschrieben wird, dann werden zunächst die Zeitkorrekturflags ausgewertet und dann der damit korrigierte aktuelle Zeitstatus zum Setzen der internen Zeit verwendet.)

#### Wichtige Hinweise zum Arbeiten mit den neuen Teifeldern der Uhrzeitkontrolle:

1. **Die Felder UR und KT enthalten immer**, wie in den älteren **TCL** Versionen, **die aktuelle, lokale Zeit und das aktuelle, lokale Datum**. Diese Zeit wird auch von der Hardwareuhr (RTC) gehalten und ist damit auch die interne Systemzeit. Sie wird nicht durch Ändern der Zeitkontrolle (Zeitkorrekturflags: GMT/UTC-Abweichung und Kontrolle der Sommer-/ Winterzeitumschaltung) beeinflusst. Vom **TCL** System wird in diesem Fall die Zeit unter Berücksichtigung der Zeitkorrekturflags in das unkorrigierte Zeit/Datum-Paar, P20+39,17, eingetragen. Wenn im Bereich der Zeitkontrolle kein Korrekturflag gesetzt ist, dann enthält das unkorrigierte Zeit/Datum-Paar dieselben Werte wie UR und KT. Das ist die Voreinstellung.

Wenn der Leitrechner nur über die lokale Zeit verfügt, dann kann er bei dieser Voreinstellung weiterhin diese lokale Zeit zur Zeitsynchronisation in die Felder UR,KT eintragen.



Aus dem oben genannten Beispiel unter der Überschrift „*Wichtiger Hinweis, um Zeit und Datum konsistent zu halten*“ ergibt sich aber, dass das **atomare** Setzen (d.h. das Schreiben mit einer **TCL** Anweisung) des unkorrigierten Zeit/Datum-Paares mit der lokalen Zeit sicherer ist.

2. Zwischen der lokalen Zeit und der unkorrigierten Zeit, das ist die Greenwich Mean Time GMT bzw. die Coordinated Universal Time UTC, gelten folgende Zusammenhänge (in Sekunden):

*unkorrigierte\_Zeit*

$$= \text{lokale\_Zeit} + \text{GMT/UTC\_Abweichung} * 60 - \text{Sommerzeit\_Flag} * 3600$$

*lokale\_Zeit*

$$= \text{unkorrigierte\_Zeit} - \text{GMT/UTC\_Abweichung}*60 + \text{Sommerzeit\_Flag}*3600$$

3. Ist die **automatische Sommerzeitumschaltung** (P20+62,1='2') ohne GMT/UTC-Abweichung aktiviert, enthält das unkorrigierte Zeit/Datum-Paar, P20+39,17, immer die Winterzeit. Die lokale Uhrzeit vom TCL System wird unter Berücksichtigung des Sommerzeitflags berechnet und in die Felder UR, KT eingetragen.

Die Zeitsynchronisation vom Leitrechner verwendet das unkorrigierte Zeit/Datum-Paar. Es muss immer mit der Winterzeit, und nicht mit der lokalen Zeit geladen werden.

4. Ist dazu noch die **GMT/UTC-Abweichung**, P20+57,5 , aktiviert, enthält das unkorrigierte Zeit/Datum-Paar, P20+39,17, immer die GMT/UTC-Zeit. Die lokale Uhrzeit vom TCL System wird unter Berücksichtigung der GMT/UTC-Abweichung und des Sommerzeitflags berechnet und in die Felder UR, KT eingetragen.

Die Zeitsynchronisation vom Leitrechner verwendet das unkorrigierte Zeit/Datum-Paar. Sie müssen immer mit der GMT/UTC-Zeit und nicht mit der lokalen Zeit geladen werden.

5. Wird entweder die GMT/UTC-Abweichung, P20+57,5, oder die Kontrolle der Sommer-/ Winterzeitumschaltung, P20+62,1, (was zum automatischen Setzen des Sommerzeit-Flags führt) geändert, dann ändert sich sowohl das unkorrigierte Zeit/Datum-Paar, als auch UR und KT.



Setzen Sie immer zuerst die Zeitkorrekturflags im Bereich der Zeitkontrolle und danach das unkorrigierte Zeit/Datum-Paar. Siehe weiter oben unter der Überschrift „*Wichtiger Hinweis, um Zeit und Datum konsistent zu halten*“.

Um nicht mit der automatisierten Zeitumschaltung in Konflikt zu kommen, sollte nach einem Einschalten der Automatik die Zeitsynchronisation nur noch über die neuen Teilfelder von P20 erfolgen und nicht mehr über UR und KT.

6. Bei einem Warm- oder Kaltstart bleiben alle eingestellten Werte des Zeitkontrollbereichs erhalten.



**Ein Eiskaltstart und ein Comstart löscht alle eingestellten Parameter im Zeitkontrollbereich, P20+57,30. Insbesondere wird damit auch die Kontrolle der Sommer-/ Winterzeitumschaltung, P20+62,1, auf '0' gesetzt und damit auch der Sommerzeitstatus, P20+56,1. Deshalb müssen nach dem Laden des TCL-Programms, d.h. schon bei jedem Kaltstart, die Korrekturfaktoren durch das TCL-Programm überprüft und eventuell neu gesetzt werden.**

7. Die Parameter des Zeitkontrollbereichs werden im Terminal-Setup (in der Datei intus.cfg ) gespeichert und können im Setup unter Test|Version/Status angezeigt, aber nicht geändert werden.

Mit den INTUS RemoteSetup kann der Zeitkontrollbereich angezeigt und modifiziert werden.

8. Das Laden eines TCL-Programms, das die neuen Feldoffsets von P20 verwendet, in eine Vorgängerversion des TCL-Interpreters führt nicht zu einem Fehler.

Soll das TCL-Programm abwärtskompatibel funktionieren, muss es nur anhand der im LS-Feld angegebenen Version entscheiden, ob es die Sommer-/ Winterzeitumschaltung nutzen kann, etwa in der Form:

`DP,LS+22,<'5',@100:!Alte Version:`

**Die Felder, die den Zeitstatus angeben, sind wie folgt aufgebaut:**

P20+39,8 unkorrigierte Uhrzeit	ist identisch zu dem Feld UR mit dem Format HH:MM:SS aufgebaut. Dieses Feld enthält die Uhrzeit ohne die Korrekturen Sommerzeit, P20+56,1 (basiert auf P20+62,23), und GMT/UTC-Abweichung, P20+57,5. Wenn sowohl eine GMT/UTC-Abweichung ungleich 0 angegeben ist, als auch eine Sommerzeitumschaltung, dann enthält P20+39,8 die zur lokalen Zeit in UR gehörende GMT/UTC-Zeit. Ist nur die Sommer-/ Winterzeitumschaltung aktiviert, dann enthält P20+39,8 die Winterzeit, egal ob aktuell die Sommerzeit gültig ist. Wenn beide Korrekturen unterdrückt sind, dann ist P20+39,8 identisch zum Feld UR. Beide Felder UR und P20+39,8 haben identische Trennzeichen (hier ':'). Das zuletzt geschriebene Trennzeichen wird in beiden Feldern verwendet.
P20+47,9 unkorrigiertes Datum	dieses Feld enthält das zur unkorrigierten Uhrzeit in P20+39,8 gehörende unkorrigierte Datum und hat dasselbe Format wie das KT-Feld. Da das unkorrigierte Datum bei Anwendung von Korrekturen von dem aktuellen Datum abweichen kann, ist es dann nicht zur Anzeige geeignet.
P20+56,1 Sommerzeit- Flag (Status)	ist '0' wenn Winterzeit gültig ist, '1' bei Sommerzeit. Dieses Feld wird automatisch neu berechnet, wenn die automatische Sommer-/ Winterzeitumschaltung in P20+62,1 mit dem Wert '2' eingeschaltet ist. Wenn die automatisierte Umschaltung in P20+62,1 nicht eingeschaltet ist, dann wird der dort eingestellte Wert '0' oder '1' hier als Sommerzeitstatus übernommen, was eine programmierte Einstellung des Winter/Sommerzeitstatus erlaubt. Ein Schreiben in dieses Feld hat keine Wirkung. <b>Anmerkung:</b> Dieses Flag wird im Uhrbaustein abgelegt. Es bleibt auch über einen Warm/Kaltstart erhalten, solange die Batterie bzw. die Netzspannung die Werte im Uhrbaustein erhält. Bei einem Eiskalt- oder Comstart wird P20+62,1 zu '0' gesetzt und erzwingt damit, dass P20+56,1 auch '0' wird.

**Tabelle 6.7 – P20-Teilfelder für den aktuellen Zeitstatus**

### 6.6.1.2 Teifelder Zeitkontrolle

Die Felder zur Kontrolle der Zeitumschaltung bzw. des Netzwerkabgleichs sind wie folgt angeordnet:

P20+57,5 GMT/UTC Abweichung	<p>Das Feld P20+57,5 gibt die Abweichung der ortsbezogenen <b>Winterzeit</b> zur GMT/UTC-Zeit an. Das erste Byte P20+57,1 enthält ein Vorzeichen '+' für eine positive (westliche) und '-' für eine negative (östliche) Differenz. Ein Leerzeichen oder eine '0' an dieser Stelle bedeutet beim Beschreiben dieses Felds ebenfalls ein positives Vorzeichen. Gelesen wird jedoch immer '+' oder '-'.</p> <p>Die folgenden vier Bytes P20+58,4 enthalten die absolute Differenz von der GMT/UTC-Zeit in Minuten. MEZ ist demnach als '0060' in dieses Feld einzutragen (wie in der UNIX TZ-Environment-Variablen).</p> <p>Wenn dieses Feld 0 (etwa '+0000') enthält, wird keine Korrektur zur GMT/UTC-Zeit verrechnet. Der korrekte Eintrag dieses Korrekturfaktors ist insbesondere dann wichtig, wenn ein Uhrzeitabgleich über ein Netzprotokoll erfolgen soll, die standardmäßig nur GMT/UTC-Zeit angeben.</p> <p>Der Wert nach einem Eiskalt- oder Comstart ist '+0000'.</p>
P20+62,1 Kontrolle der Sommer/ Win- terzeitum- schaltung	<p>Das Feld P20+62,1 kontrolliert die automatische Sommerzeitumschaltung. Es kann folgende Werte annehmen:</p> <ul style="list-style-type: none"> <li>'0'      programmierte Einstellung der Winterzeit; keine automatische Sommer-/ Winterzeitumschaltung. Das Feld P20+56,1 nimmt stets den Wert '0' an.</li> <li>'1'      programmierte Einstellung der Sommerzeit; keine automatische Sommer-/ Winterzeitumschaltung. Das Feld P20+56,1 nimmt stets den Wert '1' an.</li> <li>'2'      automatische Sommer-/ Winterzeitumschaltung; die Daten in den Feldern P20+63,11 und P20+74,11 müssen korrekt sein. Das Feld P20+56,1 nimmt automatisch den Wert '0' oder '1' an.</li> </ul> <p>Der Wert nach einem Eiskalt- oder Comstart ist '0'.</p>
P20+63,11 Start der Som- merzeit	<p>Dieses Feld enthält den Start der Sommerzeit.</p> <p>Es wird nur im Zusammenhang mit der automatischen Sommer-/ Winterzeitumschaltung verwendet.</p> <p>Ab TCL Version 6.10 steht die immerwährende, regelbasierte Sommerzeitumschaltung mit dem <b>POSIX TZ-Format</b> zur Verfügung:  <math>Mm.w.d/h</math> Dies spezifiziert den Umschaltzeitpunkt als Tag <math>d</math> der Woche <math>w</math> im Monat <math>m</math> zur Stunde <math>h</math>.</p> <p>Der Tag <math>d</math> muss zwischen 0 (Sonntag) und 6 (Samstag) liegen. Die Woche <math>w</math> muss zwischen 1 und 5 liegen; Woche 1 ist die erste Woche in der der Wochentag <math>d</math> auftritt und Woche 5 ist die letzte Woche in der der Wochentag <math>d</math> vorkommt. Der Monat <math>m</math> darf Werte zwischen 1 und 12 annehmen, die Stunde <math>h</math> Werte zwischen 0 und 23. Der gesamte Zeitpunkt muss in P20+63,11 passen und ist mit Leerzeichen aufzufüllen.</p>

	<p>Seit TCL Version 5.01 ist die jährliche Anpassung der Sommerzeitumschaltung mit <b>Datum und Stunde</b> möglich:</p> <p><i>mmttwjjjhh</i> Datumsformat <i>mmttwjjj</i> wie KT-Feld, Stundenangabe <i>hh</i> mit '00' – '23'. Der Wochentag <i>w</i> darin spielt keine Rolle.</p> <p>In einer aktuell laufenden Sommerzeit kann dieser Wert auf den Start der Sommerzeit des nächsten Jahres umgestellt werden. Falls P20+63,11 einen späteren Zeitpunkt als P20+74,11 darstellt, gilt für alle Tage bis zum Umschaltzeitpunkt in P20+74,11 die Sommerzeit.</p> <p>Der Wert in diesem Feld wird mit in der Setup-Datei INTUS.CFG abgelegt. Wenn diese Datei nicht lesbar ist (etwa nach der Softwareproduktion oder Eiskalt- bzw. Comstart), nimmt dieses Feld den Wert '000000000000' (ungültiges Datum) an.</p>
P20+74,11 Start der Winterzeit	<p>Dieses Feld enthält den Start der Winterzeit.</p> <p>Es wird nur im Zusammenhang mit der automatischen Sommer-/Winterzeitumschaltung verwendet.</p> <p>Ab TCL Version 6.10 steht die immerwährende, regelbasierte Winterzeitumschaltung mit dem <b>POSIX TZ-Format</b> zur Verfügung:</p> <p><i>Mm.w.d/h</i> Dies spezifiziert den Umschaltzeitpunkt als Tag <i>d</i> der Woche <i>w</i> im Monat <i>m</i> zur Stunde <i>h</i>.</p> <p>Der Tag <i>d</i> muss zwischen 0 (Sonntag) und 6 (Samstag) liegen. Die Woche <i>w</i> muss zwischen 1 und 5 liegen; Woche 1 ist die erste Woche in der der Wochentag <i>d</i> auftaucht und Woche 5 ist die letzte Woche in der der Wochentag <i>d</i> vorkommt. Der Monat <i>m</i> darf Werte zwischen 1 und 12 annehmen, die Stunde <i>h</i> Werte zwischen 0 und 23. Der gesamte Zeitpunkt muss in P20+74,11 passen und ist mit Leerzeichen aufzufüllen.</p>
	<p>Seit TCL Version 5.01 ist die jährliche Anpassung der Winterzeitumschaltung mit <b>Datum und Stunde</b> möglich:</p> <p><i>mmttwjjjhh</i> Datumsformat <i>mmttwjjj</i> wie KT-Feld, Stundenangabe <i>hh</i> mit '00' – '23'. Der Wochentag <i>w</i> darin spielt keine Rolle.</p> <p>In einer aktuell laufenden Winterzeit kann dieser Wert auf den Start der Winterzeit des nächsten Jahres umgestellt werden. Falls P20+74,11 einen späteren Zeitpunkt als P20+63,11 darstellt, gilt für alle Tage bis zum Umschaltzeitpunkt in P20+63,11 die Winterzeit.</p> <p>Der Wert in diesem Feld wird mit in der Setup-Datei INTUS.CFG abgelegt. Wenn diese Datei nicht lesbar ist (etwa nach der Softwareproduktion oder Eiskalt- bzw. Comstart), nimmt dieses Feld den Wert '000000000000' (ungültiges Datum) an.</p>
P20+85,1 Kontrolle des Netzwerk-Zeitabgleichs	<p>Das Feld P20+85,1 kontrolliert den automatischen Zeitabgleich über ein LAN-Netzwerk. Es kann folgende Werte annehmen:</p> <p>'0' kein Abgleich der Zeit über ein Netzwerkprotokoll.</p> <p>Weitere Werte sind für zukünftige Entwicklungen reserviert.</p> <p>Der Wert in diesem Feld wird mit in der Setup-Datei INTUS.CFG abgelegt. Wenn diese Datei nicht lesbar ist (etwa nach der Softwareproduktion oder Eiskalt- bzw. Comstart), nimmt dieses Feld den Wert '0' an.</p>

P20+86,1 Status des Netzwerk- Zeitabgleichs	Das Feld P20+82,1 erlaubt die Erkennung eines Zeitabgleichs über ein Netzwerkprotokoll. Nach einem Abgleich über ein Netzwerkprotokoll wird dieses Feld auf '1' gesetzt. Das TCL-Programm kann das Feld nach der Auswertung wieder auf '0' setzen und so den nächsten Zeitpunkt eines Zeitabgleichs in periodischen Abständen pollend erfragen bzw. überwachen.  Nach einem Warm/Kalt/Com/Eiskaltstart ist der Wert dieses Feldes '0'.
--	--

**Tabelle 6.8 – P20-Teilfelder für die Zeitkontrolle**

## 6.6.2 Anwendungsbeispiele

Die folgenden Beispiele gehen davon aus, dass die in der ANSI (C-) Norm definierte TZ-Umgebungsvariable bei einem Leitrechner korrekt mit den aktuellen Umschaltzeitpunkten für Sommer- und Winterzeit definiert ist. Des Weiteren wird davon ausgegangen, dass die ANSI-C Funktionen time(), mktime(), gmtime(), localtime() zur Verfügung stehen.

In den Beispielen wird die direkte Form der TCL-Anweisung über das Routing-Byte 'I' verwendet. Tatsächlich sollte einem an den \$7-Puffer über das Routing-Byte 'J' geschickte Datensatz der Vorzug gegeben werden.

### Sommer-/ Winterzeitumschaltung durch das Terminal

Wenn die Sommer-/ Winterzeitumschaltung durch das Terminal erfolgen soll, müssen die Umschalttage in P20+63,22 eingetragen werden. Weiterhin muss eine '2' in P20+62,1 eingetragen werden.

Es wird dringend empfohlen, einen Uhrzeitabgleich durch den Leitrechner nur über das Feld P20+39,17 und nicht mehr über die Felder UR und KT zu machen. Nicht zuletzt deshalb, weil die Uhrzeit 2:00A und 2:00B bei der Winterzeitumschaltung nicht eindeutig (im UR-Feld) ist. In P20+39,17 muss in diesem Fall aber immer die Winterzeit (und nicht die lokale Zeit) eingetragen werden.

Obwohl man auch die Winterzeit in P20+39,17 eintragen kann, wenn die GMT/UTC-Abweichung 0 ist, wird empfohlen, gleich GMT/UTC zu verwenden, weil dies einen zukünftigen Zeitabgleich über ein LAN-Netz ermöglicht. Dazu trägt man die GMT/UTC-Abweichung in P20+57,5 ein und verwendet zum Zeitabgleich in P20+39,17 die C-Funktion gmtime().

### Beispiel für Deutschland:

Abweichung zur GMT/UTC-Zeit in Zentraleuropa und Flag für die automatische Sommer-/Winterzeitumschaltung:

IK, '-00602', P20+57:

Immerwährende Sommer-/ Winterzeitumschaltung im POSIX TZ Format (Stand 2010):

Die Sommerzeit beginnt am letzten Sonntag im März um 2 Uhr:

IK, 'M3.5.0/02', P20+63:

Die Sommerzeit endet am letzten Sonntag im Oktober um 3 Uhr:

IK, 'M10.5.0/03', P20+74:

Sommer-/ Winterzeitumschaltung für 2011 mit Datum und Stunde, jährlich anzupassen:

IK, '03270201102', P20+63:

IK, '10300211103', P20+74:

### Sommer-/ Winterzeitumschaltung durch den Leitrechner

Wenn der Leitrechner die Zeitumschaltung vornehmen soll, dann verlangt dies nach einer permanenten Verbindung zum Terminal und nach einem Abgleichsvorgang der kurz (Vorschlag 10 Sekunden) nach jeder vollen Stunde stattfindet. Die in diesem Intervall aufgerufene C-Routine könnte etwa so aussehen:

```
void
intus_update_time (void)
{
    struct tm * tm;
    time_t stamp;
    int sz_flag, n;
    stamp = time (NULL);
    tm = localtime (&stamp);
    if ((sz_flag = tm->tm_dst) < 0)
        sz_flag = 0;
    tm = gmtime (&stamp);
    n = sprintf (intus_buf,
                 "IK,%1u", P20+62:" ", sz_flag);
    n += sprintf (&intus_buf [n],
                  "K,%2.2u:%2.2u:%2.2u%2.2u%1u%4.4u%\r",
                  tm->tm_hour, tm->tm_min, tm->tm_sec,
                  tm->tm_mon + 1, tm->tm_mday, tm->tm_wday,
                  tm->tm_year + 1900);
    send_intus (intus_buf, n);
}
```

Dies setzt jedoch die Kenntnis der GMT/UTC-Abweichung im Terminal voraus. Für Mitteleuropa ist dies '-0060'. Eine überall gültige Abweichung könnte etwa mit folgender Funktion (bitte vorher wegen Systemabhängigkeiten gegentesten!) berechnet und gesetzt werden:

```
void
intus_set_gmt_diff (void)
{
    struct tm local, gm;
    time_t gm_t, local_t, t;
    int n;

    t = time (NULL);
    local = *localtime (&t);
    gm = *gmtime (&t);
    gm.tm_isdst = 0;
    if (local.tm_isdst < 0)
        local.tm_isdst = 0;
    local_t = mktime (&local);
    gm_t = mktime (&gm);
    n = sprintf (intus_buf, "IK,%+5.4ld", P20+57:\n",
                 (long) ((gm_t - local_t) / 60));
    send_intus (intus_buf, n)
}
```

### **6.6.3 Zeiteinträge in Buchungssätzen**

Das Fehlen einer (ANSI-C) Bibliotheksfunction, die eine Umrechnung von GMT/UTC-Zeit zusammen mit einem Sommerzeit-Flag und einer GMT/UTC-Abweichung in eine lokale Zeit leistet, legt es zunächst nahe, bei den Buchungssätzen die in den Feldern UR und KT vorhandene, lokale Zeit zusammen mit dem Sommerzeit-Flag in P20+56,1 in den Buchungssätzen zu verwenden.

Je nachdem, wie eine tarifliche Vereinbarung die Verrechnung der fehlenden oder zusätzlichen Umschaltstunden an den Umschalttagen regelt, ist es jedoch günstiger die Buchungssätze mit der GMT/UTC-Zeit und allen Korrekturfaktoren zu versehen und abzuspeichern und sogar zu verrechnen. In diesem Fall würde man das Teilstück P20+39,23 komplett in den Buchungssatz übernehmen. Dies liegt insbesondere dann nahe, wenn Buchungen aus verschiedenen Nationen zusammengeführt werden sollen.

In diesem Fall benötigt man eine Funktion, die aus einer GMT/UTC-Zeit mit allen Korrekturen eine lokale Zeit berechnet. Dazu kann man zunächst mit `mktime()` aus der GMT/UTC-Zeit einen fehlerhaft korrigierten, lokalen Zeitstempel machen, den man dann zur GMT/UTC-Zeit oder zur lokalen Zeit (des Leitrechners oder des Terminals) mit Hilfe der Korrekturfaktoren (des Leitrechners oder des Terminals) umkorrigieren kann.

Es gelten folgende Zusammenhänge (in Sekunden):

$$\text{GMT/UTC} = \text{lokale\_Zeit} + \text{GMT/UTC\_Abweichung} * 60 - \text{Sommerzeit\_Flag} * 3600$$

oder

$$\text{lokale\_Zeit} = \text{GMT/UTC} - \text{GMT/UTC\_Abweichung} * 60 + \text{Sommerzeit\_Flag} * 3600$$



### **7 Ansteuerung des Displays**

Das TCL-Programmiersystem enthält zur Ansteuerung eines Displays einen VT100-Emulator, der es ermöglicht, alle Displays in ähnlicher Weise anzusprechen.

Die Zutrittskontroller sind optional mit 2-zeiligen Zeichendisplays ausgestattet. Bei den Terminals kommt häufig das 240x64 Pixel Display zum Einsatz, das grafikfähig ist und unterschiedliche Zeichengrößen bietet. Im folgenden Kapitel sind die Displays und ihre Einsatzmöglichkeiten beschrieben.

Einträge in das TCL Display-Feld D (Abschnitt 4.7) werden intern so in Steuersequenzen umgesetzt, dass die in das D-Feld geschriebenen Zeichen an einer entsprechenden Stelle auf dem Display erscheinen.

Die Stelle auf dem Display, an dem die in das D-Feld geschriebenen Zeichen erscheinen, wird durch den Offset des D-Felds bestimmt.

#### **Beispiel:**

Im Folgenden nehmen wir ein Display mit 40 Zeichen in einer Zeile an. Dann bewirkt die TCL-Anweisung

**K, 'Hier' ,D+45:**

eine Anzeige des Texts 'Hier' ab der sechsten Position der zweiten Zeile des Displays.

Die untenstehende TCL-Anweisung erzeugt ebenso eine Anzeige des Texts 'Hier' ab der sechsten Position der zweiten Zeile des Displays.

**K, 'Hier' ,D[1]+5:**

Hier gibt der '['-Operator die Zeile für die Ausgabe vor, wobei die erste Zeile die Nummer 0 hat.

Bei einer Veränderung des D-Felds führt das TCL-System folgende Teilschritte aus:

- Abspeicherung der Änderung im D-Feld,
- Sicherung der aktuellen Cursorposition (ESC 7),
- die zum Offset im D-Feld gehörende Cursorposition wird eingestellt (ESC[<l>;<c>H),
- die in das D-Feld geschriebenen Zeichen werden ausgegeben und
- die alte Cursorposition wird zurückgeladen (ESC 8).

Damit wird also sichergestellt, dass bei Ausgabe eines druckbaren Textes im sichtbaren Bereich des Displays, das D-Feld mit der Anzeige auf dem Display übereinstimmt.

 Das Sichern und Zurückladen der Cursorposition ist insbesondere bei einer freigegebenen Tastatureingabe notwendig, um den Cursor auf der richtigen Eingabeposition zu belassen.

## Steuersequenzen

Wenn ein Programmierer auf der Anzeige Dinge darstellen möchte, die über einen normalen Text hinausgehen, dann kann er selbst Steuersequenzen in das D-Feld schreiben.

Jedoch ist dann nicht mehr gewährleistet, dass das D-Feld den gleichen Inhalt hat, wie die auf dem Display sichtbaren Zeichen. Die Anweisung

**K,<"1B"&'[1;1HHier',D[1]+5:**

schreibt den Text 'Hier' in die erste Zeile links oben und verändert jedoch das D-Feld im Bereich D[1]+5,10, wobei D[1]+5,1 den Wert "1B" enthält.

Um diese "Verschmutzung" des D-Felds zu vermeiden, gibt es die Möglichkeit eine Steuersequenz, die mit ESC ("1B") beginnt, in einen Bereich des D-Felds zu schreiben, der nicht mehr von dem vorhandenen Display angezeigt wird. Zum Beispiel an den Offset D+1000.

Von dieser Möglichkeit sollte insbesondere bei langen Steuersequenzen Gebrauch gemacht werden, damit diese nicht unbeabsichtigt Teile des D-Felds verändern, die mit Hilfe des Aktualisierungsmechanismus (siehe Abschnitt 5.5) automatisch neu angezeigt werden.

In den meisten Fällen sollte die Steuersequenz mit einer Positionierung beginnen, die den Cursor an die gewünschte Stelle setzt.

Es wird davon abgeraten:

- Die Cursorposition selbst zu speichern und zurückzusichern. Dies ist nur **einmal** möglich und würde zu einer falschen Cursorposition bei einer freigegebenen Tastatureingabe führen.
- Die Ausgabeseite oder die Anzeigeseite (siehe Abschnitt 7.5) umzuschalten (ESC [=<n>q bzw. ESC [=<n>r]), denn die Seite 1 ist für das TCL-Programm reserviert und die Seite 2 für das Setup. Weitere Seiten sind bei den meisten Displaytypen nicht verfügbar.



Der Programmierer sollte darauf achten, dass eine Steuersequenz immer **ganz** in einer TCL-Anweisung in das D-Feld geschrieben wird. Eine Sequenz, die vollständig in einem Stück in das D-Feld geschrieben wird, wird dem VT100-Emulator auch als eine Einheit übergeben.

## 7.1 Übersicht über die eingesetzten Displays

Der VT100 Emulator setzt nur die Steuersequenzen um, die das angeschlossene Display versteht. Alle anderen Sequenzen werden ignoriert. Die Fähigkeiten der Displays differieren erheblich zwischen dem 2-zeiligen Zeichendisplay und den grafikfähigen Displays.

Displaytyp	CV+25,1 Displayvariante	Zeichensätze	CV+109,1 kann Latin-1 (Teifeld ab V5.01)	Format	Grafikdisplay	Darstellungsgröße verdoppeln	Steuersequenzen für Bit-Grafik (Kap. 7.6) GR Grafikfunktionen (ab TCL V6)
<b>2-zeilige Displays</b>	"01"	nationale Zeichensätze	'0'	eine Zeichen- größe	-	-	-
	"09" "0A"	nationale Zeichensätze, Latin-1 ab TCL V5	'1'	eine Zeichen- größe	-	-	-
<b>240 x 64 Pixel Display</b>	"07"	Zeichensätze siehe Abschnitt 7.3	'1'	verschiedene Zeichengrößen Umschaltung über P20+23,1 siehe Abschnitt 7.1.2	ja	ja	ja
<b>320 x 240 Pixel Display</b>	"08"	Zeichensätze siehe Abschnitt 7.3	'1'	eine Zeichen- größe 8 x 16 Pixel	ja	ja	ja
<b>INTUS Graph 640x480 Display</b>	"0C"	Zeichensätze siehe Abschnitt 7.3	'1'	Masken Defaultmaske wie 240x64 Pixel Display	ja	ja	-
<b>320x240 Pixel TFT Display</b>	"0D"	Zeichensätze siehe Abschnitt 7.3	'1'	TPI-Masken	ja	-	-
<b>480x272 Pixel TFT Display</b>	"0E"	Zeichensätze siehe Abschnitt 7.3	'1'	TPI-Masken	ja	-	-

*Tabelle 7.1 – Übersicht Displays*

### 7.1.1 2-zeilige Displays

Die 2-zeiligen Displays sind nicht grafikfähig; es werden nur Zeichen fester Größe dargestellt.

TCL geht bis einschließlich Version 4.28 von einem 2x40 stelligen Zeichendisplay aus, auf dem nur ein nationaler Zeichensatz dargestellt werden kann (CV+25,1 = "01"). Wenn auf einen anderen nationalen Zeichensatz umgeschaltet wird, ändern sich eventuell auch die schon auf dem Display stehenden, nationalen Zeichen, weil neue Zeichen in den Zeichensatzgenerator geladen werden müssen.

Ab Version 5.01 können auch 2x40 stellige Zeichendisplays eingesetzt werden, die den Latin-1 Zeichensatz beherrschen (CV+25,1 = "09").

Die 2x20 stelligen Zeichendisplays (CV+25,1 = "0A") haben immer diesen Latin-1 Zeichensatz.

Von der vollständigen Implementierung des Latin-1 Zeichensatzes gibt es folgende, technisch begründete Abweichungen:

Latin-1 Zeichencode			Abbildung auf Zeichencode		
Zeichen	dezi-mal	hexa-dez.	Zeichen	dezi-mal	he-xad ez.
(NBSP – nicht trennbares Leerzeichen)	160	a0	(SP – Leerzeichen)	32	20
„ (Umlautzeichen)	168	a8	(SP – Leerzeichen)	32	20
¬ (Nicht-Zeichen)	172	ac	(SP – Leerzeichen)	32	20
- (SHY – Silbentrennungsstrich)	173	ad	- (Minuszeichen)	45	2d
ˉ (Macron)	175	af	(SP – Leerzeichen)	32	20
ˊ (Akut)	180	b4	(SP – Leerzeichen)	32	20
, (Cedille)	184	b8	(SP – Leerzeichen)	32	20

**Tabelle 7.2 - Nicht darstellbare Latin-1 Zeichen**

Ab Version 5.01 gibt es ein zusätzliches CV-Feld, CV+109,1, mit dessen Hilfe ein TCL Programm erfahren kann, ob das eingebaute Display den internationalen Zeichensatz beherrscht.

Hat CV+109,1 den Wert '1', dann kann das Display den internationalen Latin-1 Zeichensatz darstellen. Wenn das Feld eine '0' enthält kann das Display nur einen nationalen Zeichensatz darstellen.



### 7.1.2 240x64 Pixel Display

Das 240x64 Pixel Grafikdisplay kennt neben verschiedenen Zeichengrößen die doppelt hohe und breite Darstellung von Zeilen (Abschnitt 7.4) und erlaubt die Darstellung von Bitmustern über die ESC[s und ESC[S Steuersequenzen (Abschnitt 7.6).

Es gibt Zeichengrößen in 6 und 8 Pixel Breite mit 8 und 16 Bit Höhe. Die Einstellung der Zeichengröße erfolgt über das Feld P20+23,1 (Display-Umschaltung). Nach einer Display-Umschaltung ist das Display gelöscht, die Ausgaben müssen wiederholt werden.

Änderungen an Zeichensätzen und Zeichenmapping sind zurückgesetzt. Siehe Abschnitt 7.3.

Die untenstehende Tabelle gibt Aufschluss über die Zeichengröße und die daraus resultierende Spalten- und Zeilenzahl des Displays.

<b>Wert in P20+23,1</b>	<b>Zeichenbreite in Pixel</b>	<b>Zeichenhöhe in Pixel</b>	<b>Anzahl Spalten</b>	<b>Anzahl Zeilen</b>
'1' (default)	6	8	40	8
'2'	6	16	40	4
'3'	6	8	40	8
'4'	8	8	30	8
'5'	8	16	30	4

**Tabelle 7.3 – Zeichengrößen beim 240x64 Pixel Display**

Alle vom Terminal darstellbaren Zeichensätze werden über VT100 (VT220) Steuersequenzen eingestellt (Abschnitt 7.3). Sie werden so gut wie möglich auf die internen Zeichensätze (Abschnitt 10) abgebildet.

Neben nationalen und internationalen Zeichensätzen können auch einfache Semigrafikzeichen, zum Beispiel für eine einfache Umrandung, ausgegeben werden. Siehe Abschnitt 7.3.

### 7.1.3 320x240 Pixel Display

Beim 320x240 Pixel große Grafikdisplay wird nur der 8x16 Pixel große Zeichensatz verwendet. Damit hat eine Änderung des Felds P20+23,1 (Display-Umschaltung) keine Wirkung.

Mit diesem Zeichensatz lassen sich 15 Zeilen zu 40 Zeichen darstellen.

Die doppelt hohe und breite Darstellung von Zeilen (Abschnitt 7.4) und die Darstellung von Bitmustern über die ESC[s und ESC[S Steuersequenzen (Abschnitt 7.6) ist möglich.

Alle vom Terminal darstellbaren Zeichensätze werden über VT100 (VT220) Steuersequenzen eingestellt (Abschnitt 7.3). Sie werden so gut wie möglich auf die internen Zeichensätze (Abschnitt 10) abgebildet.

Neben nationalen und internationalen Zeichensätzen können auch einfache Semigrafikzeichen, zum Beispiel für eine einfache Umrandung, ausgegeben werden. Siehe Abschnitt 7.3.

### **7.1.4 INTUS Graph 640x480 Display**

Das 640x480 Display des INTUS 5600 wird mit INTUS Graph angesteuert. INTUS Graph verwendet Masken, die in QML-Dateien definiert sind.

Das Handbuch *INTUS 5600 Grafik Handbuch – QML und Qt Creator (Bestellnummer D5600-003)* führt in die Erstellung der Masken und die Anbindung an TCL ein.

#### **Grundzustand: Emulation des 240x64 Pixel Displays**

Die Firmware des INTUS 5600 enthält eine interne Defaultmaske 'default'. Sie bildet ein 240x64 Pixel Display nach und stellt, je nach Wert in P20+23,1, die entsprechende Anzahl Zeilen und Spalten dar. Der voreingestellte Wert '1' in P20+23,1 führt zu 8 Zeilen und 40 Spalten auf dem Display.

Die Darstellung von Bitmustern über die ESC[s und ESC[S Steuersequenzen, die GR-Funktion und der Semigrafik-Zeichensatz stehen aber nicht zur Verfügung.

Diese Defaultmaske wird verwendet, wenn eine TCL-Applikation geladen ist, und keine anderen Masken im Terminal zur Verfügung stehen. Zwischen den Masken wird mit dem P22-Feld, das in TCL Version 6.50 eingeführt wurde, umgeschaltet. Siehe Abschnitt 4.34.

Das Defaultprogramm des INTUS 5600 verwendet die spezielle Maske 'defprog'.

 Steht ein anderer Maskensatz im Terminal zur Verfügung, kann dieser ebenfalls eine Maske 'default' enthalten, die dann an Stelle der internen Defaultmaske verwendet wird.

### **7.1.5 320x240 Pixel TFT Display**

Das 320x240 Pixel TFT Display wird in INTUS 5200 und INTUS 5205 Terminals eingebaut. Es werden ausschließlich TPI-Masken dargestellt.

Das Firmenlogo in den TPI-Masken kann durch das Laden einer Logo-Datei mit INTUS RemoteConf ausgetauscht werden.

Siehe Handbuch *INTUS TPI 3.7 Referenzhandbuch (Bestellnummer D3000-420.19)*.

### **7.1.6 480x272 Pixel TFT Display**

Das 480x272 Pixel TFT Display wird in INTUS 5540 Terminals eingebaut. Es werden ausschließlich TPI-Masken dargestellt.

Das Firmenlogo in den TPI-Masken kann durch das Laden einer Logo-Datei mit INTUS RemoteConf ausgetauscht werden.

Siehe Handbuch *INTUS TPI 3.7 Referenzhandbuch (Bestellnummer D3000-420.19)*.

## 7.2 Cursorsteuerung und Editorfunktionen

Am häufigsten werden Steuersequenzen zur Cursorsteuerung und zum Löschen von Bildschirmbereichen verwendet. Dabei steht ESC für das ASCII-Zeichen mit der hexadezimalen Kodierung "1B". Weiterhin werden folgende Platzhalter verwendet:

- <n> Ganze Zahl in druckbarer Dezimaldarstellung; meistens wird '1' angenommen, wenn sie weggelassen wird.
- <l> Angabe einer Zeilennummer in druckbarer Dezimaldarstellung. Wenn diese Angabe fehlt, wird '1' angenommen. Die oberste Zeile des Bildschirms hat die Nummer 1.
- <c> Angabe einer Spalte in druckbarer Dezimaldarstellung. Wenn diese Angabe fehlt, wird '1' angenommen. Die erste linke Spalte des Bildschirms hat die Nummer 1.

Die folgende Tabelle gibt die häufig benötigten Steuersequenzen für die Positionierung des Cursors an. Die Cursorposition ist gleichzeitig der Punkt, an dem das nächste sichtbare Zeichen auf dem Schirm dargestellt wird.

Bezeichnung	Code	Bedeutung
LF	0x0A	Line Feed: Cursor in der gleichen Spalte eine Zeile nach unten. Wenn der Cursor in der letzten Zeile ist, wird der Bildschirm gescrollt.
CR	0x0D	Carriage Return: Cursor in die erste Spalte der aktuellen Zeile
CUU	ESC [ <n> A	Cursor <n> Zeilen nach oben (ohne Scroll)
CUD	ESC [ <n> B	Cursor <n> Zeilen nach unten (ohne Scroll)
VPR	ESC [ <n> e	wie CUD
CUF	ESC [ <n> C	Cursor <n> Spalten nach rechts
HPR	ESC [ <n> a	wie CUF
CUB	ESC [ <n> D	Cursor <n> Spalten nach links
CUP	ESC [ <l> ; <c> H	absolute Cursorposition
HVP	ESC [ <l> ; <c> f	absolute Cursorposition
CUP	ESC [ H	Cursor Home
HVP	ESC [ f	Cursor Home
RI	ESC M	Cursor nach oben (mit Scroll)
IND	ESC D	Cursor nach unten (mit Scroll)
NEL	ESC E	Cursor auf Anfang nächste Zeile
	ESC 7	Cursorposition speichern
	ESC 8	Cursor auf gespeicherte Position
CHA	ESC [ <n> G	absolute Cursorposition in Spalte <n>
HPA	ESC [ <n> `	wie CHA
CPL	ESC [ <n> F	Cursor <n> Zeilen hoch und Zeilenanfang
CNL	ESC [ <n> E	Cursor <n> Zeilen nach unten und Zeilenanfang

Tabelle 7.4 - Cursorsteuerung

Funktionen, die zum Löschen von Bildschirmbereichen oder zur Verschiebung von Information auf dem Bildschirm dienen, finden sich in folgender Tabelle.

Bezeichnung	Code	Bedeutung
EL	ESC [ K	von Cursorposition bis Zeilenende löschen
	ESC [ 0 K	
EL	ESC [ 1 K	vom Zeilenanfang bis zum Cursor löschen
EL	ESC [ 2 K	ganzen Zeileninhalt löschen
ED	ESC [ J	von Cursorposition bis Bildschirmende löschen
	ESC [ 0 J	
ED	ESC [ 1 J	Bildschirm bis Cursor löschen
ED	ESC [ 2 J	ganzen Bildschirm löschen
DCH	ESC [ <n> P	<n> Zeichen ab Cursorposition löschen
IL	ESC [ <n> L	<n> Zeilen einfügen
DL	ESC [ <n> M	<n> Zeilen löschen, nachfolgende Zeilen rücken nach
	ESC [ <n> X	<n> Zeichen mit Leerzeichen überschreiben
ICH	ESC [ <n> @	<n> Leerzeichen einfügen, Rest der Zeile verschieben

**Tabelle 7.5 - Editorfunktionen**

Die Darstellung des Cursors kann mit folgenden Steuersequenzen verändert werden:

Code		Bedeutung
ESC [ 0 v	*	Cursor sichtbar
ESC [ ? 25 l		
ESC [ 1 v	*	Cursor unsichtbar
ESC [ ? 25 h		
ESC [ 2 v	*	Unterstrich-Cursor
ESC [ 34 h		
ESC [ 3 v	*	Block-Cursor
ESC [ 34 l		
ESC [ 4 v	*	Cursor nichtblinkend
ESC [ 5 v	*	Cursor blinkend

Eine Kennzeichnung mit '\*' bedeutet, dass die Sequenz eine spezielle Erweiterung für die INTUS Terminals ist.

**Tabelle 7.6 – Cursor Attribute**

Für die Veränderung der Darstellung der (druckbaren) Zeichen auf dem Schirm sind folgende Sequenzen vorgesehen:

<b>Code</b>		<b>Bedeutung</b>
ESC [ m		Attribute ausschalten
ESC [ 0 m	*	
ESC [ 1 m		volle Intensität (Bold) einschalten
ESC [ 4 m		Unterstreichen einschalten
ESC [ 5 m		Blinken einschalten
ESC [ 7 m	*	Invers einschalten
ESC [ 22 m		normal hell
ESC [ 25 m		Blinken ausschalten
ESC [ 27 m	*	Invers ausschalten

Eine Kennzeichnung mit '\*' bedeutet, diese Sequenzen werden derzeit im INTUS 3000 wirksam.

**Tabelle 7.7 - Zeichenattribute**

Die folgende Tabelle zur Displaysteuerung enthält die Sequenzen, die nicht in den folgenden Kapiteln gesondert beschrieben werden.

<b>Bezeichnung</b>	<b>Code</b>	<b>Bedeutung</b>
RIS	ESC c	Rücksetzen des Bildschirm-Treibers/ -Emulators
	ESC [ <n>; <m> r	Scrollbereich definieren: <n> oberste Zeile des Scrollbereichs, <m> unterste Zeile des Scrollbereichs (<m> > <n>)
	ESC [ ? 7 h	Automatisches Zeilenende (Line Wrap) ein
	ESC [ ? 7 l	Automatisches Zeilenende (Line Wrap) aus

**Tabelle 7.8 – Displaysteuerung**

## 7.3 Zeichensätze und Zeichenmapping

Folgende Zeichensätze stehen in TCL zur Verfügung:

Zeichensatz	Bemerkung	ab TCL Version	verfügbar bei 2-zeiligem Display
8 Bit, international	ISO 8859-1 Latin-1	Im Bereich der Kodierung "00" - "7F" identisch zu den anderen hier gelisteten ISO 8859 Zeichensätzen und zu US-ASCII	V4.22 siehe <i>Bemerkung 1</i> CV+109,1 = '1': ja CV+109,1 = '0': nein
	ISO 8859-2 Latin-2	Siehe oben	V6.04 siehe <i>Bemerkung 2</i>
	ISO 8859-5 kyrillisch	Siehe oben	V6.04 siehe <i>Bemerkung 2</i>
	ISO 8859-15 Latin-9	Siehe oben	V6.04 siehe <i>Bemerkung 2</i>
7 Bit, national und andere	ISO 646 Großbritannien		V4.22 ja
	US-ASCII	Im Bereich der Kodierung "00" - "7F" identisch zu ISO 8859-1	V4.22 ja
	ISO 646 Deutschland		V4.22 ja
	ISO 646 Frankreich		V4.22 ja
	ISO 646 Spanien		V4.22 ja
	ISO 646 Norwegen		V4.22 ja
	Semigrafik		V4.22 siehe <i>Bemerkung 3</i> nein

Tabelle 7.9 – TCL-Zeichensätze

### Bemerkung 1

Ab TCL V5.01 liegt der Latin-1 Zeichensatz überarbeitet und vervollständigt vor. Bis einschließlich V4.28 stellten die Graphikdisplays die Zeichen dar, die sich im intern verwendeten IBM 437-Zeichensatz befanden.

### Bemerkung 2

Dieser internationale Zeichensatz steht in den Modellen INTUS 3300, INTUS 3400 und INTUS 3500 nicht zur Verfügung.

**Bemerkung 3**

Der Semigrafik Zeichensatz steht beim INTUS 5600/5200/5205 nicht zur Verfügung.



Alle TCL-Zeichensätze sind in Kapitel 10 vollständig aufgelistet.

Der VT100 Emulator des TCL-Programmiersystems zur Ansteuerung eines Displays folgt der VT220 Konvention bei der Behandlung der Zeichensätze:

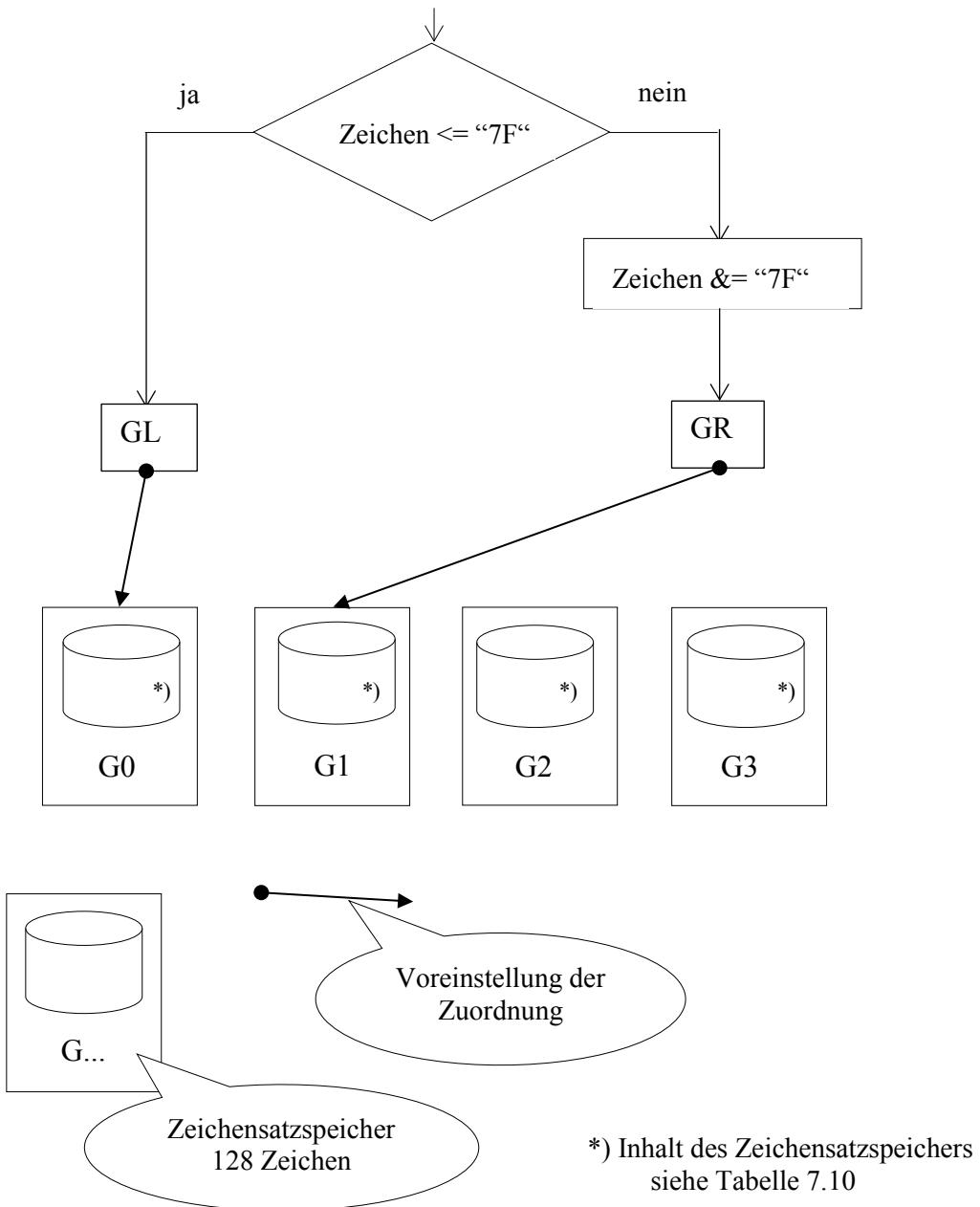


Abbildung 7.1 – 8-Bit Zeichenausgabe

Zeichensatz-einstellung in CV+39	G0	G1	G2	G3
ISO 646 * (7-Bit)	ISO 646 *	ISO 8859-1 ("80" - "FF")	ISO 646 *	ISO 8859-1 ("80" - "FF")
ISO 8859-* (8-Bit)	ISO 8859-* ("00" - "7F")	ISO 8859-* ("80" - "FF")	ISO 8859-* ("00" - "7F")	ISO 8859-* ("80" - "FF")

**Tabelle 7.10 – Inhalt der Zeichensatzspeicher**

Es gibt vier Zeichensatzspeicher zu je 128 Zeichen, die mit G0, G1, G2 und G3 bezeichnet werden. Der 8-Bit umfassende Bereich der Zeichenkodierungen wird vom VT100-Emulator in zwei Teile unterteilt:

- Der GL Bereich wird den Kodierungen 0-127 ("00" bis "7F") zugeordnet,
- der GR Bereich den Kodierungen 128-255 ("80" bis "FF").

Liegt die Kodierung eines ausgegebenen Zeichens zwischen 0 und 127, dann wird das Zeichen in dem Zeichensatz dargestellt, der dem GL Bereich zugeordnet ist; bei einer Kodierung zwischen 128 und 255 ist es der dem GR Bereich zugeordnete Zeichensatz.

#### Voreinstellung des Zeichensatzes:

- Zeichensatzspeicher G0 und G2 mit dem nationalen Zeichensatz ISO 646 Deutschland (CV+39 = "02").
- Zeichensatzspeicher G1 und G3 bekommen den oberen Bereich von "80" bis "FF" des ISO 8859-1 8-Bit Code zugewiesen.

#### Voreinstellung der Zuordnung von GL und GR:

- GL Bereich wird dem Zeichensatzspeicher G0 zugeordnet.
- GR Bereich wird dem Zeichensatzspeicher G1 zugeordnet.

### 7.3.1 Statische Änderung des Zeichensatzes

Eine statische Änderung des Zeichensatzes kann auf zwei Arten durchgeführt werden:

- Im Setupmenü „Zeichensatz“ den Zeichensatz auswählen. Wenn das Setup über Reset verlassen wird, steht er zur Verfügung.
- In CV+39,1 den gewünschten Zeichensatz einstellen und mit einem anschließenden Reset im TCL übernehmen.

#### Beispiel:

Ausgaben in kyrillischer Schrift mit TCL-Anweisungen einstellen:

```
DK,"06",CV+39:  
DWR,$1,'R':
```

### 7.3.2 Temporäre Änderung des Zeichensatzes

Die vier 128 Byte Zeichensatzspeicher G0, G1, G2 und G3 können mit folgenden Steuersequenzen mit Zeichensätzen geladen werden:

Zeichensatz	in G0	in G1	in G2	in G3
ISO 8859 (Kodierung "00" - "7F") US-ASCII	ESC ( B	ESC ) B	ESC * B	ESC + B
ISO 8859-1 (Kodierung "80" - "FF")	ESC ( <	ESC ) <	ESC * <	ESC + <
ISO 8859-2 (Kodierung "80" - "FF")	ESC ( H	ESC ) H	ESC * H	ESC + H
ISO 8859-5 (Kodierung "80" - "FF")	ESC ( G	ESC ) G	ESC * G	ESC + G
ISO 8859-15 (Kodierung "80" - "FF")	ESC ( I	ESC ) I	ESC * I	ESC + I
ISO 646 Großbritannien	ESC ( A	ESC ) A	ESC * A	ESC + A
ISO 646 Deutschland	ESC ( C	ESC ) C	ESC * C	ESC + C
ISO 646 Frankreich	ESC ( D	ESC ) D	ESC * D	ESC + D
ISO 646 Spanien	ESC ( E	ESC ) E	ESC * E	ESC + E
ISO 646 Norwegen	ESC ( F	ESC ) F	ESC * F	ESC + F
Semigrafik	ESC ( 0	ESC ) 0	ESC * 0	ESC + 0

**Tabelle 7.11 – Steuersequenzen zum Laden der Zeichensatzspeicher**

Der GL und der GR Bereich lassen sich mit Steuersequenzen den Zeichensatzspeichern G0 bis G3 zuordnen:

Zuordnung	Steuersequenz
GL zu G0	"0F" (SO)
GL zu G1	"0E" (SI)
GL zu G2	ESC n
GL zu G3	ESC .
GR zu G1	ESC ~
GR zu G2	ESC }
GR zu G3	ESC :

**Tabelle 7.12 – Steuersequenzen zur Zuordnung von GL und GR**

Falls GL nur für das nächste Zeichen zu G2 oder G3 zugeordnet werden soll, kann man die Steuersequenzen

ESC N                    ordne GL G2 für das nächste Zeichen zu

ESC O                    ordne GL G3 für das nächste Zeichen zu

verwenden. Nach der Ausgabe des Zeichens wird die temporäre Zuordnung rückgängig gemacht.



Die Änderung der Inhalte der Zeichensatzspeicher G0 bis G3 ist erheblich aufwendiger, als den GL oder GR Bereich einem anderen Zeichensatzspeicher zuzuordnen.

Zeichensatzspeicher sowie GL und GR Zuordnungen werden zurückgesetzt, wenn P20+23 beschrieben wird. Dies ist z.B. bei der Reset-Anweisung R,V: der Fall, die auch in den zusammengesetzten Reset-Anweisungen R,I: und R,S: ausgeführt wird. Siehe Abschnitt 5.27.

### 7.3.3 Beispiel 1: Ausgaben mit mehreren Zeichensätzen

Das deutsche Wort „Danke“ wird in verschiedenen Sprachen ausgegeben. Zunächst werden alle benötigten Zeichensätze in die Zeichensatzspeicher G0 bis G3 geladen:

G0: ISO 8859 (Kodierung "00" - "7F")

G1: ISO 8859-1 (Kodierung "80" - "FF")

G2: ISO 8859-2 (Kodierung "80" - "FF")

G3: ISO 8859-5 (Kodierung "80" - "FF")

Im weiteren Ablauf des Programms wird der GR Bereich dem jeweils benötigten Zeichensatz zugeordnet.

```
D#0:  
DK,<"1B"&'(B')>,D+1000:  
DK,<"1B"&')<'>,D+1000:  
DK,<"1B"&'*H'>,D+1000:  
DK,<"1B"&'+G'>,D+1000:  
DF,D,/D/, :  
DK,'DE: Danke',D[1]:  
DK,'FR: merci',D[1]+20:  
DK,<'ES: '&"1B"&'~'&"A1"&'gracias!'>,D[2]:  
DK,<'PL: '&"1B"&' }'&'dzi'&"EA"&'kuj'&"EA">,D[2]+20:  
DK,<'RU: '&"1B"&' :'&"E1DFD0E1D8D1DE">,D[3]:  
DK,<'CZ: '&"1B"&' }'&'D'&"EC"&'kuji'>,D[3]+20:  
DS:
```

Ausgabe auf dem Display:

DE: Danke	FR: merci
ES: ¡gracias!	PL: dziękuje
RU: спасибо	CZ: Děkuji

### 7.3.4 Beispiel 2: Ausgaben im ISO 8859-1 8-Bit Code mit Semigrafik

In einem Dialog-Kasten werden die Zeichen '[]{}|\~' und 'ÄÖÜäöüß' dargestellt. Dazu wird der volle ISO 8859-1 8-Bit Code verwendet.

Die unteren 128 Byte des ISO 8859-1 Zeichensatzes (Kodierung "00" – "7F", entspricht dem US-Zeichensatz) werden nach G0 geladen. Die oberen 128 Byte des ISO 8859-1 Zeichensatzes (Kodierung "80" – "FF") werden nach G1 geladen.

Der GL Bereich wird dem G0-Speicher zugeordnet (ASCII-Steuerzeichen SO "0F") und GR dem G1-Speicher (ESC ~).

Hinweis: Diese Einstellung kann auch über das Setup erreicht werden: Setupmenü „Zeichensatz“ und Auswahl „Zeichensatz:ISO 8859-1“. Setup über Reset verlassen.

Weiterhin wird der Dialog-Kasten mit Hilfe des Semigrafik-Zeichensatzes erzeugt. Dazu wird der Semigrafik-Zeichensatz nach G2 geladen. Vor dem Zeichnen des Kastens wird mit Hilfe der Steuersequenz ESC n, der GL Bereich G2 zugeordnet. Nach dem Zeichnen des Kastens wird GL wieder auf G0 zurückgeschaltet.

Das beschriebene Beispiel wird durch folgendes kleines TCL-Programm verdeutlicht. Es arbeitet nur auf einem grafikfähigen Display korrekt:

```
D#0:  
DK,'5',P20+23:F,D,/D/, :  
DK,<"1B"&'(B'&"1B"&')<'&"1B"&'*0'&"1B"&'~>,D+1000:  
DK,<"1B"&'[1;5H"&"1B"&'n'&'1qqqqqqqqk'&"0F">,D+1000:  
DK,<"1B"&'[2;5H"&"1B"&'n'&'x'&"0F"&' []{}|\~ '&"1B"&'n'&'x'&"0F">,D+1000:  
DK,<"1B"&'[3;5H"&"1B"&'n'&'x'&"0FC4D6DCE4F6FCDF"&' '&"1B"&'n'&'x'&"0F">,D+1000:  
DK,<"1B"&'[4;5H"&"1B"&'n'&'mqqqqqqqqj'&"0F">,D+1000:  
DS:
```

## 7.4 Doppelthohe und -breite Darstellung

Bei den Grafikdisplays lassen sich Zeilen in doppelt hoher und/ oder doppelt breiter Zeichenbreite darstellen. Dies kann mit Hilfe der VT100 Steuersequenzen

ESC # 3	obere Zeile, doppelt hoch und breit
ESC # 4	untere Zeile, doppelt hoch und breit
ESC # 5	Zeile in normaler Schriftgröße
ESC # 6	Zeile doppelt breit

pro Zeile eingestellt werden. Die Steuersequenzen wirken in der Zeile, in der der Cursor positioniert ist. Normalerweise muss ein Text in einer doppelt hohen und breiten Zeile sowohl in der oberen als auch in der unteren Zeile ausgegeben werden, damit diese einen kompletten, doppelt hohen und breiten Text ergeben. Da dieses Vorgehen, insbesondere im Zusammenspiel mit der Aktualisierungsanweisung A (Abschnitt 5.5), umständlich ist, ist für das INTUS eine weitere Steuersequenz

ESC # 7

definiert worden, die den VT100 Emulator in einen Modus versetzt, in dem er beim Beschreiben einer unteren, doppelt hohen und breiten Zeile zugleich die darüber befindliche, obere Zeile auch mit aktualisiert. Man beachte, dass dieser Modus mit der nächsten ESC#5 Sequenz wieder ausgeschaltet wird.

### Beispiel:

Das folgende Programmfragment gibt die Uhrzeit doppelt hoch und doppelt breit in den oberen beiden Zeilen des Displays aus.

```
DK,<"1B"&'#7'&"1B"&'#3'>,D[0]:K,<"1B"&'#4'>,D[1]:  
DK,UR,D[1]:A:
```

## 7.5 Anzeige- und Ausgabeseite

Die VT100 Emulation verfügt über mehrere, unabhängige Speicherbereiche, die am Display zur Anzeige gebracht werden können. Diese Bereiche werden Seiten genannt. Dabei kann in einer Seite ein Dialogschritt aufgebaut werden, während eine andere Seite angezeigt wird. Nach Aufbau der Seite kann diese dann umgeschaltet werden. Auf diese Weise lässt sich auch das Flackern beim Aufbau der Anzeige vermeiden.

Im TCL-System ist die Seite 1 für das TCL-Programm reserviert, während Seite 2 nur für die Ausgaben des Setup benutzt wird. Da das Setup nach einer Ausgabe fest die Seite 1 als Ausgabeziel einstellt, so dass die laufende TCL-Ausgabe wieder dorthin gelangt, sollte kein TCL Programm eine andere Ausgabeseite verwenden, selbst wenn diese im VT100 Emulator für den jeweiligen Displaytyp vorhanden sind.

Der Vollständigkeit halber werden hier die Steuersequenzen für die Seitensteuerung angegeben:

ESC [ = <n> q	Seite <n> wird durch den folgenden Ausgabestrom beschrieben.
ESC [ = <n> r	Seite <n> wird angezeigt.

## 7.6 Steuersequenzen für Bit-Grafik

Zur Ausnutzung der Grafikfähigkeiten der beiden Grafikdisplays 240x64 Pixel und 320x240 Pixel wurden zwei ESC-Sequenzen für die Ausgabe von Bitmustern auf das Display definiert.

### 7.6.1 Ausgabe von Bit-Grafik an Zeichenpositionen

ESC [=<Bit-Muster>{ ; <Bit-Muster>}]\*S

Die Sequenz erlaubt Bitmuster auf das Display auszugeben.

<Bit-Muster> kann eine 6-Bit, 8-Bit, 12-Bit und 16-Bit große Dezimalzahl sein.

#### 240x64 Pixel Display:

Die Anzahl der Bits hängt vom Betriebsmodus des 240x64 Grafikdisplays ab, der sich über das P20+23,1 Feld einstellen lässt (Abschnitt 7.1.2 und Abschnitt 4.32, Tabelle 3).

8 Bit sind es bei einer Zeilen/Spaltenzahl von 4x30 oder 8x30.

6 Bit sind es bei einer Zeilen/Spaltenzahl von 4x40 oder 8x40.

#### 320x240 Pixel Display:

Beim 320x240 Grafikdisplay werden immer 8 Bit breite Muster verwendet.

Die doppelte Breite von 12 bzw. 16 Bit muss verwendet werden, wenn die Zeile über eine ESC#6, ESC#3 oder ESC#4 Sequenz auf eine doppelt-breite und/ oder doppelt-hohe Darstellung umgestellt wurde.

Über ';' abgetrennt lassen sich <Bit-Muster> bis zu 16 Mal wiederholen. Dabei wird die Zeichen-Matrix an der aktuellen Cursorposition überschrieben. Sinnvollerweise wird <Bit-Muster> also 8 bzw. 16 Mal wiederholt, abhängig von der Anzahl der Scanlines des Standardzeichens im jeweiligen Displaymodus.

Um das D-Feld der Anzeige nicht mit langen ESC-Sequenzen zu "verschmutzen" mit lästigen Seiteneffekten, wie Stillstand der Aktualisierung, sollte beim INTUS 3000 eine ESC-Sequenz im D-Feld außerhalb des sichtbaren Bereichs ausgegeben werden. Um eine solche Ausgabe zu erlauben, muss die Sequenz mit einem ESC-Zeichen ("1B") beginnen. Sinnvollerweise beginnt eine Sequenz immer mit einer Positionierung.

Man beachte, dass die angezeigten Bitmuster derzeit bei einem Scroll des Bildschirms verschwinden, weil nur die Zeichen einer Zeile neu angezeigt werden. Bei einem Wechsel in den Setup (Ausgaben auf Seite 1) bleibt das Bitmuster jedoch erhalten.

#### Beispiel:

Im Displaymodus 4x30 bei einer Zeile mit doppelt hoher und breiter Darstellung zeichnet folgende ESC-Sequenz ein 16 Scanlines großes Dreieck in die untere rechte Ecke des Displays:

DK,<"1B5B"&'4;15H'&"1B5B"&'=1;3;7;15;31;63;127;255;511

D;1023;2047;4095;8191;16383;32767;65535s',D+1000:

## 7.6.2 Ausgabe von Bit-Grafik an Pixelpositionen

Die oben beschriebene 's' ESC-Sequenz erlaubt nur eine Ausgabe an Zeichenpositionen. Die folgende 'S' ESC-Sequenz erlaubt eine Ausgabe an Pixelpositionen:

`ESC [=<bit_line>;<bit_offset>;<bits>{;<bits>}*S`

schreibt 8-Bit oder 6-Bit Werte `<bits>` an der angegebenen Position `<bit_offset>` in die Y-Richtung aufsteigend (im Display nach unten), angefangen von der Y-Position `<bit_line>`.

### 240x64 Pixel Display:

Die Anzahl der Bits hängt vom Betriebsmodus des 240x64 Grafikdisplays ab, der sich über das P20+23,1 Feld einstellen lässt (Abschnitt 7.1.2 und Abschnitt 4.32, Tabelle 3).

8 Bit sind es bei einer Zeilen/Spaltenzahl von 4x30 oder 8x30.

6 Bit sind es bei einer Zeilen/Spaltenzahl von 4x40 oder 8x40.

Beim 240x64 Grafikdisplay kann `<bit_offset>` einen Wert von 0 bis 232 (eventuell auch 234) haben, `<bit_line>` den Wert 0 bis 63.

### 320x240 Pixel Display:

Beim 320x240 Pixel Grafikdisplay werden immer 8 Bit große Muster verwendet. `<bit_offset>` hat dabei einen Wert von 0 bis 312 und `<bit_line>` einen Wert von 0 bis 239.

Die Schreiboperation ist ein "XOR" (exklusives Oder) mit der im Display befindlichen Information an der angegebenen Position. Ein Bit 0 im Bitmuster lässt das im Display befindliche Pixel unverändert, ein Bit 1 invertiert das Pixel. Ist das Display an der Schreibstelle gelöscht (alle Pixel enthalten 0), dann wird das Bitmuster in der ESC-Sequenz unverändert geschrieben (Beispiel siehe nächste Seite).

### **Beispiel:**

Das folgende Programmfragment invertiert das Beispiel eines Dreiecks vom letzten Abschnitt 7.6.1.

Man beachte, dass man durch Verlängerung der ESC-Sequenzen weniger Sequenzen benötigt. Die maximale Länge einer ESC-Sequenz sollte 128 Zeichen nicht überschreiten.

```
DK,<"1B5B"&'=32;208;255;255;255;255;255;255;255;255S'>,D+1000:  
DK,<"1B5B"&'=40;208;255;255;255;255;255;255;255;255S'>,D+1000:  
DK,<"1B5B"&'=48;208;255;255;255;255;255;255;255;255S'>,D+1000:  
DK,<"1B5B"&'=56;208;255;255;255;255;255;255;255;255S'>,D+1000:  
DK,<"1B5B"&'=32;216;255;255;255;255;255;255;255;255S'>,D+1000:  
DK,<"1B5B"&'=40;216;255;255;255;255;255;255;255;255S'>,D+1000:  
DK,<"1B5B"&'=48;216;255;255;255;255;255;255;255;255S'>,D+1000:  
DK,<"1B5B"&'=56;216;255;255;255;255;255;255;255;255S'>,D+1000:  
DK,<"1B5B"&'=32;224;255;255;255;255;255;255;255;255S'>,D+1000:  
DK,<"1B5B"&'=40;224;255;255;255;255;255;255;255;255S'>,D+1000:  
DK,<"1B5B"&'=48;224;255;255;255;255;255;255;255;255S'>,D+1000:  
DK,<"1B5B"&'=56;224;255;255;255;255;255;255;255;255S'>,D+1000:  
DK,<"1B5B"&'=32;232;255;255;255;255;255;255;255;255S'>,D+1000:  
DK,<"1B5B"&'=40;232;255;255;255;255;255;255;255;255S'>,D+1000:  
DK,<"1B5B"&'=48;232;255;255;255;255;255;255;255;255S'>,D+1000:  
DK,<"1B5B"&'=56;232;255;255;255;255;255;255;255;255S'>,D+1000:
```

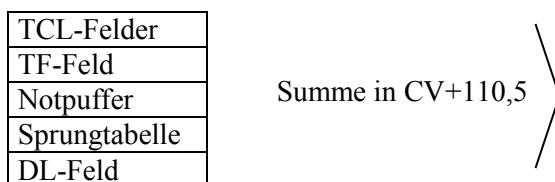


## 8 TCL-Programmentwicklung

### 8.1 Speicheraufteilung

Bevor man ein TCL Programm erstellt oder in das INTUS 3000 Terminal lädt, sollte man sich über die Aufteilung des Speichers im Klaren sein. Eine Änderung der Speicheraufteilung bewirkt einen Kaltstart, wonach das TCL-Programm wieder geladen werden muss.

Der im Terminal verfügbare Speicher kann in den Downloadbereich, DL, das Tabellenfeld, TF, den Notpuffer, \$4, und eine interne Sprungzieltabelle aufgeteilt werden. Die Größe der letzten drei Speicherkomponenten lassen sich per Setup oder im CV-Feld einstellen. Der Downloadbereich belegt den verbleibenden Speicher nach Abzug des Speicherbedarfs dieser drei Komponenten und der TCL-Felder. Rechnen Sie für TCL 6 mit 13 kB Speicher für die TCL-Felder.



*Abbildung 8.1 - Speicheraufteilung*

Zunächst einmal sollte man sich über die Anzahl der benötigen Sprungziele klar werden. Die Voreinstellung von 1024 Sprungzielen kann bei „Modul“-weiser Vergabe von Sprungziel-Bereichen zu klein sein. Per Setup unter dem Punkt **TCL:Label-Anzahl** oder in CV+63,1 kann die Anzahl der Sprungziele bis auf 4352 erhöht werden. Jedes Sprungziel benötigt 4 Byte Speicherplatz.

Die für das Tabellenfeld benötigte Größe hängt von der Anzahl der darin zu speichernden Datensätze, Steuerparameter und Ausgabetexte ab. Bei bekannter Anzahl von Datensätzen und bekannter Datensatzgröße lässt sich ein Mindestbedarf an Speicherplatz für das TF-Felds berechnen. Alle anderen internen Steuerfelder und Texte, die dort auch gespeichert werden, müssen dann noch dazugerechnet werden. Die Größe des TF-Felds stellt man im Setup unter **TCL:Tabellenfeld** oder in CV+6,1/CV+108,1 ein. Die Voreinstellung ist der Zahlenwert 16, was eine Größe von 48kB ergibt. Bis zur TCL Version 5.01 konnte des Tabellenfeld maximal 765kB groß sein. Jetzt ist die Größe des verfügbaren Speichers das Limit. Jedoch macht ein TF-Feld Größe von mehr als 1MB nur Sinn, wenn TF-Feld Offsets mit 8 Byte Größe und das SP-Feld mit 8 Byte Größe konsistent durch das ganze TCL Programm verwendet wird (siehe P20+38,1).

Der Speicherbedarf des Notpuffers berechnet sich aus der Anzahl der zu speichernden Sendedatensätze zum Host und der dazugehörigen Satzlänge. Die Anzahl der Sendesätze bestimmt sich aus Vorgaben wie z.B. Anzahl der Buchungen über einen Zeitraum, in dem der Leitrechner ausfallen darf. Die Größe des Notpuffers stellt man im Setup unter **TCL:Notpuffer** oder in CV+5,1/CV+107,1 ein. Die Voreinstellung ist der Zahlenwert 16, was eine Größe von 48kB ergibt. Vor TCL Version 5.01 konnte der Notpuffer maximal 765kB groß sein. Derzeit ist nur der verfügbare Speicher das Limit.

Für größere TCL Programme sollte als Downloadfeld noch 30-40kB Speicher übrig sein.

Nach einer Einstellung der Setup-Parameter im Setup-Mechanismus muss das Setup über den Punkt **Reset:Ja** verlassen werden, damit die neuen Einstellungen wirksam werden. Wenn die Einstellung über das CV-Feld erfolgt, dann muss ein Terminal-Reset über den MONIN-

Datensatz mit Adressbyte 'R' erfolgen (siehe Hinweise im Abschnitt 4.6 *CV Setup*). Der Anlaufmodus (**Setup:Anlaufmodus** bzw. CV+8,1) sollte auf Warmstart stehen.

Nach dem Wiederhochlaufen des Terminals sollte überprüft werden, ob das Terminal die neu eingestellten Werte akzeptiert hat. Einstellungen, die einen Speicherbedarf über den verfügbaren Platz hinaus ergeben, führen beim INTUS 3000 dazu, dass das System wieder die Voreinstellungen einsetzt.

Als nächstes sollte die Größe des DL-Felds überprüft werden. Dies kann etwa mit der Anweisung

**IF,D,/D/, :AD,0,/DL/,D:**

geschehen, die die Größe des DL-Felds ins Display ausgibt. Größere TCL-Programme benötigen ca. 30-40kB Speicherplatz. Dies ist jedoch stark abhängig von der Länge und Anzahl der darin vorhandenen Texte. Kommentare benötigen keinen Speicherplatz.

Nach dem Herunterladen eines TCL-Programms kann der verbliebene Platz im DL-Feld mit der Anweisung

**IF,D,/D/, :AD,0,/@/,D:**

ermittelt werden. Die Größe des übersetzten TCL-Programms wird durch

**IF,D,/D/, :SB,/DL/,/@/,D:**

ausgegeben.

Für ein Programm, das beim Kaltstart verwendet werden soll, und deshalb im EEPROM abgespeichert werden muss, stehen ca. 10kB zur Verfügung. Sollte mehr Platz benötigt werden, dann sollten Sie sich mit PCS in Verbindung setzen.

## 8.2 **TCL-Programm schreiben**

Ein TCL-Programm ist eine ASCII-Datei, die an jedem Rechner mit einem beliebigen Editor erstellt werden kann. Das TCL-Programm wird als Datei vom Leitrechner über die Kommunikationsschnittstelle in das Terminal geladen.

Der lexikalische Aufbau eines TCL-Programms ist bereits im Abschnitt 3.1 beschrieben.

Die Konzepte der TCL-Sprache sind in den Abschnitten

- Datenfluss
- Reaktion auf Ereignisse
- Parallele Vorgänge

in der Einleitung dieses Handbuchs dargelegt und sollten an dieser Stelle des Handbuchs nochmals in Erinnerung gerufen werden.

Das folgende kleine Beispiel zeigt den Grundaufbau eines TCL-Programms:

**Beispiel:**

1. IR,S:
2. D#0:T1,@20:
3. D#10:K,'Eingabe: \_\_\_\_ ',D:E,(D+9,4N,@11):S:
4. D#11:SR,D+9,4:@10:
5. D#20:SR,UR:T6000,@20:S:
6. I@0:

### Erklärung:

Das jeweils erste Zeichen jeder Zeile ist das Adressbyte, das vom MONIN-Prozess ausgewertet wird und angibt, was mit dem jeweiligen Datensatz zu geschehen hat. 'T' bezeichnet eine Anweisung für den Interpreter, die sofort ausgeführt wird. 'D' bezeichnet eine TCL-Programmzeile für den Programmreich.

1. R,S: löscht Eingabefreigaben, Zeitüberwachungen, Ringpuffersprünge, deaktiviert DI-Sprünge und zugehörige Zählerereignisse und löscht die TCL-Felder, bis auf wenige Ausnahmen. Schließlich wird der Programmstartzeiger an den Anfang des Programmreichs gesetzt. (Siehe Abschnitt 5.27 *R Reset*).
2. #0 ist ein Sprungziel, auf das von anderen Stellen im Programm verzweigt werden kann. Das Sprungziel #0 sollte in jedem Programm vorhanden sein. Es wird automatisch angesprungen, wenn das Terminal nach dem Einschalten oder nach einem Terminal-Reset im Warmstart anläuft. Deshalb werden in dieser Zeile Initialisierungen durchgeführt. Die Anweisung T1,@20: setzt eine Zeitüberwachung auf. Der ausführende Timeout-Prozess soll 100 Millisekunden warten und dann eine Sprunganweisung (Sprung an die Adresse #20) in den Interpreter-Anweisungspuffer \$3 eintragen.
3. Die Eingabeschleife für die Tastatureingabe besteht aus den Sprungzielen #10 und #11. In #10 wird eine Eingabe angefordert, die im Display angezeigt wird. Mit der Kopieranweisung K wird "Eingabe: \_\_\_\_" in das Display D kopiert, und anschließend die Eingabe für 4 Ziffern freigegeben. Nach der Stopanweisung S: wird auf Ereignisse gewartet, im Beispiel auf die Tastatureingabe und den Timeout.
4. Wenn eine vollständige Eingabe erfolgt ist, führt der Interpreter den Sprung nach #11 aus. Die eingegebenen Daten werden über den Sendepuffer Hostschnittstelle \$2 an den Host gesendet. Anschließend erfolgt der Sprung nach #10, wo wieder eine neue Eingabe aktiviert wird.
5. Wenn der Timeout von #0 abgelaufen ist, erfolgt der Sprung nach #20. Dort wird als "Lebensmeldung" die Uhrzeit an den Leitrechner gesendet. Diese Lebensmeldung wird alle 10 Minuten ( $6000 * 0.1$  Sekunden) wiederholt, indem der Timeout wieder aktiviert wird. Nach der Stopanweisung wird wieder auf Eingabe und Timeout gewartet.
6. @0: startet das Programm vom Host aus, sofort nachdem es geladen wurde.

Die zwei Anweisungsfolgen #10 zusammen mit #11 und #20 laufen völlig unabhängig voneinander ab und werden nur abgearbeitet, wenn das jeweilige Ereignis eingetreten ist. Die Ereignisse unterbrechen den Interpreter nicht. Erst nach einer Stopanweisung wartet der Interpreter auf Anweisungen im Ringpuffer \$3 und findet dann die Ereignissprünge vor.

### Unterschiedliche Terminalmodelle:

TCL-Programme können so geschrieben werden, dass sie auf allen INTUS 3000 Modellen laufen. Dies ist durch folgende zwei Eigenschaften möglich:

- Es wird keine Fehlermeldung erzeugt, wenn eine nicht vorhandene Hardwarekomponente (Funktionstaste, Lampe, Hupe usw.) angesprochen wird. Es erfolgt auch sonst keine Reaktion.
- Über das Feld CV, beschrieben in Abschnitt 4.6, ist es möglich, die INTUS 3000 Terminalkonfiguration abzufragen. So kann im Programm entsprechend der vorhandenen Hardware verzweigt werden.

**Verwendung des TCL Präprozessors:**

Wenn ein großes TCL Programm erstellt wird, sollte der TCL Präprozessor TCLPRE verwendet werden. Der Präprozessor unterstützt die Aufteilung des TF-Felds in logische Datensätze und Tabellen. Außerdem ermöglicht er symbolische Sprungziele und führt eine Syntaxprüfung durch, bevor das Programm in das Terminal geladen wird.

## **8.3   TCL-Programm laden**

Zunächst müssen die Protokollparameter für die Hostschnittstelle, wie z.B. Datenformat und Baudrate, im Setup oder im CV-Feld eingestellt werden. Anschließend ist ein Terminal-Reset durchzuführen. Das Defaultprogramm im Terminal sendet '77' an den Rechner. Danach kann ein TCL-Programm geladen und gestartet werden.

### **8.3.1   Laden in den Programmbereich**

Datensätze, die mit dem Adressbyte 'D' beginnen, werden vom MONIN-Prozess in den Programmbereich DL geladen.

Der MONIN-Prozess, der für den Empfang der Daten vom Rechner zuständig ist, enthält einen Ladecompiler, der jeden Datensatz auf syntaktische Richtigkeit überprüft und alle TCL-Anweisungen in einen speziellen Zwischencode kompiliert. Dieser Code wird lückenlos im Programmbereich gespeichert.

Ist im Programmbereich nicht mehr genügend Platz, so wird jeder weitere Datensatz verworfen und die Fehlermeldung "DE<CR>" (Download Error) zum Rechner gesendet.

Das Übersetzen von jedem Datensatz beim Laden benötigt Rechenzeit. Deshalb verzögert sich die Datenübertragung zum Terminal. Dies wirkt sich auf Protokollebene so aus, dass das Terminal häufig XOFF (TTY) oder WACK (BSC) sendet, und die Datensätze nicht mit der erwarteten Geschwindigkeit geladen werden.

Beim TTY-Protokoll sollte am Rechner die Übertragung so eingestellt sein, dass er auf empfangene XOFF-Zeichen reagiert. Im anderen Fall erhalten Sie im Terminal die Fehlermeldung "Receive Buffer Overrun", da Zeichen verloren gehen.

**Programm-Ladezeiger**

Datensätze mit TCL-Anweisungen können jederzeit an das Terminal gesendet werden. Die Anweisungen werden immer ab dem aktuellen Stand des Ladezeigers geladen. Alte Anweisungen werden nicht überschrieben, es sei denn, der Zeiger wird zurückgesetzt. Dies kann durch folgende Aktionen geschehen:

- Kaltstart nach Terminal-Reset
- TCL-Anweisung R,D:
- TCL-Anweisung R,S: (enthält die TCL-Anweisung R,D:)

**Doppeltes Sprungziel**

Wird ein Sprungziel mehrfach vergeben, wird keine Fehlermeldung erzeugt, sondern nur die zuletzt geladene Programmzeile wird verwendet.

Die Mehrfachverwendung von Sprungzielen sollte vermieden werden, da auch die eventuell nicht mehr ansprechbaren Anweisungen Platz im Speicher belegen.

### Overlay-Technik

Der Programmablezeiger kann mit der TCL-Anweisung R,D{,<Label>}: zum Nachladen auf das Sprungziel <Label> gesetzt werden. Das Sprungziel muss im Programmreich vorhanden sein. Das Nachladen erfolgt im unmittelbaren Anschluss an das Sprungziel. Das Sprungziel selbst muss auch nachgeladen zu werden.

### Löschen eines geladenen Programms

Vor dem Download eines neuen TCL Programms sollte der Interpreter und der Ladezeiger mit der Reset-Anweisung IR,S: zurückgesetzt werden.

### 8.3.2 Laden des Defaultprogramms

Datensätze, die mit dem Adressbyte 'E' beginnen, werden vom MONIN-Prozess in den EEPROM-Programmbereich geladen.

Ein EEPROM ist ein Speicherbaustein, der die in ihn geladenen Daten auch dann speichert, wenn keine Stromversorgung (weder Netz noch Batterie) vorhanden ist. Dadurch ist das geladene Programm immer verfügbar.

Ein Programm, das in das EEPROM geladen werden soll, muss mit dem Sprungziel "#0:" beginnen, und es muss in aufeinander folgenden MONIN-Datensätzen geladen werden.

Ein geladenes EEPROM-Programm kann nicht gelöscht werden, es kann nur durch ein Neues überladen werden.

Da der EEPROM Speicherbaustein nur begrenzt oft beschrieben werden darf, wird empfohlen, ein Programm erst vollständig zu testen, bevor es in das EEPROM geladen wird.

### 8.3.3 Defaultprogramm

Werksseitig wird folgendes TCL-Programm in den EEPROM-Bereich geladen:

```
E#0:UN,CV+67,"04",H1:P,H1,<"00",@99:  
E#1:  
ERS,$7,1,@87:  
EK,'PCS Defaultprogramm v5.1',D:K,'0080',PO+1:T20,@80:S:  
E#80:F,D,40,:@81+(PO,1):S:  
E#81:T1,@83:S:  
E#82:P,CV+25,="0A",@89:  
EK,'Warten auf Rechnerverbindung',D:S:  
E#83:SR,'77':K,'Ladeanforderung ',D:T50,@84:S:  
E#84:F,D,40,:K,'Nicht bereit ',D:  
EE,(F0,@85/F1,@86):  
EP,CV+25,="0A",@90:K,UR,D[1]+32:A:S:  
E#85:F,D,80,:K,'DL= ',D:AD,0,/DL/,D+4,8:T20,@84:S:  
E#87:R,F,$7:RS,$7,1,@87:S:  
E#86:P,CV+28,<"04",@84:K,'Modembetrieb ',D:K,UR,D+16:A:  
EK,'6',MI+6:SR,'AT':T30,@88:S:  
E#88:F,D,80,:K,'0',MI+6:@84:  
E#89:K,'Warten auf Rechner-',D[0]:K,'verbindung',D[1]:S:  
E#90:K,UR,D[1]+12:A:S:  
E#99:@1:
```

## 8.4 TCL-Programm starten

Es gibt zwei Möglichkeiten ein TCL-Programm zu starten, die in den folgenden beiden Abschnitten beschrieben werden.

### 8.4.1 Programmstart vom Leitrechner

Die Anweisungen im Programmreich werden erst dann ausgeführt, wenn der Interpreter eine Sprunganweisung mit einem Sprungziel in den Programmreich erhält.

Dies kann nach dem Laden mit einer Interpreteranweisung (Datensatz mit MONIN-Adressbyte 'I') vom Host

I@<Label>:

geschehen. Üblicherweise beginnt ein Programm bei Sprungziel #0, d.h. I@0:

### 8.4.2 Programmstart nach Terminal-Reset

Es gibt drei Aktionen, die ein Terminal-Reset auslösen können:

- vom Leitrechner durch einen Datensatz mit MONIN-Adressbyte 'R'.
- im Setup durch **Reset:Ja**
- durch Aus- und Einschalten der Netzversorgung

Die nach einem Reset ausgeführten Aktionen sind nicht davon abhängig, wie der Reset ausgelöst wurde. Sie werden durch den im Setup eingestellten Anlaufmodus (CV+8,1) bestimmt.

### 8.4.3 Anlaufmodi: Warm-, Kalt- , Neu- Eiskalt- und Comstart

#### 8.4.3.1 Warmstart

Beim Einschalten bzw. beim Reset des Terminals wird üblicherweise das geladene TCL-Programm aktiviert, wobei dieses auf die weitgehend unveränderten TCL-Felder zugreift, die sich im batteriegepufferten Speicher befinden. Die Reaktivierung des Programms geschieht durch einen Sprung an das Sprungziel #0. Diesen Vorgang nennt man Warmstart.

Ein Warmstart wird unter folgenden Bedingungen durchgeführt:

- Die Speicheraufteilung, TF-Feldgröße und Notpuffer-Größe, wurde nicht geändert (Setup oder CV+6,1 mit CV+108,1 und CV+5,1 mit CV+107,1).
- Die Batterien haben genug Ladung, um den Erhalt der Daten im (PCMCIA) SRAM zu garantieren.
- Es ist ein TCL Programm geladen, das ein Sprungziel #0 enthält.
- Der Parameter Anlaufmodus ist per Setup oder CV+8,1 auf Warmstart "00" gesetzt.

Bei einem Warmstart bleiben alle TCL-Felder auf dem Zustand vor dem Reset bzw. Einschalten, bis auf folgende Ausnahmen:

- P2 wird auf '0' gesetzt.
- Die Leserfelder M, I, B werden gelöscht.
- Die Sx-Felder werden aktualisiert.
- Das Display und das D-Feld werden mit Leerzeichen gefüllt, der Cursor wird unsichtbar geschaltet und in die linke obere Ecke positioniert.

Dies gilt auch für die Displays der abgesetzten Leser.

- Die lokalen LEDs und Ausgänge werden entsprechend der Werte in den dazugehörigen Ox- und Lx-Feldern geschaltet.

Die LEDs und digitalen Ausgänge der abgesetzten Leser werden nicht gesetzt.

Im Laufe der Reinitialisierung nach dem Sprungziel #0 wird das TCL-Programm das P2-Feld auf '1' setzen, um die DI-Sprünge zu aktivieren. Weiterhin müssen die Eingaben und alle notwendigen Zeitüberwachungen sowie Ringpuffer-Sprünge wieder aktiviert werden.

Es wird versucht, das Prozedurstatusflag PO zu ermitteln. Alle Ringpuffer sind zunächst leer; der Notpuffer enthält alle nicht quittierten Datensätze. Der MONOUT-Prozess beginnt mit dem Senden der Datensätze, wenn das Betriebsstatusflag P3,1 kleiner als '3' ist.

#### **8.4.3.2 Kaltstart und Neustart**

Wenn kein Warmstart möglich ist oder der Anlaufmodus auf Kaltstart, "01", oder Neustart, "02", eingestellt ist, dann wird ein Kaltstart durchgeführt. Das TCL-Programm wird entfernt und alle Felder werden zu '0' gelöscht, bis auf die Felder

MI	MONIN-Steuerfeld
P10	MONOUT-Steuerfeld
P20+2,2	Datenübertragungsmodus der Ringpuffer
P20+23,1	Display-Umschaltung
P21,1	Setupsteuerung
UR+2,1 und UR+5,1	Trennzeichen der Uhrzeit
CE	Ersetzungszeichen
TR	Trace

die auf die Voreinstellungen gesetzt werden. Damit werden die Ausgänge Ox und Leuchtdioden Lx auch auf inaktiv geschaltet.

Die logische Satznummer, die der MONOUT-Prozess gegebenenfalls für die Sendedatensätze verwendet, wird zurückgesetzt.

Dann wird die Warmstart-Initialisierung aus dem vorangegangenen Abschnitt durchgeführt. Das CV-Feld enthält die eventuell vom Benutzer geänderten Werte der Setup-Parameter.

Danach wird das Defaultprogramm geladen und ausgeführt, wenn der Setup-Parameter EEPROM-TCL bzw. CV+67,1 dies nicht verbietet, "01". Man beachte, dass das Defaultprogramm mit dem Sprungziel #0 beginnen muss, also mit

**E#0:...**

da es sonst nicht als vorhanden erkannt wird. Wenn kein Defaultprogramm vorhanden ist, wird die Anweisung

**IK, 'NICHT BEREIT ; DL=' ,D:AD,0,/DL/,D+18,6:SR,'77':**

ausgeführt.

#### **Erweiterte Benutzerschnittstelle:**

Das Maskenarchiv, Logo-Datei und das INTUS Sound Audioarchiv bleiben erhalten.

### **8.4.3.3 Eiskaltstart**

Ein Eiskaltstart wird nur dann ausgeführt, wenn die Setup-Parameter im EEPROM nicht gefunden werden bzw. korrupt sind oder wenn der Setup-Parameter Anlaufmodus bzw. CV+8,1 auf Eiskaltstart, "03", steht. In diesem Fall werden die Setup-Parameter auf die Default-Werte gesetzt (siehe Betriebshandbuch) und ein Kaltstart durchgeführt.

Ein Linux Systemstart wird nicht ausgeführt, sondern muss bei Bedarf eingestellt werden.

**Erweiterte Benutzerschnittstelle:**

Das Maskenarchiv, Logo- Datei und das INTUS Sound Audioarchiv werden gelöscht.

### **8.4.3.4 Comstart**

Beim Comstart werden wie beim Eiskaltstart die Setup-Parameter auf die Default-Werte gesetzt, bis auf die Einstellungen für die Hostschnittstelle Kanal A, die TCP/IP Wartungsschnittstelle und die Wartungsgruppe (siehe Betriebshandbuch). Somit bleiben die Kommunikationseinstellungen unangetastet. Der Setup-Parameter Anlaufmodus bzw. CV+8,1 muss für diese Anlaufmodus auf Comstart, "04", stehen.

**Erweiterte Benutzerschnittstelle:**

Das Maskenarchiv, Logo- Datei und das INTUS Sound Audioarchiv werden gelöscht.

## **8.5 TCL-Programm testen**

### **8.5.1 Format der TCL-Fehlermeldungen**

Fehler, die in TCL-Programmen auftreten, werden in einer einheitlichen Form angezeigt.

**Prozess : Fehlernr : Zusatzinfo : Sprungziel : Anweisung : Syserror**

Dabei bedeuten:

Feld	Bedeutung	Bemerkung
Prozess	Name des Prozesses, der den Fehler feststellt.	Die wichtigsten Prozesse sind: MONIN: Fehlermeldung des Ladecompliers beim Laden eines TCL-Programms. INTERP: Fehlermeldung des Interpreters bei der Ausführung von TCL-Anweisungen.
Fehlernr	Die Fehlernummer zeigt die Ursache des Fehlers an.	Siehe in den Abschnitten 9.1 bis 9.3.2
Zusatzinfo	Zusatzinformation zur Fehlermeldung	Z.B. Ringpuffernummer usw.
Sprungziel	Letztes Sprungziel, bevor der Fehler aufgetreten ist.	Diese Angabe ist nur relevant, wenn als Prozess MONIN oder INTERP angegeben ist.
Anweisung	Nummer die angibt, in welcher Anweisung nach dem Sprungziel der Fehler aufgetreten ist.	Diese Angabe ist nur relevant, wenn als Prozess MONIN oder INTERP angegeben ist.

Syserror	Syserror zeigt einen Fehler bei einem System Call an.	Zusatzinformation, bitte bei unerwarteten Fehlern für Anfragen bei PCS notieren.
----------	---	--

**Tabelle 8.1 – Format der TCL-Fehlermeldungen**

Die Anzeige erfolgt normalerweise im Display, kann aber umgelenkt werden. Im Tracefeld TR+6,1 wird die Ausgabe der Fehlermeldungen gesteuert. Mehrere Angaben sind gleichzeitig möglich:

- Fehlermeldung ins Display (Voreinstellung)
- Fehlermeldung in den Sendepuffer Hostschnittstelle \$2
- Fehlermeldung in den Sendepuffer Kanal B \$6
- Fehlermeldung unterdrücken

Die Fehlermeldung steht auch im Tracefeld TR von TR+9 bis TR+26, sodass im Programm auf sie reagiert werden kann. Siehe Abschnitt 4.47 *TR Trace*.

### 8.5.2 Syntaktische Fehler beheben

Syntaxfehler entstehen durch unzulässige Zeichenfolgen im TCL-Programm. Sie werden beim Laden des TCL-Programms durch den Ladecompiler im MONIN-Prozess festgestellt.

**Syntaktische Fehlermeldungen** haben den Zahlenbereich **1-199**, sie sind im Abschnitt 9.1 aufgelistet. MONIN wird in der Fehlermeldung im Prozess-Feld angegeben.

Eine fehlerhafte Anweisung wird ignoriert, alle weiteren werden normal ausgeführt.

Achten Sie darauf, dass zuerst alle syntaktischen Fehler behoben sind, bevor Sie das Programm ausführen. Ein nicht behobener syntaktischer Fehler führt in jedem Fall zu einem fehlerhaften Programmablauf.

### 8.5.3 Semantische Fehler beheben

Semantische Fehler sind etwa ein unzulässiger Gebrauch von Werten in Feldern oder Anweisungen, z.B. nicht vorhandene Sprungziele, zu große Offsetwerte usw. Sie werden in der Regel vom Interpreter, manchmal auch vom Ladecompiler des MONIN-Prozesses festgestellt.

**Semantische Fehlermeldungen** haben den Zahlenbereich **200-400**, sie sind im Abschnitt 9.2 aufgelistet.

Die fehlerhafte Anweisung wird nicht ausgeführt, das Programm läuft aber anschließend weiter.

#### Testmöglichkeiten:

Es gibt folgende Testmöglichkeiten, um die Ursache von semantischen Fehlern zu finden:

- **Tracefeld TR**  
Es gibt eine Reihe von Testmöglichkeiten, die über das Feld TR gesteuert werden. Dazu müssen mit einer Kopieranweisung bestimmte Werte in TR eingetragen werden. Siehe Abschnitt 4.47. Die Kopieranweisungen können an jeder Stelle des Programms eingefügt werden, oder jederzeit vom Testrechner gesendet werden (MONIN-Adressbyte 'T').
- **Einfügen von Testausgaben**  
Der einfachste und oft schnellste Weg, Fehler zu finden, besteht darin, die Werte von Feldern ständig ins Display auszugeben, oder Datensätze mit der Anweisung SR an den Leitrechner zu senden und dort anzuzeigen.

- **Testen vom Leitrechner aus**

Ein sehr effizienter Weg besteht darin, das geladene Programm vom Leitrechner aus zu testen. Dies setzt allerdings am Leit- oder Testrechner ein Testprogramm voraus, mit dem es möglich ist, Datensätze am Host einzugeben und an das INTUS 3000 Terminal zu senden und umgekehrt jeden Datensatz vom INTUS 3000 Terminal auf dem Host anzuzeigen.

Es können dann, während das normale Programm läuft, TCL-Anweisungen mit Adressbyte 'T' direkt vom Host aus eingegeben werden. Damit können jederzeit Felder angezeigt und verändert, der Programmablauf unterbrochen und fortgesetzt, oder bestimmte Unterprogramme direkt angesprungen werden. Außerdem können die Testmöglichkeiten über das Feld TR ein- und ausgeschaltet werden.

- **Wenn das Programm stehenbleibt**

Wenn ein TCL-Programm aus unbekannten Gründen nicht mehr weiterläuft, liegt dies daran, dass der Interpreter auf eine Stopanweisung gelaufen ist und im Interpreter-Anweisungspuffer \$3 auf neue Anweisungen wartet, ohne dass vorher dafür gesorgt wurde, dass Ereignissprünge in \$3 eingetragen werden können, oder dass die Ereignissprünge an falscher Stelle versehentlich zurückgesetzt wurden.

In diesem Fall kann man zum Beispiel die Anweisung C4 (Abschnitt 5.11) vom Leitrechner an das Terminal senden. Sie bewirkt, dass der Tracepuffer, der die 5 zuletzt durchlauften Sprungziele enthält, im Display angezeigt wird.

- **Scheinbarer Terminalabsturz**

Oft hat es den Anschein, dass das Terminal "abgestürzt" ist, d.h. es sind keine Eingaben mehr möglich und das Terminal sendet nicht mehr an den Leitrechner. In diesem Fall sollte zuerst geprüft werden, ob sich das Setup noch bedienen lässt. Wenn dies nicht der Fall ist, liegt entweder ein Hardwaredefekt oder eine schwere Störung vor, die nur durch ein Terminal-Reset zu beheben ist.

Wenn sich das Setup jedoch bedienen lässt, ist die Ursache für den "Absturz" in der Regel in einem fehlerhaften TCL-Programm zu suchen.

#### **8.5.4 Laufzeitfehler beheben**

Es gibt Fehlerursachen, die nicht in einem fehlerhaften Programm zu suchen sind, sondern nur in bestimmten Betriebssituationen auftreten. In diesem Fall ist es mitunter sinnvoll, im TCL Programm auf sie zu reagieren, anstatt eine Fehlermeldung auf dem Display ausgeben zu lassen.

Wenn in TR+27,4 ein Sprungziel ungleich '0000' eingetragen wird, dann wird im Fehlerfall ein **Unterprogrammsprung** auf dieses Sprungziel in den Interpreter-Anweisungspuffer \$3 eingetragen. Dort kann dann die Fehlersituation bearbeitet werden. Die Fehlermeldung steht dazu in TR+9 bis TR+26.

##### **Programmstop bei Fehler**

Über diesen Unterprogramm-Fehlersprung ist es auch möglich, das Programm nach einem Fehler anzuhalten. Am Sprungziel xxx muss (je nachdem, welche Ereignisse aktiv sind) folgendes TCL-Unterprogramm stehen:

```
D#xxx: R,T: R,E: R,R: K,'0',P2: S:  
D%:
```

Damit werden folgende Ereignissprünge deaktiviert:

- R,T: Zeitüberwachungen
- R,E: Eingabefreigaben
- R,R: Ringpuffersprünge
- K,'0',P2: DI-Sprünge und Zählerereignisse

### **8.5.5 Systemfehlermeldungen**

Neben den Fehlern, die durch fehlerhafte TCL-Programme hervorgerufen werden und in den vorherigen Abschnitten beschrieben wurden, gibt es Systemfehlermeldungen, die im normalen Betrieb nicht auftreten sollten. Sie weisen in der Regel nicht auf einen TCL Programmfehler oder einen Bedienfehler, sondern auf ein Problem in der Firmware hin. Diese Fehler sollten PCS mitgeteilt werden.

Eine Ausnahme bilden die beiden Systemfehlermeldungen **401** und **402**, die sich auf Ringpufferzugriffe beziehen. Sie entstehen in der Regel dadurch, dass ein Ringpuffer nicht ausgelesen wird, z.B. weil die Anweisung RD nicht ausgeführt wird, oder weil die Daten wegen XOFF nicht über die Schnittstelle gesendet werden können.

Die Systemfehlermeldung **BSCM-TX:403** nimmt ebenfalls eine Sonderstellung ein: das Absetzen eines LBus Lese- oder Schreibkommandos „ESCw...“ an den internen oder externen LBus-Leser ist nicht möglich, da dem Terminal die benötigte RW-Leserlizenz (siehe CV+261,1 in Abschnitt 4.6) fehlt.

Die **Systemfehlermeldungen** sind im Abschnitt 9.3 aufgeführt. Es gibt folgende Arten:

- **Prozess-Fehlermeldungen in 9.3.2:** Diese Fehler werden von den Prozessen des Laufzeitsystems festgestellt und werden in der TCL-Fehlermeldung im Feld "Fehlernr" angezeigt.
- **Abbruchfehlermeldungen in 9.3.3:** Diese Meldungen werden vom Betriebssystem bei einem schweren Systemfehler ausgegeben und führen entweder zu einem Halt oder Reset des Terminals. Sie unterliegen nicht der Steuerung über das Feld TR und haben ein eigenes Format: "SYSTEM-ERROR: x"



## 9 Fehlermeldungen

### 9.1 Syntaktische Fehlermeldungen

Fehler, die in TCL-Programmen auftreten, werden in einer einheitlichen Form angezeigt.

**Prozess : Fehlernr : Zusatzinfo : Sprungziel : Anweisung : Syserror**

In Abschnitt 8.5.1 ist das Format der Fehlermeldung beschrieben.

Die syntaktischen Fehler werden vom MONIN-Prozess festgestellt. Der Zahlenbereich der Fehlernummer "*Fehlernr*" liegt von 1-199.

#### Indizierte Adressierung: <Zahlenzeiger>

5	'<Feldadr>' oder '<Feldadr>,<Zahlenwert>)' erwartet
6	',<Zahlenwert>)' erwartet
7	'<Zahlenwert>)' erwartet
8	')' erwartet

#### # Sprungziel:

10	<Dezimalzahl> erwartet
----	------------------------

Siehe Abschnitt 5.3.

#### AD, DI, MU, SB Arithmetikanweisungen:

11	','(1. Komma) erwartet
12	'(1. Zahlenwert) erwartet
13	','(2. Komma) erwartet
14	'(2. Zahlenwert) erwartet
15	'( <Feldadr> erwartet

Siehe Abschnitte 5.6 (AD), 5.13 (DI), 5.22 (MU), 5.33 (SB).

#### I, D Inkrementier-/Dekrementieranweisung:

16	','(1. Komma) erwartet
17	<Feldadr> (1. Feld) erwartet
18	','(2. Komma) erwartet
19	<Zahlenwert> (1. Wert) erwartet
20	','(3. Komma) erwartet
21	<Zahlenwert> (2. Wert) erwartet
22	'-' erwartet
23	<Zahlenwert> (3. Wert) erwartet

Siehe Abschnitte 5.12 (D), 5.18 (I).

**F Fillanweisung:**

24	', (1. Komma) erwartet
25	<Feldadr> erwartet
26	', oder <Zahlenwert> (2. Komma) erwartet
27	', (3. Komma) erwartet
28	<Zahlenwert> (2. Wert) erwartet

Siehe Abschnitt 5.16.

**K Kopieranweisung:**

29	', (1. Komma) erwartet
30	<kQop> erwartet
31	', (2. Komma) erwartet

Siehe Abschnitt 5.19.

**M1 Modulo 10/11 Fehlerprüfung:**

32	', (1. Komma) erwartet
33	<kQop> (1. Feld) erwartet
34	', (2. Komma) erwartet
35	<Zahlenwert> oder '*' erwartet
36	', oder <Label> (3. Komma) erwartet

Siehe Abschnitt 5.21.

**P Vergleich oder Prüfanweisung:**

37	', (1. Komma) erwartet
38	<kQop> (1. Feld) erwartet
39	', (2. Komma) erwartet
40	'> oder < oder = erwartet
41	<eQop> (2. Feld) erwartet
42	', (3. Komma) erwartet

Siehe Abschnitt 5.24.

**PS Prüfsumme anhängen:**

43	', (1. Komma) erwartet
44	<Feldadr> erwartet
45	', (2. Komma) erwartet
46	<Zahlenwert> erwartet

Siehe Abschnitt 5.25.

**PT Tabellenvergleich:**

47	', (1. Komma) erwartet
48	<kQop> erwartet
49	', (2. Komma) erwartet
50	<Zahlenwert> (1. Wert) erwartet
51	', (3. Komma) erwartet
52	<Zahlenwert> (2. Wert) erwartet
53	', (4. Komma) erwartet
54	<Zahlenwert> (3. Wert) erwartet
55	', (5. Komma) erwartet
56	<Zahlenwert> oder '*' (4. Wert) erwartet
57	Sprung erwartet

Siehe Abschnitt 5.26.

**R Resetanweisung:**

58	', (1. Komma) erwartet
59	Buchstabe erwartet

Siehe Abschnitt 5.27.

**RD Ringpuffer lesen:**

60	Timeout erwartet
61	', (1. Komma) erwartet
62	'\$' Puffer erwartet
63	', (2. Komma) erwartet
64	Feld erwartet

Siehe Abschnitt 5.28.

**SE, SR Sendeanweisungen:**

65	', (1. Komma) erwartet
66	<Zeichenkette> erwartet

Siehe Abschnitte 5.34 (SE), 5.35 (SR).

**E Eingabeanweisung:**

67	Eingabeliste erwartet
68	Eingabeliste erwartet
69	', (1. Komma) erwartet
70	'/' (in Eingabeliste) erwartet
71	'/' (in Eingabeliste) erwartet
72	Eingabe erwartet

Siehe Abschnitt 5.14.

**WR, WO Schreibanweisungen:**

73	','(1. Komma) erwartet
74	'\$'Puffer erwartet
75	Feld erwartet
76	','(2. Komma) erwartet

Siehe Abschnitt 5.39 (WR), 5.38 (WO).

**RS Ringpuffer-Sprung:**

77	','(1. Komma) erwartet
78	'\$'Puffer erwartet
79	','(2. Komma) erwartet
80	Wert erwartet
81	','(3. Komma) erwartet

Siehe Abschnitt 5.31.

**XA, XS, EO, OR, UN binäre Arithmetikanweisungen:**

82	','(1. Komma) erwartet
83	1. Feld erwartet
84	','(2. Komma) erwartet
85	2. Feld erwartet
86	3. Feld erwartet

Siehe Abschnitte 5.40 (XA), 5.41 (XS), 5.15 (EO), 5.23 (OR), 5.37 (UN).

**RL, RR Shiftanweisungen:**

87	','(1. Komma) erwartet
88	1. Feld erwartet
89	','(2. Komma) erwartet
90	2. Feld erwartet
91	3. Feld erwartet

Siehe Abschnitte 5.29 (RL), 5.30 (RR).

**R,E Anweisung:**

92	Lesertyp erwartet
93	Lesertyp erwartet

Siehe Abschnitt 5.27.

**R,R Anweisung:**

94	'\$'Puffer erwartet
95	'\$'Puffer erwartet
96	Sprung erwartet
197	Sprung Fehler
198	TCL-Anweisung ':' erwartet
199	':' erwartet

Siehe Abschnitt 5.27.

## 9.2 Semantische Fehlermeldungen

Fehler, die in TCL-Programmen auftreten, werden in einer einheitlichen Form angezeigt.

**Prozess : Fehlernr : Zusatzinfo : Sprungziel : Anweisung : Syserror**

In Abschnitt 8.5.1 ist das Format der Fehlermeldung beschrieben.

Die semantischen Fehler werden vom Interpreter INTERP oder auch vom MONIN-Prozess festgestellt. Der Zahlenbereich der Fehlernummer "Fehlernr" liegt von 200-400.

Nummer	Bedeutung
200	Falscher Zahlenwert
201	Hexzahl: ungerade Anzahl von Hexziffern, oder zu lang
202	String oder Anweisungscode zu lang
203	Anweisungscode zu lang
204	Anweisungscode zu lang
205	Routing-Byte 'I' beinhaltet eine Sprungziel Anweisung
206	Falsches Sprungziel, z.B. da die Sprungtabelle zu klein ist.
207	Falsche Funktionstasten-Nummer
208	Falscher Index bei Symbol(...)
209	Falscher Offset bei Symbol+Offset
210	Länge bei Strings nicht erlaubt
211	Unterprogramm Returnstack leer
212	Sprungziel nicht definiert

Die folgenden Fehlermeldungen zeigen einen falschen Code im DL Feld an

Nummer	Bedeutung
214	Falsche Anweisung in DL Feld
215	Falscher Index
216	Task benutzt die Parameter falsch!
217	Task benutzt die Parameter falsch!
218	falsche Anweisung in DL Feld

219	HW-Tabelleneintrag ist falsch (semantisch)
220	falsche Anweisung in DL Feld (OP_TYPE zu OP_LOAD)
221	falsche Anweisung in DL Feld (OP_TYPE zu OP_PUSH)
222	Zu viele Parameter
223	falsche Unterprogrammadresse im DL Feld
224	falsche Anweisung in DL Feld

Mit '\*' gekennzeichnete Fehlermeldungen werden nur erzeugt, wenn TR+31 auf '1' gesetzt wird (Abschnitt 4.47).

Nummer	Bedeutung	
225	Verkettungsstring zu lang	
226	Speicherreservierungsfehler (malloc, interp)	
227	Verschachtelungstiefe für Subroutinen überschritten	
228	Inkrementieranweisung	
229	Dekrementieranweisung	
230	Fehler in Datumsberechnung (C1)	
231	Falsche Puffernummer	
232	Eingabeanweisung	
233	Adresse außerhalb des TCL-Feldes	
234	Prüfstring außerhalb des TCL-Feldes (M1)	
235	Dividend ist Null	
236	Resultat zu groß	
237	freigegebener Leser nicht konfiguriert	
238	unzulässiger Codetyp	
239	Lesernummer_Low Lesernummer_High	
240	Falsches Adress-Byte	
241	Feld DL zu klein zum Laden	
242	unzulässiger Modus in Anweisung KW	
243	optionale Länge in Prüfanweisung zu groß oder Null	
244	KW Anweisung: Feld (Quelle oder Ziel) zu klein für die Umwandlung	
245	PS Anweisung: Anzahl ist falsch	
246	Prüfanweisung: Anzahl ist falsch	
247	*	optionale Länge in Kopieranweisung zu groß
248	Falscher Inhalt in H3 ( C0 )	
249	Falscher Inhalt in Gx oder ND ( C0 )	
250	Falscher Inhalt in Gx,ND oder ST Überlauf ( C0 )	
251	Falscher Inhalt in AB,AE,H1,H2 ( C0,C1 )	
252	*	Lampen nicht mit 'A', 'E' oder 'B' belegt
253	Taktausfallzeit <= Obergrenze	
254	'N'-Routing mit falscher Längenangabe	

<b>Nummer</b>		<b>Bedeutung</b>
255		Fehler beim Schreiben auf das EEPROM
256		Fehler beim Lesen vom EEPROM
257		Shiftparameter > 32
258	*	Uhr-String falsch
259	*	Datum-String falsch
260	*	Eingabealternative doppelt aktiviert
261		Fehlerhafte Eingabeanweisung
262		INTUS 1000 antwortet nicht
263		Wert für Ox und Lx nicht A,B,E
264	*	DNCIN: Sprungziel nicht vorhanden
265	*	DNCIN: TF-Feld Offset zu hoch
266		Inhalt von Zx-Feld nicht dezimal (ASCII)
267		RAC-Fehler (nur INTUS 2000)
268		Timernummer in Timeoutanweisung fehlerhaft
269	*	Länge in RD-Anweisung ohne Timeout zu groß (>256)
270		Parameterübergabe ist nicht korrekt
271		kein INTUS 2600 Grafikdisplay
272		Fehler in Grafikfunktion (nur INTUS 2600)
273		Sprache nicht korrekt
274		Setup-Sprachtexte nicht korrekt
275		Unzulässiger Inhalt in ER, EZ bei der C3-Anweisung
276		Unzulässiger Code (>7) in PS-Anweisung
277		Fehler im ETH/ AS/400-Controller (nur INTUS 2000)
278		Fehler beim Arbeiten mit VI-Feld (nur INTUS 2000)
279		Fehler im P10-Feld
280		Fehler in P20+23 (INTUS 2400/ LCD-Modus)
281		Mehr als 10 Aktualisierungen angegeben
282		Falsche LBus-Adresse
283	*	Freigabe für nicht konfiguriertes Terminal verworfen
284		Versuch ein geschütztes Feld (CV) zu lesen gescheitert
285		Versuch ein geschütztes Feld zu schreiben ist gescheitert
286	*	Ungültiger Wert für Feld
287		Maskenumschaltung fehlgeschlagen (Grafikprozess)

## 9.3 Systemfehlermeldungen

### 9.3.1 Konfigurationsfehler des Programmbereichs

Der MONIN-Prozess sendet

**DE<CR>**

als „Download Error“ Fehlermeldung an die Hostschnittstelle, wenn im Programmbereich DL nicht mehr ausreichend Speicherplatz für den empfangenen Datensatz mit Adressbyte 'D' ist. Der Datensatz wird verworfen. Die Behebung des Konfigurationsfehlers wird in Abschnitt 4.9 DL Programmbereich beschrieben, siehe auch Abschnitt 8.3.

### 9.3.2 Prozess-Fehlermeldungen

Diese Fehler werden von den Prozessen des Laufzeitsystems festgestellt.

Sie werden in der TCL-Fehlermeldung (siehe Abschnitt 8.5.1)

**Prozess:Fehlernr:Zusatzinfo:Sprungziel:Anweisung:Syserror**

unter "Fehlernr" angezeigt.

Die Funktion des Prozesses "Prozess" ist in der Tabelle 6.1 beschrieben.

**Sonderfall Ringpufferzugriffe:**

Nummer	Bedeutung
401	Fehler beim Schreiben auf Ringpuffer, Ringpuffernummer in "Zusatzinfo"
402	Fehler beim Lesen von Ringpuffer, Ringpuffernummer in "Zusatzinfo"
403	Fehler den der Prozess BSCM-TX (Verarbeitung der Daten für den SendeRingpuffer \$D) beim Absetzen eines LBus Lese- oder Schreibkommandos „ESCw...“ generiert, da die RW-Leser Lizenz fehlt, siehe CV+261,1 in Abschnitt 4.6

Die beiden Fehler 401 und 402 entstehen in der Regel dadurch, dass ein Ringpuffer nicht ausgelesen wird (siehe auch Abschnitt 8.5.5).

**Probleme der TCL-Firmware:**

Die folgenden Fehler zeigen keinen TCL-Programm- oder Anwendungsfehler an, sondern ein Problem in der TCL-Firmware. Sie sollten PCS mitgeteilt werden.

Nummer	Bedeutung
4	Fehler bei Speicheranforderung (Compiler )
500	Fehler bei Ringpufferanforderung
503	Fehler beim Schreiben auf ein Device
504	Fehler beim Lesen aus einem Device
507	Fehler beim Senden einer Message
508	Fehler beim Empfangen einer Message oder zu viele Aktualisierungsfelder (vom AKTUELL-Prozess)
511	Fehler beim Öffnen von tty-Kanal
515	STACK-Fehler (Compiler)

516	STACK-Fehler (Compiler)
518	HW-Tabelleneintrag vom Symbol fehlt

### 9.3.3 Abbruch-Fehlermeldungen

Während der Selbstkonfiguration oder Initialisierung kann es passieren, dass das System feststellt, dass Systemressourcen nicht ausreichen. In diesem Fall wird die Meldung

**SYSTEM ERROR:X**

auf das Display ausgegeben und das System verbleibt in diesem Zustand. Im Gegensatz zu den Abbrüchen wegen der defekten Hardware ist das Setup des Terminals jedoch bedienbar. Dadurch lassen sich Konfigurationsfehler durch einen Eiskalt- oder Comstart beheben, den man nach Einwahl des Setup einstellt und danach einen Reset über **Reset:Ja** durchführt. Das X wird in der Systemmeldung durch einen Buchstaben ersetzt, der eine genauere Ursache beschreibt.

Über die derzeit möglichen Ursachen gibt folgende Tabelle Auskunft:

Systemfehler	Status LED blinkt, Hupe tönt	Ursache und Behebung
G	7x	<p><b>TCL</b> Firmware und Textdatei <b>INTUS.TXT</b>, mit den sprachlich abhängigen Meldungs- und Setuptexten, passen nicht zusammen.</p> <p><b>Behebung:</b> Laden Sie die Textdatei, die zur TCL Version gehört mit INTUS RemoteSetup ins Terminal. Das Setup ist noch benutzbar, wenn auch für nicht vorhandene Texte die Zeichenkette '????' angezeigt wird.</p>
H	8x	<p>Die Verbindung zum Leitrechner konnte nicht geöffnet werden.</p> <p><b>Behebung:</b> Versuch eines Eiskaltstarts. Wenn das keinen Erfolg hat, dann liegt ein Hardwareproblem vor, das repariert werden muss.</p>
I	9x	<p>Die Hardwarekonfiguration konnte nicht aus dem EEPROM geladen werden.</p> <p><b>Behebung:</b> Das Terminal muss mit der Produktions- und Wartungssoftware neu produziert werden. Wenn das keinen Erfolg hat, liegt vermutlich ein Hardwarefehler vor.</p>
J	10x	<p>Es wurden mehr Software-Timer angefordert als angelegt sind.</p> <p>Interner Softwarefehler, der nicht vorkommen sollte.</p>
K	11x	<p>Eine interne Speicheranforderung zur Anlage einer Tabelle im RAM konnte nicht erfüllt werden. Die Ursache kann in einer zu großen Puffervorgabe für die seriellen Kanäle liegen.</p> <p><b>Behebung:</b> Eiskalt- oder Comstart und Neukonfiguration. Wenn das keinen Erfolg hat, liegt vermutlich ein Hardwarefehler vor.</p>

System-fehler	Status LED blinkt, Hupe tönt	Ursache und Behebung
L	12x	Ein Softwaremodul konnte sich nicht für eine De-Initialisierung eintragen. Interner Softwarefehler, der nicht vorkommen sollte.
M	13x	Speichermangel beim Anlegen einer Realzeitkomponente. <b>Behebung</b> wie unter K.
N	14x	Speichermangel beim Anlegen eines Ringpuffers. <b>Behebung</b> wie unter K.
O	15x	Fehler in der SRAM Verwaltung. Interner Fehler, der nicht vorkommen sollte.
P	16x	Fehler in der SRAM Verwaltung. Interner Fehler, der nicht vorkommen sollte.
Q	17x	Speichermangel beim Anlegen eines Realzeitprozesses. <b>Behebung</b> wie unter K.
R	18x	Notpuffer-Konfiguration zu groß. Dieser Fehler sollte weitgehend vermieden werden durch die automatische Neukonfiguration, die die Notpuffergröße auf die Voreinstellung von 48kB reduziert. <b>Behebung:</b> Eiskalt- oder Comstart mit Neukonfiguration, wobei Notpuffer und Tabellenfeld so konfiguriert werden sollten, dass mindestens 30kB für den Downloadbereich, DL, übrig bleiben.

*Tabelle 9.1 - Systemfehler – Ursache und Behebung*

## 9.4 Fehlercodes bei Lesereingaben

Wenn in den Lesefeldern B, I, oder M folgende Eintragungen stehen:

- das Konfigurationsbyte (+80,1 bzw. +107,1) ist auf '1' gesetzt, d.h. Eingabesprung auch bei einer Fehllesung,
- das Lesefehlerbyte (+81,1 bzw. +108,1) ist auf '1' gesetzt, d.h. Lesung war nicht fehlerfrei,

dann enthält das Codekennungsbyte (+82,1 bzw. +109,1) einen der folgenden Fehlercodes:

### Allgemeine Fehler

Anzeige im TCL	Bedeutung
1	falsche Leserkennung
2	weniger Zeichen als Minimalanzahl
3	mehr Zeichen als Maximalanzahl
4	nicht die gewünschte Anzahl von Zeichen
5	unvollständige Eingabe (Timeout)
6	unvollständige Eingabe
7	zu viele Zeichen in Eingabe
8	Eingabe ist nicht numerisch
9	Eingabe ist nicht alphanumerisch
0	Eingabe enthält nichtdarstellbare Zeichen
:	Seriell angeschlossener interner Leser oder abgesetzte Eingabeeinheit ausgefallen

### B-Feld: Barcodeleser

Anzeige im TCL	Anzeige im Lesertest des Setup	Bedeutung
A ("41")	001	1. keine Ruhezone gefunden 2. zu wenig Striche
B ("42")	002	Kein Startzeichen gefunden
D ("44")	004	Kein Stopzeichen gefunden
H ("48")	008	Weder Start- noch Stopzeichen gefunden, falsche Strichanzahl
P ("50")	016	Zu breiter/schmaler Strich
"60"	032	Nicht dekodierbares Zeichen
"80"	064	Code 93 Prüfsummenfehler
"C0"	128	Pufferüberlauf

**M-Feld: Proximity-, Speicherchipkarten- und Magnetkartenleser**

Anzeige im TCL	Anzeige im Lesertest des Setup	Bedeutung
A	001	Magnetkartenleser: Parity Fehler
B	002	Magnetkartenleser: LRC Fehler Speicherchipkartenleser: unbekannter Typ
C	003	Proximityleser: 1. Karte aus Feld (nur bei kundenspezifischer Parametrierung des INTUS 600) 2. Nur INTUS 300h und INTUS 340h: Modus „ID“ ausgewählt, aber keine ID-Nummer auf der Karte vorhanden.
E	005	Proximityleser: Authentifizierungsfehler (nur bei kunden-spezifischer Parametrierung des INTUS 600DF)
H	008	Magnetkartenleser: Keine Magnetspur gefunden
I	009	Nur INTUS 610 Moto: Kartenentnahme während eines wiederholten Leseversuchs nach Parity Fehler (001)
J	010	Nur INTUS 610 Moto: Kartenentnahme während eines wiederholten Leseversuchs nach LRC Fehler (002)
K	011	Nur INTUS 610 Moto: Kartenentnahme während eines wiederholten Leseversuchs nach dem Fehler „keine Magnetspur“ (008)
L	012	Nur INTUS 610 Moto: Kartenauswurf nach Timeout
Q-Z	017-026	Wenn ein Biometrieleser zur Verifikation gegen Karte verwendet wird, sind auch die Fehlercodes aus der folgenden Tabelle „I-Feld: Biometrieleser“ möglich.

**I-Feld: Biometrieleser (Fingerprint und Handvenenscan)**

Anzeige im TCL	Anzeige im Lesertest des Setup	Bedeutung
Q	017	Finger / Handvenen nicht erkannt
R	018	Template nicht vorhanden (Verifikation)
S	019	Sensorfehler oder ungültige Sensordaten
T	020	Scanqualität nicht ausreichend
U	021	Timeout beim Scannen
V	022	Timeout beim Vergleichen
W	023	Karte aus Feld / Lesefehler
X	024	Biometriesegment leer
Y	025	Ungültige Biometriedaten
Z	026	Abbruch durch Benutzer

**I-Feld: RENA Induktivleser**

Anzeige	Bedeutung
A	Vor/Rück Vergleichsfehler
B	Prüfsummenfehler
D	Vergleichsfehler bei Lagemarke
H	Karte nicht vollständig

**I-Feld: HENGSTLER Induktivleser**

Anzeige	Bedeutung
A	Parity Fehler
B	Lagemarken unplausibel
D	Karte nicht vollständig



## **10    TCL Zeichensätze**



Der Zeichensatz im Display wird in CV+39,1 (Abschnitt 4.6, Tabelle 2i) eingestellt.

Siehe Abschnitt 7.3 für eine ausführliche Beschreibung, wie die Zeichensätze in einem TCL-Programm eingesetzt werden können.

<b>Zeichencode</b>	<b>ISO8859-1, -2, -5, -15</b>	<b>ISO646 Deutschland</b>	<b>ISO646 Frankreich</b>	<b>ISO646 Norwegen</b>	<b>ISO646 Spanien</b>	<b>ISO646 UK</b>	<b>Semigrafik</b>
00	NUL	NUL	NUL	NUL	NUL	NUL	NUL
01	SOH	SOH	SOH	SOH	SOH	SOH	SOH
02	STX	STX	STX	STX	STX	STX	STX
03	ETX	ETX	ETX	ETX	ETX	ETX	ETX
04	EOT	EOT	EOT	EOT	EOT	EOT	EOT
05	ENQ	ENQ	ENQ	ENQ	ENQ	ENQ	ENQ
06	ACK	ACK	ACK	ACK	ACK	ACK	ACK
07	BEL	BEL	BEL	BEL	BEL	BEL	BEL
08	BS	BS	BS	BS	BS	BS	BS
09	HT	HT	HT	HT	HT	HT	HT
0A	NL	NL	NL	NL	NL	NL	NL
0B	VT	VT	VT	VT	VT	VT	VT
0C	NP	NP	NP	NP	NP	NP	NP
0D	CR	CR	CR	CR	CR	CR	CR
0E	SOH	SOH	SOH	SOH	SOH	SOH	SOH
0F	SI	SI	SI	SI	SI	SI	SI
10	DLE	DLE	DLE	DLE	DLE	DLE	DLE
11	DC1	DC1	DC1	DC1	DC1	DC1	DC1
12	DC2	DC2	DC2	DC2	DC2	DC2	DC2
13	DC3	DC3	DC3	DC3	DC3	DC3	DC3
14	DC4	DC4	DC4	DC4	DC4	DC4	DC4
15	NAK	NAK	NAK	NAK	NAK	NAK	NAK
16	SYN	SYN	SYN	SYN	SYN	SYN	SYN
17	ETB	ETB	ETB	ETB	ETB	ETB	ETB
18	CAN	CAN	CAN	CAN	CAN	CAN	CAN
19	EM	EM	EM	EM	EM	EM	EM
1A	SUB	SUB	SUB	SUB	SUB	SUB	SUB
1B	ESC	ESC	ESC	ESC	ESC	ESC	ESC
1C	FS	FS	FS	FS	FS	FS	FS
1D	GS	GS	GS	GS	GS	GS	GS
1E	RS	RS	RS	RS	RS	RS	RS
1F	US	US	US	US	US	US	US
20	SP	SP	SP	SP	SP	SP	SP



	<b>Zeichencode</b>																				
79	y	ISO8859-1, -2, -5, -15	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	
7A	z		z	z	z	z	z	z	z	z	z	z	z	z	z	z	z	z	z	z	
7B	{	ä	é	æ	°	{	π	π	π	π	π	π	π	π	π	π	π	π	π	π	
7F	DEL	DEL	DEL	DEL	DEL	DEL	DEL	DEL	DEL	DEL	DEL	DEL	DEL	DEL	DEL	DEL	DEL	DEL	DEL	DEL	

	<b>Zeichencode</b>																				
80	XXX	ISO8859-1, -2, -5, -15		A0	ISO8859-1		ISO8859-2	ISO8859-5	ISO8859-15												
81	XXX		A1	í	À	È	í														
82	BPH		A2	¢		Ђ	¢														
83	NBH		A3	£	Ł	Ѓ	£														
84	IND		A4	¤	¤	€	€														
85	NEL		A5	¥	Ł	S	¥														
86	SSA		A6	‘	Ś	I	Ś														
87	ESA		A7	§	§	Ї	§														
88	HTS		A8			J	š														
89	HTJ		A9	©	Š	Љ	©														
8A	VTS		AA	ª	Ѕ	Њ	ª														
8B	PLD		AB	«	҆	Ћ	«														
8C	PLU		AC		Ž	Ќ															
8D	RI		AD	-	-	-	-														
8E	SS2		AE	®	Ž	Ў	®														
8F	SS3		AF		Ž	Џ															
90	DCS		B0	º	º	А	º														
91	PU1		B1	±	ќ	Б	±														
92	PU2		B2	²		В	²														
93	STS		B3	³	ќ	Г	³														
94	CCH		B4			Д	Ž														
95	MW		B5	µ	љ	Е	µ														
96	SPA		B6	¶	ѕ	Ж	¶														
97	EPA		B7	.		З	.														
98	SOS		B8			И	ž														
99	XXX		B9	¹	š	Ї	¹														
9A	SCI		BA	º	§	К	º														
9B	CSI		BB	»	ќ	Л	»														
9C	STS		BC	¼	ž	М	Œ														
9D	OSC		BD	½		Н	œ														
9E	PM		BE	¾	ž	О	Ý														
9F	APC		BF	¸	ž	П	¸														

<b>Zeichencode</b>	<b>ISO8859-1</b>	<b>ISO8859-2</b>	<b>ISO8859-5</b>	<b>ISO8859-15</b>
E0	à	í	پ	à
E1	á	â	с	á
E2	â	â	т	â
E3	ã	ă	у	ã
E4	ä	ää	ֆ	ä
E5	å	í	ҳ	å
E6	æ	ć	҃	æ
E7	ç	ç	Ч	ç
E8	è	č	Ш	è
E9	é	é	Щ	é
EA	ê	ę	ъ	ê
EB	ë	ë	ы	ë
EC	ì	ě	ь	ì
ED	í	í	э	í
EE	î	î	ю	î
EF	ï	đ'	я	ï
F0	ð	đ	№	ð
F1	ñ	ń	ë	ñ
F2	ò	њ	ђ	ò
F3	ó	ó	ѓ	ó
F4	ô	ô	€	ô
F5	õ	ő	ѕ	õ
F6	ö	ö	і	ö
F7	÷	÷	ї	÷
F8	ø	ř	ј	ø
F9	ù	ü	љ	ù
FA	ú	ú	њ	ú
FB	û	ű	ћ	û
FC	ü	ü	ќ	ü
FD	ý	ý	§	ý
FE	þ	ť	њ	þ
FF	ÿ	·	Џ	ÿ



### 11 TCL Adressen der Zutrittskontrollmanager

Bei den Zutrittskontrollmanagern kann die Verkabelung der externen Leser an LBus1 und LBus2 im Point-to-Point *PP* oder im Multipoint *MP* Verfahren ausgeführt werden. Ab TCL Version 5.5 wird die gewählte Verkabelung von LBus1 und LBus2 in CV+9,1 hinterlegt. Die Kombination der Werte für LBus1 und LBus2 hat Einfluss auf die Adresse *<Lbusadr>* des externen Lesers und auf die Nummerierung der digitalen Ein- und Ausgänge des Zutrittskontrollmanagers, die *Ex* und *Ox* Felder.

In den folgenden Abschnitten verdeutlichen Abbildungen die Zuordnung der Hardware zu den TCL-Feldern und Adressen.

Für die Modelle INTUS ACM8, INTUS ACM8(0)e Rack und INTUS ACM8(0)e Wand ist die Voreinstellung für LBus 1 PP und für LBus 2 nicht belegt: CV+9,1 enthält “02” (PP/MP(nicht vorhanden)). Für den LBus 1 ist die einfache Adressierung voreingestellt: CV+224,1 enthält “02”.

Der INTUS ACM4 und der INTUS ACM40 haben die Voreinstellung PP/PP. Damit sind sie kompatibel zu TCL 5 Versionen. CV+9,1 enthält “06” und CV+224,1 für die einfache Adressierung enthält “06”.

Der INTUS 3000 ACM hat die Voreinstellung MP/MP: CV+9,1 enthält “00”. Außerdem wird keine einfache Adressierung voreingestellt: CV+224,1 enthält “00”. In der Einstellung MP/MP unterscheiden sich der INTUS 3000 ACM und die Modelle ACM8 in der Anzahl bzw. Addressierung der Relais. Dies hat historische Gründe und ist zur Aufrechterhaltung der Abwärtskompatibilität des INTUS 3000 ACM erforderlich. Die im INTUS 3000 ACM Betriebshandbuch beschriebenen Konfigurationen lassen sich in der Konfiguration MP/MP wie gewohnt einstellen.



In den Konfigurationen PP/PP und PP/MP (nicht vorhanden) muss auf dem optionalen Leser-DI/DO-Modul zwingend die Adresse 2 eingestellt werden. Danach verhält sich der INTUS 3000 ACM wie ein INTUS ACM8 mit gleicher Konfiguration.

## 11.1 INTUS ACM40, INTUS ACM40 AKKU

### 11.1.1 LBus 1: PP / LBus 2: PP (Voreinstellung)

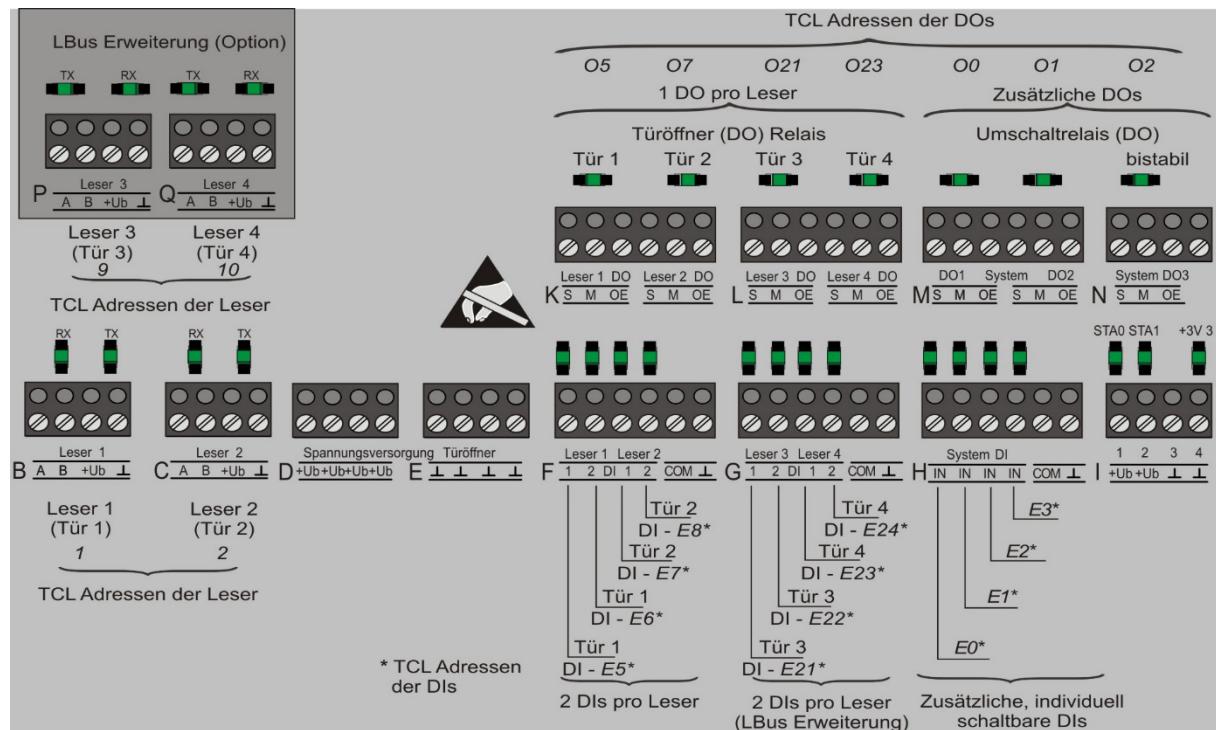


Abbildung 11.1 – TCL-Adressen INTUS ACM40 LBus 1:PP/LBus 2: PP

### 11.1.2 LBus 1: PP / LBus 2: nicht belegt

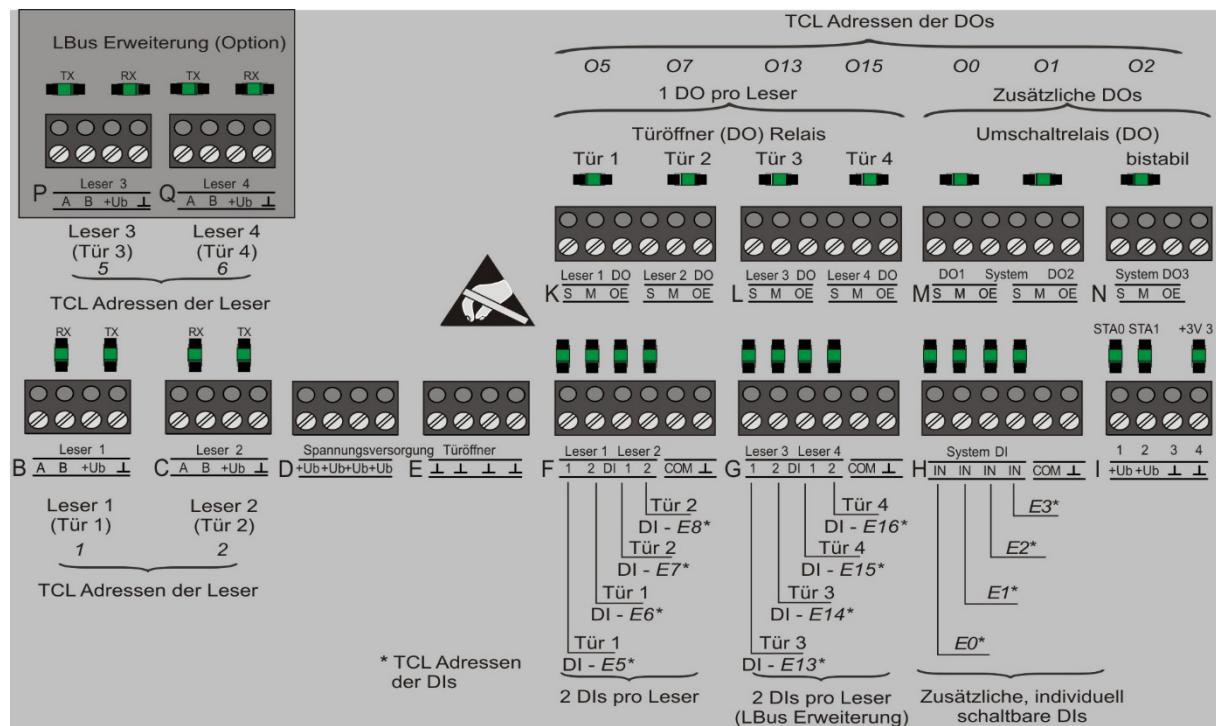


Abbildung 11.2 – TCL-Adressen INTUS ACM40 LBus 1:PP/LBus 2: nicht belegt

## 11.2 INTUS ACM4, INTUS ACM4 AKKU

### 11.2.1 LBus 1: PP / LBus 2: PP (Voreinstellung)

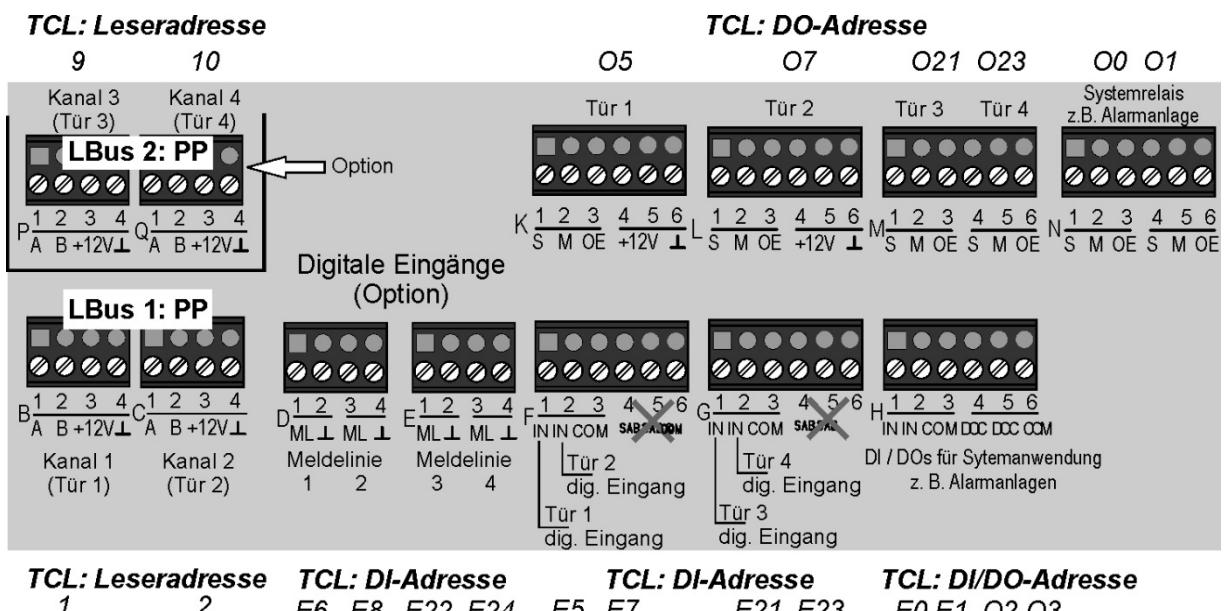


Abbildung 11.3 – TCL- Adressen INTUS ACM4 LBus 1:PP/LBus 2: PP

### 11.2.2 LBus 1: PP / LBus 2: nicht belegt

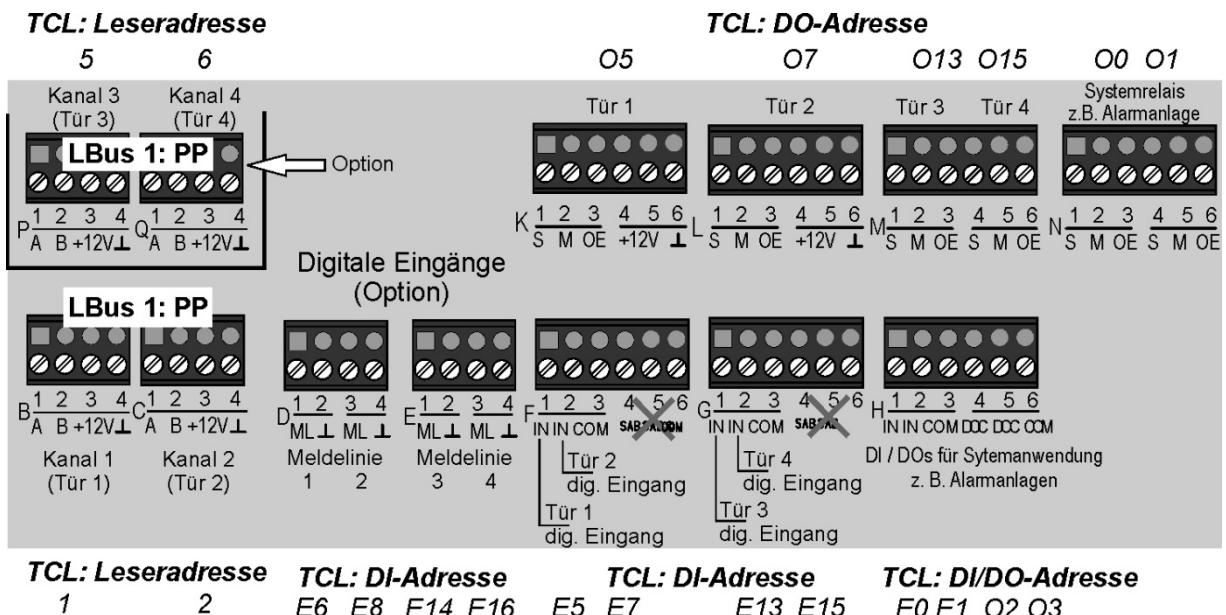


Abbildung 11.4 – TCL- Adressen INTUS ACM4 LBus 1:PP/LBus 2: nicht belegt

## 11.3 INTUS ACM8, INTUS ACM8(0)e Rack

### 11.3.1 Point-to-Point Verkabelung

**LBus 1: PP / LBus 2: nicht belegt (Voreinstellung)**

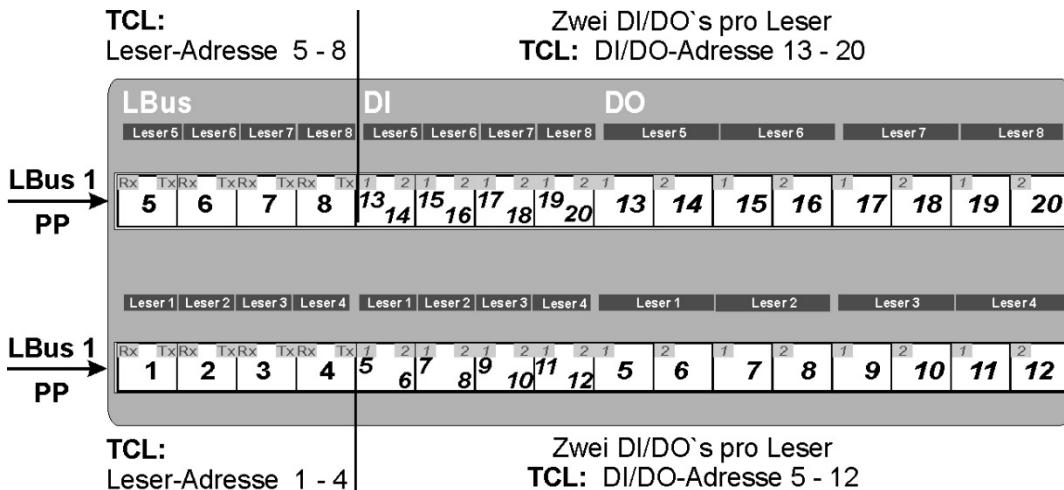


Abbildung 11.5 - TCL- Adressen INTUS ACM8(0)e Rack LBus 1:PP/LBus 2: nicht belegt

**Alternative: LBus 1: PP / LBus 2: PP**

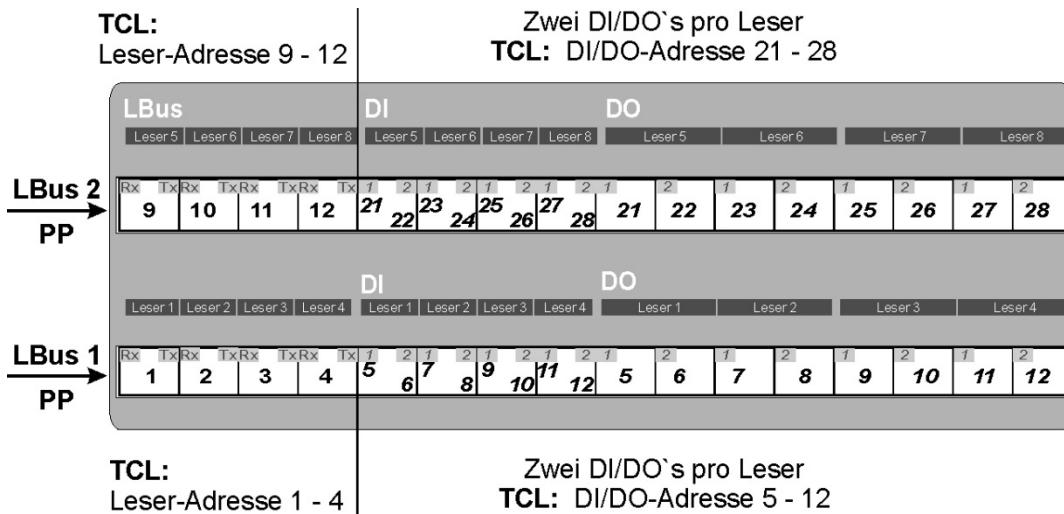


Abbildung 11.6 – TCL- Adressen INTUS ACM8(0)e Rack LBus 1:PP/LBus 2: PP

### 11.3.2 Multipoint Verkabelung

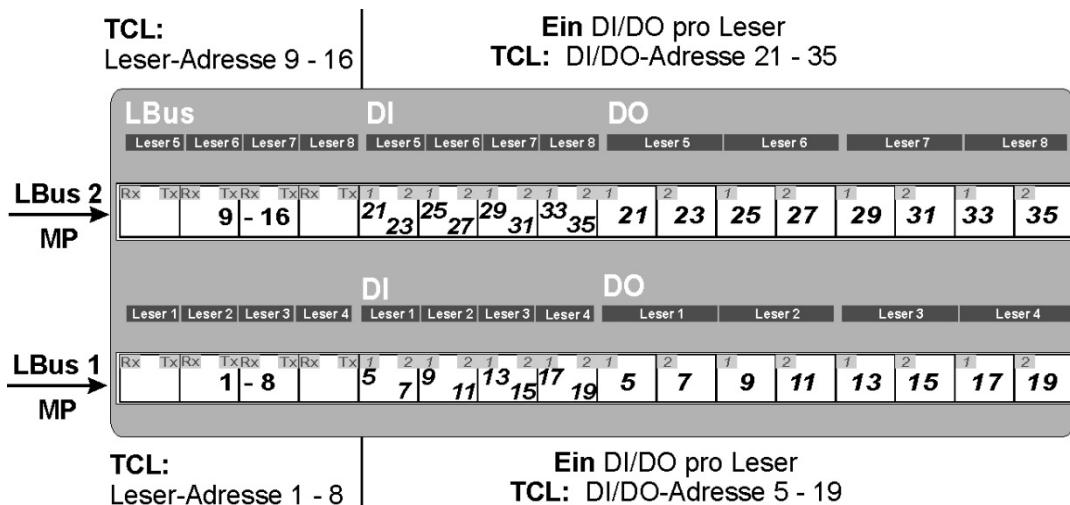


Abbildung 11.7 – TCL- Adressen INTUS ACM8(0)e Rack LBus 1:MP/LBus 2: MP

### 11.3.3 DI /-DO-Adressen für Systemanwendungen

Die Anordnung der RJ45-Buchsen ist bei allen 3 Modellen gleich, die Beschriftung ist unterschiedlich.

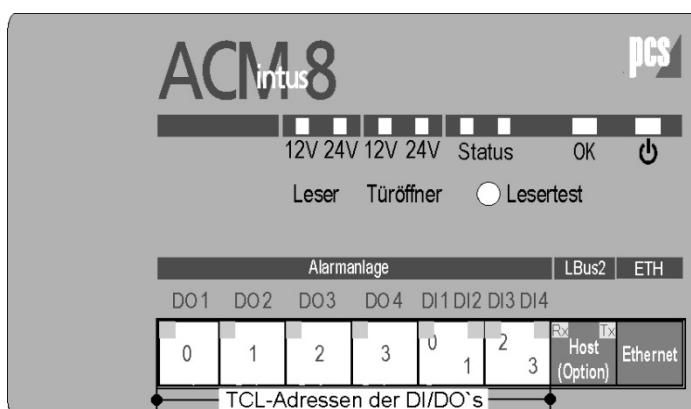


Abbildung 11.8 – TCL- Adressen INTUS ACM8(0)e Rack für Systemanwendungen

## 11.4 INTUS ACM80e Wand

### 11.4.1 Point-to-Point Verkabelung

**LBus 1: PP / LBus 2: nicht belegt (Voreinstellung)**

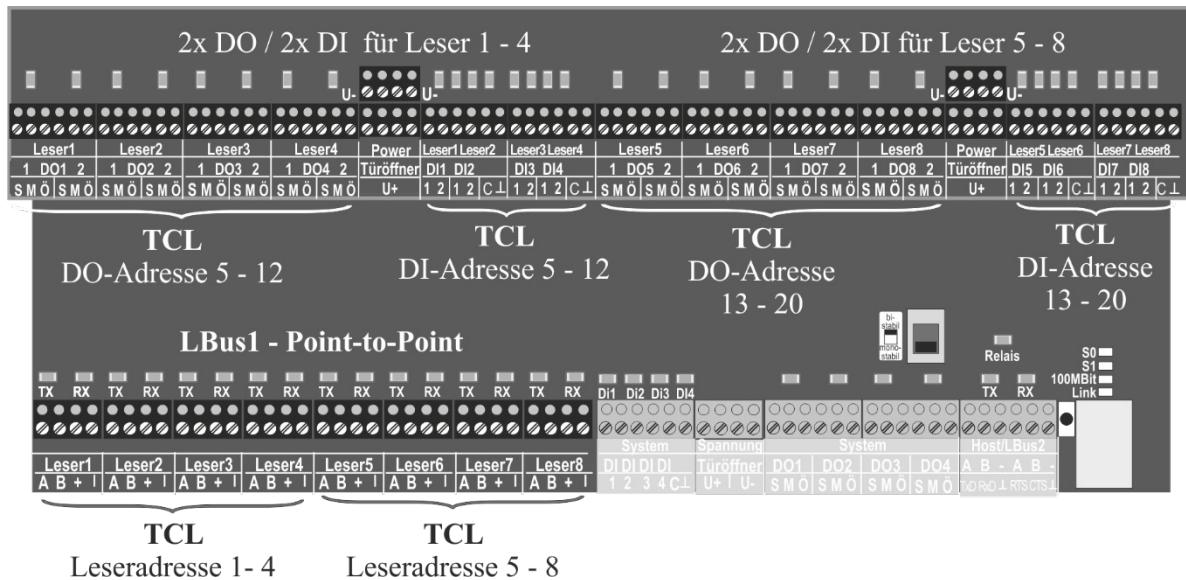


Abbildung 11.9 – TCL- Adressen INTUS ACM80e Wand LBus 1:PP/LBus 2: nicht belegt

**Alternative: LBus 1: PP / LBus 2: PP**

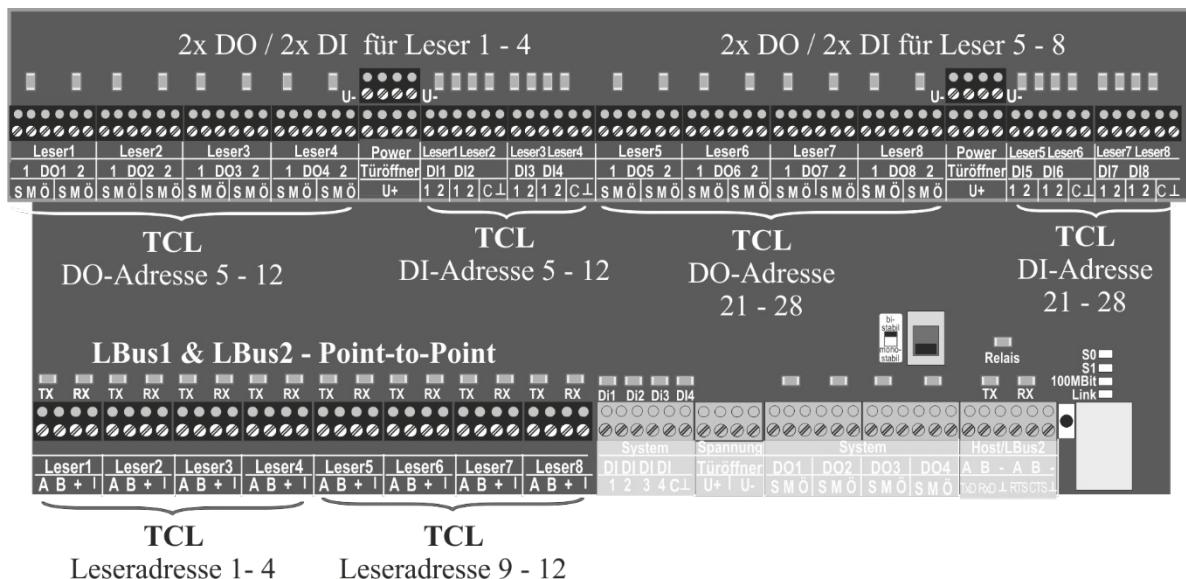


Abbildung 11.10 – TCL- Adressen INTUS ACM80e Wand LBus 1:PP/LBus 2: PP

### 11.4.2 Multipoint Verkabelung

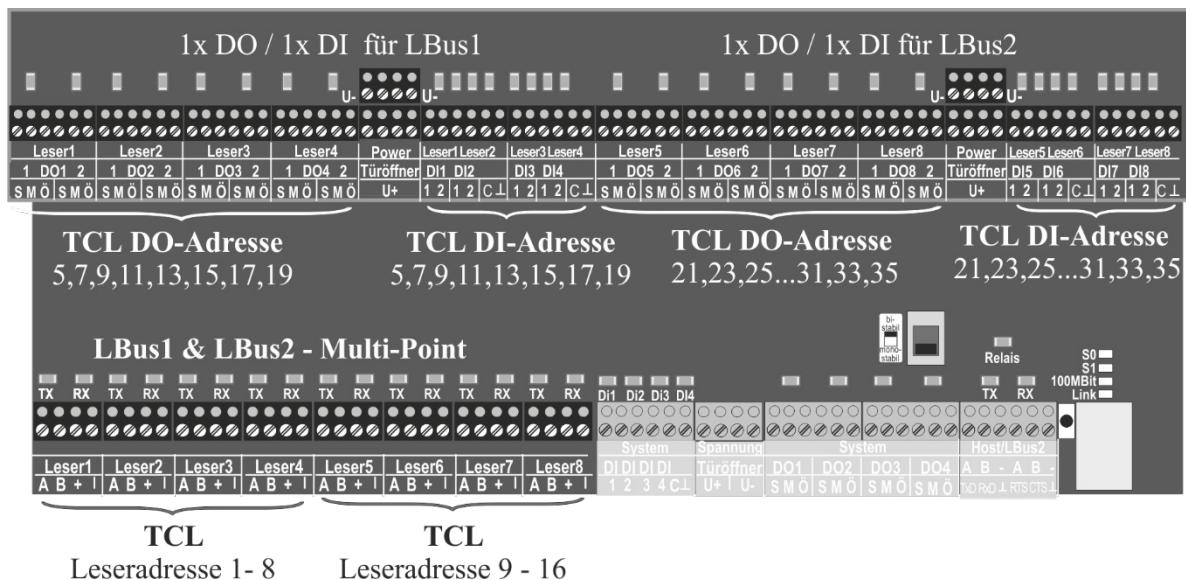


Abbildung 11.11 – TCL- Adressen INTUS ACM80e Wand LBus 1:MP/LBus 2: MP

### 11.4.3 DI /-DO-Adressen für Systemanwendungen

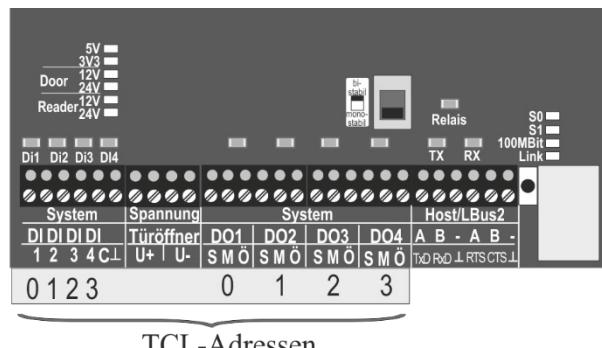


Abbildung 11.12 – TCL- Adressen INTUS ACM80e Wand für Systemanwendungen

## 11.5 INTUS ACM8e Wand

### 11.5.1 Point-to-Point Verbindung

**LBus 1: PP / LBus 2: nicht belegt (Voreinstellung)**

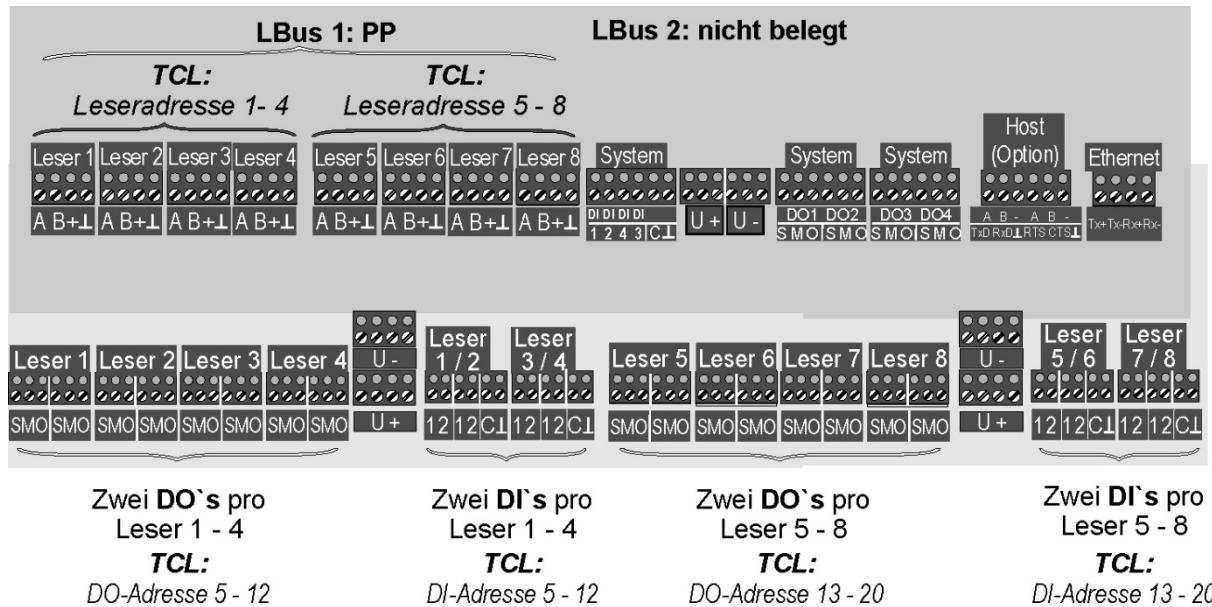


Abbildung 11.13 – TCL- Adressen INTUS ACM8e Wand LBus 1:PP/LBus 2: nicht belegt

**Alternative: LBus 1: PP / LBus 2: PP**

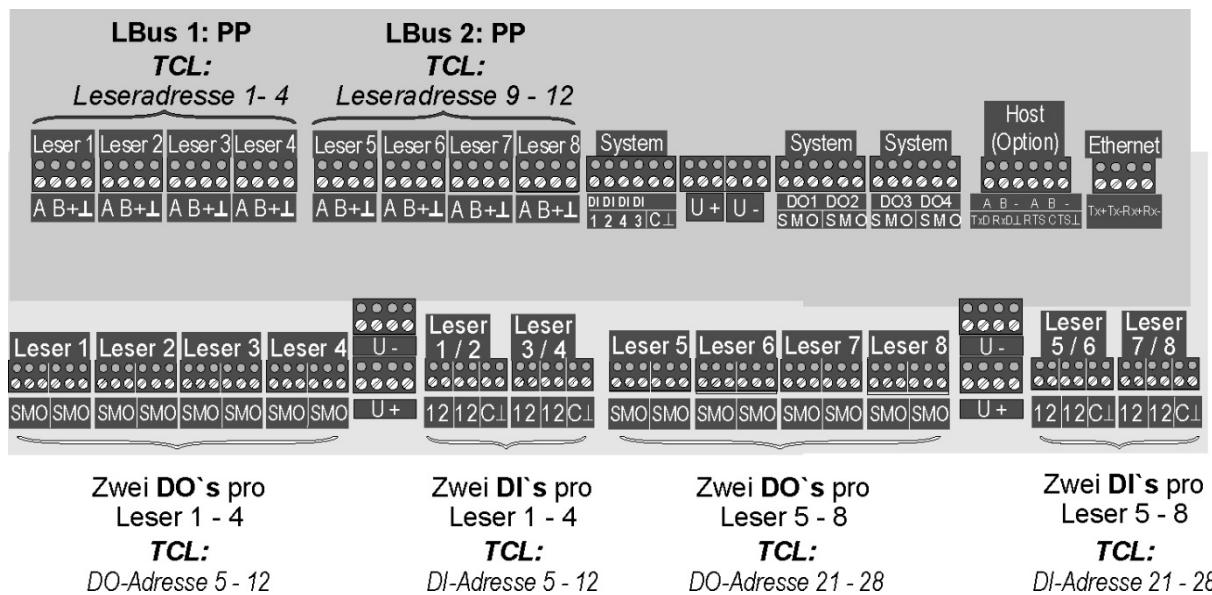


Abbildung 11.14 – TCL- Adressen INTUS ACM8e Wand LBus 1:PP/LBus 2: PP

### 11.5.2 Multipoint Verbindung

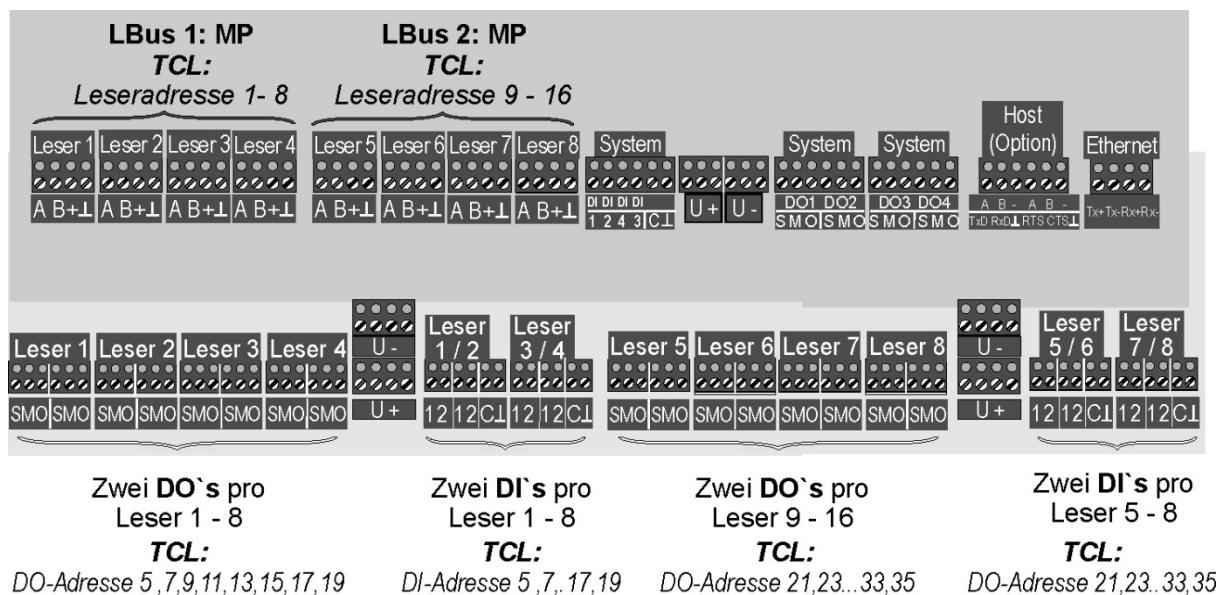


Abbildung 11.15 – TCL- Adressen INTUS ACM8e Wand LBus 1:MP/LBus 2: MP

### 11.5.3 DI /-DO-Adressen für Systemanwendungen

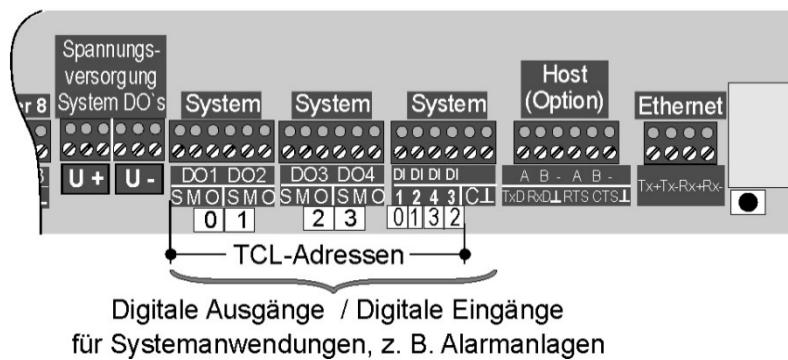


Abbildung 11.16 – TCL- Adressen INTUS ACM8e Wand für Systemanwendungen

## 11.6 INTUS 3000 ACM

### 11.6.1 LBus 1: MP / LBus 2: MP (Voreinstellung)

#### Moduladresse 0 (Voreinstellung)

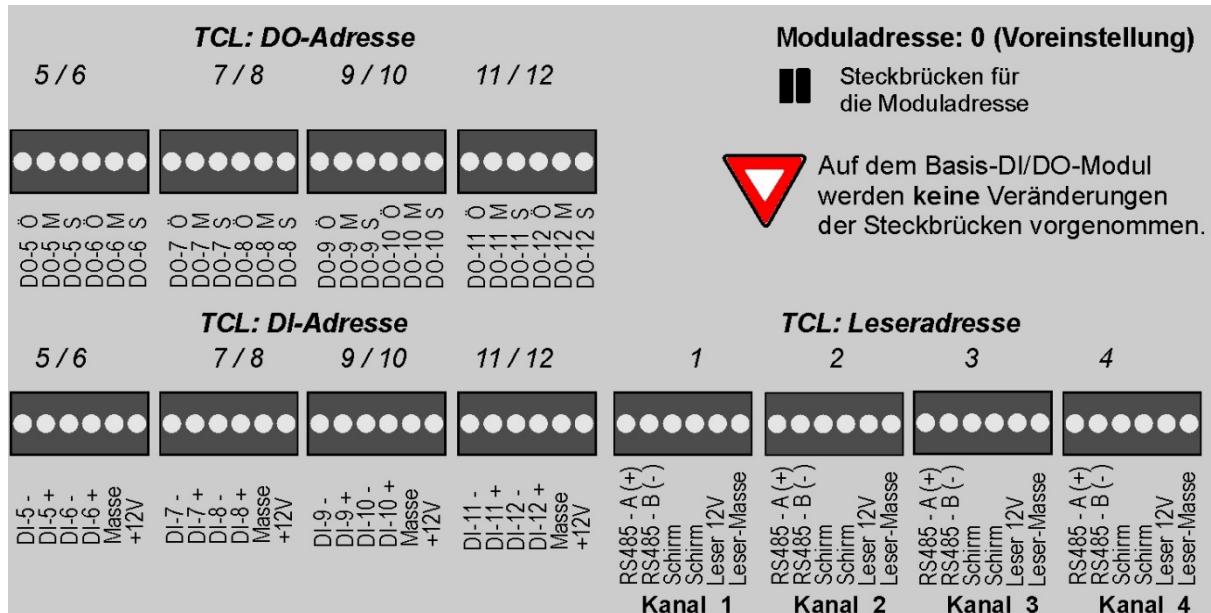


Abbildung 11.17 – TCL-Adressen INTUS 3000 ACM - MP/MP Moduladresse 0

#### Moduladresse 1

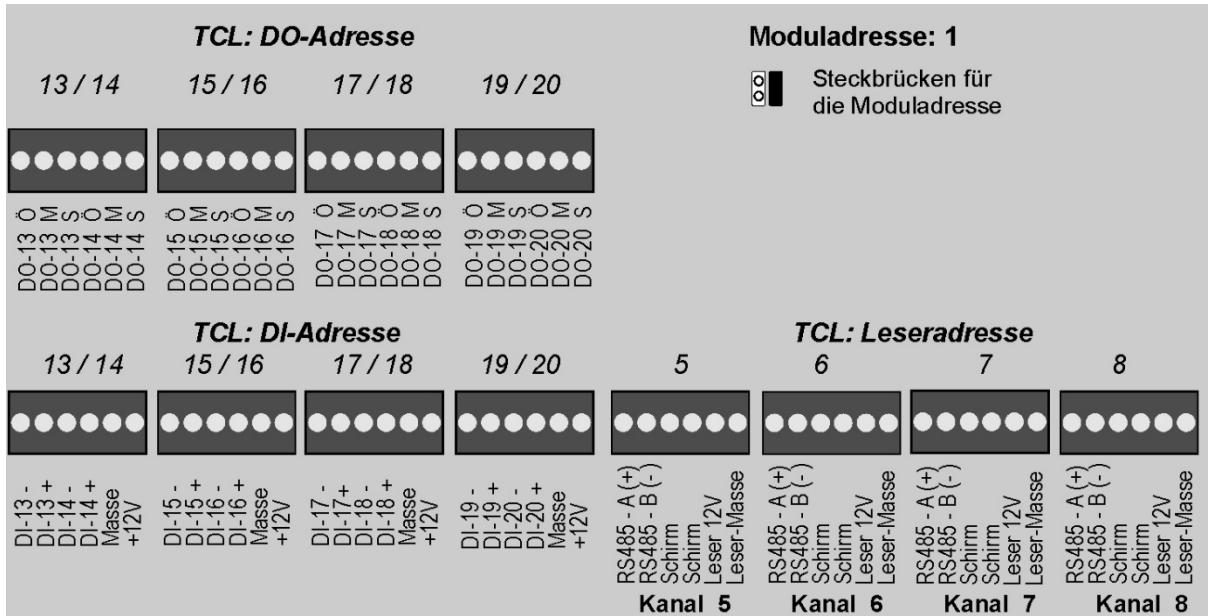


Abbildung 11.18 – TCL-Adressen INTUS 3000 ACM - MP/MP Moduladresse 1



### Moduladresse 2 – LBus 1: MP / LBus 2: MP sowie LBus 1: PP / LBus 2: PP

Die TCL Adressen gelten ebenso für LBus 1: PP / LBus 2: PP, wenn die Moduladresse 2 eingestellt ist.

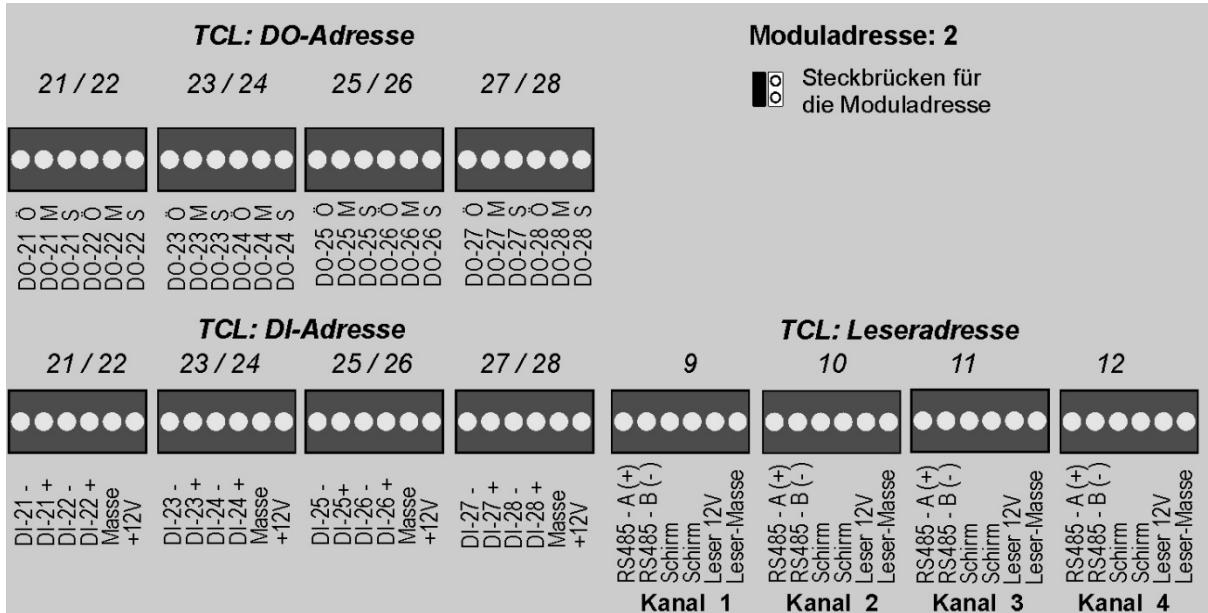


Abbildung 11.19 - TCL-Adressen INTUS 3000 ACM - MP/MP Moduladresse 2

### 11.6.2 LBus 1: PP / LBus 2: MP

#### Moduladresse 2

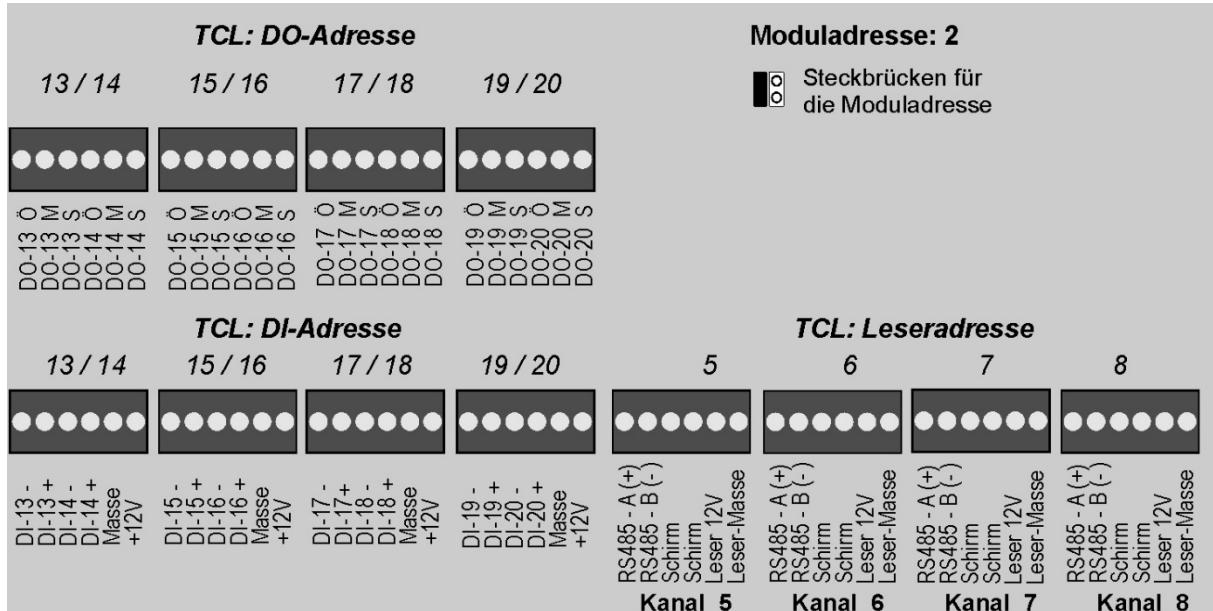


Abbildung 11.20 - TCL-Adressen INTUS 3000 ACM - PP/MP Moduladresse 2



## **12 Anhang**

### **Tabellenverzeichnis**

Tabelle 4.1 – Feldaufteilung beim digitalen Eingang .....	4-32
Tabelle 4.2 – Digitale Eingänge / Vandalenkontakt der Terminals .....	4-34
Tabelle 4.3 - Digitale Eingänge / Vandalenkontakt der abgesetzten Leser.....	4-35
Tabelle 4.4 – Hupe und Lampen der Terminals.....	4-46
Tabelle 4.5 – Hupe und Lampen der abgesetzten Leser.....	4-46
Tabelle 4.6 – Digitale Ausgänge der Terminals.....	4-65
Tabelle 4.7 – Digitale Ausgänge der abgesetzten Leser.....	4-66
Tabelle 5.1 – Zuordnung TCL Funktionstaste zu Beschriftung .....	5-16
Tabelle 5.2 –INTUS LBus Modi der Subterminals.....	5-18
Tabelle 5.3 – Subterminals mit Tastatur und Display, INTUS LBus Modi .....	5-19
Tabelle 5.4 – Zeichentypen für Terminals und Subterminals mit INTUS LBus Modus C .....	5-20
Tabelle 5.5 – Zeichentypen für verdeckte Eingabe am Terminal.....	5-20
Tabelle 5.6 – Zeichentyp für Subterminals mit INTUS LBus Modus A und B .....	5-21
Tabelle 6.1 – Prozesse im INTUS TCL.....	6-1
Tabelle 6.2 – Ringpuffer in INTUS TCL .....	6-2
Tabelle 6.3 – Format der LBus-Message .....	6-4
Tabelle 6.4 – Adressbytes für Datensätze an den MONIN .....	6-5
Tabelle 6.5 – Sendedatensatz mit optionalen Bestandteilen.....	6-10
Tabelle 6.6 – Zusammenfassung der P20-Teilfelder zur Uhrzeitkontrolle .....	6-25
Tabelle 6.7 – P20-Teilfelder für den aktuellen Zeitstatus .....	6-28
Tabelle 6.8 – P20-Teilfelder für die Zeitkontrolle .....	6-31
Tabelle 7.1 – Übersicht Displays .....	7-3
Tabelle 7.2 - Nicht darstellbare Latin-1 Zeichen.....	7-4
Tabelle 7.3 – Zeichengrößen beim 240x64 Pixel Display .....	7-5
Tabelle 7.4 - Cursorsteuerung .....	7-7
Tabelle 7.5 - Editorfunktionen .....	7-8
Tabelle 7.6 – Cursor Attribute.....	7-8
Tabelle 7.7 - Zeichenattribute .....	7-9
Tabelle 7.8 – Displaysteuerung .....	7-9
Tabelle 7.9 – TCL-Zeichensätze .....	7-10
Tabelle 7.10 – Inhalt der Zeichensatzspeicher .....	7-12
Tabelle 7.11 – Steuersequenzen zum Laden der Zeichensatzspeicher .....	7-13
Tabelle 7.12 – Steuersequenzen zur Zuordnung von GL und GR.....	7-13
Tabelle 8.1 – Format der TCL-Fehlermeldungen.....	8-9
Tabelle 9.1 - Systemfehler – Ursache und Behebung .....	9-10

## **Abbildungsverzeichnis**

Abbildung 1.1 – INTUS TCL Terminalfamilie.....	1-12
Abbildung 2.1 - Datenfluss zwischen Leitrechner und Terminal.....	2-2
Abbildung 2.2 - Datenfluss zwischen Terminal und Gerät an Kanal B .....	2-5
Abbildung 2.3 – Programm Konzept.....	2-6
Abbildung 2.4 – Beispiel eines parallelen Vorgangs .....	2-7
Abbildung 6.1 - Übersicht über die Funktionen der Taktüberwachung.....	6-23
Abbildung 7.1 – 8-Bit Zeichenausgabe .....	7-11
Abbildung 8.1 - Speicheraufteilung .....	8-1
Abbildung 11.1 – TCL- Adressen INTUS ACM40 LBus 1:PP/LBus 2: PP .....	11-2
Abbildung 11.2 – TCL- Adressen INTUS ACM40 LBus 1:PP/LBus 2: nicht belegt.....	11-2
Abbildung 11.3 – TCL- Adressen INTUS ACM4 LBus 1:PP/LBus 2: PP .....	11-3
Abbildung 11.4 – TCL- Adressen INTUS ACM4 LBus 1:PP/LBus 2: nicht belegt.....	11-3
Abbildung 11.5 - TCL- Adressen INTUS ACM8(0)e Rack LBus 1:PP/LBus 2: nicht belegt.....	11-4
Abbildung 11.6 – TCL- Adressen INTUS ACM8(0)e Rack LBus 1:PP/LBus 2: PP .....	11-4
Abbildung 11.7 – TCL- Adressen INTUS ACM8(0)e Rack LBus 1:MP/LBus 2: MP .....	11-5
Abbildung 11.8 – TCL- Adressen INTUS ACM8(0)e Rack für Systemanwendungen .....	11-5
Abbildung 11.9 – TCL- Adressen INTUS ACM80e Wand LBus 1:PP/LBus 2: nicht belegt .....	11-6
Abbildung 11.10 – TCL- Adressen INTUS ACM80e Wand LBus 1:PP/LBus 2: PP.....	11-6
Abbildung 11.11 – TCL- Adressen INTUS ACM80e Wand LBus 1:MP/LBus 2: MP .....	11-7
Abbildung 11.12 – TCL- Adressen INTUS ACM80e Wand für Systemanwendungen.....	11-7
Abbildung 11.13 – TCL- Adressen INTUS ACM8e Wand LBus 1:PP/LBus 2: nicht belegt .....	11-8
Abbildung 11.14 – TCL- Adressen INTUS ACM8e Wand LBus 1:PP/LBus 2: PP.....	11-8
Abbildung 11.15 – TCL- Adressen INTUS ACM8e Wand LBus 1:MP/LBus 2: MP .....	11-9
Abbildung 11.16 – TCL- Adressen INTUS ACM8e Wand für Systemanwendungen.....	11-9
Abbildung 11.17– TCL-Adressen INTUS 3000 ACM - MP/MP Moduladresse 0 .....	11-10
Abbildung 11.18 – TCL-Adressen INTUS 3000 ACM - MP/MP Moduladresse 1 .....	11-10
Abbildung 11.19 - TCL-Adressen INTUS 3000 ACM - MP/MP Moduladresse 2.....	11-11
Abbildung 11.20 - TCL-Adressen INTUS 3000 ACM - PP/MP Moduladresse 2 .....	11-11

## **Dokumenthistorie**

### **Änderungsstand 11 (September 2014)**

Der Änderungsstand 11 beschreibt **TCL Version 6.60**.

**TCL Version 6.60** unterstützt INTUS Sound, so dass folgende Abschnitte erweitert wurden:

- 4.34 (P22-Feld): Vergrößerung des P22-Felds für INTUS Sound.
- 4.6 (CV-Feld): Erweiterung des Wertebereichs von CV+8,1 und CV+22,1.

Zahlreiche Abschnitte wurden um Hinweise und Erläuterungen ergänzt:

#### **Kapitel 1: Einleitung**

- 1.4: Neu: Beschreibung der INTUS Terminalfamilie.

#### **Kapitel 4: TCL-Felder**

- 4.5 (CE-Feld): Klarstellung bezüglich CE+1,1.
- 4.20: (LS-Feld): Verbesserung der Beschreibung.
- 4.47: (TR-Feld): TR+9,1 Prozesskennung erweitert um BSCM-RX und BSCM-TX.

#### **Kapitel 5: TCL-Anweisungen**

- 5.11 (C4-Anweisung): Erklärung erweitert und Beispiel angefügt.
- 5.14 (E-Anweisung): Erklärung der verdeckten Zeicheneingabe und der neuen Formate P und Q verbessert.
- 5.15 (EO-Anweisung), 5.23 (OR-Anweisung), 5.37 (UN-Anweisung): Erklärung bzgl. Byte-Anzahl verbessert.
- 5.20 (KW-Anweisung): Beispiel 3 eingefügt (Quine).
- 5.26 (PT-Anweisung): Erklärung zum Offset im SP-Feld verbessert, zusätzliches Beispiel 4.
- 5.29 (RL- Anweisung), 5.30 (RR- Anweisung): Voreinstellung Byte-Anzahl erläutert.
- 5.36 (T-Anweisung): Timer-Nummern 0-99 seit TCL V6.0 möglich.
- 5.38 (WO-Anweisung) und 5.39 (WR-Anweisung) : Syntax-Beschreibung korrigiert.
- 5.40 (XA-Anweisung), 5.41 (XS-Anweisung): Erklärung bzgl. Voreinstellung Byte-Anzahl verbessert.

#### **Kapitel 6: Laufzeitsystem**

- Tabelle 6.1: TCL-Prozesstabelle um BSCM-RX und BSCM-TX erweitert.

#### **Kapitel 7: Displayansteuerung**

- 7.1 (Displaytypen) Abschnitt wurde überarbeitet.
- 7.3 (Zeichensätze und Zeichenmapping): Überarbeitet und weiteres Beispiel.

#### **Kapitel 8: TCL Programmentwicklung**

- 8.4.3 Anlaufmodi: Löschen der Archive beschrieben.

#### **Kapitel 9: Fehlermeldungen**

- Tabelle 9.1: Systemfehler um Status-LED und Hupe ergänzt.

Es wurden im gesamten Dokument Querverweise ergänzt und der Text auf die neue deutsche Rechtschreibung umgestellt. In den Abbildungen werden jetzt die aktuellen Terminal-Modelle verwendet.

## **Änderungsstand 12 (Oktober 2015)**

Der Änderungsstand 12 beschreibt **TCL Version 6.62** mit den Modellen **INTUS 5205** und **INTUS 5200**.

Zahlreiche Abschnitte wurden um Hinweise und Erläuterungen ergänzt:

### **Kapitel 4: TCL-Felder**

- 4.6 (CE-Feld): INTUS 5200/5205 ergänzt.
- 4.6 (CV-Feld): Klarstellung zu CV+46,1, CV+68,2 und CV+226,2.  
INTUS 5200/5205 führen zu neuen Werten und Zuordnungen in CV+25, 1, - CV+27,1, CV+76,1 und CV+228,3. Beschreibungen zum Sound in CV+8,1 und CV+22,1 um INTUS 5200 erweitert. Löschen der Logo-Datei in CV+8,1 ergänzt. CV+261,1 ist für die RW-Leserlizenz hinzugekommen.
- 4.10 (Ex-Feld): Tabelle 4.2 um INTUS 5200/5205 ergänzt, Beschreibung von L3 bei INTUS 5300/5500/5600 verbessert.
- 4.19 (Lx-Feld): Tabelle 4.4 um INTUS 5200/5205 ergänzt.
- 4.20: (LS-Feld): Erweiterung der Tabelle 1 um Werte für den Austausch von Compact-Flashes bei INTUS ACM8e.
- 4.25 (Ox-Feld): Tabelle 4.6 um INTUS 5200/5205 ergänzt.

### **Kapitel 5: TCL-Anweisungen**

- 5.14 (E-Anweisung): im Abschnitt 5.14.3 Funktionstasteneingabe ist die Tabelle zur Zuordnung der TCL-Funktionstaste zur Beschriftung der Tastauren ersetzt.

### **Kapitel 7: Ansteuerung des Displays**

- 7.1 Übersicht über die e Displays: 320x240 Pixel TFT Display in der Tabelle ergänzt.
- 7.1.5 320x240 Pixel TFT Display: Abschnitt angefügt.

### **Kapitel 8: TCL Programmentwicklung**

- 8.4.3 Anlaufmodi: Behandlung der Logo-Datei bei den einzelnen Anlaufmodi ergänzt.
- 8.5.5 Systemfehlermeldungen: fehlende RW-Leserlizenz führt zu neuem Fehler BSCM-TX:403.

### **Kapitel 9: Fehlermeldungen**

- 9.3.1 Konfigurationsfehler des Programmabreichs: Erklärung zu der Meldung „DE“ auf der Hostschnittstelle in neuem Abschnitt aufgenommen.
- 9.3.2 Prozess-Fehlermeldungen: fehlende RW-Leserlizenz führt zu neuem Fehler BSCM-TX:403.

## **Änderungsstand 13 (April 2017)**

Der Änderungsstand 13 beschreibt **TCL Version 6.20** und **6.70** mit den Modellen **INTUS ACM80e Rack**, **INTUS ACM80e Wand** und **INTUS 5540**.

Zahlreiche Abschnitte wurden um Hinweise und Erläuterungen ergänzt:

### **Kapitel 4: TCL-Felder**

- 4.6 (CE-Feld): INTUS 5540 ergänzt.
- 4.6 (CV-Feld): INTUS 5540 und die beiden ACM80e Modelle führen zu neuen Werten und Zuordnungen in CV+25, 1, CV+27,1, CV+76,1 und CV+228,3. Beschreibungen in CV+8,1 und CV+22,1 um INTUS 5540 erweitert.  
CV+262,46 neu für die PIN-Code Verifikation
- 4.10 (Ex-Feld): Tabelle 4.2 um INTUS 5540 und ACM80e Modelle ergänzt.
- 4.17 (I-Feld): Beschreibung zur Identifikation mit Biometrieleser verbessert.
- 4.19 (Lx-Feld): Tabelle 4.4 um INTUS 5540 und ACM80e Modelle ergänzt.
- 4.20: (LS-Feld): Erweiterung der Tabellen 4 und 6 zu Treiberfehlern um Werte für den Online/Offline-Sprung des BSC-Slave-Treibers für INTUS 5600/5540/5500/5200 ab TCL 6.70.
- 4.22 (M-Feld): Beschreibung zur Verifikation mit Biometrieleser in Kombination mit einem Proximityleser verbessert.
- 4.25 (Ox-Feld): Hinweis zu bistabilem Relais. Tabelle 4.6 um INTUS 5540 und ACM80e Modelle ergänzt.
- 4.32 (P20-Feld): Display Kontrast für 240x64 Display ergänzt.
- 4.38 (Sx-Feld): für INTUS 5600/5540/5500/5200 ab TCL 6.70 Sx+60,1 beim BSC-Protokoll ergänzt.

### **Kapitel 5: TCL-Anweisungen**

- 5.14 (E-Anweisung): im Abschnitt 5.14.3 Zuordnung der CLEAR-Taste zu F17 dokumentiert.
- 5.24 (P-Anweisung): neues Kapitel eingeführt, um die Syntaxerweiterung des Kommandos für die PIN-Code Verifikation zu beschreiben.
- 5.29 (RL-Anweisung): Erklärung des optionalen Parameters <Byteanz> präzisiert.
- 5.30 (RR-Anweisung): Erklärung des optionalen Parameters <Byteanz> präzisiert.

### **Kapitel 7: Ansteuerung des Displays**

- 7.1 Übersicht über die Displays: 480x272 Pixel TFT Display in der Tabelle ergänzt.
- 7.1.6 480x272 Pixel TFT Display: Abschnitt angefügt.

### **Kapitel 9: Fehlermeldungen**

- 9.4 mehrere Tabellenüberschriften verbessert.

### **Kapitel 11: TCL Adressen der Zutrittskontrollmanager**

- 11.3 Überschrift und Texte nur um ACM80e Rack ergänzt, da die RJ45-Plugs an denselben Positionen wie bei den älteren Modellen sind.
- INTUS ACM80e Wand als neues Kapitel 11.4 vor dem Kapitel für INTUS ACM8e Wand eingeschoben, da das Layout des Modells neu ist.

## **Haben Sie noch Fragen?**

Haben Sie einen Fehler entdeckt, vermissen Sie Informationen oder verstehen Sie etwas nicht?

Haben Sie einen Verbesserungsvorschlag oder eine Idee für eine Ergänzung?

Wir bemühen uns, das Handbuch so hilfreich wie möglich zu machen. Trotzdem kann einmal etwas vergessen oder übersehen werden. Und weil Handbuchbenutzer immer am besten wissen, was an einem Handbuch gut oder schlecht ist:

## **Rufen Sie uns an**

und sagen Sie uns, was wir noch besser machen können. Wir bedanken uns schon jetzt für die kleine Mühe.

Ihre PCS Systemtechnik GmbH

**PCS-Hotline: +49/ (0)89/68004-666**  
**Email: support@pcs.com**

*PCS. The terminal people.*®



[www.pcs.com](http://www.pcs.com)

PCS Systemtechnik GmbH

Pfälzer-Wald-Str. 36

81539 München

Fon +49-89-68004-550

[intus@pcs.com](mailto:intus@pcs.com)

Ruhrallee 311

45136 Essen

Fon +49-201-89416-0

Hofzeile 24

1190 Wien

Fon +43-1-3670-302

