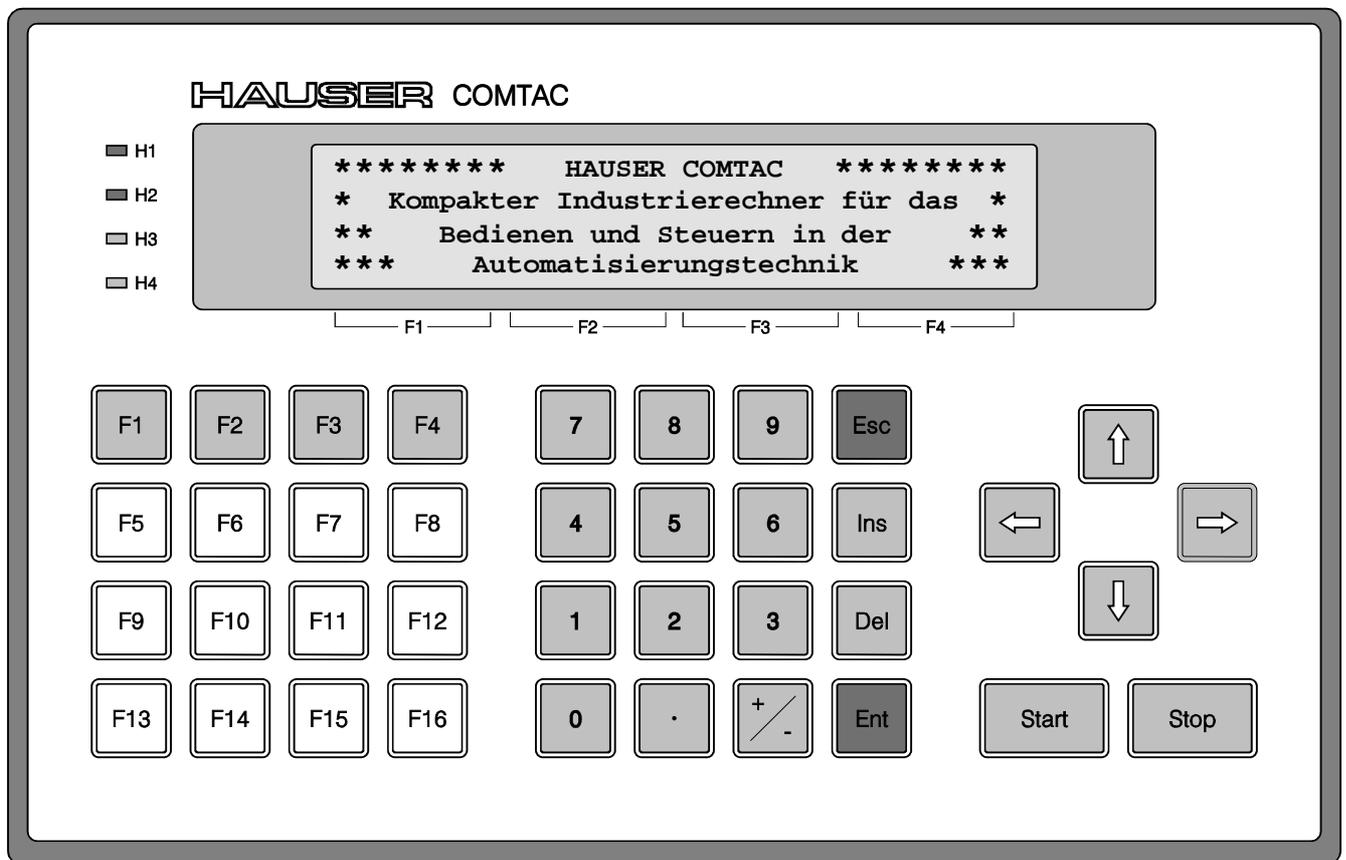


Befehlsbeschreibung COMTAC

Kompakter Industrierechner



Ab Softwareversion 2.60

März 2001

HAUSER
Wir automatisieren Bewegung



Parker Hannifin GmbH
EMD HAUSER
Postfach: 77607-1720
Robert-Bosch-Str. 22
D-77656 Offenburg, Germany
Tel.: +49 (0)781 509-0
Fax: +49 (0)781 509-176
<http://www.parker-emd.com>



1. Inhalt

1. Inhalt 2	
2. Gerätezuordnung	5
3. Befehlssyntax	6
4. Alphabetische Befehlsübersicht	7
5. Allgemeines	17
5.1 Die Betriebsarten	17
5.2 Die Programmiersprache	17
5.3 Daten-Format und Genauigkeit	17
5.4 Numerische Variablen	17
5.4.1 Variablennamen	17
5.4.2 Indizierte Variable	17
5.5 Zeichenketten	18
5.6 COMTAC-Basic Schlüsselwörter	18
5.7 Begriffe	18
5.8 Befehlskürzel	19
5.9 Arithmetische Operatoren u. Funktionen	19
5.10 Trigonometrische Funktionen	20
5.11 Vergleichende und logische Operatoren	20
5.12 Bit-Abfrage	20
5.13 Rangfolge der Operatoren und Funktionen	20
6. Programmstruktur	21
6.1 Unterprogramme	21
6.2 Labels	22
7. Programmerstellung	23
7.1 Terminal-Programm	23
7.2 Editierfunktionen	25
7.2.1 NEW	25
7.2.2 AUTO	25
7.2.3 COPY [C.] Basic-Zeilen	25
7.2.4 DEL Basic-Zeilen	26
7.2.5 COPYDEL	26
7.2.6 LIST	26
7.2.7 REN	27
7.3 Programmablauf	27
7.3.1 RUN	27
7.3.2 GOTO [GT.]	28
7.3.3 PSTEP [PS.]	28
7.3.4 CONT	28
7.3.5 ENABLE [EN./]DISABLE [DI.] STOP	28
7.4 Systemvariable	28
7.4.1 FREE	28
7.4.2 LEN()	29
7.4.3 MTOP	29
7.4.4 XTAL	29
8. Abspeichern und Laden von Programmen und Daten	30
8.1 Speichermedien/ Dateinamen	30
8.2 Checksummenbildung	30
8.3 Dateinamen	31
8.3.1 Programm-File	31
8.3.2 Daten-File	31
8.4 Format	31
8.5 DIR	31
8.6 STORE [S.]	31
8.7 LOAD [L.]	32
8.8 COPY [C.] Datei	32
8.9 COPY Disk	32
8.10 DEL Datei	32

8.11 RENAME	32
8.12 AUTO-START-Funktion	33
8.13 DEBUG-Funktion	33
9. Parameter	34
9.1 CONTROL [CT.]	34
9.2 Parameter - Übersicht	34
9.3 System-Parameter	34
9.4 Variable Überwachungszeit für den Up/Download	35
9.5 Passwortschutz	35
10. Allgemeine Anweisungen	36
10.1 CLEAR [CL.]	36
10.2 CLEARI	36
10.3 CLEARS	36
10.4 DIM	36
10.5 DIRDIM	37
10.6 DEL Variable	37
10.7 DEL DIM	37
10.8 DATA	38
10.9 READ	38
10.10 RESTORE	38
10.11 PUSH	38
10.12 POP	38
10.13 STOP	39
10.14 WAIT	39
10.15 REM [!]	39
11. Programmverzweigungen und Programmschleifen	40
11.1 GOTO [GT.] Zeilennummer	40
11.2 GOSUB [GS.]	40
11.3 ON	40
11.4 ON-Interrupt	40
11.5 RETURN [R.]	41
11.6 RETI	41
11.7 IF - THEN - (ELSE)	41
11.8 DO ... UNTIL	41
11.9 DO ... WHILE	42
11.10 FOR ... NEXT	42
12. Terminal-Ausgabe	43
12.1 CLEAR	43
12.2 PRINT [P.]	43
12.3 PH	43
12.4 PB.@	43
12.5 PRINT@ [P.@]	43
12.6 PH.@	44
12.7 PB	44
12.8 BEEP	44
12.9 SCALE	44
12.10 BARGRAPH	44
12.11 VLINE [V.]	44
12.12 HLINE [H.]	45
12.13 RECTANGLE [RECT.]	45
12.14 GCHR	45
12.15 CURSOR [CU.]	45
13. LCD-Ausgabe	46
13.1 CLEAR	46
13.2 DISP [D.]	46
13.3 DISPVAR	46
13.4 DH	46

13.5	DB	46	19.2	IN (digitaler Eingang)	63
13.6	CURSOR_D	47	19.3	BCDIN	64
13.7	SETLED	47	19.4	ONINP [OP.]	64
13.8	CLRLED	47	19.5	ENABLE [EN.]/DISABLE [DI.] ONINP [OP.]	65
14.	Ausgabeformatbefehle	48	19.6	SETOUT	65
14.1	TAB	48	19.7	CLROUT	65
14.2	TABXY [T.]	48	19.8	BINOUT [B.]	65
14.3	SPC	48	19.9	BCDOUT	66
14.4	CR	48	19.10	REQOUT	67
14.5	USING-Befehle	48	19.11	DYNOUT	68
14.6	USING [U.] (#.#)	49	19.12	KEYSWITCH	68
14.7	USING [U.] (Fx)	49	19.13	ONKEY 19 (Schüsselschalter)	68
14.8	USING [U.] (0)	49	19.14	ENABLE [EN.]/DISABLE [DI.] ONKEY 19 (Schüsselschalter)	68
14.9	USING [U.] (B)	49	19.15	EMY_STOP	68
15.	Terminal-Eingabe	51	19.16	ONEMY	69
15.1	INPUT [I.] (Tastatureingabe)	51	19.17	ENABLE [EN.]/DISABLE [DI.] ONEMY	69
15.2	INPUT [I.] TABXY [T.]	51	19.18	RDY	69
15.3	GET	51	19.19	ONRDY	69
15.4	ASK	52	19.20	ENABLE [EN.]/DISABLE [DI.] ONRDY	69
15.5	Anwendung der Funktionstasten	52	20.	Analoge Ein-/Ausgänge	70
16.	Folientastatur-Eingabe	53	20.1	VIN	70
16.1	INPKBD	53	20.2	VOUT	70
16.2	INPKBD TABXY [T.]	53	20.3	SLOPE	70
16.3	EDITVAR	53	21.	Interrupt Verarbeitung	71
16.3.1	STOP DISP	54	21.1	Interrupt Quellen	71
16.3.2	CONT DISP	54	21.1.1	Interrupt Initialisierung	72
16.4	ONKEY [OK.]	54	21.1.2	Interrupt Überwachung	72
16.5	ENABLE [EN.]/DISABLE [DI.] ONKEY [OK.]	54	21.1.3	Interrupt Überwachung freigeben	72
16.6	KBDCODE	55	21.1.4	Interrupt Überwachung sperren	72
16.7	ASKKBD	55	21.1.5	Interrupt-Verzweigung	72
16.8	ONKBD	56	21.1.6	Verzweigung sperren	72
16.9	ENABLE [EN.]/DISABLE [DI.] ONKBD	56	21.1.7	Interrupt-Verzweigung freigeben	72
17.	Echtzeituhr und Timer	57	21.1.8	Interrupt rücksetzen	72
17.1	TIME	57	21.1.9	Interrupt-Rücksprung	73
17.2	TIMEH	57	21.2	Interrupt-Priorität	73
17.3	TIMEM	57	21.2.1	Fehler Interrupt	75
17.4	TIMES	57	21.2.2	COMPAX Fehler Interrupt	75
17.5	ONTIME	57	21.2.3	Not-Stop Eingang Interrupt	75
17.6	ENABLE [EN.]/DISABLE [DI.] ONTIME	58	21.2.4	Timer Interrupt	76
17.7	DATE	58	21.2.5	Folientastatur Interrupt	76
17.8	DATEW	58	21.2.6	Schnittstellen Interrupt	76
17.9	DATED	58	21.2.7	Echtzeit-Uhr Interrupt	76
17.10	DATEM	58	21.2.8	Bereit Eingang Interrupt	76
17.11	DATEY	58	21.2.9	Schüsselschalter Interrupt	76
17.12	TIMER	59	21.2.10	Funktionstasten Interrupt	77
17.13	ENABLE [EN.]/DISABLE [DI.] ONTIMER [OC.]	59	21.2.11	Dig. Eingang Interrupt	77
17.14	ONTIMER [OC.]	59	21.3	IDLE (Warten auf Interrupt)	77
17.15	OFFTIMER	59	22.	RS485-Schnittstellen	78
18.	Verarbeitung von Zeichenketten	60	22.1	Funktionsbeschreibung	78
18.1	Zeichenketten oder "Strings"	60	22.2	Parameter der RS485/1-Schnittstelle (#1)	78
18.2	Speicherung von Zeichenketten	60	22.3	Parameter der RS485/2-Schnittstelle (#3) (Option F6)	79
18.2.1	Verkettung von Zeichenketten	60	22.4	Empfangen	80
18.3	Teilzeichenketten	60	22.5	Senden	80
18.4	Vordefinierte Zeichenketten	61	22.6	BCC RS485	80
18.5	Vergleich von Zeichenketten	61	22.7	OUTPUT [O.] RS485	81
18.6	LEN(\$x)	61	22.8	ENTER [E.] (RS485)	81
18.7	CHR\$	61	22.9	ON#1/#3	82
18.8	ASC	61	22.10	ENABLE [EN.] / DISABLE [DI.] (RS485)	82
18.9	VAL\$	62	22.11	RESET [RS.] RS485	82
18.10	VAL	62	22.12	STCTR#1/#3	82
19.	Digitale Ein/Ausgänge	63	23.	RS232-Schnittstelle	83
19.1	Feldbus E/A-Belegung	63	23.1	Funktionsbeschreibung	83

23.2	Parameter der RS232/1-Schnittstelle (#0)	83	26.21	FBUPD1	99
23.3	Parameter der RS232/2-Schnittstelle (#2)	84	27. COMPAX - Befehle		100
23.4	Parameter der RS232/3-Schnittstelle (#4) (Option F6)	84	27.1	DISPCPXS	100
23.5	Empfangen	85	27.2	DISPCXP	100
23.6	Senden	85	27.3	DISPCPXZ	100
23.7	BCC (RS232)	85	27.4	CPXSTEXT	101
23.8	OUTPUT [O.] (RS232)	85	27.5	CPXPTEXT	101
23.9	ENTER [E.] (RS232)	86	27.6	CPXPOSA [PA.]	101
23.10	ON#0 / #2 / #4	86	27.7	CPXPOSR [PR.]	101
23.11	ENABLE [EN.]/DISABLE [DI.] RS232	86	27.8	CPXPH [ZP.]	101
23.12	RESET [RS.] RS232	87	27.9	CPXSPEED [SD.]	101
23.13	STSCTR#0 / #2 / #4	87	27.10	CPXACCEL [AL.]	101
24. DATA-TEXT-FELD		88	27.11	CPXPARA [PAR.]	102
24.1	Allgemeines	88	27.12	CPXSTA [STA.]	102
24.2	Data-Text-Editor: DTFEDIT	88	27.13	CPXST [ST.]	102
24.3	DTFLLEN	88	27.14	CPXSP [SP.]	102
24.4	DTFLCNT	88	27.15	CPXBK [BK.]	102
24.5	DTFNFCT	88	27.16	CPXQT [QT.]	102
24.6	DTFVFCT	89	27.17	CPXON [P1.]	102
24.7	UMEM(x) [;(x)]	89	27.18	CPXOFF [P0.]	103
24.8	UMEMB(x) [;B(x)]	89	27.19	CPXBOFF [P0B.]	103
24.9	UMEM\$(Zeile) [;(Zeile)]	89	27.20	CPXIMASK [CIM.]	103
24.10	UMEM\$(Adresse,Länge) [;(Adresse,Länge)]	90	27.21	CPXCTR [CC.]	103
25. Fehlerbehandlung		91	27.22	CPXSTS [CS.]	104
25.1	ERRSTS [ES.]	91	27.23	CPXOMASK [COM.]	104
25.2	ERRSTS\$	91	27.24	CPXOUT [CO.]	104
25.3	ERRSTSL	91	27.25	CPXINP [CI.]	105
25.4	ERRSTS#	91	27.26	CPXOVR [CV.]	105
25.5	ONERR [OE.]	91	27.27	CPXPOS [CP.]	105
25.6	ENABLE [EN.]/DISABLE [DI.] ONERR [OE.]	91	27.28	WAITPOS	105
25.7	Fehlerbeschreibung	92	27.28.1	Warten auf "Position erreicht"	105
26. Feldbus-Schnittstelle		94	27.28.2	Abfrage auf "Position erreicht"	106
26.1	Funktionsbeschreibung	94	27.29	WAITMN	106
26.2	Parameter der Feldbus - Schnittstelle	94	27.29.1	Warten auf "Maschinennull erreicht"	106
26.3	Zykluszeiten	95	27.29.2	Abfrage auf "Maschinennull erreicht"	106
26.4	OUTPUT [O.](Feldbus)	95	27.30	ONCPXERR	106
26.5	ENTER [E.] (Feldbus)	95	27.31	DISABLE [DI.] / ENABLE [EN.] ONCPXERR	107
26.6	FBUS_S [FS.]	96	27.32	CPX	107
26.7	FBUS_I [FI.]	97	27.33	CPXERRADR	107
26.8	FBUS_O [FO.]	97	28. Beispielprogramm		108
26.9	FBUS_P [FP.]	97	29. Parameter, nach Nummern sortiert		109
26.10	FBUS_D [FD.]	97	Parameter - Übersicht	109	
26.11	On#1	98	System-Parameter	109	
26.12	ENABLE [EN.]/DISABLE [DI.] Feldbus	98	Parameter der RS232/3-Schnittstelle (#4) (Option F6)	110	
26.13	FBINTADR	98	Parameter der RS232/1-Schnittstelle (#0)	110	
26.14	FBINTREG	98	Parameter der RS485/1-Schnittstelle (#1)	111	
26.15	FBDSRQ(x)	98	Parameter der Feldbus - Schnittstelle	111	
26.16	FBDRSP(x)	99	22.3	Parameter der RS485/2-Schnittstelle (#3) (Option F6)	112
26.17	RESET [RS.] Feldbus	99	23.3	Parameter der RS232/2-Schnittstelle (#2)	112
26.18	FBDINFO(x)	99	30. Sachwortverzeichnis		114
26.19	FBDRES(x)	99			
26.20	FBUPD0	99			

2. Gerätezuordnung

Diese Dokumentation gilt für die Geräte:

COMTAC 2000
COMTAC 3000

Grundsätzlicher Aufbau des HAUSER-Typenschilds



Bezeichnung
der Optionen

Identnummer
des Geräts

Bezeichnung
des Geräts

Artikel-Nr.
des Geräts

Weitere Dokumentationen:

- ◆ Gerätebeschreibung
- ◆ Bedienungsanleitung zum COMTAC-Programmierool.

Begriffdefinition

Controlwörter: Parameter von COMTAC (Kurzform: CWs). Teilweise wird der Begriff "Controlwort" verwendet; auch der Befehl "Control" zum Lesen und Einstellen von Parametern ist vom Begriff "Controlwort" abgeleitet. Auf Parameter wurde umgestellt, um eine Einheitlichkeit zu COMPAX herzustellen.

Unterschiede zu früheren Softwareversionen kleiner V2.01:

Die Betriebssoftware ist bezüglich der Interruptverarbeitung nicht 100%ig abwärtskompatibel:

- Ein Interruptprogramm muß grundsätzlich mit der Anweisung RETI verlassen werden! Vorher war, zumindest teilweise, auch der Befehl RETURN möglich.
- Alle Interrupts haben nach dem Einschalten und Starten eines Programms die gleiche Priorität. Somit kann ein Interruptprogramm kein anderes mehr unterbrechen. Vorher hatte der Counter-Interrupt und die RS232-Schnittstellen höhere Priorität gegenüber anderen Interrupts, die Funktionstasteninterrupts konnten alle Interrupts unterbrechen. Dies ist insbesondere beim Aufbau von bedienergeführten Ein-Ausgabemasken zu berücksichtigen. Für jede Interruptquelle kann jetzt eine Priorität programmiert werden.

Sonstige Änderungen:

- Die Befehlsyntax des Befehls COUNTER wurde um den Begriff TIMER erweitert. Beide Begriffe sind zugelassen, bei einem Listing wird jedoch COUNTER in TIMER umgewandelt.
- Erhöhung des Systemtakt von 11,059 Mhz auf 24,576Mhz. Es ist zu beachten, daß sich dadurch alle Befehlsausführungszeiten verkürzen.

Neue Softwareversion V2.50

1. Bekannte Fehler behoben

2. Erweiterungen

Programmerstellung

- ◆ Während dem Compilervorgang blinkt die Led H4.
- ◆ Gezieltes Ein/Ausschalten der Uhrzeitanzeige (Infozeile des Terminals).
 - ◆ Mit Ctrl+W wird die Uhrzeitanzeige in der Infozeile aktiviert und mit Ctrl+O deaktiviert.
 - ◆ Mit Ctrl+P wird die Uhrzeitanzeige nicht mehr automatisch aktiviert.

Schnittstellen

- ◆ OUTPUT x
Vor einer neuen Zeichenausgabe prüft das Betriebssystem nun selbständig das Bit OUTPUT-Ready mit dem Befehl OUTPUT. Wie bei der ENTER-Anweisung wird nach Ablauf der vorgegebenen Timeoutzeit der Befehl abgebrochen, wenn das OUTPUT-Ready Bit nicht innerhalb dieser Zeit gesetzt wird. Das OUTPUT-Ready Bit muß nicht mehr vor jeder OUTPUT-Anweisung im STSCTR#x abgefragt werden.

Terminal/Folientastatur-Eingabe

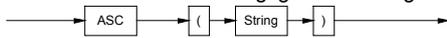
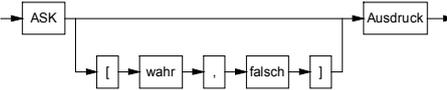
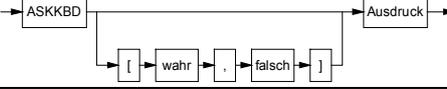
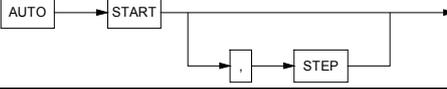
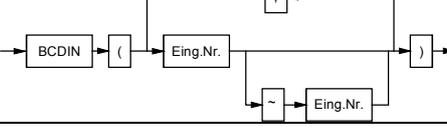
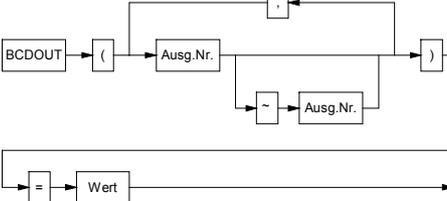
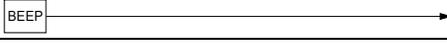
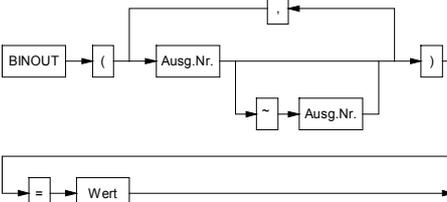
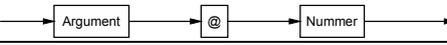
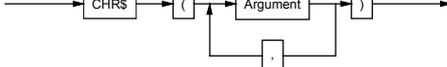
- ◆ INPUT, INPUT TABXY, INPKBD, INPKBD TABXY
Mit P13 Bit 0 = 1 kann die parallele Terminal/Folientastatur-Eingabe aktiviert werden; d.h. die Eingabe kann sowohl über die Tastatur des Terminals als auch über die Folientastatur erfolgen. Die Anzeige erfolgt entsprechend der programmierten Basic Anweisung; entweder auf dem Terminal (INPUT, INPUT TABXY) oder auf dem LCD (INPKBD, INPKBD TABXY).

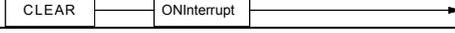
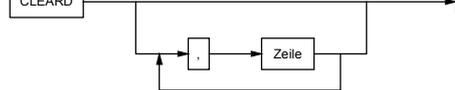
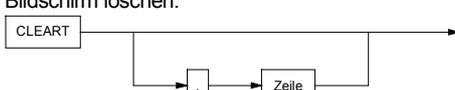
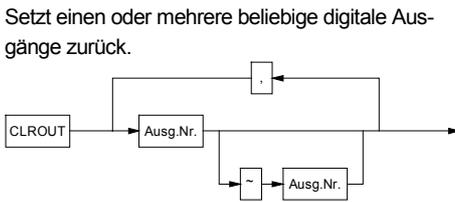
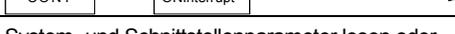
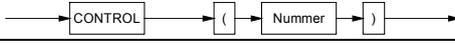
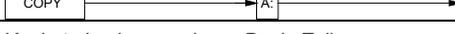
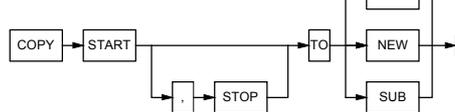
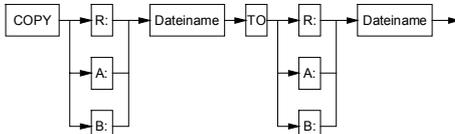
3. Befehlssyntax

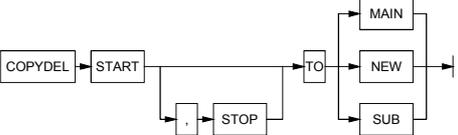
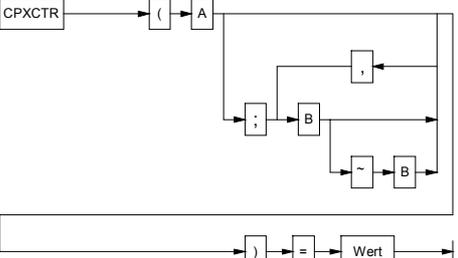
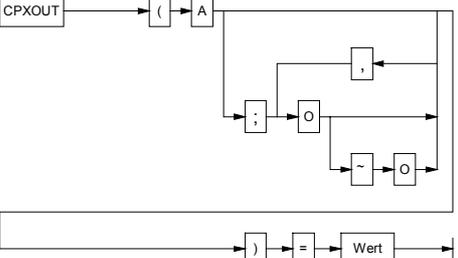
COMTAC verwendet die zeilenorientierte Programmiersprache BASIC. Die Syntax wird mit speziellen Strukturbildern dargestellt. Die Bedeutung der einzelnen Blöcke wird nachfolgend an Hand von Beispielen erläutert.

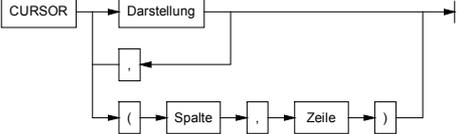
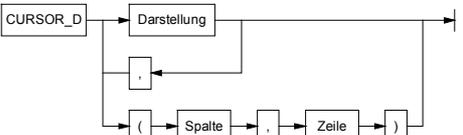
Befehl	Funktion Syntax	Bedeutung der Syntax-Strukturbilder
GOTO Zeilen- nummer	<p>Programmsprung zur angegebenen Zeilennummer.</p>	<p>Nach dem Befehl "GOTO" muß die Zeilennummer, SUB Zeilennummer oder MAIN Zeilennummer folgen. Der Strich als Abschluß bedeutet, daß nach der Zeilennummer der Befehl abgeschlossen ist. Beispiel: GOTO 30 oder GOTO 400</p>
RUN	<p>Löscht alle Variablen, setzt alle Interrupts zurück und startet das Anwender-Programm.</p>	<p>Die Befehlssyntax kann folgendermaßen aussehen:</p> <ul style="list-style-type: none"> ◆ RUN ◆ RUN 10 ◆ RUN 10-100 <p>D. h., jeder Zweig, der nach rechts führt kann verwendet werden.</p>
OUTPUT (RS232)	<p>Gibt Zeichen, Zeichenketten und/oder den Wert eines numerischen Ausdrucks über die RS232-Schnittstelle aus.</p>	<p>Nach OUTPUT muß die Schnittstellennummer "S" angegeben werden. Wahlweise kann nun eine Adresse mit vorstehendem Punkt oder Komma (unterschiedliche Bedeutung) angegeben werden. Nun muß ein ";" oder ein "\" folgen (unterschiedliche Bedeutung). Danach sind mehrere Anweisungen getrennt durch "," möglich:</p> <ul style="list-style-type: none"> ◆ numerischer Ausdruck ◆ Zeichenkette und ◆ 3 Funktionen (zu Erkennen an den Großbuchstaben) die separat beschrieben werden. <p>Beispiel: OUTPUT 0; "WINKEL=",W OUTPUT 2,5; "SPA",XPOS,CHR\$(10)</p>
TABXY	<p>Wird in der PRINT/DISP-Anweisung verwendet um den Cursor an die angegebenen Koordinaten des Bildschirms/Displays zu verschieben und die Darstellungsart der Ausgabe zu definieren.</p>	<p>Diese Anweisung wird für weitere Befehle verwendet; dies zeigt der Pfeil vor dem "TABXY"-Rechteck an. "Spalte" und "Zeile" muß angegeben werden; die "Darstellung" nur nach Bedarf. Der offene Pfeil am Schluß zeigt an, daß nachfolgende Anweisungen möglich sind. Beispiel: PRINT TABXY(10,10,4),A PRINT TABXY(10,15),B</p>

4. Alphabetische Befehlsübersicht

Befehl	Funktion. Syntax	Seite
ASC	Liefert den dezimalen Wert aus dem ASCII-Code für das erste Zeichen des angegebenen String. 	61
ASK	Tastaturabfrage (PC, Terminal) auf "Ja" oder "Nein". 	52
ASKKBD	COMTAC- Tastaturabfrage auf "Ja" oder "Nein". 	55
AUTO	Aktiviert die automatische Zeilennummerierung zur Programmerstellung. 	25
BARGRAPH	Bildschirm Ausgabe einer Balkengrafik. 	44
BCDIN	Zahlenwert im BCD-Format von den Eingängen lesen. 	64
BCDOUT	Zahlenwert im BCD-Format einem Ausgang zuweisen. 	66
BEEP	Terminal Tastatur-Beep. 	44
BINOUT	Zahlenwert im Binär-Format einem Ausgang zuweisen. 	65
Bitabfrage	Ein Bit aus einer Variablen auswählen. 	20
CHR\$	Numerischer Wert in ASCII-Zeichen umwandeln. 	61

CLEAR	Setzt alle Variablen auf 0, alle Strings und Interrupts werden gelöscht. 	36
CLEAR ONInterrupt	Interrupt in einen definierten Zustand setzen. 	72
CLEARD	Display löschen. 	46
CLEARI	Löscht alle Interrupts. 	36
CLEAR\$	Setzt den Control- und den Argument-Stack zurück. 	36
CLEART	Bildschirm löschen. 	43
CLRLLED	Schaltet die mit Nummer definierte LED (H1 - H4) des COMTAC aus. 	47
CLROUT	Setzt einen oder mehrere beliebige digitale Ausgänge zurück. 	65
CONT	Setzt das unterbrochene Programm fort. 	28
CONT DISP	Display - Inhalt wieder zur Anzeige bringen 	54
CONT ONInterrupt	Verriegelung desr Interrupts wieder aufheben 	72
CONTROL	System- und Schnittstellenparameter lesen oder verändern. 	34
COPY (Disk)	Laufwerk A auf Laufwerk B kopieren. 	32
COPY Basic-Zeilen	Kopiert eine bzw. mehrere Basic-Zeilen. 	25
COPY Datei	Kopiert Programm- und Datenfiles von Laufwerk 1 nach Laufwerk 2. 	32

COPYDEL	Kopiert eine bzw. mehrere Basic-Zeilen und löscht anschließend die Quell-Basic-Zeilen. 	26
CPX	Abfrage, ob ein COMPAX mit bestimmter Adresse am Bus angeschlossen ist. → CPX () Adresse)	107
CPXACCEL	Geschwindigkeitsvorgabe für COMPAX. → CPXACCEL () Adresse) = Wert	101
CPXBK	Break an COMPAX. CPXBK () Adresse)	102
CPXBOFF	Antrieb stromlos bei geschlossener Bremse (COMPAX). CPXBOFF () Adresse)	103
CPXCTR	Funktionen der COMPAX-Eingänge über das COMPAX-Steuerswort auslösen. 	103
CPXERRADR	Liefert die Adresse des COMPAX, das den ONCPXERR-Interrupt ausgelöst hat. → CPXERRADR	98
CPXIMASK	Freigabe der COMPAX-Eingangs-Funktionen E1 bis E16. CPXIMASK () Adresse) = Wert	103
CPXINP	Einlesen der Eingänge E16 bis E1 von einem COMPAX. → CPXINP () Adresse)	105
CPXOFF	Antrieb stromlos bei offener Bremse (COMPAX). CPXOFF () Adresse)	103
CPXOMASK	Maske für die Ausgänge A1 bis A16 von COMPAX. CPXOMASK () Adresse) = Wert	104
CPXON	Antrieb unter Moment bei offener Bremse (COMPAX). CPXON () Adresse)	102
CPXOUT	COMPAX-Ausgänge beschreiben. 	104
CPXOVR	Einstellen der COMPAX-Drehzahlreduzierung. CPXOVR () Adresse) = Wert	105

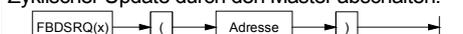
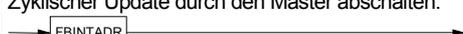
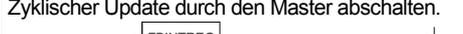
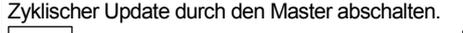
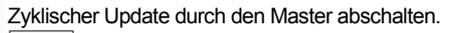
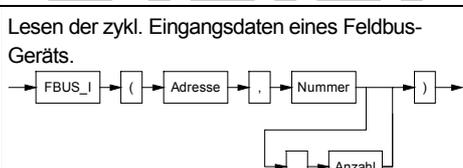
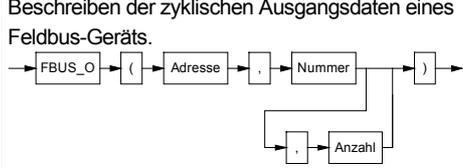
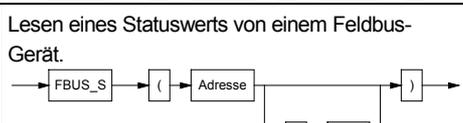
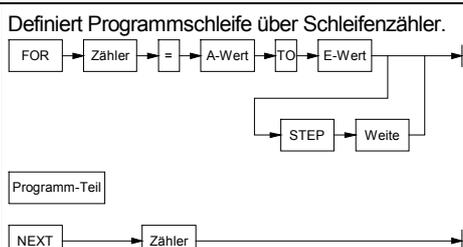
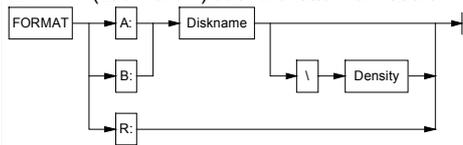
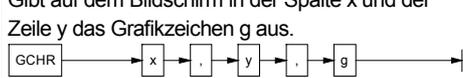
CPXPARA	Lesen oder Ändern eines COMPAX - Parameters → CPXPARA () Adresse , Parameternummer)	102
CPXPH	Maschinennullpunkt anfahren (COMPAX) CPXPH () Adresse)	101
CPXPOS	Einlesen der COMPAX-Istposition. → CPXPOS () Adresse)	105
CPXPOSA	Absolute Positionierung (COMPAX) → CPXPOSA () Adresse) = Position	101
CPXPOSR	Relative Positionierung (COMPAX) → CPXPOSR () Adresse) = Position	101
CPXPTEXT	Beschreibung für einen COMPAX-Parameter lesen. → CPXPTEXT Parameter-Nr.	101
CPXQT	Quit (COMPAX) CPXQT () Adresse)	102
CPXSP	STOP (COMPAX) CPXSP () Adresse)	102
CPXSPEED	SPEED (COMPAX) → CPXSPEED () Adresse) = Wert	101
CPXST	START (COMPAX) CPXST () Adresse)	102
CPXSTA	Lesen eines COMPAX - Statuswerts → CPXSTA () Adresse , Statusnummer)	102
CPXSTEXT	Beschreibung für einen COMPAX-Status lesen. → CPXSTEXT Status-Nr.	101
CPXSTS	COMPAX-Ausgangswort (A1.A16) lesen. → CPXSTS () Adresse)	104
CR	Gibt ein Carriage Return (ODH) aus. → CR	48
CURSOR	Verändert die Darstellung des Bildschirm-Cursors und / oder dessen Position. 	45
CURSOR_D	Verändert die Darstellung des LCD-Display-Cursors und / oder dessen Position. 	47
DATA	Anlegen eines Datenfelds. DATA () Wert	38
Data-Text-Editor: DTFEDIT	Data-Text-Editor aufrufen (zum Beschreiben und Lesen des Daten-Text-Felds). DTFEDIT	88
DATE	Liefert den formatierten Wert des aktuellen Datums. → DATE	58
DATED	Auslesen / Setzen des Tags des aktuellen Datums. → DATED	58

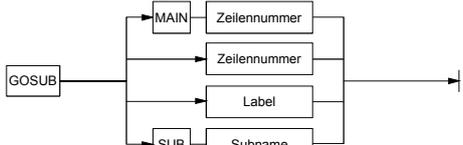
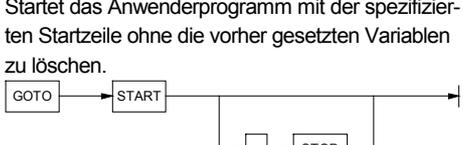
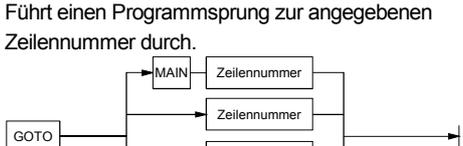
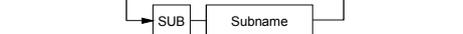
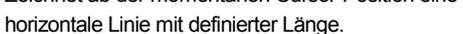
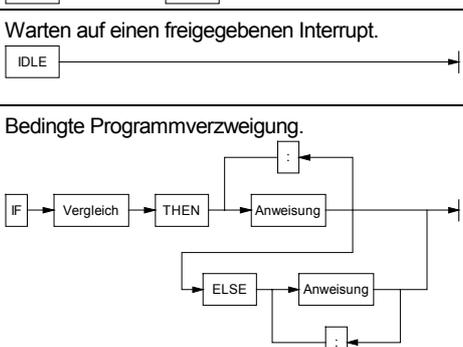
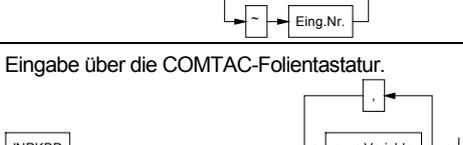
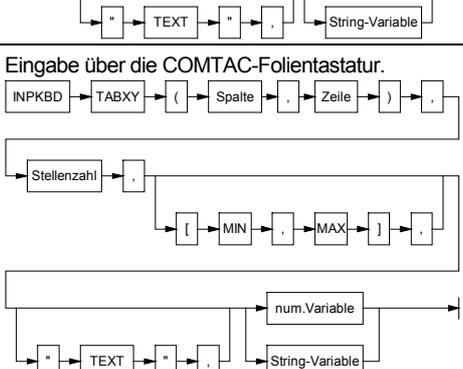
DATEM	Auslesen / Setzen des Monats des aktuellen Datums. 	58
DATEW	Auslesen / Setzen des aktuellen Wochentags. 	58
DATEY	Auslesen und/oder Setzen der Jahreszahl des aktuellen Datums. 	58
DB.	Gibt den Wert des Arguments im Binär-Format COMTAC-Display aus. 	46
DEL Basic-Zeilen	Löscht eine bzw. mehrere Basic-Zeilen. 	26
DEL Datei	Löscht Programm- oder Datenfiles auf dem angegebenen Laufwerk (A, B, oder R). 	32
DEL Variable	Löscht Matrizen die sich im ZP-RAM (Laufwerk R) befinden oder eine bzw. mehrere Basic-Zeilen. 	37
DELDIM	Löscht alle Matrizen in Laufwerk R. 	37
DH.	Gibt den Wert des Arguments im hexadezimalen Format im COMTAC-Display aus. 	46
DIM	Reserviert Speicherplatz für Zeichenketten oder für ein- / zweidimensionale Matrizen. 	36
DIR	Gibt das Inhaltsverzeichnis des angegebenen Speichermediums aus. 	31
DIRDIM	Abfrage von Größe und Speicherbelegung von Matrizen in Laufwerk R:. 	37

DISP	Gibt Zeichen, Zeichenketten und/oder den Wert eines numer. Ausdrucks im COMTAC-Display aus. 	46
DISPCXP	Einmaliges Lesen eines COMPAX-Parameters und Anzeige im Display. 	100
DISPCXS	Einmaliges Lesen eines COMPAX-Status und Anzeige im Display. 	100
DISPCXZ	Zyklisches Lesen eines COMPAX-Status und Anzeige im Display. 	100
DISPVAR	Zyklische Anzeige einer Basic-Variablen im Display 	46
DO ... UNTIL	Bedingte Programmschleife. 	41
DO ... WHILE	Bedingte Programmschleife. 	42
DTFLCNT	Systemvariable: Zeilenanzahl des Data-Text-Felds. 	88
DTFLLEN	Systemvariable: Zeilenlänge (Anzahl der Zeichen/Zeile) des Data-Text-Felds. 	88
DTFNFCT	Auslesen der n-ten Zahl aus der angegebenen Zeile. 	88

DTFVFC	Weist der, in der angegebenen Zeile des Data-Text-Felds stehenden Variable den unmittelbar folgenden Zahlenwert zu. 	89
DYNOUT	Setzt einen digitalen Ausgang zeitgesteuert. 	68
EDITVAR	Folientastatureingabe ohne Programmunterbrechung. 	53
EMY_STOP	Abfrage des Eingangs "Not Stop". 	68
Enable	Interrupte freigeben. 	72
ENABLE / DISABLE (RS232)	Freigabe/Sperren des RS232/1, /2, /3 - Interrupt. 	86
ENABLE / DISABLE (RS485)	Freigabe/Sperren des RS485/1, /2 - Interrupt. 	82
ENABLE / DISABLE ONCPXERR	Freigabe/Sperren des ONCPXERR-Interrupt. 	107
ENABLE/ DISABLE Feldbus	Freigabe/Sperren des Feldbus - Interrupts. 	98
ENABLE/ DISABLE ONEMY	Freigabe/Sperren des Not-Stop - Interrupts. 	69
ENABLE/ DISABLE ONERR	Freigabe/Sperren des ONERR-Interrupt. 	91
ENABLE/ DISABLE ONINP	Freigabe/Sperren des ONINP-Interrupt. 	65
ENABLE/ DISABLE ONKBD	Freigabe/Sperren des ONKBD-Interrupt. 	56

ENABLE/ DISABLE ONKEY	Freigabe/Sperren des ONKEY-Interrupts. 	54
ENABLE/ DISABLE ONRDY	Freigabe/Sperren des Bereit - Interrupts. 	69
ENABLE/ DISABLE ONTIME	Freigabe/Sperren des Uhrzeit Interrupt. 	59
ENABLE/ DISABLE ONTIMER	Freigabe/Sperren des internen Timers. 	59
ENABLE/ DISABLE STOP	Freigabe/Sperren von Ctrl-C, F8. 	28
ENTER (Feldbus)	Liest Zeichen oder eine Zeichenkette von einem Feldbus-Gerät in den String \$(#1). 	95
ENTER (RS232)	Übergibt ein Zeichen oder eine Zeichenkette aus dem Ringpuffer der RS232 in den String \$(#0) \$(#2) bzw. \$(#4). 	86
ENTER (RS485)	Übergibt ein bereitstehendes Zeichen oder eine Zeichenkette aus dem Ringpuffer der RS485-Schnittstelle in den String \$(#1) bzw. \$(#3). 	81
ERRSTS	Systemvariable: Fehlernummer des zuletzt aufgetretenen Fehlers. 	91
ERRSTS#	Systemvariable: Fehlermeldung des zuletzt aufgetretenen Fehlers. 	91
ERRSTS\$	Systemvariable: Fehlermeldung des zuletzt aufgetretenen Fehlers. 	91
ERRSTSL	Systemvariable: Zeilennummer der Basic-Zeile, in der ein Fehler festgestellt wurde. 	91
FBDINFO	Klartextinfo über einen Feldbus-Slave-Gerät. 	99

FBDRES	Klartextinfo über einen Feldbus-Slave-Gerät. 	99
FBDRSP	Zyklischer Update durch den Master abschalten. 	99
FBDSRQ	Zyklischer Update durch den Master abschalten. 	98
FBINTADR	Zyklischer Update durch den Master abschalten. 	98
FBINTREG	Zyklischer Update durch den Master abschalten. 	98
FBUPD0	Zyklischer Update durch den Master abschalten. 	99
FBUPD1	Zyklischer Update durch den Master abschalten. 	99
FBUS_D	Feldbus-Slave-Gerät zurücksetzen. 	97
FBUS_I	Lesen der zykl. Eingangsdaten eines Feldbus-Geräts. 	97
FBUS_O	Beschreiben der zyklischen Ausgangsdaten eines Feldbus-Geräts. 	97
FBUS_P	Lesen und Beschreiben des durch Nummer selektierten Parameters eines Feldbus-Geräts. 	97
FBUS_S	Lesen eines Statuswerts von einem Feldbus-Gerät. 	96
FOR ... NEXT	Definiert Programmschleife über Schleifenzähler. 	42
FORMAT	ZP-RAM (Laufwerk R) oder Disketten formatieren 	31
FREE	Systemvariable: Verfügbare Bytes im Arbeitssp. 	28
GCHR	Gibt auf dem Bildschirm in der Spalte x und der Zeile y das Grafikzeichen g aus. 	45
GET	Abfrage der Terminal-Tastatur. 	51

GOSUB	Ruft ein Unterprogramm auf. 	40
GOTO	Startet das Anwenderprogramm mit der spezifizierten Startzeile ohne die vorher gesetzten Variablen zu löschen. 	28
GOTO Zeilennummer oder SUB	Führt einen Programmsprung zur angegebenen Zeilennummer durch. 	40
HLINE	Zeichnet ab der momentanen Cursor-Position eine horizontale Linie mit definierter Länge. 	45
IDLE (Warten auf Interrupt)	Warten auf einen freigegebenen Interrupt. 	77
IF - THEN - (ELSE)	Bedingte Programmverzweigung. 	41
IN (digitaler Eingang)	Logischen Zustand eines digitalen Eingangs oder einer def. Eingangsgruppe als Zahlenwert einlesen. 	63
INPKBD	Eingabe über die COMTAC-Folientastatur. 	53
INPKBD-TABXY	Eingabe über die COMTAC-Folientastatur. 	53

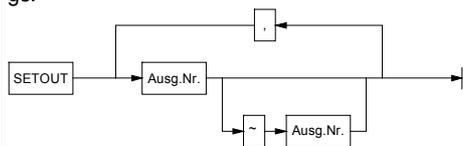
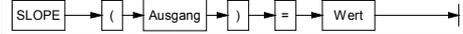
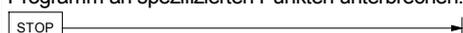
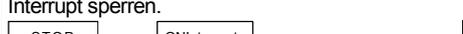
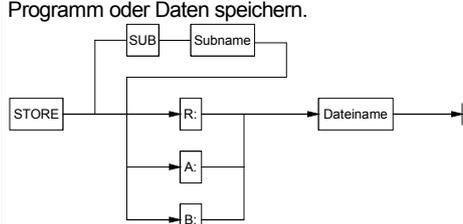
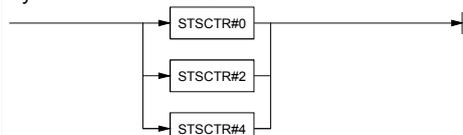
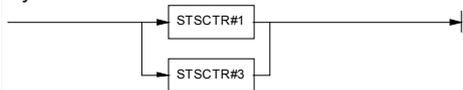
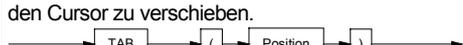
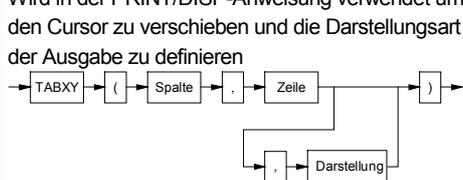
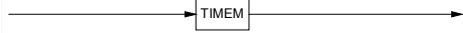
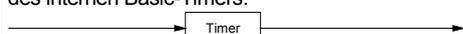
INPUT (Tastatur-eingabe)	Eingabe über die Tastatur. 	51
INPUT-TABXY	Eingabe über die Tastatur. 	51
KBDCODE	Abfrage der COMTAC-Folien-Tast. (beliebige Taste). 	55
KEYSWITCH	Abfrage des Schlüsselschalters (Stecker X7, S1). 	68
LEN()	Systemvariable: Länge des aktuellen Programms im Arbeitsspeicher (RAM). 	29
LEN(String)	Liefert die aktuelle Länge eines Strings. 	61
LIST	Gibt das aktuelle Programm im Arbeitsspeicher an das Terminal oder an eine Schnittstelle aus. 	26
LOAD	Lädt ein Programm- oder Datenfile aus dem angegebenen Laufwerk (R, A, oder B) in den Arbeitsspeicher (RAM) von COMTAC. 	32
MTOP	Systemvariable: Adresse des letzten verfügbaren Byte im Arbeitsspeicher (RAM). 	29
NEW	Löscht das aktuelle Programm im Arbeitsspeicher, alle Strings und Interrupts und setzt alle Variablen auf 0. 	25
OFFTIMER	Löscht den Befehl ONTIMER, sperrt jedoch nicht den internen Zähler TIMER. 	59
ON	Bedingte mehrfache Programmverzweigung. 	40

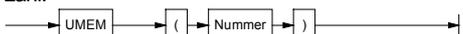
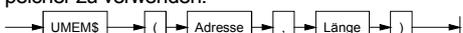
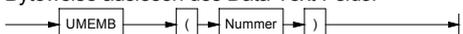
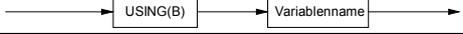
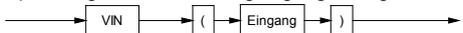
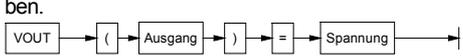
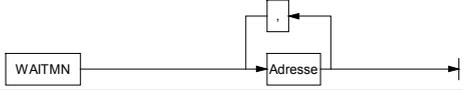
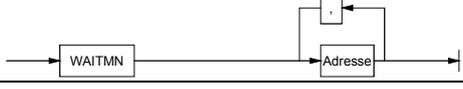
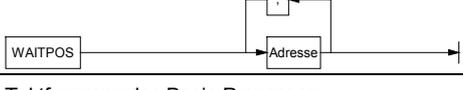
ON#0 / #2 / #4	Programmverzweigung, wenn ein String im Zwischenpuffer einer RS232-Schnittstelle bereitsteht und dieser Interrupt freigegeben ist. 	86
ON#1	Programmverzweigung, wenn von einem Feldbus-Gerät eine Interrupt-Meldung ansteht. 	98
ON#1/#3	Programmverzweigung, wenn ein String im Zwischenpuffer der RS485-Schnittstelle bereitsteht und dieser Interrupt freigegeben ist. 	98
ON-Interrupt	Programmverzweigung durch ein Interrupt. 	40
ONCPXERR	Programmverzweigung, wenn ein COMPAX einen Fehler meldet und der ONCPXERR-Interrupt freigegeben ist. 	106
ONEMY	Programmverzweigung, bei Flanke am Not-Stop-Eingang. 	69
ONERR	Programmverzweigung, wenn ein Fehler während der Programmabarbeitung erkannt wird und der ONERR-Interrupt freigegeben ist. 	91

<p>ONINP</p>	<p>Programmverzweigung, wenn der entsprechende Eingang den Interrupt - Zustand angenommen hat und der ONINP-Interrupt freigegeben ist.</p>	<p>64</p>
<p>ONKBD</p>	<p>Programmverzweigung, wenn eine Taste der COMTAC-Folien-Tastatur betätigt oder losgelassen wird und der ONKBD-Interrupt freigegeben ist.</p>	<p>56</p>
<p>ONKEY 19</p>	<p>Programmverzweigung, wenn die entsprechende Funktionstaste betätigt wird und der ONKEY-Interrupt freigegeben ist.</p>	<p>54</p>
<p>ONRDY</p>	<p>Programmverzweigung, bei Signal am Bereit-Eingang.</p>	<p>69</p>
<p>ONTIME</p>	<p>Programmverzweigung, wenn die Echtzeit-Uhr die angegebene Uhrzeit erreicht und der ONTIME-Interrupt freigegeben ist.</p>	<p>59</p>
<p>ONTIMER</p> <p>TCR: Timer-Compare-Register</p>	<p>Unterprogramm-Aufruf, wenn der interne Zähler TIMER den Vorgabewert überschreitet.</p>	<p>59</p>
<p>OUTPUT (Feldbus)</p>	<p>Gibt Zeichen, Zeichenketten und/oder den Wert eines numer. Ausdrucks über den Feldbus aus.</p>	<p>95</p>
<p>OUTPUT (RS232)</p>	<p>Gibt Zeichen, Zeichenketten und/oder den Wert eines numer. Ausdrucks über die RS232 aus.</p>	<p>85</p>
<p>OUTPUT (RS485)</p>	<p>Gibt Zeichen, Zeichenketten und/oder den Wert eines numer. Ausdrucks über die RS485 aus.</p>	<p>81</p>
<p>PB.</p>	<p>Gibt einen binären Wert an den Bildschirm aus.</p>	<p>44</p>
<p>PB.@</p>	<p>Gibt einen binären Wert an den Bildschirm in Zeile 22 aus.</p>	<p>43</p>
<p>PH.</p>	<p>Gibt einen hexadezimalen Wert an den Bildschirm aus.</p>	<p>43</p>

PH.@	Gibt einen hexadezimalen Wert an den Bildschirm in Zeile 22 aus. 	44
POP	Weist Werte aus dem Stack einer Variablen zu. 	38
PRINT	Gibt Zeichen, Zeichenketten und/oder den Wert eines numer. Ausdrucks über Bildschirm oder Drucker aus. 	43
PRINT@	Gibt Zeichen, Zeichenketten und/oder den Wert eines numer. Ausdrucks in Zeile 22 des Bildschirms aus. 	43
PSTEP	Programm im Arbeitsspeicher schrittweise abarbeiten. 	28
PUSH	Stackübergabe: zur Parameterübergabe an Unterprogramme und zum Zwischenspeichern. 	38
RDY	Abfrage des Eingangs "Bereit" an X7. 	69
READ	Lesen eines Datenfelds. 	38
RECTANGLE	Rechteck auf den Bildschirm zeichnen. 	45
REM	Einfügen von Kommentarzeilen in das Basic-Programm. 	39

REN	Numeriert die Basic-Zeilen neu. 	27
RENAME	Dateiname ändern. 	32
REQOUT	Setzt einen dig. Ausgang, bis ein frei wählbarer dig. Eingang in einer def. Zeit einen best. Zustand hat. 	67
RESET Feldbus	Setzt Feldbus in den Power-On Zustand zurück. 	99
RESET RS232	Setzt die RS232/1 /2 bzw. /4 in den Power-On Zustand. 	87
RESET RS485	Setzt RS485 in den Power-On Zustand zurück. 	82
RESTORE	Setzt den READ-Zeiger auf den 1. DATA-Wert. 	38
RETI	Abschluß für Unterprogramme, die durch einen Interrupt aufgerufen wurden. 	41
RETURN	Abschluß eines Unterprogramms (allg.). 	41
RUN	Löscht alle Variablen, setzt alle Interrupts zurück und startet das Anwender-Programm. 	27
SCALE	Gibt auf dem Bildschirm einen Maßstab aus. 	44
SETLED	Schaltet eine LED (H1 - H4) des COMTAC ein. 	47

SETOUT	Setzt einen o. mehrere beliebige digitale Ausgänge. 	65
SLOPE	Vorgabe einer Rampenzeit für Analogausgänge. 	70
SPC	Ausgabe von Leerzeichen (20H). 	48
STOP	Programm an spezifizierten Punkten unterbrechen. 	39
STOP DISP	Anzeige zwischenspeichern. 	54
STOP ON Interrupt	Interrupt sperren. 	72
STORE	Programm oder Daten speichern. 	31
STSCTR#0 /#2 /#4	Systemvariable: Statusinformation der RS232. 	87
STSCTR#1 /#3	Systemvariable: Statusinformation der RS485. 	82
TAB	Wird in der PRINT/DISP-Anweisung verwendet um den Cursor zu verschieben. 	48
TABXY	Wird in der PRINT/DISP-Anweisung verwendet um den Cursor zu verschieben und die Darstellungsart der Ausgabe zu definieren 	48
TIME	Liefert die aktuelle Uhrzeit. 	57
TIMEH	Auslesen oder Setzen der Stunden der akt. Uhrzeit. 	57
TIMEM	Auslesen oder Setzen der Minuten der akt. Uhrzeit. 	57
TIMER	Funktionsvariable zum Auslesen und / oder Setzen des internen Basic-Timers. 	59
TIMES	Auslesen oder Setzen der Sekunden der akt. Uhrzeit. 	57

UMEM (X)	Auslesen des Data-Text-Felds als Fließkommazahl. 	89
UMEM\$ (Adresse, Länge)	String-Variable um das Data-Text-Feld als Stringspeicher zu verwenden. 	90
UMEM\$ (Zeile)	String-Variable um jede Zeile des Data-Text-Felds als Stringspeicher zu behandeln. 	89
UMEMB (X)	Byteweise auslesen des Data-Text-Felds. 	90
USING(##)	Dezimalen Zahlenformat definieren. 	50
USING(0)	Freies Zahlenformat definieren. 	50
USING(B)	Internes binäres Zahlenformat definieren. 	50
USING(Fx)	Exponentielles Zahlenformat definieren. 	50
VAL	Wandelt einem String in einen numerischen Wert. 	63
VAL\$	Wandelt einen numerischen Wert in einen String um. 	63
VIN	Spannung an einem Analogeingang abfragen. 	71
VLINE	Zeichnet eine vertikale Linie auf den Bildschirm. 	45
VOUT	Spannung über Analogausgang (Option) ausgeben. 	71
WAIT	Wartezeit programmieren. 	40
WAITMN	Warten auf "Maschinennull erreicht". 	107
WAITMN	Warten auf "Position erreicht". 	107
WAITPOS	Abfrage von "Position erreicht". 	106
XTAL	Taktfrequenz des Basic-Prozessors. 	30

5. Allgemeines

5.1 Die Betriebsarten

Das COMTAC unterscheidet zwischen den beiden Betriebsarten COMMAND-Mode und RUN-Mode. Die Eingabe von Programmzeilen und die direkte Ausführung von Kommandos oder Anweisungen ist nur im COMMAND-Mode möglich. Im RUN-Mode arbeitet COMTAC das im Arbeitsspeicher vorhandenen aktuelle Basic-Programm ab.

5.2 Die Programmiersprache

COMTAC verwendet die zeilenorientierte Sprache BASIC. Neben den Standard-Basic-Befehlen gibt es spezielle Anweisungen für die Bedienung der COMTAC - Hardware z.B. für die dig. Ein-/Ausgänge. Außerdem existieren besondere Befehle zur Steuerung von COMPAX (Kompakte Servo-Steuerung) via Feldbus, welche den Programmieraufwand um einiges minimieren. Wird eine solche Anweisung zusammen mit einer Zeilennummer per RETURN Taste eingegeben, so wird daraus eine Programmzeile. Wird diese Anweisung hingegen ohne Zeilennummer per RETURN-Taste direkt ausgeführt, so nennen wir dies ein Kommando. Es gibt einige Kommandos, die nicht als Programmzeilen eingegeben werden können, wie etwa LIST und NEW. Umgekehrt gibt es Anweisungen z.B. GOSUB und RETURN, die nicht als Kommandos zur Verfügung stehen. Jedoch sind die meisten Anweisungen sowohl programmierbar wie auch unmittelbar ausführbar, z.B. PRINT und DISP.

5.3 Daten-Format und Genauigkeit

Der gültige Zahlenbereich für COMTAC-Basic ist: $\pm 1E-127$ bis $\pm 0,999\ 999\ 99\ E+127$
 Floating point (Fließkomma-Arithmetik) berücksichtigt 8 Dezimalstellen und rundet intern die Zahlen zur Anpassung an diese Genauigkeit. Zahlen können in 5 verschiedenen Formaten eingegeben werden:

ganzzahlig	dezimal	hexadezimal	binär	Floating point
Beispiel:				
129	88.32	0E7B5H	01011B	1.2345E+3

◆ Für die hexadezimale und binäre Ein- und Ausgabe ist der Zahlenbereich begrenzt auf 0 - 65 535.

- ◆ Eine hexadezimale Zahl muß mit einer Ziffer beginnen (z.B. die Zahl A000H muß als 0A000H eingegeben werden).
- ◆ Alle Zahlen werden intern in Fließkomma-Zahlen (Floating point) gespeichert und benötigen dafür 6 Byte Speicherplatz. Die Zahl PI (3.1415926) z.B. würde im Speicher an der Stelle x wie folgt abgelegt:

Speicherplatz	Wert	Bedeutung
X	81H	Exponent 81H=10 ¹ ,82H=10 ² 80H=10 ⁰ ,7FH=10 ⁻¹ etc.
X-1	00H	Vorzeichen 00H = positiv 01H = negativ
X-2	26H	7. und 8. Stelle der Mantisse
X-3	59H	5. und 6. Stelle der Mantisse
X-4	41H	3. und 4. Stelle der Mantisse
X-5	31H	1. und 2. Stelle der Mantisse

5.4 Numerische Variablen

5.4.1 Variablennamen

Der Name einer Variablen kann aus 1 bis 10 Buchstaben, Zahlen oder dem Unterstrich bestehen, z.B. POSITION1 oder POS1. Das erste Zeichen des Namens muß ein Buchstabe sein.

Bei der Namensgebung ist zu beachten, daß COMTAC-Basic intern nur das erste Zeichen, das letzte Zeichen und die Anzahl der Zeichen des Namens verarbeitet.

Das bedeutet, daß z.B. die Variablen MAUS, MIES, MUSS, M_1S oder MXXS identisch sind, weil das erste Zeichen, das letzte Zeichen und die Anzahl der Zeichen dieser Namen gleich sind.

Beachten Sie, daß Sie den Befehlssyntax der Befehle nicht als Variablennamen oder als Teile von Variablennamen verwenden. Weitere Variablennamen, die Sie nicht verwenden dürfen sind im nachfolgenden Kapitel unter COMTAC-Basic-Schlüsselwörter beschrieben.

5.4.2 Indizierte Variable

COMTAC-Basic kann Variable mit einem oder zwei Indices verarbeiten; eindimensionale bzw. zweidimensionale Matrizen. Der Index wird in Klammern dem Variablen-Namen angehängt; bei Variablen mit 2 Indices werden diese durch Komma getrennt, z.B.:

A(7) , B(9,4) , ZX(I) , VPOS(I,7) , WINKEL(2,N)
 Nicht explizit dimensionierte Matrizen sind immer zweidimensional und haben die Standardanzahl von 121 Elementen;

Element(0,0) Element(10,10).
 Matrizen in anderen Größen müssen vor ihrer Verwendung dimensioniert werden, wie es in folgenden Beispielen gezeigt wird:

DIM A(30)
 ♦ erzeugt die Matrize A mit 31 Elementen A(0) ... A(30).
 DIM B(20,15)
 ♦ erzeugt die Matrize B mit 126 Elementen B(0,0) ... B(20,5).
 Ein Index darf nicht größer als 254 sein ! ;
 d. h. eindimensionale Matrizen haben max. 255 Elemente;
 Element x(0) - x(254).

Ein Index kann als Zahl, Variable oder numerischer Ausdruck angegeben werden.
 Mit der DIM-Anweisung ist es möglich Matrizen zu definieren, die im ZP-RAM (Laufwerk R) netzausfallsicher gespeichert werden.
 Außerdem können Matrizen erzeugt werden, die ein anderes Datenformat haben als das oben beschriebene Floating-point-Format.

5.6 COMTAC-Basic Schlüsselwörter

	Beachten Sie, daß Sie den Befehlssyntax der Befehle nicht als Variablenamen oder als Teile von Variablenamen verwenden. Weitere Variablenamen, die Sie nicht verwenden dürfen sind nachfolgend beschrieben.
--	--

- | | |
|---|---|
| CBY(
CKCON
COUNT
CWO(
DBY(
DWO(
IEREG
OFF
ONCOUNT
PBY(
PREG | PWO(
RCAP2
T1REG
T2CON
T2REG
TMOD
TOREG
WDPCON
XBY(
XWO(
 |
|---|---|

5.5 Zeichenketten

Für die Verarbeitung von nicht-numerischen Informationen werden Zeichenketten- oder String-Variable verwendet. COMTAC-Basic benutzt eindimensionale String-Variable, die mit dem Dollar-Zeichen und einem in runden Klammern stehenden Index benannt werden (\$ (Index)).

Der Index, dessen max. Wert 255 sein darf, kann durch eine Zahl, Variable oder numerischen Ausdruck dargestellt werden.

z. B. \$(7) ; \$(A) : \$(A-8/4)
 COMTAC-Basic erlaubt für Zeichenketten eine dimensionierte Länge zwischen 1 und 255, während die aktuelle Länge zwischen Null und der dimensionierten Länge liegen darf. Zeichenketten mit mehr als 16 Zeichen (Standardlänge) müssen vor ihrer Verwendung mit der DIM - Anweisung dimensioniert werden.

5.7 Begriffe

BCD	Binär-Codierte-Dezimalzahl jeweils 4bit einer binären Zahl werden als Dezimalstelle ausgewertet. Beispiele:							
Dekaden	10 ¹				10 ⁰			
Binärstellen	2 ³	2 ²	2 ¹	2 ⁰	2 ³	2 ²	2 ¹	2 ⁰
Wertigkeit der Binärstellen	8	4	2	1	8	4	2	1
Zahlenwert 35 im BCD-Format	0	0	1	1	0	1	0	1

5.8 Befehlskürzel

➡ Die Befehlskürzel stehen zusätzlich bei allen Befehlen (in der Überschrift), zu denen es eine Abkürzung gibt.

Anweisung	Kürzel
BINOUT	B.
CLEAR	CL.
CONTROL	CT.
COPY	C.
CPXACCEL	AL.
CPXBK	BK.
CPXBOFF	POB.
CPXCTR	CC.
CPXCTR	CC.
CPXIMASK	CIM.
CPXIMASK	CIM.
CPXINP	CI.
CPXINP	CI.
CPXOFF	OFF.
CPXOMASK	COM.
CPXON	ON.
CPXOUT	CO.
CPXOVR	CV.
CPXPARA	PAR.
CPXPH	ZP.
CPXPOS	CP.
CPXPOSA	PA.
CPXPOSR	PR.
CPXQT	QT.
CPXSP	SP.
CPXSPEED	SD.
CPXST	ST.
CPXSTA	STA.
CPXSTS	CS.
CURSOR	CU.
DISABLE	DI.
DISP	D.
ENABLE	EN.
ENTER	E.
ERRSTS	ES.
FBUS_D	FD.
FBUS_I	FI.
FBUS_O	FO.
FBUS_P	FP.
FBUS_S	FS.
GOSUB	GS.
GOTO	GT.
HLINE	H.
INPUT	I.
LOAD	L.
ONERR	OE.
ONINP	OP.
ONKEY	OK.
ONTIMER	OC.
OUTPUT	O.
PRINT	P.
PRINT	?
PSTEP	PS.
RECTANGLE	RECT.
REM	!
RESET	RS.
RETURN	R.
STORE	S.
TABXY	T.
UMEM	
USING	U.
VLINE	V.

5.9 Arithmetische Operatoren u. Funktionen

Operator	Beschreibung/Funktion	Beispiel
+	Addition	A = B + C
-	Subtraktion	X = B - 8
*	Multiplikation	Z = 6 * 6
/	Division	T = X / Y
'DIV'	ganzzahliger Teil einer Division	D = X'DIV'Y
'MOD'	Rest einer Division	M = X'MOD'Y
**	Potenzfunktion (x ^y)	A = X ** Y
SQR(x)	Quadratwurzel von x	V = SQR(9)
EXP(x)	Exponentialfunktion (e ^x) e = 2.7182818	Q = EXP(T)
LOG(x)	natürlicher Logarithmus (log _e x)	T = LOG(Q)
ABS(x)	Betrag von x	B = ABS(X/Y-8)
INT(x)	ganzzahliger Anteil von x	Z = INT(W/12)
SGN(x)	Vorzeichen von x SGN(x) = 1 wenn x > 0 SGN(x) = 0 wenn x = 0 SGN(x) = -1 wenn x < 0	V = SGN(A-B)
NOT(x)	16 Bit Einerkomplement von x. x muß ganzzahlig und im Bereich von 0 - 65535 liegen. NOT(0) = 65535 NOT(1) = 65534	C = NOT(S*4)
RND	Zufallszahl im Bereich 0 .. 1	Z = RND*100

➡ Die Berechnungen werden immer im Floating-Point - Format durchgeführt.

5.10 Trigonometrische Funktionen

COMTAC-Basic arbeitet im Bogenmaß (0 - 2 Π). Zur Berechnung der trigonometrischen Funktionen werden Taylor-Reihen verwendet. Dafür wird das Argument zuerst reduziert auf einen Wert zwischen 0 und Π/2.

Es empfiehlt sich daher das Argument so klein wie möglich anzugeben, damit bei der Reduzierung keine signifikante Stellen des Arguments verloren gehen.

Operator	Beschreibung/Funktion	Beispiel
SIN(x)	Sinus von x x muß zwischen +/- 200 000 liegen	U=SIN(PI/4)
COS(x)	Cosinus von x x muß zwischen +/- 200 000 liegen	C=COS(A*B+4)
TAN(x)	Tangens von x x muß zwischen +/- 200 000 liegen	N=TAN(D+V)
ATN(x)	Arcus Tangens von x das Ergebnis liegt zwischen +/-PI/2	T=ATN(Z/8)
PI	Kreiszahl (3.1415927)	A=PI/4
RAD	die Zahl 57.295779 (180/PI) Konstante zur Umwandlung von Altgrad in Bogenmaß	W=SIN(30/RAD)

5.11 Vergleichende und logische Operatoren

In COMTAC-Basic ist das Ergebnis eines Vergleichs 65535, wenn der zu vergleichende Ausdruck wahr ist, oder 0, wenn der zu vergleichende Ausdruck falsch ist.

Das Ergebnis eines Vergleichs wird auf den Argument-Stack gelegt und kann somit angezeigt oder auch einer Variablen zugewiesen werden.

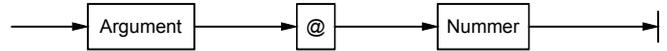
Zudem ist es möglich das Ergebnis mit einem weiteren Vergleich logisch zu verknüpfen (z.B. IF A<B 'OR' A>C THEN).

Operator	Beschreibung/Funktion	Beispiel
=	gleich	IF C=4 THEN ...
<>	ungleich	IF X<>Y THEN ...
>	größer	PRINT X > 4
>=	größer gleich	T = 5 >= X
<	kleiner	WHILE I < 10
<=	kleiner gleich	UNTIL N <= 30
'AND'	logische UND-Verknüpfung	X = B'AND'C
'OR'	logische ODER-Verknüpfung	PRINT 4'OR'Z
'NOT'	Bitwechsel	IF NOT (A) ...
'XOR'	logische EXKLUSIV-ODER-Verknüpfung	DISP Q'XOR'30
@	Bit-Abfrage	DISP Q@3

Die logischen Operatoren ('AND','OR','XOR') führen eine Bit-für-Bit Verknüpfung durch. Deshalb müssen beide Argu-

mente dieser Operatoren ganzzahlig sein und im Bereich von 0 - 65535 liegen.

5.12 Bit-Abfrage



Funktion:

Dieser Operator ermittelt den logischen Zustand des einzelnen Bits in dem angegebenen Argument.

Die Angabe Position definiert, welches Bit des Arguments abgefragt wird.

Die Bit-Abfrage übergibt die Information wahr = 65535, wenn das entsprechende Bit im Argument gesetzt ist oder falsch = 0, wenn das Bit nicht gesetzt ist.

Parameter	Angabe	Bereich	Beschreibung
Argument	num. Ausdr.	0 - 65535	Wert in dem ein Bit abgefragt wird
Nummer	num. Ausdr.	0 - 15	Nummer des abzufragenden Bits (15 = MSB, 0 = LSB)

Beispiel: IF CPXSTS(1)@4 then GOTO 100
WHILE A@1

Anmerkung:

Das Bit 0 hat die kleinste Wertigkeit, das Bit 15 hat die höchste Wertigkeit.

5.13 Rangfolge der Operatoren und Funktionen

COMTAC-Basic verwendet folgende mathematische Hierarchie zur Berechnung eines numerischen Ausdrucks:

- 1) Funktionen deren Argument in Klammern steht
- 2) Bit-Abfrage (@)
- 3) Potenzieren (**)
- 4) Multiplikation (*) und Division (/)
- 5) Addition (+) und Subtraktion (-)
- 6) Vergleichende Operatoren (=,<>,>,>=,<,<=)
- 7) logische UND-Verknüpfung
- 8) logische ODER-Verknüpfung
- 9) logische EXKLUSIV-ODER-Verknüpfung

Anmerkung:

Im Zweifelsfalle sollten bei einem komplexen mathematischen Ausdruck immer Klammern verwendet werden.

6. Programmstruktur

6.1 Unterprogramme

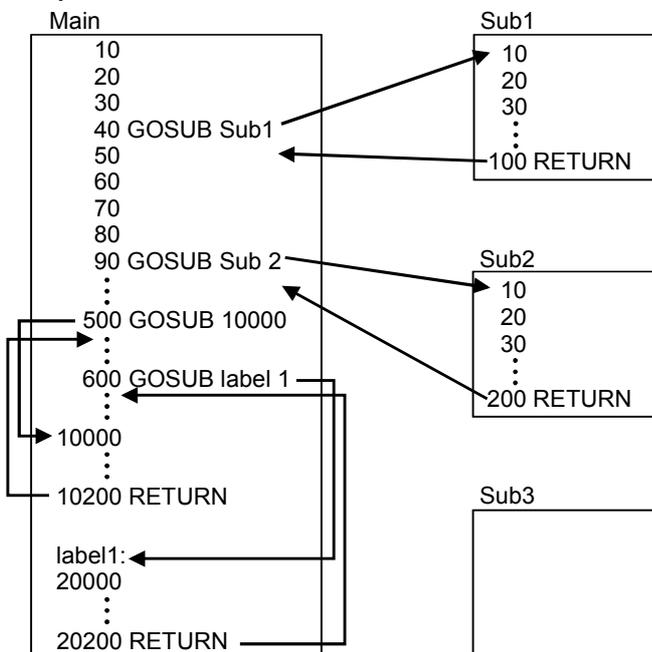
Sie haben die Möglichkeit in einer Datei

- ◆ ein Hauptprogramm (Main) und
- ◆ mehrere Unterprogramme (Subs) abzuspeichern.

Sämtliche Programmzeilen müssen mit Zeilennummern versehen sein (Wertebereich 0-65535).

Die Programmzeilennummern stehen für das Hauptprogramm sowie für jedes Unterprogramm zur Verfügung. D. h., in Haupt- und Unterprogramm ist der gleiche Wertebereich an Zeilennummern vorhanden.

Beispiel:



➡ Aus den Subs können weitere Subs aufgerufen werden. Zur Tiefe der Sub-Verschachtelung siehe am Ende des Kapitels.

Erstellen eines SUBs

Mit der Anweisung **SUB "Subname"** wird ein neues Unterprogramm erzeugt.

- ◆ Die Länge des Namen ist nur durch die Zeilenlänge begrenzt.
- ◆ Es wird zwischen Groß- und Kleinbuchstaben unterschieden.
- ◆ Ein SUB wird automatisch erstellt, wenn mit der COPY-Anweisung bestehende Programmzeilen des angewählten SUB/MAIN in ein noch nicht vorhandenes SUB kopiert werden.
z.B. COPY 1000,1999 TO SUB "Handbetrieb".

Speichern eines SUBs

Mit der Anweisung:

STORE SUB "Subname" Laufwerk: "Filename.Extention" wird das angegebene Sub als Datei gespeichert.

Mit der Anweisung:

STORE Laufwerk: "Filename.Extention"

wird das Hauptprogramm mit allen SUBs gespeichert.

Hinweis zum Programmierool

- ◆ Im DIR-Mode wird immer das aktuell angewählte SUB gespeichert.
- ◆ Ist das Hauptprogramm angewählt, werden immer auch alle SUBs mit abgespeichert.
- ◆ Die Auswahl, was gespeichert werden soll, ist vor dem Aufruf des DIR-Mode zu treffen.

Laden eines SUBs

Datei als SUB laden:

- ◆ Neues SUB erstellen
- ◆ Befehl: **LOAD Laufwerk: "Dateiname"** eingeben.

➡ Ein kompiliertes Programm kann nicht geladen werden (siehe Seite 28 unter 7.3.1 RUN).

Aufruf eines Subs

GOSUB SUB "Subname"

Mit RETURN wird das SUB wieder verlassen.

Verzweigen in ein SUB

GOTO SUB "Subname"

Verzweigen zurück zum MAIN

Mit der Anweisung **GOTO MAIN ZNr.** wird zu der angegebenen Zeile im MAIN verzweigt.

Aufruf eines Unterprogramms im MAIN

Mit der Anweisung **GOSUB MAIN ZNr.** wird ein Unterprogramm im MAIN von einem Sub aus aufgerufen, das bei der angegebenen Zeile beginnt.

Beim nächsten Return springt das Programm zum Sub zurück.

Editieren eines bestehenden SUB

Mit der Anweisung **SUB "Unterprogrammname"** wird das gewünschte SUB angewählt.

Dann kann mit dem LIST-Befehl das Programm angezeigt werden.

Ist das SUB noch nicht vorhanden wird es neu erzeugt.

Löschen eines bestehenden SUB

Mit der Anweisung **DEL SUB "Unterprogrammname"** wird das gewünschte SUB gelöscht.

Kopieren von Programmzeilen in ein bestehendes SUB

Mit der Anweisung **COPY/COPYDEL Znr, ZNr TO SUB "Unterprogrammname"** wird der angegebene Programm-Block vom aktuellen SUB/MAIN in das gewünschte SUB kopiert.

Kopieren von Programmzeilen eines SUB zum MAIN

Mit der Anweisung **COPY/COPYDEL ZNr,ZNr TO MAIN** wird der angegebene Programm-Block vom aktuellen SUB in das MAIN kopiert.

Editieren des Hauptprogramms

Mit der Anweisung **MAIN** wird das Hauptprogramm ausgewählt und sofort angezeigt.

Anzeigen aller vorhandenen SUB-Namen

Mit der Anweisung **LIST SUB** werden alle SUB-Namen aufgelistet.

Maximale Programmverschachtelung

Zur Verwaltung von Programmstrukturen steht intern ein Speicher von 254 Byte zur Verfügung. Diese dürfen nicht überschritten werden.

Einzelne Programmschleifen sowie Unterprogrammaufrufe benötigen folgenden Speicher

GOSUB: 4 Byte

FOR/ NEXT-Schleife: 18 Byte

DO UNTIL-Schleife: 4 Byte

DO WHILE-Schleife: 4 Byte

Es zählen nur die gleichzeitig aktiven Schleifen und Unterprogramme.

6.2 Labels

Programmsprünge mit **GOTO** oder mit **GOSUB** können sich auf Zeilennummern oder auf Labels beziehen.

Die Labels sind wie die Zeilennummern nur im lokalen Programm (Main oder Sub) gültig.

Label setzen

Um ein Label zu setzen ist in eine bestehende Basic-Zeile nach der Zeilen-Nr. die Anweisung **LABEL "Name"**: einzufügen.

Nach dem Doppelpunkt muß mindestens ein weiterer Basic-Befehl stehen.

Beispiel:

Eingabe:

```
100 label "marke 1": Print "Test"
```

Sie erhalten damit folgender Aufbau:

```
marke1:
```

```
100 Print "Test"
```

Label löschen

Ein Label wird gelöscht, wenn die Zeile ohne die Anweisung **LABEL "Name"** eingegeben oder editiert wird.

D. h., wenn Sie zum Beispiel die obige Zeile

```
marke1:
```

```
100 Print "Test"
```

editieren.

Wird die Zeile 100 mit Enter (↵) neu bestätigt, dann wird das Label "marke1" entfernt. Der Befehl label "marke 1" muß bei jedem Editieren der Zeile neu eingegeben werden.

Empfehlung: Schreiben Sie als nachfolgende Anweisung einen Kommentar in die Zeile.

Bsp.: Eingabe:

```
100 label "marke 1": ! marke 1
```

Label suchen

Mit der Anweisung **LABEL "Name"** wird das gewünschte LABEL gesucht und zusammen mit den folgenden Zeilen aufgelistet.

Ist das LABEL nicht vorhanden wird ein Fehler erzeugt.

Mit der Funktionstaste F12 kann das LABEL angewählt werden, das in der aktuellen Zeile in Hochkomma angegeben ist

Aufruf eines Unterprogrammes mit LABEL (lokales Unterprogramm)

Mit der Anweisung **GOSUB "Name"** wird das Unterprogramm mit dem angegebenen Label aufgerufen.

Verzweigen zu einem LABEL

Mit der Anweisung **GOTO "Name"** wird zu dem angegebenen Label verzweigt.

7. Programmerstellung

Die Programme für COMTAC können direkt über ein angeschlossenes Terminal erstellt werden. Das Terminal-Programm ist in COMTAC enthalten und unterstützt folgende Terminal bzw. Terminalemulationen:

- ◆ TV905
- ◆ VT100

Desweiteren können Sie bei uns die Software "COMTAC - Programmierool" oder "Win TV905" beziehen. Diese läuft auf PCs unter DOS und bietet folgende Funktionserweiterungen gegenüber dem Terminalprogramm:

- ◆ Speichern von Programm- und Datenfiles auf Platte.
- ◆ Laden von Programm- und Datenfiles von der Platte.
- ◆ Enthält einen Parametereditor zum Editieren der COMTAC - Parameter.
- ◆ Enthält einen Makro-Editor.

➡ Die Software "COMTAC - Programmierool" ist in einer separaten Anleitung beschrieben.

Einstellungen:

- ◆ Baudrate (standard): 9600
(durch Drücken von F12 während dem Einschalten von COMTAC werden alle Parameter aus Standardwerte gesetzt)
- ◆ Schnittstelle: RS232/1
- ◆ und den Typ des Terminal in Parameter (CONTROL 2); standard ist TV905 (CT.2=0)
Mit (CT.2=3) wird ein VT100 Terminal eingestellt.

Funktionsübersicht

- ◆ Direkte Befehle eingeben.
Ohne Zeilennummer eingegebene Befehle werden mit dem Drücken von Return (↵) von COMTAC ausgeführt.
- ◆ Programme erstellen.
An der vorgestellten Zeilennummer erkennt COMTAC eine Zeile als Programmzeile.
Z.B.:
10 GOSUB 1000
- ◆ Programme starten.
- ◆ Programme vom eingebauten ZP-RAM oder vom angeschlossenen Diskettenlaufwerk (HFM2) laden /speichern.

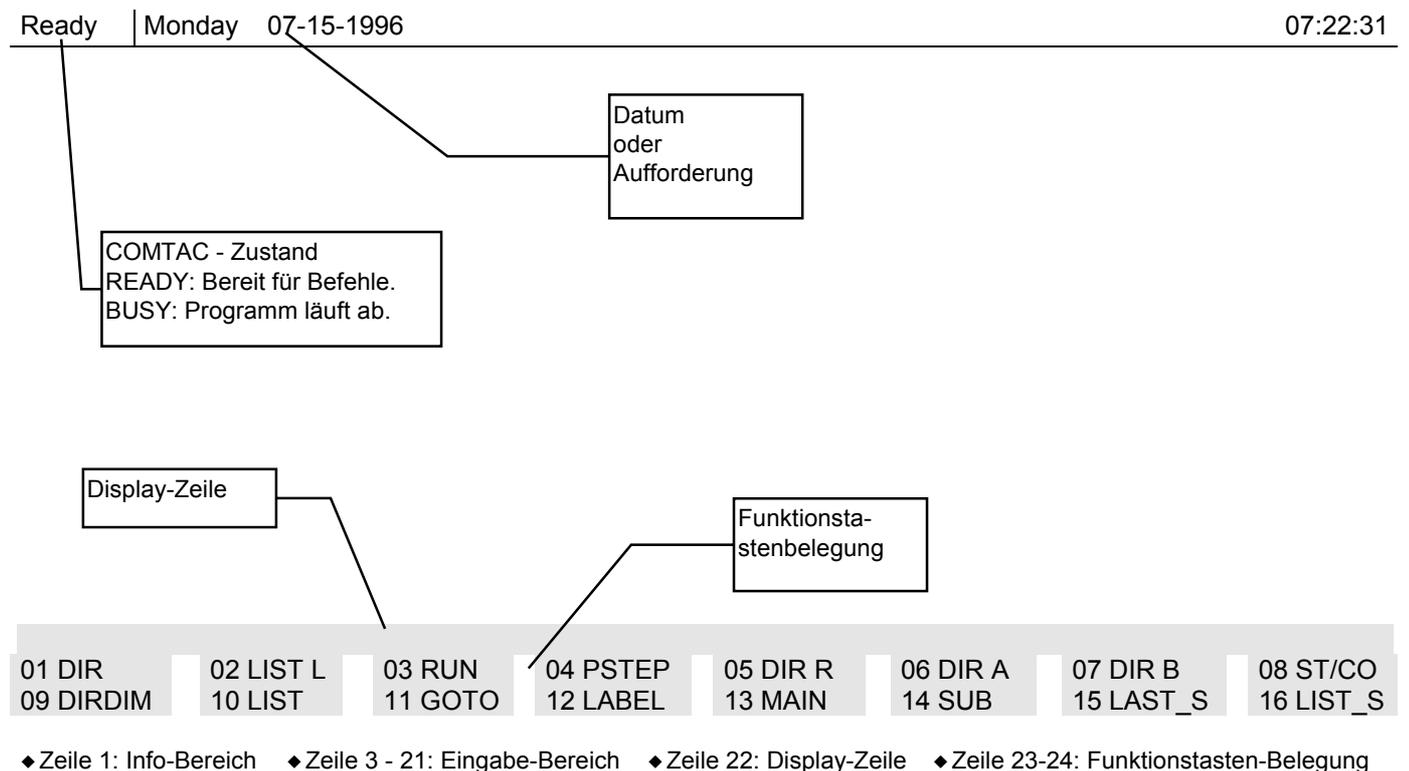
➡ Innerhalb des Eingabebereichs können Sie den Cursor mit der Pfeiltaste frei bewegen und eine beliebige Zeile editieren (ändern oder ergänzen). Die Zeile wird dann durch Return (↵) von COMTAC ausgeführt bzw. als Programmzeile übernommen.

7.1 Terminal-Programm

Bei einem PC als Terminal arbeiten Sie bei allen Aktionen über COMTAC, d.h. z. B. bei der Programmerstellung direkt im Arbeitsspeicher von COMTAC.

Bildschirmaufbau

Befindet sich das COMTAC nach POWER ON im COMMAND-Mode, wird der Bildschirm des angeschlossenen Terminals wie folgt in 4 Bereiche aufgeteilt:



Info-Bereich

Im 1. Teil der 1. Zeile wird die Betriebsbereitschaft des COMTAC angezeigt.??
 Der 2. Teil wird vom COMTAC als Info-Zeile benutzt für die Anzeige von Datum mit Uhrzeit, Fehlermeldungen oder Eingabeanweisungen.

Eingabebereich

Dieser Bereich dient zur Eingabe von Kommandos, Eingabe von Programmzeilen, Auflisten und Verändern von Programmzeilen.

Editierfunktionen im Eingabebereich

Der Eingabebereich verfügt über 197 Eingabezeilen; davon sind die ersten 18 Zeilen gepuffert. Die aktuelle Eingabezeile wird durch den Cursor gekennzeichnet. Durch Drücken der Taste RETURN wird der Inhalt der aktuellen Eingabezeile übernommen. Beginnt die Zeile mit einer Zahl, wird diese Zeile als Programmzeile abgespeichert; sonst wird die Zeile als Kommando interpretiert und ausgeführt. Programmfehler werden erst beim Programmablauf entdeckt. Mit den Cursor-Tasten können bereits ausgeführte Eingabezeilen angewählt werden, um Änderungen vorzunehmen oder ein Kommando zu wiederholen. Durch die Übernahme der 19. Eingabezeile wird der Inhalt des Eingabebereiches um eine Zeile nach oben verschoben, wodurch die erste Zeile vom Bildschirm verschwindet und damit nicht mehr editiert werden kann.
 Für die Eingabe bietet das COMTAC eine Vielfalt von Editierfunktionen, die im folgenden beschrieben werden:

Display-Zeile

Ausgabezeile für die DISP@-Anweisung

Funktionstasten-Belegung

Diese Belegung gilt für Terminals bei denen es möglich ist die Funktionstasten mit dem entsprechenden Tastencode zu programmieren (z.B. TV905, TV910, TV925, TV950, TV9065).

Taste	Code		Funktion	Beschreibung
	Tastatur	hex		
F1	Ctrl_A 0	0x01 0x30	DIR	Aufruf des Directory-Mode (DIR-Mode)
F2	Ctrl_A 1	0x01 0x31	LIST_L	Ausgabe des aktuellen Programmlistings (SUB/MAIN) an das Terminal ab der Zeilen-Nr. die beim letzten LIST-Befehl eingegeben wurde
F3	Ctrl_A 2	0x01 0x32	RUN	Programmstart ab der SUB/MAIN-Zeilen-Nr. die in der aktuellen Edit-Zeile steht. Ist in der akt. Edit-Zeile (aktuelle Cursor-Position) keine Zeilen-Nr. startet das Programm mit der 1. Zeile des MAIN-Programmes. Variable werden gelöscht.
F4	Ctrl_A 3	0x01 0x33	PSTEP	Programmschritt ab der SUB/MAIN-Zeilen-Nr. die in der aktuellen Edit-Zeile steht. Ist in der akt. Edit-Zeile (aktuelle Cursor-Position) keine Zeilen-Nr. wird die nächste Basic-Anweisung des Programms ausgeführt..
F5	Ctrl_A 4	0x01 0x34	DIR R	Aufruf des DIR-Mode. Anzeige des Inhaltsverzeichnis von Laufwerk R
F6	Ctrl_A 5	0x01 0x35	DIR A	Aufruf des DIR-Mode. Anzeige des Inhaltsverzeichnis von Laufwerk A
F7	Ctrl_A 6	0x01 0x36	DIR B	Aufruf des DIR-Mode. Anzeige des Inhaltsverzeichnis von Laufwerk B
F8	Ctrl_A 7	0x01 0x37	ST/CO	Ein laufendes Programm wird unterbrochen (STOP) und ein unterbrochenes Programm wird fortgesetzt (CONTinue). Die Funktion kann mit dem Systemparameter 13 ausgeschaltet werden.
F9	Ctrl_A 8	0x01 0x38	DIRDIM	Anzeige von Größe und Speicherbelegung der im Laufwerk R vorhandenen Arrays (Matrizen).
F10	Ctrl_A 9	0x01 0x39	LIST	Ausgabe des aktuellen Programmlistings (SUB/MAIN) an das Terminal ab der Zeilen-Nr. die in der aktuellen Edit-Zeile steht. Ist in der akt. Edit-Zeile (aktuelle Cursor-Position) keine Zeilen-Nr. wird das Listing fortgesetzt mit der Zeile die der zuletzt angezeigten Zeile folgt.
F11	Ctrl_A :	0x01 0x3A	GOTO	Programmstart ab der SUB/MAIN-Zeilen-Nr. die in der aktuellen Edit-Zeile steht. Ist in der akt. Edit-Zeile (aktuelle Cursor-Position) keine Zeilen-Nr. startet das Programm mit der 1. Zeile des MAIN-Programmes. Variable werden nicht gelöscht.
F12	Ctrl_A ;	0x01 0x3B	LABEL	Ausgabe des aktuellen Programmlistings (SUB/MAIN) an das Terminal ab dem LABEL dessen Name in der aktuellen Edit-Zeile steht. Ist in der akt. Edit-Zeile (aktuelle Cursor-Position) keine LABEL-Name wird nichts ausgeführt.
F13	Ctrl_A <	0x01 0x3C	MAIN	Ausgabe des MAIN-Programmlistings an das Terminal ab der 1. Zeile
F14	Ctrl_A =	0x01 0x3D	SUB	Ausgabe des Programmlistings an das Terminal von dem SUB dessen Name in der aktuellen Edit-Zeile steht. Ist in der akt. Edit-Zeile (aktuelle Cursor-Position) keine SUB-Name wird das nächste SUB gelistet.
F15	Ctrl_A >	0x01 0x3E	LAST_S	Ausgabe des Programmlistings an das Terminal von dem SUB/MAIN das vor dem aktuellen SUB/MAIN ausgegeben wurde.
F16	Ctrl_A ?	0x01 0x3F	LIST_S	Ausgabe aller SUB-Namen an das Terminal

HOME

Der Cursor wird in der HOME-Position (1. Zeile des Eingabe-Bereiches) gebracht. Der Inhalt des Eingabebereiches wird nicht gelöscht.

PFEILE

Der Cursor wird entsprechend der Pfeilrichtung um eine Spalte bzw. eine Zeile verschoben.

Wird unmittelbar aufeinanderfolgend eine PFEIL-Taste und die HOME-Taste gedrückt, wird der Cursor entsprechend der Pfeilrichtung bis zum letzten oder dem ersten Zeichen der Zeile bzw. zum Anfang der letzten beschriebenen Zeile oder dem Anfang der 1. Zeile verschoben.

{CLEAR SPACE}¹ oder Ctrl+T oder Ctrl+Z

Der Inhalt des Eingabe-Bereiches wird gelöscht und der Cursor in die HOME-Position gebracht.

Der Bildschirm wird ganz gelöscht und neu aufgebaut, wenn sich das COMTAC vorher im RUN-Mode befunden hat.

PAGE ERASE oder Ctrl+P

Der Bildschirm wird ganz gelöscht und neu aufgebaut.

DEL oder CHAR DELETE

Das Zeichen auf der aktuellen Cursor-Position wird gelöscht.

CHAR INSERT oder Ctrl+B

Ermöglicht das Einfügen eines oder mehrerer Zeichen vor die aktuelle Cursor-Position.

Ein nochmaliges Drücken dieser - Taste löscht diese Funktion.

{LINE DELETE} oder Ctrl+Y

Die aktuelle Zeile wird gelöscht.

{LINE INSERT} oder Ctrl+I oder TAB

Eine neue Zeile wird in den Text eingefügt.

{LINE ERASE} oder Ctrl+E

Die aktuelle Zeile wird ab der Cursorposition bis zum Zeilenende gelöscht.

Ctrl+C

Abbruch eines laufenden BASIC-Programms (Run-Mode) und Rückkehr in den COMMAND-Mode

Ctrl+R

Abbruch aller momentanen Aktivitäten und Neuinitialisierung aller Schnittstellen.

Das Anwenderprogramm im Arbeitsspeicher bleibt erhalten.

Ctrl+A dann R

Reset des COMTAC Neuinitialisierung wie nach Power ON.

Ctrl+S

Stop einer LISTING-Ausgabe.

Ctrl+Q

Fortsetzung der LISTING-Ausgabe nach Control+S.

Ctrl+U

Rückcompilieren des aktuellen Programms.

PAGE oder Ctrl+X

Der DATA-TEXT-EDITOR wird aufgerufen.

Ctrl+O

Unterbricht vorübergehend die Uhranzeige in der Infozeile.

¹ Die Tasten in den Klammern "{}" gelten für ein TV905 - Terminal

7.2 Editierfunktionen

7.2.1 NEW



Funktion:

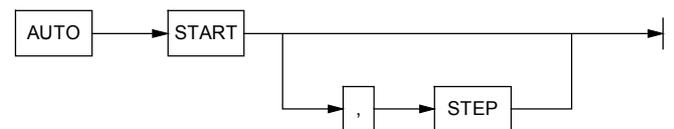
Mit dieser Anweisung wird das aktuelle Programm im Arbeitsspeicher gelöscht, alle Variablen auf 0 gesetzt, alle Strings und Interrupts gelöscht. Bereits ausgeführte DIM-Anweisungen sind nicht mehr gültig. Control- und Argument-Stack werden zurückgesetzt.

Beispiel: NEW

Anmerkung:

Der Inhalt des Data-Text-Feldes, der interne Zähler TIMER, die Strings \$(#0), \$(#1), \$(#2), \$(#3), \$(#4) und Matrizen im ZP-RAM (Laufwerk R) werden nicht gelöscht.

7.2.2 AUTO



Funktion:

Diese Anweisung aktiviert die automatische Zeilennummerierung im aktuellen Programm. Nach jedem Return (↵) steht in die neuen Zeile die nächste Zeilennummer.

Parameter	Angabe	Bereich	Beschreibung
Start	Zahl	0 - 65535	1. neue Basic-Zeilen-Nr.
Step	Zahl	1 - 65535	Schrittweite

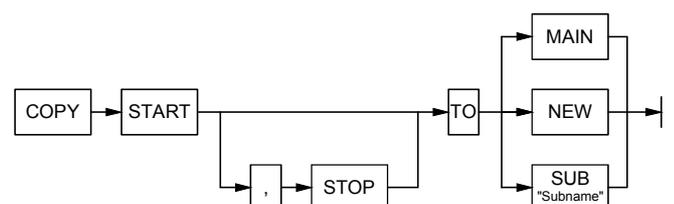
Beispiel: AUTO 100

AUTO 100,5

Anmerkung:

Wird keine Schrittweite angegeben ist die Schrittweite 10. Diese Funktion kann mit Ctrl-C abgebrochen werden.

7.2.3 COPY [C.] Basic-Zeilen



Funktion:

Diese Anweisung kopiert eine bzw. mehrere Basic-Zeilen.

Parameter	Angabe	Bereich	Beschreibung
Start	Zahl	0 - 65535	1. Basic-Zeile, die kopiert werden soll
Stop	Zahl	0 - 65535	letzte Basic-Zeile, die kopiert werden soll
New	Zahl	0 - 65535	Zeilen-Nr., wohin kopiert werden soll oder Sub "Subname" in welches die Zeilen kopiert werden.

Beispiel: COPY 30 TO 200 oder COPY 100, 260 TO 1000

Anmerkung:

Vorhandene Zeilen an der Zieladresse werden überschrieben.
Die Schrittweite bleibt erhalten.
Beim Kopieren der Zeilen in ein nicht vorhandenes Sub, wird dieses neu erstellt.

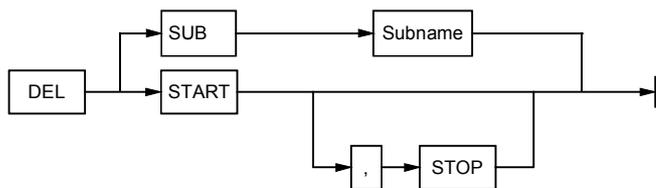
Parameter	Angabe	Bereich	Beschreibung
Start	Zahl	0 - 65535	1. Basic-Zeile, die kopiert werden soll
Stop	Zahl	0 - 65535	letzte Basic-Zeile, die kopiert werden soll
New	Zahl	0 - 65535	Zeilen-Nr., wohin kopiert werden soll oder Sub "Subname" in welches die Zeilen kopiert werden.

Beispiel: COPYDEL 100 TO 500
COPYDEL 230,440 TO 20000

Anmerkung:

Vorhandene Zeilen an der Zieladresse werden überschrieben.
Die Schrittweite bleibt erhalten.
Beim Kopieren der Zeilen in ein nicht vorhandenes Sub, wird dieses neu erstellt.

7.2.4 DEL Basic-Zeilen



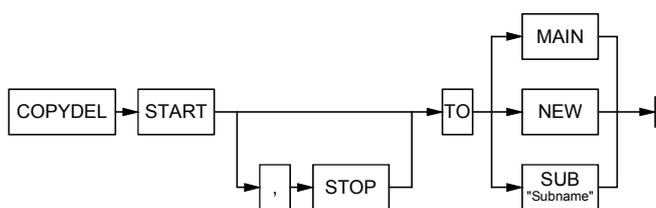
Funktion:

Diese Anweisung löscht
◆ eine bzw. mehrere Basic-Zeilen im aktiven Programm (Main oder ein Sub) oder
◆ ein ganzes Sub.

Parameter	Angabe	Bereich	Beschreibung
Start	Zahl	0-65 535	1. Basic-Zeile, die gelöscht werden soll
Stop	Zahl	0-65 535	letzte Basic-Zeile, die gelöscht werden soll

Beispiele: DEL 30 oder
DEL 70,200
DEL SUB "Test"

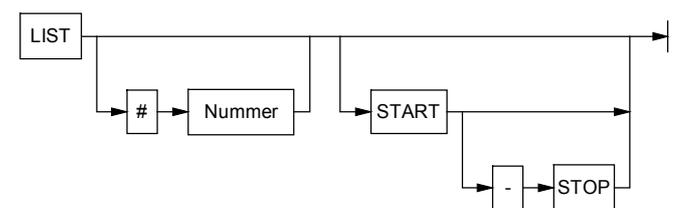
7.2.5 COPYDEL



Funktion:

Diese Anweisung kopiert eine bzw. mehrere Basic-Zeilen und löscht anschließend die Quell-Basic-Zeilen.

7.2.6 LIST



Funktion:

Diese Anweisung gibt das aktuelle Programm (Main oder Sub) im Arbeitsspeicher an das Terminal oder an die entsprechend angewählte Schnittstelle aus.

Parameter	Angabe	Bereich	Beschreibung
Nummer	Zahl	0 - 9	Nummer der Schnittstelle
Start	Zahl	0 - 65535	Zeilen-Nummer Listing-Start
Stop	Zahl	0 - 65535	Zeilen-Nummer Listing-Stop

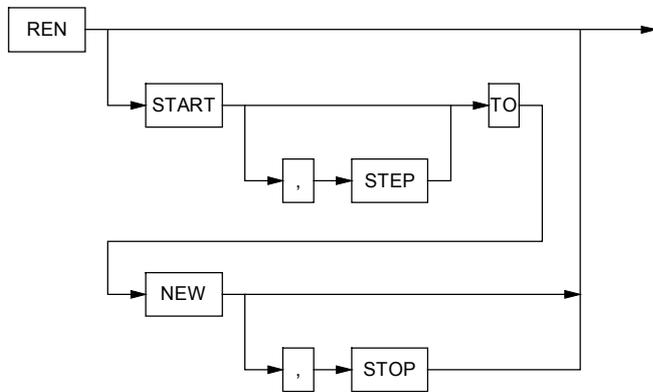
Beispiel: LIST oder LIST#1 10-200

Anmerkung:

- ◆ Wird explizit keine Schnittstelle ausgewählt erfolgt die Ausgabe an die Terminal - Schnittstelle.
- ◆ Mit Ctrl-S kann die List-Anweisung angehalten und mit Ctrl-Q fortgesetzt werden, Ctrl-C bricht die List-Anweisung ab.
- ◆ Die List-Anweisung wird nach der Ausgabe von 18 Zeilen automatisch angehalten.
Ein weitere LIST-Anweisung bringt die nächsten 18 Zeilen auf den Bildschirm.
- ◆ Mit SHIFT F2
 - ◆ erfolgt ein Listing oder
 - ◆ ein Listing wird fortgesetzt
 - ◆ oder, wenn in der aktuellen Zeile eine Zeilennummer steht, erfolgt das Listing ab dieser Zeile.

- ◆ Durch Eingabe einer Zeilennummer und Enter (↵), erfolgt ein Listing ab dieser Zeilennummer.
- ◆ F2 wiederholt die letzte LIST Anweisung.

7.2.7 REN



Funktion:

Diese Anweisung numeriert die Basic-Zeilen im aktuellen Programm neu (Main oder Sub).

Parameter	Angabe	Bereich	Beschreibung
Start	Zahl	0 - 65535	1. alte Zeilen-Nummer, die neu numeriert werden soll
Step	Zahl	1 - 65535	Schrittweite der neuen Zeilen-Nummern
New	Zahl	0 - 65535	1. neue Zeilen-Nummer
Stop	Zahl	0 - 65535	letzte Basic-Zeile, die neu numeriert werden soll

Beispiel: REN oder REN 10,5 TO 100,90

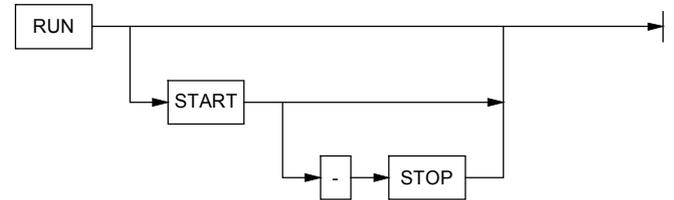
Anmerkung:

Wird nur die Anweisung REN gegeben, gilt:
 START = erste Zeile des Programms
 STEP = 10
 NEW = 10
 STOP = letzte Zeile des Programms

➡ Sprünge (GOTO, GOSUB) auf Zeilennummern werden automatisch korrigiert!

7.3 Programmablauf

7.3.1 RUN



Funktion:

Diese Anweisung löscht alle Variablen, setzt alle Interrupt's zurück kompiliert (siehe unten) und startet das Anwender-Programm.

Wird eine Start-Adresse angegeben, beginnt das Programm mit dieser Zeile.

Wird eine Stop-Adresse angegeben, unterbricht das Programm an dieser Zeile. Die Stop-Zeile wird nicht ausgeführt.

Parameter	Angabe	Bereich	Beschreibung
Start	Zahl	0 - 65535	Zeilen-Nr. -Programm-Start
Stop	Zahl	0 - 65535	Zeilen-Nr. -Programm-Stop

Beispiel: RUN oder RUN 10 - 100

Anmerkung:

- ◆ Das Programm kann über die Folientastatur mit der Funktionstaste F8 unterbrochen werden, wenn diese nicht gesperrt sind.
- ◆ Das Programm kann über die Terminal-Tastatur mit der Funktionstaste F8 oder mit Ctrl_C unterbrochen werden, wenn diese nicht gesperrt sind.
- ◆ Mit Ctrl-R wird das Programm immer abgebrochen und COMTAC neu initialisiert ohne das Programm im Arbeitsspeicher zu löschen.
- ◆ Mit dem Befehl RUN (ohne Zeilennummer) wird das Hauptprogramm (Main) gestartet.
- ◆ Starten eines Subs
 - ◆ Zum Sub wechseln.
 - ◆ Run Zeilennummer eingeben.

Compilieren des Programms

Die Zeit für das Compilieren eines großen Programms kann lange dauern. Speichern Sie ein Programm jedoch nach dem Compilieren ab (nach Ausführen von RUN), dann wird es in kompilierter Form abgespeichert und wird beim nächsten RUN sofort ausgeführt.

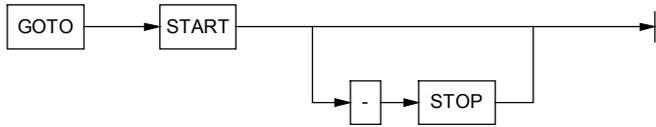
Ein kompiliertes Programm kann jedoch nicht in ein SUB geladen werden; es muß zuvor rückkompiliert werden.

Rückcompilieren des Programms

Laden Sie das Programm als MAIN (Laden ohne zuvor mit SUB ein Unterprogramm zu öffnen).

Mit [Ctrl+U] wird das Programm rückkompiliert und kann anschließend gespeichert werden.

7.3.2 GOTO [GT.]



Funktion:

Diese Anweisung startet das Anwender-Programm mit der spezifizierten Start-Zeile ohne die vorher gesetzten Variablen zu löschen.
Das Programm wird automatisch an der angegebenen Stop-Zeile unterbrochen.
Die Anweisung in der Stop-Zeile wird nicht ausgeführt.

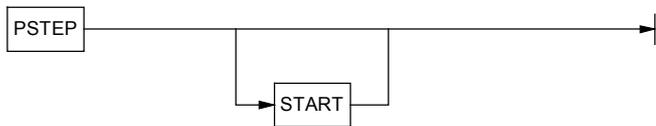
Parameter	Angabe	Bereich	Beschreibung
Start	Zahl	0 - 65535	Zeilen-Nr.-Programm-Start
Stop	Zahl	0 - 65535	Zeilen-Nr.-Programm-Stop

Beispiel: GOTO 30
GOTO 30 - 400

Anmerkung:

- ◆ Das Programm kann über die Folientastatur mit der Funktionstaste F8 unterbrochen werden, wenn diese nicht gesperrt sind.
- ◆ Das Programm kann über die Terminal-Tastatur mit der Funktionstaste F8 oder mit Ctrl_C unterbrochen werden, wenn diese nicht gesperrt sind.
- ◆ Mit Ctrl-R wird das Programm immer abgebrochen und COMTAC neu initialisiert ohne das Programm im Arbeitsspeicher zu löschen.
- ◆ Starten eines Subs
 - ◆ Zum Sub wechseln.
 - ◆ GOTO Zeilennummer eingeben.

7.3.3 PSTEP [PS.]



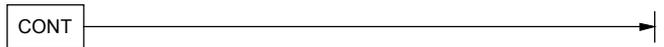
Funktion:

Mit dieser Anweisung kann das Programm im Arbeitsspeicher schrittweise abgearbeitet werden. Mit jedem PSTEP wird die nächste BASIC-Anweisung des Programmes ausgeführt.

Beispiel: PSTEP oder PSTEP 50

- ◆ Schrittweise abgearbeitet eines Subs
 - ◆ Zum Sub wechseln.
 - ◆ PSTEP Zeilennummer eingeben.

7.3.4 CONT



Funktion:

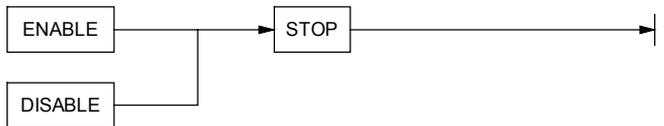
Diese Anweisung setzt das unterbrochene Programm fort.

Beispiel: CONT

Anmerkung:

Das Programm wird nicht fortgesetzt, wenn es zwischenzeitlich verändert wurde, oder ein Fehler erzeugt wurde.

7.3.5 ENABLE [EN.]/DISABLE [DI.] STOP



Funktion:

Freigabe/Sperren von Ctrl-C, F8.

Beispiel: ENABLE STOP
DISABLE STOP

Anmerkung:

Nach der Ausführung der Anweisung DISABLE STOP kann das laufende Programm mit F8 und Ctrl-C nicht mehr unterbrochen werden.

7.4 Systemvariable

7.4.1 FREE



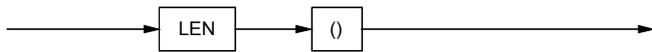
Funktion:

In dieser Systemvariablen steht die Anzahl der noch verfügbaren Bytes des Arbeitsspeichers (RAM) die dem Anwender zur Programmierung zur Verfügung stehen.

FREE = MTOP - LEN()

Beispiel: PRINT FREE
DISP FREE

7.4.2 LEN()



Funktion:

In dieser Systemvariablen steht die Länge des aktuellen Programms im Arbeitsspeicher (RAM).

Beispiel: PRINT LEN()
DISP LEN()

7.4.3 MTOP



Funktion:

In dieser Systemvariablen wird die Adresse des letzten verfügbaren Byte des Arbeitsspeichers (RAM) abgelegt.

Nach Power-On überprüft das COMTAC den Arbeitsspeicher (RAM).

Sind alle Speicherplätze in Ordnung hat MTOP den Wert 65535.

Beispiel: PRINT MTOP
IF MTOP <> 65535 THEN...

7.4.4 XTAL



Funktion:

In dieser Systemvariablen steht die Taktfrequenz des Basic-Prozessors.

8. Abspeichern und Laden von Programmen und Daten

8.1 Speichermedien/Dateinamen

Das COMATC-Basic bietet die Möglichkeit Programme und Daten unter einem frei wählbaren Namen, dem sogenannten Datei- oder Filenamen, im internen ZP-RAM (Laufwerk R) oder auf Diskette (externes Diskettenlaufwerk) abzuspeichern.

Die Selektierung des Speichermediums wird durch die Angabe eines Laufwerk-Kennbuchstabens vorgenommen.

R = ZP-RAM

A = Diskettenlaufwerk A (HFM2)

B = Diskettenlaufwerk B (HFM2)

Speicherkapazität

Nichtflüchtiger Speicher für Programme und Daten:

ZP-RAM (Laufwerk R) mit 128kByte.

Arbeitsspeicher:

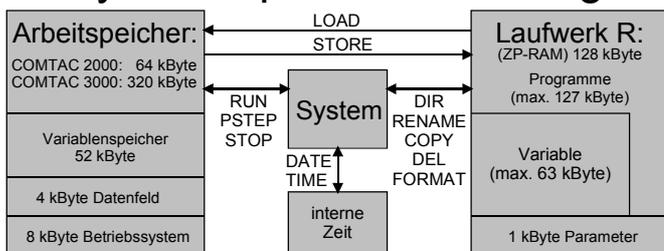
S-RAM mit 128kByte Speicher (384kByte bei COMTAC 3000). Dabei gilt:

◆ Maximale Größe des Basic-Programms: 64kByte (320kByte bei COMTAC 3000).

◆ 64kByte stehen für Variable zur Verfügung.

Speicheraufteilung

System/Speicheraufteilung



- ◆ jede Matrize (Array)
 - ◆ die System Parameter (System CW's; Ct. 0 ... Ct. 100)
 - ◆ die Benutzer Parameter (User CW's; Ct. 101 ... Ct. 200)
- Diese werden von COMTAC unterschiedlich behandelt.

Programm-file	Datenfile	Matrizen/ User CW's	System CW's
Berechnung der Prüfsumme			
immer beim Abpeichern	1. immer beim Ab-speichern 2. autom. bei jedem Schreibzugriff auf ein Element *1 3. per Befehl	1. autom. bei jedem Schreibzugriff auf ein Element *1 2. per Befehl	1. autom. bei jedem Schreibzugriff auf ein Element
Prüfung der Prüfsumme *1			
beim Laden	1. bei Power-On oder Reset 2. beim Laden	1. bei Power-On oder Reset	1. bei Power-On oder Reset
Fehlermeldung			
Checksum Error in Program	Checksum Error in Data	Checksum Error in Data / User CWs	Checksum Error in System CWs

*1 Funktion ist abschaltbar

Betriebsarten der Checksummenbildung

Einstellung von CT.(13)	Programm/ System CWs	Datenfile /Matrizen	User CWs
CHECKSUM-Prüfung aus	Bit 11=0	Bit 4=0	Bit 6=0
CHECKSUM-Prüfung ein	Bit 11=1	Bit 4=1	Bit 6=1
CHECKSUM-Bildung automatisch	-	Bit 5=0	Bit 7=0
CHECKSUM-Bildung per Befehl	-	Bit 5=1	Bit 7=1

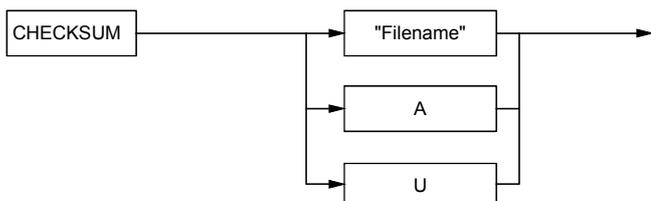
Wenn Sie mehrere Änderungen in Datenfiles, Matrizen oder Parameter (Cws) vornehmen, dann wird bei jeder einzelnen Änderung die Checksumme gebildet.

Durch manuelle Checksummenbildung per Befehl nach dem aller Änderungen wird die Programmlaufzeit verkürzt.

Fehlererkennung

Wir bei Power On/Reset in einem Datenfile oder in einem Array (Matrix) ein Prüfsummenfehler erkannt, wird der Name des fehlerhaften Datenfile/Array im String \$(#0) gespeichert.

8.2 Checksummenbildung



Für die Anzeige von eventuellem Datenverlust im ZP-Ram bildet COMTAC sogenannte Prüfsummen für die einzelnen Dateien bzw. Daten die im ZP-Ram gespeichert sind.

Es wird jeweils eine eigene Prüfsumme gebildet für:

- ◆ jedes Programmfile
- ◆ jedes Datenfile

8.3 Dateinamen

8.3.1 Programm-File

Der Name für ein Basic-Programmfile besteht aus max. 8 Zeichen. Dieser kann mit einer Namensweiterung (Dateityp) ergänzt werden. Sie besteht aus einem Punkt und darauf folgend höchstens drei Zeichen.

COMTAC-Basic hat zwei Namensweiterungen reserviert für Dateien mit besonderer Bedeutung:

- ◆ '.ASP' kennzeichnet ein Basic-Programmfile mit Auto-Start-Funktion.
 - ◆ '.DAT' kennzeichnet ein COMTAC-Datenfile.
- Ansonsten ist die Dateinamenserweiterung frei.

8.3.2 Daten-File

Der Name für ein COMTAC-Daten-File besteht, wie ein Basic-Programmfile, aus max. 8 Zeichen, jedoch muß die Namensweiterung (Dateityp) '.DAT' immer angehängt werden, damit COMTAC-Basic diese Datei als Datenfile erkennt und entsprechend verarbeitet.

Für die Dateinamen können Buchstaben, Zahlen und folgende andere Zeichen verwendet werden:

) \$ ^ ! (- @ } ' & _ { ~ ^ %

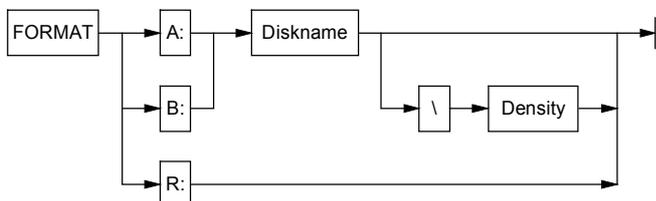
Kleinbuchstaben werden automatisch in Großbuchstaben übersetzt.

Die COMTAC-Basic-Befehle

FORMAT; DIR; STORE; LOAD; DEL; COPY; RENAME

unterstützen den Umgang mit Programm- und Datenfiles. Ihre Funktion wird auf den nachfolgenden Seiten näher erläutert.

8.4 Format



Funktion:

Mit dieser Anweisung werden das ZP-RAM (Laufwerk R) oder eine Diskette formatiert, d.h. neu initialisiert und alle Dateien gelöscht.

Density-Angabe bei Diskettenlaufwerk

- ◆ 'N' = normal Density (720k Byte) und
- ◆ 'H' = High Density (1.44 M Byte).

Keine Angabe bedeutet normal Density.

Beispiel :

FORMAT B: "COMTAC.001" \ N

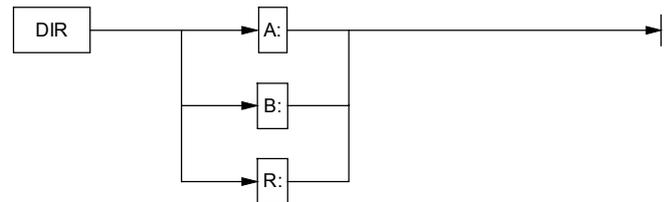
FORMAT A: \$(4)

FORMAT R:

Anmerkung:

Der Diskettenname kann direkt in Hochkomma oder indirekt in einem String angegeben werden.

8.5 DIR



Funktion:

Diese Anweisung gibt das Inhaltsverzeichnis (Directory) des angegebenen Speichermediums (Laufwerk) R, A, oder B auf dem Bildschirm und dem COMTAC-Display aus. Das COMTAC befindet sich nach Ausführung dieser Anweisung im Directory-Mode (DIR-Mode).

Beispiel: DIR A: (externes Diskettenlaufwerk)

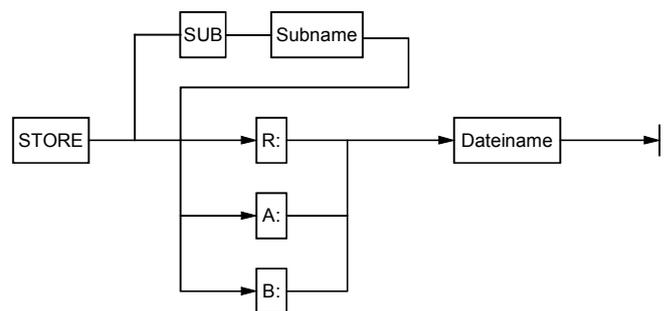
DIR B: (externes Diskettenlaufwerk)

DIR R: (ZP-RAM)

Anmerkung:

Im Directory-Mode kann das gewünschte File mit der Cursor-Taste 'UP' oder 'DOWN' angewählt werden. Die Funktionen COPY, DEL, STORE, LOAD, DIR, RENAME und Format werden bediengeführt.

8.6 STORE [S.]



Funktion:

Mit dieser Anweisung wird das Programm oder die Daten aus dem Arbeitsspeicher (RAM) des COMTAC unter dem entsprechenden Dateinamen in das angegebene Laufwerk (R, A, oder B) abgespeichert.

Beispiel: STORE A: "BAHN.TXT"

STORE R: UMEM\$(1) [1,8]

Anmerkung:

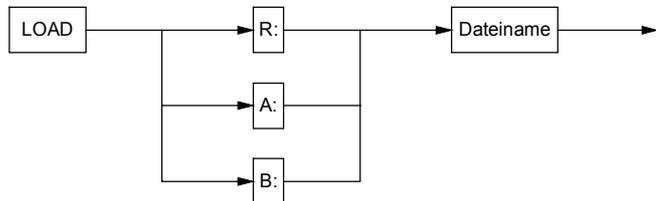
Diese Anweisung ist programmierbar.

Der Dateiname kann direkt in Hochkomma oder indirekt in einem String angegeben werden.

COMTAC 8. Abspeichern und Laden von Programmen und Daten

Das 4 Kbyte-Datenfeld wird abgespeichert, wenn als Dateinamenserweiterung "DAT" verwendet wird.

8.7 LOAD [L.]



Funktion:

Diese Anweisung lädt das Programm- oder Datenfile mit dem entsprechenden Dateinamen aus dem angegebenen Laufwerk (R, A, oder B) in den Arbeitsspeicher (RAM) des COMTAC.

Beispiel: LOAD A:"DEMO.BAS"
LOAD R:"TEST.DAT"
LOAD A:\$(7)

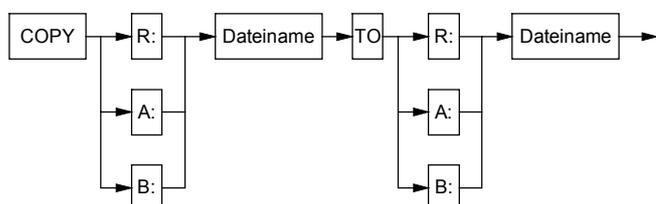
Anmerkung:

Diese Anweisung ist programmierbar.
Der Dateiname kann direkt in Hochkomma oder indirekt in einem String angegeben werden.
Im Run-Mode wird nach Ausführung des LOAD-Befehls das neu geladene Programm automatisch gestartet.

Achtung: Variablen werden gelöscht.

Hinweis: Tritt während dem Laden eines BASIC-Programmes von Diskette ein Fehler auf, wird der Ladevorgang bis zu 5 Mal wiederholt. Sollte danach das Laden nicht erfolgreich beendet worden sein, wechselt das Betriebssystem automatisch vom RUN-Mode in den COMMAND-Mode und zeigt die entsprechende Fehlermeldung des zuletzt aufgetretenen Fehlers an.

8.8 COPY [C.] Datei



Funktion:

Diese Anweisung kopiert Programm- und Datenfiles von Laufwerk zu Laufwerk.

Beispiel: COPY R: "DEMO.TXT TO A: "TEST2.TXT"
COPY R: \$(1) TO A: \$(1)

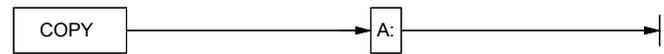
Anmerkung:

Diese Anweisung ist nicht programmierbar.

Das aktuelle Programm im Arbeitsspeicher wird durch den COPY-Befehl von dem zu kopierenden Programm überschrieben.

Ebenso wird das Data-Text-Feld überschrieben wenn ein Datenfile kopiert wird.

8.9 COPY Disk



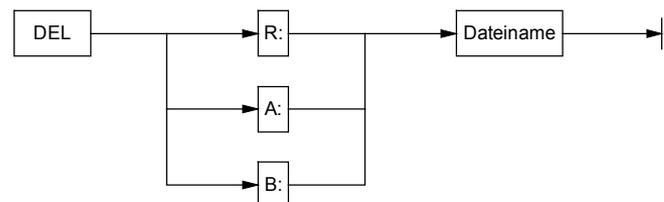
Funktion:

Diese Anweisung kopiert den Inhalt von Laufwerk A auf Laufwerk B.

Anmerkung:

Diese Anweisung ist programmierbar.

8.10 DEL Datei



Funktion:

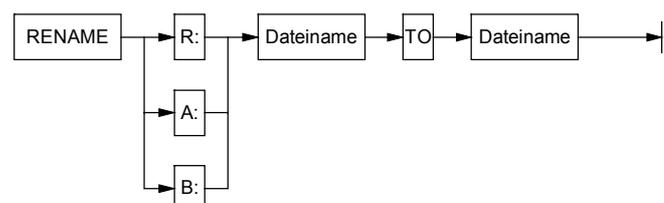
Diese Anweisung löscht Programm- oder Datenfiles mit dem entsprechenden Dateinamen auf dem angegebenen Laufwerk (A, B, oder R).

Beispiel: DEL A:"AUTOKOM.BAS"
DEL R:\$(A)

Anmerkung:

Diese Anweisung ist programmierbar.
Der Dateiname kann direkt in Hochkomma oder indirekt in einem String angegeben werden.

8.11 RENAME



Funktion:

Mit dieser Anweisung kann der Dateiname verändert werden.

Beispiel: RENAME R: "MASCH" TO "MASCH1"

RENAME B: "RTEST" TO "RTEST.TXT"

Anmerkung:

Diese Anweisung ist programmierbar.

Der Dateiname kann direkt in Hochkomma oder indirekt in einem String angegeben werden.

Findet das COMTAC kein AUTO-START-Programm im ZP-RAM (Laufwerk R) sucht es weiter auf der Diskette im Laufwerk A.

Die AUTO-START-Funktion kann explizit freigegeben oder gesperrt werden mit dem Autostart-Flag im Parameter 6.

8.12 AUTO-START-Funktion

Das COMTAC sucht nach POWER ON automatisch im ZP-RAM (Laufwerk R) nach einem Basic-Programmfile mit der Namensweiterung '.ASP'.

Dieses Programm wird in den Arbeitsspeicher kopiert und gestartet.

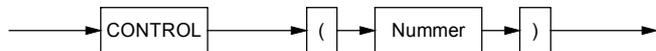
8.13 DEBUG-Funktion

Das COMTAC bietet für den Programmtest die Möglichkeit, daß während das Basic-Programm läuft, die aktuelle Zeilennummer im COMTAC-Display angezeigt wird.

Mit Parameter 16 kann die Position dieser Anzeige eingestellt werden. Bei Parameter P16 = 0 ist die DEBUG-Funktion gesperrt.

9. Parameter

9.1 CONTROL [CT.]



Funktion:

Parameter 0..100 dienen dem COMTAC zur Speicherung der System- und Schnittstellenparameter. Diese Parameter können ausgelesen und in ihrem zulässigen Bereich verändert werden.

Die Beschreibung der Parameter finden Sie auf den folgenden Seiten.

Parameter 101...200 sind frei. Sie können diesen Bereich zum Abspeichern von Programm- oder Maschinendaten verwenden.

Parameter	Angabe	Bereich	Beschreibung
Nummer	num. Ausdr.	0 - 200 0-100 101-200	Parameter -Nummer COMTAC System-Parameter Anwender-Parameter

Beispiel 1: PRINT CONTROL(10)

Beispiel 2: CONTROL(X) = 100

9.2 Parameter - Übersicht

Nr.	Verwendung
00...09	COMTAC System-Parameter
10...19	COMTAC - Systemparameter
20...29	reserviert
30...39	reserviert
40...49	RS232/3 - Schnittstellen-Parameter
50...59	RS232/1 - Schnittstellen-Parameter
60...69	RS485/1 - Schnittstellen-Parameter
70...79	RS485/1 - Schnittstellen-Parameter (Feldbus)
80...89	RS485/2 - Schnittstellen-Parameter
90...99	RS232/2 - Schnittstellen-Parameter
100	Terminal-Schnittstellenzuordnung
101...200	Anwender spezifisch

9.3 System-Parameter

Nr.	Funktion	Standard
00	COMTAC-Typ und Softwareversion; wird bei "Power on" gesetzt Beispiel: Mit der Software Version 2.53 ist bei einem COMTAC 2000 CT.(0) = 2253 und bei einem COMTAC 3000 Ct.(0)=3253.	2000/ 3000
01	Datum der Betriebssoftware	-
02	Terminal-Typ 0 = TV 905 3 = VT100	0
03	Fabrikations-Nummer	-
04	Prüfdatum	-
05	Power-ON-Wert von A1 ... A16 Nach Power-ON oder Ctrl_R erfolgt die automatische Zuweisung: BINOUT(16~1)-=Parameter 5	0
06	Power-ON-Wert von A17 ... A32 Nach Power-ON oder Ctrl_R erfolgt die automatische Zuweisung: BINOUT(32~17)-=Parameter 6	0
07	Einschaltverzögerung 1 - 255 1=100ms	50
08	REQOUT Time-Out 1 - 255 1 = 100 ms	10
09	Repeat-Zeit für COMTAC-Tastatur 1 - 255 1 = 100 ms	1
10	Schnittstellenzuordnung zum Diskettenlaufwerk 0 = RS232/1 ist das Diskettenlaufwerk 1 = RS485/1 ist das Diskettenlaufwerk 2 = RS232/2 ist das Diskettenlaufwerk 3 = RS485/2 ist das Diskettenlaufwerk 4 =RS232/3 ist das Diskettenlaufwerk 99 = kein Diskettenlaufwerk	99
11	Floppy-Timeout-Wert 0 - 255 (1 = 0.1s) 0 = keine Timeout-Überwachung	10
12	Floppy-Baudrate 4 = 2400 5 = 4800 6 = 9600 7 = 19200 8 = 38400	7

9.4 Variable Überwachungszeit für den Up/Download

13	System-Flags mit folgender Bedeutung:	0
	Bit	Funktion
0	Parallel-Eingabe (Terminal / Folientastatur)	0 = off, 1 = on
1	Listing-Option; 0 = Normal, 1 = Befehlskürzel	0
2	Tastatur-Repeat;	0 = off, 1 = on
3	F8-Funktion 0=Stop 1:ohne Funktion	0
4	Checksumme-Daten	0=off 1=on
5		0=auto 1=Befehl
6	Checksumme-User Param.	0=off 1=on
7		0=auto 1=Befehl
8	Autostartfunktion ZP-RAM;	0 = off, 1 = on
9	Autostartfunktion Diskette;	0 = off, 1 = on
10	Autostartfunktion NV-Ram	0 = off, 1 = on
11	Checksumme-Programm und Systemparameter;	0 = off, 1 = on
12	NV-Ram	0 = off, 1 = on
13	Datumsausgabe;	0 =engl. 1 = dt
14		0
15	Ready-Beep;	0 = off, 1 = on
14	Positionswert für die automatische Anzeige der Uhrzeit im LCD 0 = Anzeige gesperrt 1...160 = Position des 1. Zeichens im LCD	33
15	reserviert	-
16	Positionswert für die automatische Zeilenr.-Anzeige im LCD bei der DEBUG-Funktion 0: Anzeige gesperrt 1...160: Position des 1. Zeichens im LCD.	0
17	Überwachungszeit (0 ≡ 0,5s)	0
18	Passwortschutz	-
19	Erweiterter Passwortschutz	-

➡ Mit den Parametern P10-12 ist die Schnittstelle zum HFM2 (Doppeldiskettenlaufwerk) eingestellt. Die Einstellung weiterer Schnittstellendaten der verwendeten Schnittstelle übernimmt COMTAC selbst.

➡ Durch Drücken von F12 während dem Einschalten von COMTAC werden die COMTAC - Parameter auf Standardwerte gesetzt.

9.4 Variable Überwachungszeit für den Up/Download

Mit P17 kann die Überwachungszeit (Timeout) für den Empfang von Zeichen vom PC in 0,1s Schritten im Bereich von 0,1 bis 25,5s eingestellt werden.
Mit P17 = 0 ist der default Wert von 0,5s eingestellt.

Dies ist notwendig um den Up/Download auch über ein Modem zu ermöglichen.

9.5 Passwortschutz

Mit P18 kann ein sogenannter Passwortschutz aktiviert werden, um das Basicprogramm vor unerlaubtem Zugriff zu schützen.

P18 = 0 ⇨ Passwortschutz ausgeschaltet.

P18 <> 0 ⇨ Passwortschutz eingeschaltet.

Passwortschutz aktivieren / deaktivieren

Der Passwortschutz wird aktiviert, indem P18 mit einem Wert (dem Passwort) beschrieben wird.

Durch erneutes beschreiben von P18 mit dem selben Wert (Passwort) wird der Passwortschutz deaktiviert.

Funktionsbeschränkungen bei aktiviertem Passwortschutz

Mit aktiviertem Passwortschutz sind folgende Funktionen gesperrt:

- ◆ das Editieren des Programmes
- ◆ das Listen des Programmes (LIST)
- ◆ der Download eines Programmes

Erweiterter Passwortschutz

Mit P19 wird der erweiterte Passwortschutz aktiviert.

P19 = 0 ⇨ erweiterter Passwortschutz ausgeschaltet.

P19 <> 0 ⇨ erweiterter Passwortschutz eingeschaltet.

Erweiterter Passwortschutz aktivieren / deaktivieren

Der erweiterte Passwortschutz wird aktiviert, indem P19 mit einem Wert (dem Passwort) beschrieben wird.

Durch erneutes beschreiben von P19 mit dem selben Wert (Passwort) wird der Passwortschutz deaktiviert.

Funktionsbeschränkungen bei aktiviertem erweitertem Passwortschutz

Mit aktiviertem erweitertem Passwortschutz sind folgende Funktionen gesperrt:

- ◆ der Upload eines Programmes.

➡ P18 und P19 liefern den Wert TRUE (65535) wenn diese ein Passwort enthalten.

NV-Ram als Arbeitsspeicher in COMTAC 3000 mit Gerätenr.: > 0982600000

Das Betriebssystem ist in der Lage auch ein NV-Ram (ZP-Ram) als Arbeitsspeicher zu verwenden. Dies hat den Vorteil, daß das aktuelle Programm nicht immer wieder im ZP-Ram oder auf Diskette gespeichert werden muß.

Dem Betriebssystem muß das NV-Ram als Arbeitsspeicher angezeigt werden mit CT.(13) Bit 12 = 1.

Es ist möglich das Programm im NV-Ram nach Power On mit der Autostartfunktion zu starten.

Diese Funktion wird aktiviert mit CT.(13) Bit 10 = 1.

Für die Anzeige von eventuellem Datenverlust im Arbeitsspeicher (NV-Ram) kann mit der Anweisung CHECKSUM P eine Prüfsumme gebildet werden. Diese wird nach Power On neu berechnet und mit der alten verglichen, sofern in CT.(13) Bit 10 und Bit 11 = 1 sind.

10. Allgemeine Anweisungen

10.1 CLEAR [CL.]

CLEAR →

Funktion:

Diese Anweisung setzt alle Variablen auf 0, alle Strings und Interrupt's werden gelöscht. Bereits ausgeführte DIM-Anweisungen sind nicht mehr gültig. Control- und Argument-Stack werden zurückgesetzt.

Ausnahme: Arrays (Matrizen) im ZP-RAM bleiben unverändert.

Beispiel: CLEAR

Anmerkung:

Der Inhalt des Data-Text-Feldes, der Basic - TIMER und die Strings \$(#0), \$(#1), \$(#2), \$(#3), \$(#4) werden **nicht** gelöscht.

10.2 CLEARI

CLEARI →

Funktion:

Diese Anweisung löscht alle Interrupts.

Beispiel: CLEARI

Anmerkung:

Der Basic TIMER wird durch diese Anweisung nicht gesperrt.

10.3 CLEARS

CLEARS →

Funktion:

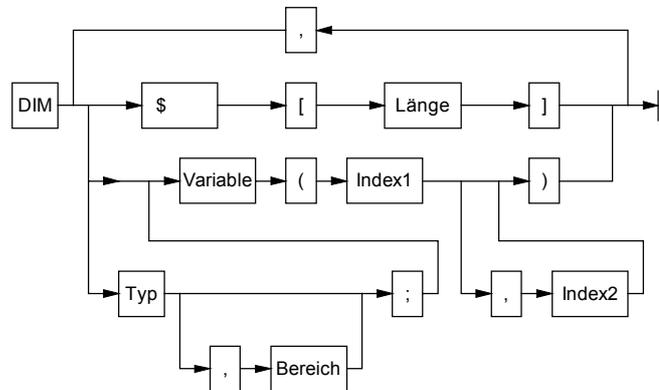
Diese Anweisung setzt den Control- und den Argument-Stack zurück.

Beispiel: CLEARS

Anmerkung:

Diese Anweisung kann verwendet werden um einen bevorstehenden Stack Überlauf zu vermeiden, weil im Programm z.B. FOR NEXT-Schleifen nicht ordnungsgemäß verlassen werden.

10.4 DIM



Funktion:

1. reserviert Speicherplatz für Zeichenketten.
2. reserviert Speicherplatz für ein- oder zweidimensionale Matrizen.

Mit der Option "Typ" kann einer der unten aufgeführten Datentypen für die zu dimensionierende Matrize gewählt werden. Wird der Datentyp nicht angegeben, ist die Matrize automatisch vom Typ Floating Point.

Mit der Option "Bereich" kann festgelegt werden, wo die Matrize gespeichert werden soll. Wird kein Bereich angegeben, wird die Matrize automatisch im Variablenspeicher angelegt.

Parameter	Angabe	Bereich	Beschreibung
\$	-	-	gültiger Stringname
Länge	num. Ausdr.	1-255	max. Länge des Strings
Variable	-	-	gültiger Variablenname
Index 1	num. Ausdr.	1 - 254	Anzahl der Matrizenel- emente -1
Index 2	num. Ausdr.	1 - 254	Anzahl der Matrizenel- emente -1
Typ	Zahl	0 (1 Byte) 1 (1 Byte) 2 (2 Byte) 3 (2 Byte) 4 (4 Byte) 5 (4 Byte) 6 (6 Byte) 7 (4 Byte) 8 (4 Byte) 9 (6 Byte)	unsigned character character (+/- 127) unsigned integer integer long unsigned integer long integer Floating point Festpunktzahl mit 3 Nachkommastellen Festpunktzahl mit 6 Nachkommastellen DSP Real
Bereich	Zahl	0 1 2 3 4 5 6	Variablenspeicher Arbeitsspeicher Bereich 0...63k ZP-RAM (Laufwerk R) (netzausfallsicher) Arbeitsspeicher Bereich 64...127k Arbeitsspeicher Bereich 128...191k Arbeitsspeicher Bereich 192...255k Arbeitsspeicher Bereich 256...319k

Beispiel: DIM \$(5)[80]

reserviert Speicher für den String \$(5) der max. 80 Zeichen aufnehmen kann.

DIM A(20)

definiert eine eindimensionale Matrize mit 21 Elementen:

A(0), A(1), ... A(20).

DIM 1;POS(10,2)

definiert eine zweidimensionale Matrize mit 33 Elementen;

die Elemente POS(0,0) ... POS(10,0)

POS(0,1) ... POS(10,1)

POS(0,2) ... POS(10,2).

DIM 6,2;Z(5,0)

definiert eine eindimensionale Matrize mit 6 Elementen; die Elemente Z(0), Z(1), ... Z(5).

Diese Matrize hat den Datentyp Floating Point und wird im ZP-RAM (Laufwerk R) netzausfallsicher gespeichert.

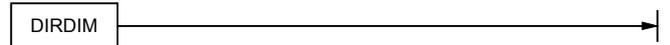
Anmerkung:

Die DIM-Anweisung muß angewendet werden, für:

- ◆ Zeichenketten die eine max. Länge ungleich der Standardlänge haben sollen (Standardlänge = 16).
- ◆ eindimensionale Matrizen.
- ◆ zweidimensionale Matrizen mit einer Anzahl von Elementen die ungleich der Standardanzahl sein soll (Standardanzahl = 121) d. h. eine Matrize mit (0,0) bis (10,10) muß nicht dimensioniert werden.

Die NEW- und CLEAR-Anweisung wirken nicht auf Matrizen, die im ZP-RAM (Laufwerk R) liegen. Deshalb müssen solche Matrizen explizit mit der DEL-Anweisung gelöscht werden.

10.5 DIRDIM

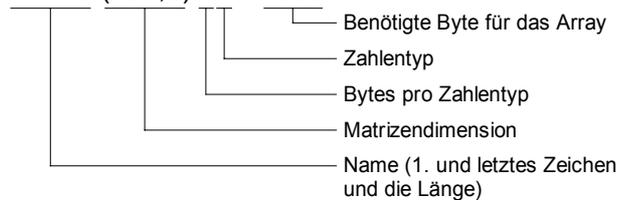


Funktion:

Abfrage von Größe und Speicherbelegung der im Laufwerk R vorhandenen Arrays (Matrizen).

Antwort für alle Arrays:

PxxxS (250,0) 6F - 1506



Bedeutung des Zahlentyps:

Zahlentyp	Kurzform
unsigned character	UC
character	C
unsigned integer	UI
integer	I
long unsigned integer	UL
long integer	L
Floating point	F
long int. 3 Nachkommast.	L.3
long int. 6 Nachkommast.	L.6
DSP Real	D

10.6 DEL Variable



Funktion:

Diese Anweisung löscht eine Matrize aus dem ZP-RAM (Laufwerk R).

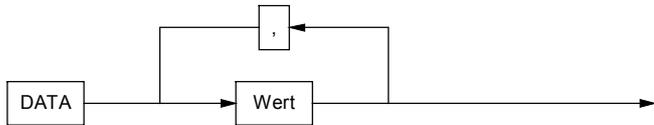
10.7 DEL DIM



Funktion:

Diese Anweisung löscht alle Matrizen die sich im ZP-RAM (Laufwerk R) befinden.

10.8 DATA



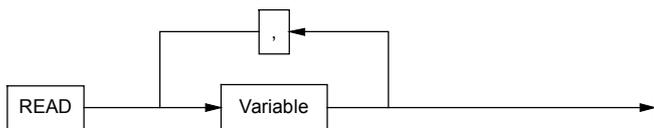
Funktion:

Diese Anweisung dient zum Anlegen eines Datenfelds im Basic-Hauptprogramm (In einem Sub dürfen keine DATA-Anweisungen stehen). Die Daten können mit der READ-Anweisung gelesen werden.

Parameter	Angabe	Bereich	Beschreibung
Wert	num. Ausdr.		programmspezifische Daten

Beispiel: DATA 10,PI/2,A*B
DATA 50,100,150,200,250,300

10.9 READ



Funktion:

Diese Anweisung dient zum Lesen eines Datenfelds im Basic-Programm. Der Ausdruck im Datenfeld wird berechnet und der Variablen zugewiesen.

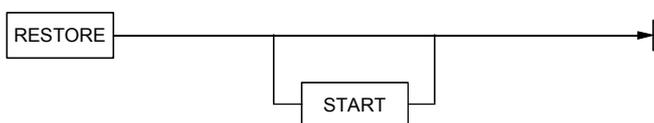
Parameter	Angabe	Bereich	Beschreibung
Variable	gültiger Variablenname		Variable der ein Wert vom Datenfeld zugewiesen wird

Beispiel: READ X,Y,Z oder READ D1,D2,D3,D4,D5

Anmerkung:

Nach dem Programmstart steht der interne READ-Zeiger auf dem 1. DATA-Wert. Nach dem Lesevorgang steht der READ-Zeiger auf dem nächsten DATA-Wert. Mit der RESTORE-Anweisung kann der READ-Zeiger auf den 1. DATA-Wert oder auf eine beliebige Zeilennummer (mit DATA-Anweisung) gesetzt werden.

10.10 RESTORE

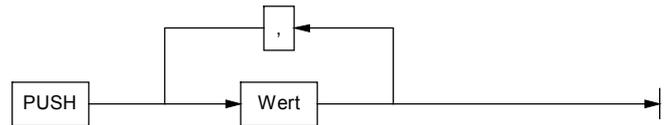


Funktion:

Diese Anweisung setzt den READ-Zeiger auf den 1. DATA-Wert zurück oder auf die angegebene Zeilennummer.

Beispiel: RESTORE
RESTORE 100

10.11 PUSH



Funktion:

Diese Anweisung dient zur Übergabe von Parametern an Unterprogramme und zum Zwischenspeichern von Variablen.

Der angegebene Wert wird berechnet und auf dem Argument-Stack abgelegt.

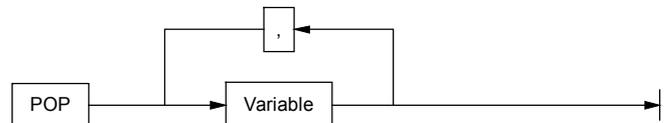
Parameter	Angabe	Bereich	Beschreibung
Wert	num. Ausdr.		Wert der auf dem A-Stack abgelegt werden soll

Beispiel: PUSH A,B,C

Anmerkung:

Der Argument-Stack arbeitet nach dem Prinzip 'first in - last out' und kann max. 39 Werte speichern. Er wird auch vom Betriebssystem benutzt. Mit der POP-Anweisung kann ein auf dem Argument-Stack abgelegter Wert einer Variablen zugewiesen werden.

10.12 POP



Funktion:

Diese Anweisung dient zur Übernahme von Werten die auf dem Argument Stack liegen. Die übernommenen Werte werden direkt Variablen zugewiesen.

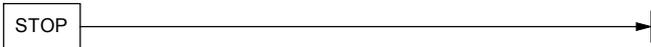
Parameter	Angabe	Bereich	Beschreibung
Variable	gültiger Variablenname		Variable der ein Wert vom A-Stack zugewiesen wird

Beispiel: POP X,Y,Z
FOR I=0 TO 10:POP X(I):NEXT I

Anmerkung:

Der Argument-Stack arbeitet nach dem Prinzip 'first in - last out'. und kann max. 39 Werte speichern.

10.13 STOP



Funktion:

Diese Anweisung ermöglicht dem Anwender das Programm an spezifizierten Punkten zu unterbrechen, um Variable anzuzeigen und/oder zu verändern.

Beispiel: STOP

Anmerkung:

Das Programm kann mit der CONT-Anweisung fortgesetzt werden.

Parameter	Angabe	Bereich	Beschreibung
Zeit	num. Ausdr.	1 - 65535	Angabe in Millisekunden

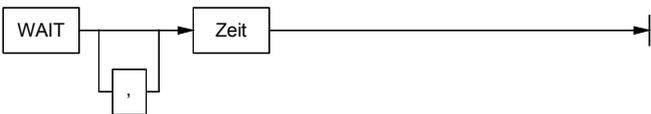
Beispiel: WAIT 1000 oder WAIT Z1

Anmerkung:

Diese Anweisung kann durch Ctrl-C oder einen Interrupt unterbrochen werden.

Wird vor die Zeitangabe ein Komma gesetzt, dann wird die Anweisung nicht durch einen Interrupt unterbrochen.

10.14 WAIT



Funktion:

Diese Anweisung veranlaßt das COMTAC für die angegebene Zeit zu warten, bevor die nächste Anweisung ausgeführt wird.

10.15 REM [!]



Funktion:

Diese Anweisung ermöglicht das Einfügen von Kommentarzeilen in das Basic-Programm.

Parameter	Angabe	Bereich	Beschreibung
Text			beliebiger Text

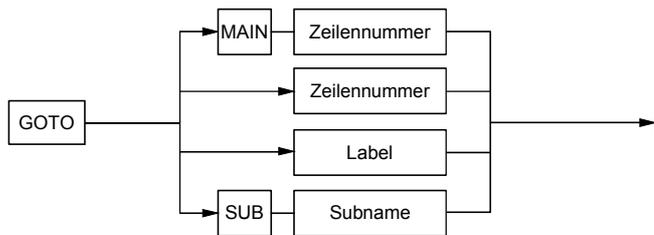
Beispiel: REM Dies ist eine Kommentarzeile
! Positionsabfrage Unterprogramm

Anmerkung:

Basic-Anweisungen die **nach REM** oder ! stehen (selbe Zeile) werden **nicht** abgearbeitet (REM und ! haben die gleiche Bedeutung).

11. Programmverzweigungen und Programmschleifen

11.1 GOTO [GT.] Zeilennummer



Funktion:

Diese Anweisung führt einen Programmsprung zur angegebenen Zeilennummer oder zum angegebenen Sub durch.

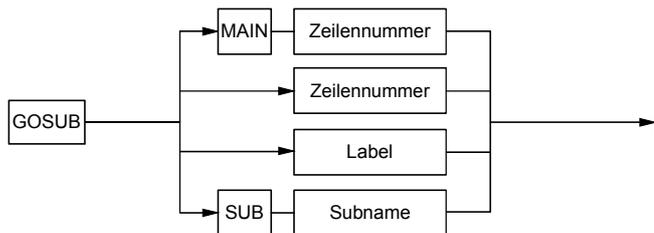
Parameter	Angabe	Bereich	Beschreibung
Zeilennummer	Zahl	0 - 65535	Zieladresse der Programmverzweigung

Beispiel 1: GOTO 30

Beispiel 2: GOTO SUB "Test"

Beispiel 3: GOTO MAIN 2000

11.2 GOSUB [GS.]



Funktion:

Diese Anweisung ruft ein Unterprogramm auf, das mit der angegebenen Zeilennummer oder dem angegebenen Sub beginnt.

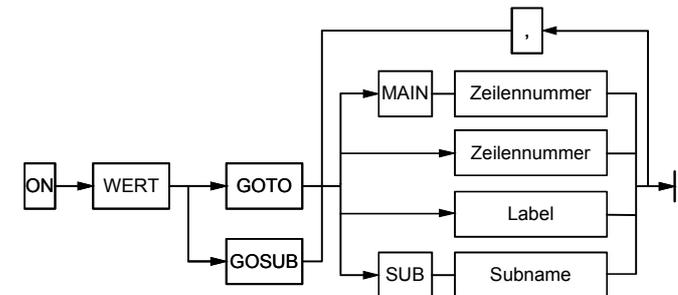
Parameter	Angabe	Bereich	Beschreibung
Zeilennummer	Zahl	0 - 65535	1. Zeilen-Nr. des Unterprogrammes

Beispiel: GOSUB 500 oder GOSUB 30000

Anmerkung:

Ein Unterprogramm wird mit der Anweisung RETURN abgeschlossen.

11.3 ON



Funktion:

Diese Anweisung ermöglicht eine bedingte Programmverzweigung.

Der angegebene Wert bestimmt, an welche Zeilennummern das Programm verzweigt.

Ist der Wert 0, verzweigt das Programm an die 1. Zeilennummer, die in der Anweisung angegeben ist.

Ist der Wert 1, verzweigt das Programm an die 2. Zeilennummer.

Ist der Wert 2, verzweigt das Programm an die 3. Zeilennummer usw.

Parameter	Angabe	Bereich	Beschreibung
Wert	num. Ausdr.	0 - 100	Kriterium für die Programmverzweigung
Zeilennummer	Zahl	0 - 65535	Zieladresse der Programmverzweigung

Beispiele:

ON Q GOSUB 1000,2000,3000,4000,5000,6000,7000,8000

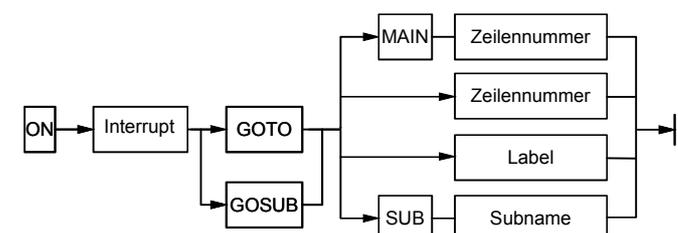
ON V-100 GOTO 300,400,500,600,10000

Anmerkung:

Ist der angegebene Wert < 0 wird ein Bad Argument Fehler erzeugt.

Ist der angegebene Wert > als die Anzahl der Zeilennummern wird ein Bad Syntax Fehler erzeugt.

11.4 ON-Interrupt



Funktion:

Eine Programmverzweigung wird durch ein besonderes Ereignis ausgelöst. Näheres siehe im Kapitel "Interrupt-Verarbeitung".

11.5 RETURN [R.]



Funktion:

Diese Anweisung ist der Abschluß eines Unterprogrammes. Es erfolgt ein Programmsprung an die Stelle, von wo das Unterprogramm aufgerufen wurde.

Beispiel: RETURN

11.6 RETI

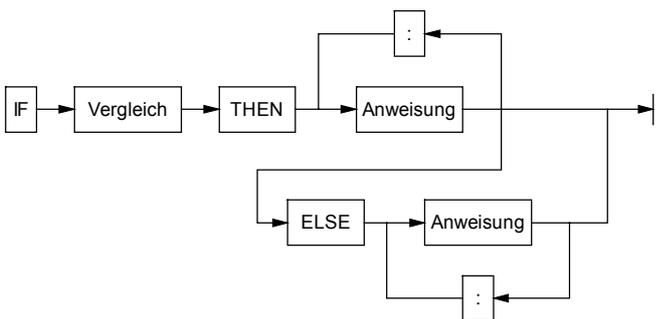


Funktion:

Diese Anweisung ist der Abschluß für Unterprogramme, die durch ein Interrupt aufgerufen wurden. Es erfolgt ein Programmsprung an die Stelle, an der das Hauptprogramm von dem Interrupt unterbrochen wurde.

Beispiel: RETI

11.7 IF - THEN - (ELSE)



Funktion:

Diese Anweisung ermöglicht eine bedingte Programmverzweigung.

1. Ist der angegebene Vergleich wahr werden die Basic-Anweisungen des THEN-Zweiges ausgeführt, der ELSE-Zweig wird übersprungen.
2. Ist der angegebene Vergleich falsch werden die Basic-Anweisungen des ELSE-Zweiges ausgeführt, der THEN-Zweig wird übersprungen. Ist kein ELSE-Zweig angegeben, wird die nächste Basic-Zeile ausgeführt.

Für Vergleich kann jeder gültige Vergleich von numerischen Ausdrücken und/oder der Vergleich von Strings angegeben werden. Im Vergleich können logische Operatoren verwendet werden und Vergleiche können mit den logischen Operatoren verknüpft werden.

Außerdem können für Vergleich direkt die speziellen COMTAC-Basic Funktionen ASK-,REQOUT, CPX_PE oder CPX_MN angegeben werden.

Für Anweisung können alle ausführbaren Basic-Befehle angegeben werden.

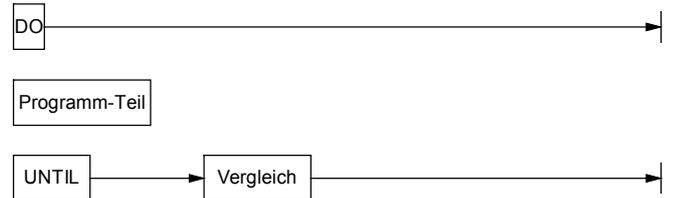
Beispiele:

```
IF A=10 THEN B=1 ELSE B=2
IF X>5 'AND' X<10 'OR' $(3)[1,1]="U" THEN GOTO 500
IF WFS 2 THEN GOSUB 1000
IF ROUT 1,5 THEN PRINT "OK" ELSE PRINT "Error"
IF (IN(1)=1'AND'IN(2)=0) THEN 200 ELSE 300
```

Anmerkung:

Nach der THEN- und ELSE-Anweisung kann direkt eine Zeilen-Nummer angegeben werden, die von COMTAC-Basic als GOTO-Anweisung interpretiert wird. Umfangreiche Vergleiche sollten mit der entsprechenden Klammerung versehen werden um Fehlinterpretationen zu vermeiden.

11.8 DO ... UNTIL



Funktion:

Diese Anweisung ermöglicht eine bedingte Programmschleife.

Der Programm-Teil, der zwischen der DO-Anweisung und der UNTIL-Anweisung steht, wird solange wiederholt, bis der angegebene Vergleich wahr wird.

Für Vergleich kann jeder gültige Vergleich von numerischen Ausdrücken und/oder der Vergleich von Strings angegeben werden. Im Vergleich können logische Operatoren verwendet werden und Vergleiche können mit den logischen Operatoren verknüpft werden.

Außerdem können für Vergleich direkt die speziellen COMTAC-Basic Funktionen ASK-,REQOUT, CPX_PE oder CPX_MN angegeben werden.

Beispiel:

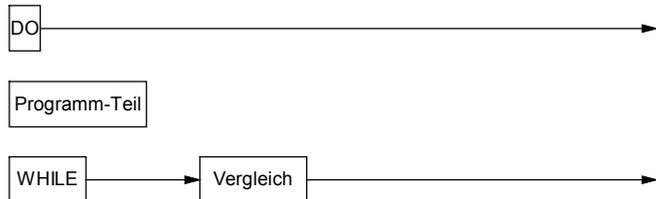
```
10 DO
20 A=A+1
30 PRINT A
40 UNTIL A=10 ! Sprung nach 10, solange die Bedingung nicht erfüllt ist.
```

Anmerkung:

Der angegebene Programm-Teil wird mindestens einmal durchlaufen.

Umfangreiche Vergleiche sollten mit der entsprechenden Klammerung versehen werden, um Fehlinterpretationen zu vermeiden.

11.9 DO ... WHILE



Funktion:

Diese Anweisung ermöglicht eine bedingte Programmschleife. Der Programm-Teil, der zwischen der DO-Anweisung und der WHILE-Anweisung steht, wird wiederholt solange der angegebene Vergleich wahr ist.

Für Vergleich kann jeder gültige Vergleich von numerischen Ausdrücken und/oder der Vergleich von Strings angegeben werden. Im Vergleich können logische Operatoren verwendet werden und Vergleiche können mit den logischen Operatoren verknüpft werden.

Außerdem können für Vergleich direkt die speziellen COMTAC-Basic Funktionen ASK-, REQOUT, CPX_PE oder CPX_MN angegeben werden.

Beispiel:

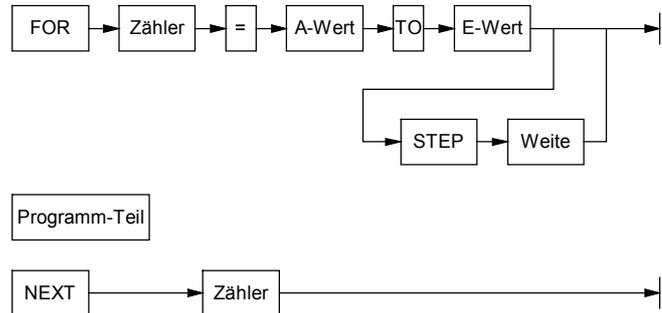
```
10 DO
20 A=A+1
30 PRINT A
40 WHILE A<10 ! Sprung nach 10, bis die Be-
! dingung erfüllt ist.
```

Anmerkung:

Die angegebene Programm-Teil wird mindestens einmal durchlaufen.

Umfangreiche Vergleiche sollten mit der entsprechenden Klammerung versehen werden, um Fehlinterpretationen zu vermeiden.

11.10 FOR ... NEXT



Funktion:

Diese Anweisung definiert eine Programmschleife.

Der Programm-Teil zwischen der FOR- und NEXT-Anweisung wird solange wiederholt, bis der Schleifenzähler den spezifizierten Endwert überschreitet.

Zu Beginn wird der Schleifenzähler auf den angegebenen Anfangswert gesetzt. In der NEXT-Anweisung wird zum Schleifenzähler die Schrittweite addiert und mit dem Endwert verglichen.

Parameter	Angabe	Bereich	Beschreibung
Zähler	Variable		gültiger Variablenname
A-Wert	num. Ausdr.		Anfangswert
E-Wert	num. Ausdr.		Endwert
Weite	num. Ausdr.		Schrittweite

Beispiel: 10 FOR I=1 TO 100 STEP 10
20 PRINT I
30 NEXT I

FOR-NEXT-Schleifen können auch verschachtelt werden:

```
10 FOR I=1 TO 10
20 FOR N=1 TO 10
30 PRINT I,N
40 NEXT N
50 NEXT I
```

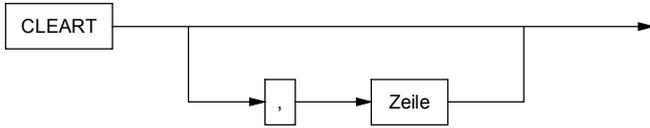
Anmerkung:

Wird keine Schrittweite angegeben ist die Schrittweite 1. Die Schrittweite kann positiv oder negativ sein.

Der angegebene Programm-Teil wird mindestens einmal durchlaufen.

12. Terminal-Ausgabe

12.1 CLEART

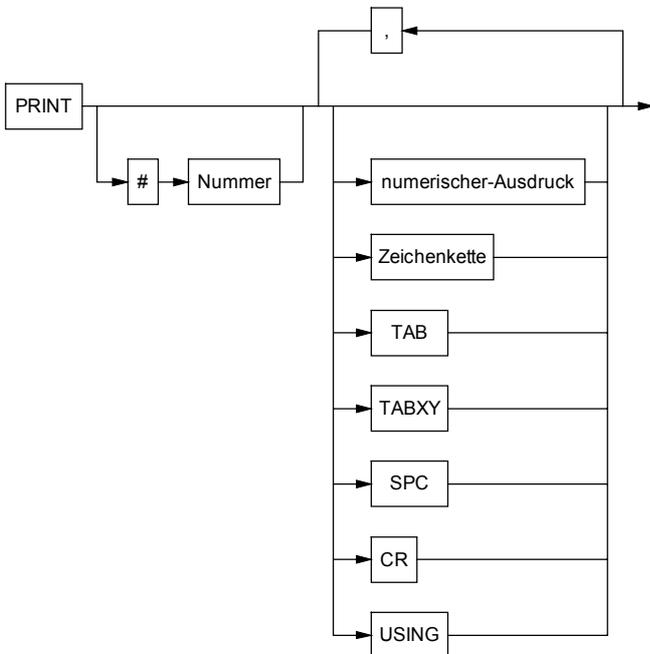


Funktion:
CLEART
 Löschen des gesamten Bildschirms
CLEART,
 Löschen des gesamten Bildschirms ab CURSOR-Position
CLEART,Zeile
 Löschen einer definierten Zeile.
 Hat die Zeilenangabe den Wert 0, so wird ab Cursor-Position bis Zeilenende gelöscht.

Parameter	Angabe	Bereich	Beschreibung
Zeile	num. Ausdr.	1-24	-

Beispiel: CLEART
 CLEART,7

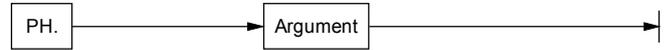
12.2 PRINT [P.]



Funktion:
 Diese Anweisung gibt Zeichen, Zeichenketten und/oder den Wert eines numerischen Ausdrucks an das Terminal oder an die entsprechend angewählte Druckerschnittstelle aus.

Beispiel: PRINT A/30
 PRINT#2 "COMTAC 2000"

12.3 PH.



Funktion:
 Diese Anweisung gibt den Wert des Arguments im Hexadezimalen-Format an das Terminal aus.

Parameter	Angabe	Bereich	Beschreibung
Argument	num. Ausdr.	0-65535	anzuweisender Wert

Beispiel: PH. 55
 PH. M/330

12.4 PB.@

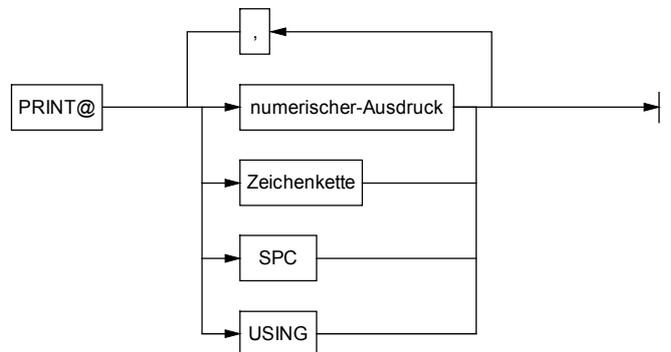


Funktion:
 Diese Anweisung gibt den Wert des Arguments im Binär-Format in der Zeile 22 des Bildschirms aus.

Parameter	Angabe	Bereich	Beschreibung
Argument	num. Ausdr.	0-65535	anzuweisender Wert

Beispiel: DB. A*B+300
 DB. 45000

12.5 PRINT@ [P.@]



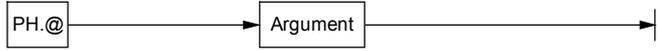
Funktion:
 Diese Anweisung gibt Zeichen, Zeichenketten und/oder den Wert eines numerischen Ausdrucks in der Zeile 22 des Bildschirms aus.

Beispiel: PRINT@ A
 PRINT@ 800*33

Anmerkung:

Die Funktion der Anweisungen SPC und USING finden Sie auf den nachfolgenden Seiten.

12.6 PH.@



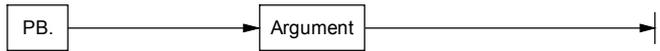
Funktion:

Diese Anweisung gibt den Wert des Arguments im hexadezimalen Format in der Zeile 22 des Bildschirms aus.

Parameter	Angabe	Bereich	Beschreibung
Argument	num. Ausdr.	0-65535	anzuweisender Wert

Beispiel: PH. 34
PH. X/2

12.7 PB.



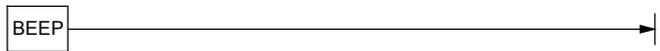
Funktion:

Diese Anweisung gibt den Wert des Arguments im Binären-Format an das Terminal aus.

Parameter	Angabe	Bereich	Beschreibung
Argument	num. Ausdr.	0-65535	anzuweisender Wert

Beispiel: PB. V-845
PB. 377

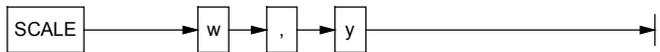
12.8 BEEP



Funktion:

Terminal Tastatur-Beep.
Sendet das Zeichen 07h (BEL) an das Terminal.

12.9 SCALE



Funktion:

Diese Anweisung gibt auf dem Bildschirm in der angegebenen Zeile y und der Zeile y-1 einen Maßstab aus für die Beschriftung der Balkengrafik- Funktion BARGRAPH.

Diese Skala beginnt in der Spalte 30 und wird beschriftet mit 0 bis zum angegebenen Maximalwert w*10.

Parameter	Angabe	Bereich	Beschreibung
w	num. Ausdr.	1-10	Maximalwert der Skala
y	num. Ausdr.	2-24	Angabe der Zeile

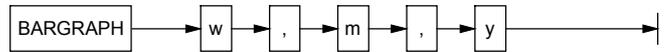
Beispiel: SCALE 3,10

Mit w = 3 und y = 10 wird auf dem Bildschirm in der Zeile 10 und 9, beginnend in der Spalte 30, folgende Skala dargestellt:

```

0   3   6   9  12  15  18  21  24  27  30
|...|...|...|...|...|...|...|...|...|...|
    
```

12.10 BARGRAPH



Funktion:

Diese Anweisung gibt auf dem Bildschirm in der Zeile y, beginnend in der Spalte 31, eine Balkengrafik des Meßwertes m aus, in Abhängigkeit des angegebenen 100%-Wertes w

Parameter	Angabe	Bereich	Beschreibung
w	num. Ausdr.	-	100%-Wert
m	num. Ausdr.	-	Meßwert
y	num. Ausdr.	1-24	Angabe der Zeile

Beispiel: BARGRAPH 30,15,11

Mit w = 30, m = 15 und y = 11 wird auf dem Bildschirm in der Zeile 11, beginnend in der Spalte 31 folgende Balkengrafik dargestellt:

Anmerkung:

Eine passende Skala zur Balkengrafik kann mit der Anweisung SCALE gemacht werden.

12.11 VLINE [V.]



Funktion:

Diese Anweisung zeichnet ab der momentanen Cursor-Position eine vertikale Linie mit definierter Länge.

Parameter	Angabe	Bereich	Beschreibung
Länge	num. Ausdr.	1 - 24	-

Beispiel: VLINE Y
VLINE 8

12.12 HLINE [H.]



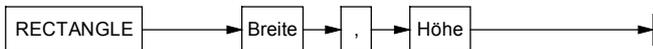
Funktion:

Diese Anweisung zeichnet ab der momentanen Cursor-Position eine horizontale Linie mit definierter Länge.

Parameter	Angabe	Bereich	Beschreibung
Länge	num. Ausdr.	1 - 80	-

Beispiel: HLINE 5
HLINE X

12.13 RECTANGLE [RECT.]



Funktion:

Diese Anweisung zeichnet ab der momentanen Cursorposition ein Rechteck mit einer definierten Breite und Höhe.

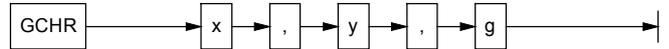
Parameter	Angabe	Bereich	Beschreibung
Breite	num. Ausdr.	1 - 79	
Höhe	num. Ausdr.	1 - 23	

Beispiel: RECTANGLE X,Y
RECTANGLE 7,9

Anmerkung:

Das Rechteck wird von links nach rechts und von oben nach unten gezeichnet.

12.14 GCHR



Funktion:

Diese Anweisung gibt auf dem Bildschirm in der Spalte x und der Zeile y das Grafikzeichen g aus. Folgende Grafikzeichen stehen zur Auswahl:

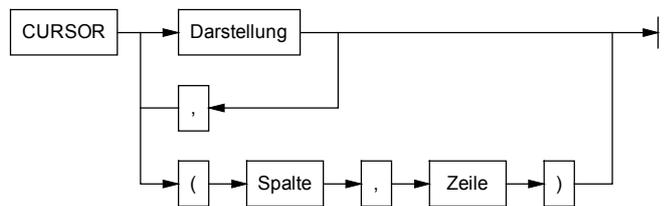
g = 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

Parameter	Angabe	Bereich	Beschreibung
x	num. Ausdr.	1 - 80	Angabe der Spalte
y	num. Ausdr.	1 - 24	Angabe der Zeile
g	num. Ausdr.	1 - 15	Nummer des Grafikzeichen

Beispiel: GCHR 5,5,10

Mit x = 5, y = 5 und g = 10 wird auf dem Bildschirm in der Zeile 5 und Spalte 5 das Grafikzeichen | dargestellt.

12.15 CURSOR [CU.]



Funktion:

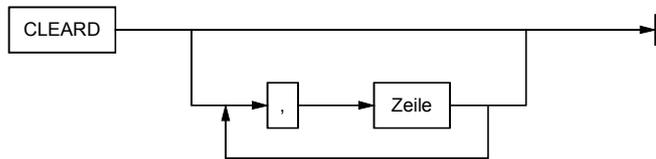
Diese Anweisung verändert die Darstellung des Bildschirm-Cursors und / oder dessen Position.

Parameter	Angabe	Bereich	Beschreibung
Darstellung	num. Ausdr.	0 - 4	Darstellung des Cursors 0 = unsichtbar 1 = Block blinkend 2 = Block statisch 3 = Unterstrich blinkend 4 = Unterstrich statisch
Spalte		1 - 80	-
Zeile	num. Ausdr.	1 - 24	-

Beispiel: CURSOR 4, (10,10)
CURSOR 1

13. LCD-Ausgabe

13.1 CLEAR



Funktion:

CLEAR

Löschen des gesamten Displays

CLEAR,

Löschen des gesamten Displays ab CURSOR-Position

CLEAR,Zeile

Löschen einer oder mehrerer definierten Zeilen des Displays.

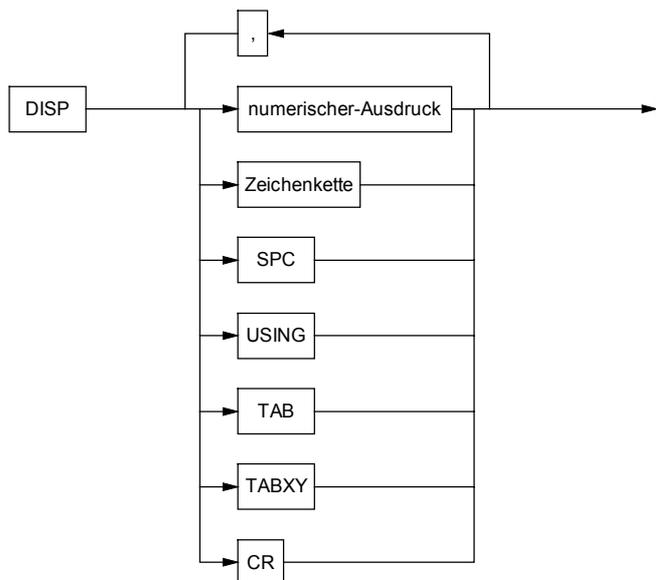
Hat die Zeilenangabe den Wert 0, so wird ab Cursor-Position bis Zeilenende gelöscht.

Parameter	Angabe	Bereich	Beschreibung
Zeile	num. Ausdr.	1-4	-

Beispiel:

CLEAR
 CLEAR,2
 CLEAR,2,3

13.2 DISP [D.]



Funktion:

Diese Anweisung gibt Zeichen, Zeichenketten und/oder den Wert eines numerischen Ausdrucks im COMTAC-Display aus.

Beispiel:

DISP A
 DISP 800*33

13.3 DISPVAR



Funktion:

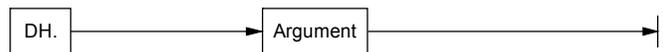
Zyklische Anzeige einer Basic-Variablen im Display.

Achtung: VK + NK muß ≤ 8 sein.

Die Anzeige kann mit STOP DISP zwischengespeichert und mit CONT DISP wieder angezeigt werden (siehe Seite 54)

Beispiel: DISPVAR A, 20, 2, 2, 4

13.4 DH.



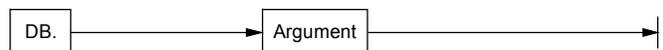
Funktion:

Diese Anweisung gibt den Wert des Arguments im hexadezimalen Format im COMTAC-Display aus.

Parameter	Angabe	Bereich	Beschreibung
Argument	num. Ausdr.	0-65535	anzuweisender Wert

Beispiel: DH. 34
 DH. X/2

13.5 DB.



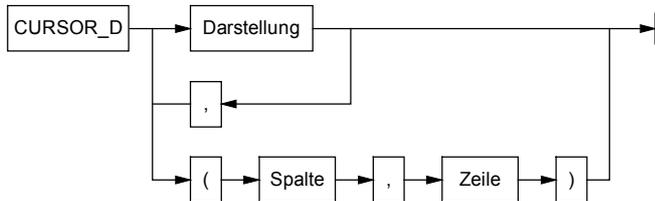
Funktion:

Diese Anweisung gibt den Wert des Arguments im binären Format im COMTAC-Display aus.

Parameter	Angabe	Bereich	Beschreibung
Argument	num. Ausdr.	0-65535	anzuweisender Wert

Beispiel: DB. A*B+300
 DB. 45000

13.6 CURSOR_D



Funktion:
Diese Anweisung verändert die Darstellung des Display-Cursors und / oder dessen Position.

Parameter	Angabe	Bereich	Beschreibung
<i>Darstellung</i>	num. Ausdr.	0 - 4	Darstellung des Cursors 0 = unsichtbar 1 = Block blinkend 2 = Unterstrich statisch 3 = Unterstrich statisch 4 = Unterstrich statisch
<i>Spalte</i>		1 - 40	-
<i>Zeile</i>	num. Ausdr.	1 - 4	-

Beispiel: CURSOR_D 4, (10,1)
CURSOR_D 1

13.7 SETLED



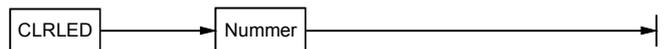
Funktion:
Schaltet die mit Nummer definierte LED (H1 - H4) des COMTAC ein.

Parameter	Angabe	Bereich	Beschreibung
<i>Nummer</i>	num. Ausdr.	1...4	Nummer der LED

Beispiel: SETLED 2 : REM H2 einschalten

Anmerkung:
Folgt der Nummer ein Komma "SETLED nr," dann blinkt die entsprechende LED.

13.8 CLRLED



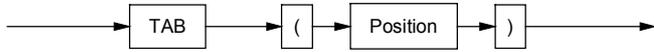
Funktion:
Schaltet die mit Nummer definierte LED (H1 - H4) des COMTAC aus.

Parameter	Angabe	Bereich	Beschreibung
<i>Nummer</i>	num. Ausdr.	1...4	Nummer der LED

Beispiel: CLRLED 2 : REM H2 ausschalten

14. Ausgabeformatbefehle

14.1 TAB



Funktion:

Diese Funktion wird in der PRINT-Anweisung verwendet, um den Cursor an die angegebene Position zu verschieben, durch das Senden der entsprechenden Anzahl von Leerzeichen (20H).

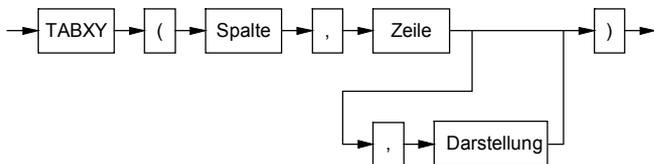
Parameter	Angabe	Bereich	Beschreibung
Position	num. Ausdr.	1 - 79	Position des Cursors

Beispiel: PRINT TAB(30), "X"
PRINT#2 TAB(71),A

Anmerkung:

Ist die momentane Cursor-Position größer oder gleich der angegebenen TAB-Position, wird die TAB-Funktion ignoriert.

14.2 TABXY [T.]



Funktion:

Diese Funktion wird in der PRINT/DISP-Anweisung verwendet um den Cursor an die angegebenen Koordinaten des Bildschirms/Displays zu verschieben und die Darstellungsart der Ausgabe auf den Bildschirm zu definieren.

Parameter	Angabe	Bereich	Beschreibung
Spalte	num. Ausdr.	1 - 80 (1-40)	Werte in Klammer gelten für die LCD-Anzeige
Zeile	num. Ausdr.	1 - 24 (1-4)	Werte in Klammer gelten für die LCD-Anzeige
Darstellung (bei LCD-Anzeige ohne Bedeutung)	num. Ausdr.	0 - 14	Darstellung der Ausgabe 0= normal 2= blinkend 4= invers 6= invers,blinkend 8= unterstrichen 10= unterstrichen, blinkend 12= unterstrichen, invers 14= unterstrichen, invers blinkend

Beispiel: PRINT TABXY(10,10,4),A
PRINT TABXY(10,15),B

Anmerkung:

Der Parameter "Darstellung" ist bei der Ausgabe an das Display ohne Funktion. Diese Anweisung kann nur in Verbindung mit einer PRINT-, oder DISP-Anweisung verwendet werden.

14.3 SPC



Funktion:

Diese Anweisung gibt eine bestimmte Anzahl von Leerzeichen (20H) aus.

Parameter	Angabe	Bereich	Beschreibung
Anzahl	num. Ausdr.	0 - 14	Anzahl der auszugehenden Leerzeichen

Beispiel: PRINT SPC(90)
PRINT% A, SPC(5),3

Anmerkung:

Diese Anweisung kann nur in Verbindung mit einer PRINT-, DISP-, oder OUTPUT-Anweisung verwendet werden.

14.4 CR



Funktion:

Diese Anweisung gibt ein Carriage Return (0DH) aus.

Beispiel: PRINT CR
PRINT% A,CR,X

Anmerkung:

Diese Anweisung kann nur in Verbindung mit einer PRINT-, DISP oder OUTPUT-Anweisung verwendet werden.

14.5 USING-Befehle

Allgemeines

Die Format-Befehle USING (##), USING (Fx) und USING (0) werden gespeichert und gelten für alle weiteren Ausgaben bis ein neuer USING-Befehl erfolgt. Mit dem Befehlssyntax USING (;##), USING (;Fx) und USING (;0) wird das aktuelle Format gespeichert und kann mit dem Befehl USING (*) wieder abgerufen werden.

Anwendung:

Wird eine Ausgaberroutine durch ein Interrupt unterbrochen, dann können Sie im Interruptprogramm ein anderes Ausgabeformat anwählen (mit USING (;...)) und vor dem Verlassen der Interruptroutine mit USING (*) das Ausgabeformat auf das ursprüngliche Format der Ausgaberroutine zurückstellen.

14.6 USING [U.] (##)



Funktion:

Diese Anweisung aktiviert das dezimale Zahlenformat; d.h. Zahlen werden dezimal dargestellt bzw. ausgegeben. Die Anzahl der '#' vor dem Dezimalpunkt gibt an, aus wievielen Stellen der ganzzahlige Teil der Zahl bestehen soll. Die Anzahl der '#' nach dem Dezimalpunkt gibt an, aus wievielen Stellen der gebrochene Teil der Zahl bestehen soll. Es können max. 8 '#' angegeben werden, mindestens ein '#' muß vor dem Dezimalpunkt stehen.

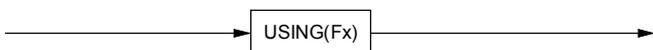
Beispiel: Die Variable A hat den Wert 12.3456

Anweisung	Darstellung
PRINT USING (##),A	12
(##.##),A	12.3
(##.#####),A	12.34560
(#.##),A	? 12.3456

Anmerkung:

USING(##) bleibt aktiv bis zur nächsten USING-Anweisung. Ist der ganzzahlige Teil der Zahl zu groß für das vorgegebene Zahlenformat gibt COMTAC-Basic ein '?' und anschließend die Zahl im freien Zahlenformat an. USING(##) kann nur in Verbindung mit einer PRINT-, DISP-, oder OUTPUT-Anweisung verwendet werden.

14.7 USING [U.] (Fx)



Funktion:

Diese Anweisung aktiviert das exponentielle Zahlenformat, d.h. Zahlen werden exponentiell dargestellt bzw. ausgegeben. Der Wert x gibt an, wieviele Stellen der Mantisse dargestellt werden. Ist x = 0 werden nachfolgende Nullen der Mantisse unterdrückt. COMTAC-Basic stellt immer mindestens drei Stellen der Mantisse dar, auch bei x=1 oder 2.

Parameter	Angabe	Bereich	Beschreibung
x	Zahl	0 - 8	Anzahl der Stellen der Mantisse

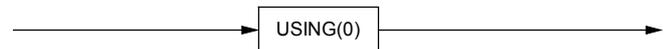
Beispiel: Die Variable hat den Wert 12345

Anweisung	Darstellung
PRINT USING (F0),A	1.2345 E+4
(F1)	1.23E+4
(F2)	1.23E+4
(F3)	1.23E+4
(F4)	1.234E+4
(F5)	1.2345E+4
(F6)	1.23450E+4
(F7)	1.234500E+4
(F8)	1.2345000E+4

Anmerkung:

USING(Fx) bleibt aktiv bis zur nächsten USING-Anweisung. USING(Fx) kann nur in Verbindung mit einer PRINT-, DISP-, DISP@- und OUTPUT-Anweisung verwendet werden.

14.8 USING [U.] (0)



Funktion:

Diese Anweisung aktiviert das "freie" Zahlenformat; d.h. COMTAC-Basic entscheidet in welchem Format Zahlen dargestellt bzw. ausgegeben werden. Ist die Zahl zwischen ± 9999 9999 und ± .1 wird das dezimale Format verwendet. Ist die Zahl außerhalb dieses Bereiches wird das exponentielle Format verwendet, bei dem führende und nachfolgende Nullen unterdrückt werden (USING(F0)).

Beispiel: dezimales Format: 12.345
exponentielles Format: 1.0 E+8

Anmerkung:

Nach POWER ON ist das "freie" Zahlenformat aktiviert. USING(0) bleibt aktiv bis zur nächsten USING-Anweisung. USING(0) kann nur in Verbindung mit einer PRINT-, DISP-, oder OUTPUT-Anweisung verwendet werden.

14.9 USING [U.] (B)



Funktion:

Diese Anweisung aktiviert zur Ausgabe der angegebenen Variable das interne binäre Zahlenformat. Auf weitere Ausgaben hat dieser Befehl keine Wirkung; das zuvor eingestellte Ausgabeformat bleibt erhalten. Die Variable wird Byte für Byte so ausgegeben, wie sie, je nach Datentyp, intern im COMTAC gespeichert sind. Folgende Tabelle zeigt in welcher Reihenfolge die Bytes der einzelnen Datentypen ausgegeben werden.

Datentyp	1.Byte	2.Byte	3.Byte	4.Byte	5.Byte	6.Byte
Floating Point	1/2. Ste.	3/4. Ste.	Mantisse 5/6. Ste. 7/8. Ste.		Vorzeich.	Exponent
DSP Fractional	Nachkomma LSB		MSB	Vorkomma LSB MSB		
long Integer	LSB			MSB		
Integer	LSB	MSB				

Zeichen	dez	hex	Bezeichnung
1	52	34h	Adresse (im ASCII-Format)
2	136	88h	CHR\$(88h)
3	65	41h	"A"
4	0	00h	POS(1) (Nachkommastelle LSB)
5	0	00h	POS(1) (Nachkommastelle)
6	0	00h	POS(1) (Nachkommastelle MSB)
7	232	e8h	POS(1) (Vorkommastelle LSB)
8	3	03h	POS(1) (Vorkommastelle)
9	0	00h	POS(1) (Vorkommastelle MSB)

Beispiel:

OUTPUT 1,4;CHR\$(88h),"A",USING(B),POS(1)

Hier wird über die RS485-Schnittstelle an das Gerät mit der Adresse 4 ein Binär-String in folgender Form übertragen: (Annahme: POS(1) ist vom Datentyp DSP-Fractional und hat den Wert 1000.)

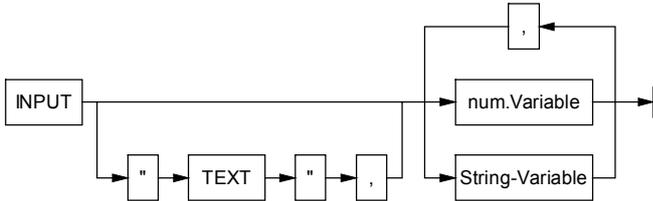
Anmerkung:

USING(B) kann nur in Verbindung mit einer PRINT-, DISP- oder OUTPUT-Anweisung verwendet werden.

USING(B) gilt nur für die eine angegebene Variable.

15. Terminal-Eingabe

15.1 INPUT [I.] (Tastatureingabe)



Funktion:
Diese Anweisung ermöglicht die Zuweisung eines Werts an eine Programm-Variable über die Tastatur.

Beispiel: INPUT "DICHTE=",D
INPUT \$(5)

Anmerkung:
Einer numerischen Variable kann nur eine Zahl zugewiesen werden.
Bei falscher Eingabe wird ein Beep gesendet und der Befehl wiederholt.

Sollen mit einer INPUT-Anweisung mehrere Werte eingelesen werden, dann müssen Sie die numerischen Variablen mit Komma und die Stringvariablen mit der Funktion "Carriage return" (↵) trennen.

Beispiel: INPUT a,b,\$(0),D
Programmablauf
?10,20,Hallo↵
?30

Achtung: Ein String wird mit ↵ (CR) abgeschlossen!

Funktion:

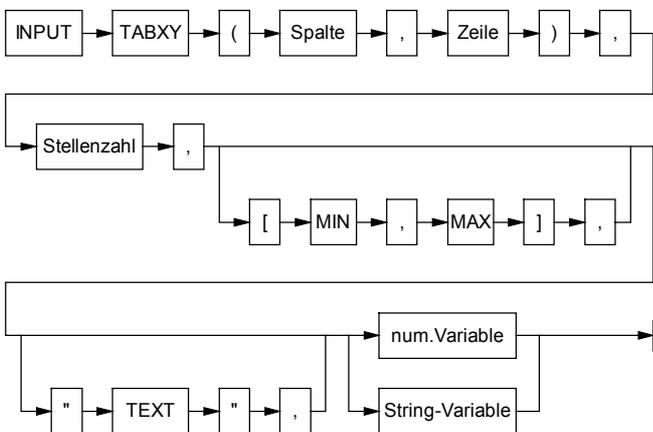
Diese Anweisung ist eine Erweiterung des INPUT-Befehls. Der aktuelle Wert der zu verändernten Programm-Variable wird an der durch 'TABXY' definierten Stelle des Terminals linksbündig angezeigt und kann nun überschrieben werden. Die Eingabe wird begrenzt durch die Stellenzahl. Nach der Quittierung der Eingabe, durch 'CR' wird der Wert auf seine Gültigkeit überprüft (Einschaltung der MIN-, MAX-Angabe), sofern diese definiert ist. Ist die Eingabe korrekt, wird der neue Wert angezeigt und die Funktion beendet. Ist die Eingabe falsch, wird der alte Wert angezeigt und die Funktion wiederholt.

Parameter	Angabe	Bereich	Beschreibung
Spalte	num. Ausdr.	1 - 80	X-Position
Zeile	num. Ausdr.	1 - 24	Y-Position
Stellenzahl	num. Ausdr.	1 - 80	Begrenzung der Eingabe
MIN	num. Ausdr.		minimaler Wert der Eingabe
MAX	num. Ausdr.		maximaler Wert der Eingabe

Beispiel: INPUT TABXY (X,Y),5,[100.M],"Vorschub = ",V
INPUT TABXY (10,10),S,[0.4,0.97],"Tiefe = ",T

Anmerkung:
Bei der Angabe der Stellenzahl muß das Vorzeichen und der Dezimalpunkt mit berücksichtigt werden.

15.2 INPUT [I.] TABXY [T.]



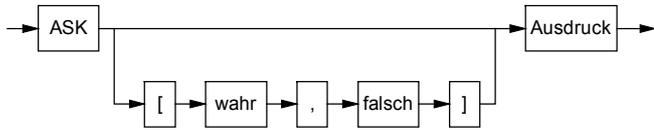
15.3 GET



Funktion:
Diese Anweisung ermöglicht die Abfrage der Terminal-Tastatur.
Ist eine Taste gedrückt, wird der entsprechende ASCII-Code dieser Taste übergeben.
Ist keine Taste gedrückt wird der Wert 0 übergeben.

Beispiel: T = GET
10 IF GET = 0 THEN GOTO 10

15.4 ASK



Funktion:

Diese Anweisung sendet den Ausdruck nach ASK zum Terminal und fragt anschließend die Tastatur solange ab, bis die Taste "JA" = Y/y oder die Taste "NEIN" = N/n gedrückt wird.

Es wird die Information wahr = 65535 übergeben, wenn die Taste y,Y gedrückt wird.

Es wird die Information falsch = 0 übergeben, wenn die Taste n,N gedrückt wird.

Jede andere Taste erzeugt ein Beep.

Sollen andere Tasten als Y und N für "Ja" und "Nein" gelten, dann muß dies explizit in eckigen Klammern angegeben werden.

Parameter	Angabe	Bereich	Beschreibung
wahr	num. Ausdr.	0 - 255	ASCII-Code der entsprechenden Taste
falsch	num. Ausdr.	0 - 255	ASCII-Code der entsprechenden Taste

Beispiel:

IF ASK TABXY(10,5), "Wollen Sie den Wert ändern?(Y/N)" THEN...
 X = ASK [49,50]"Wählen Sie Achse 1 oder 2"
 (49/50 ist der ASCII-Code von 1 und 2)

Anmerkung:

Der Ausdruck nach ASK bzw. ASK[w,f] wird wie eine PRINT-Anweisung behandelt.

Achtung!

Die ASK-Anweisung kann durch einen Interrupt abgebrochen werden.

In diesem Fall erhalten Sie den Wert 1 als Abfrageergebnis, welcher ebenfalls als wahr interpretiert wird.

Um dennoch einen Abfragewert zu erhalten können Sie in die nächste Zeile das Ergebnis auf 1 abfragen, um dann zur Eingabeanweisung zurückzuspringen.

Beispiel:

100 A = ASK [49,50]"Wählen Sie Achse 1 oder 2"
 110 If a=1 Then 100

15.5 Anwendung der Funktionstasten

Das COMTAC ermöglicht die Bedienung von max. 18 Funktionstasten.

Allgemein gelten folgende Regeln:

◆ Jeder Funktionstaste wird ein separater Programmabschnitt bzw. ein Unterprogramm zugeordnet.

Beispiel: ONKEY 1 GOTO 500
 ONKEY 12 GOSUB 1000

◆ Ab der spezifizierten Zeilennummer hat der Anwender eine Programm- Routine zu erstellen, die der gewünschten Tastenfunktion entspricht.

◆ Mit den Anweisungen ENABLE ONKEY x bzw. DISABLE ONKEY x kann jederzeit eine Tastenroutine freigegeben bzw. gesperrt werden.

◆ Das Betätigen einer freigegebenen Funktionstaste löst unmittelbar eine Interruptmeldung aus. Diese Interruptmeldung wird bis zur Abarbeitung einer laufenden BASIC-Zeile gespeichert. Danach erfolgt automatisch eine Verzweigung in die spezifizierte Funktionsroutine.

Beispiel:

50 ONKEY 1 GOTO 500:ENABLE ONKEY 1
 60 ONKEY 12 GOSUB 1000:ENABLE ONKEY 12

```
500  DISABLE ONKEY.1
510
.
.
.
580  ENABLE ONKEY 1:GOTO xxx
```

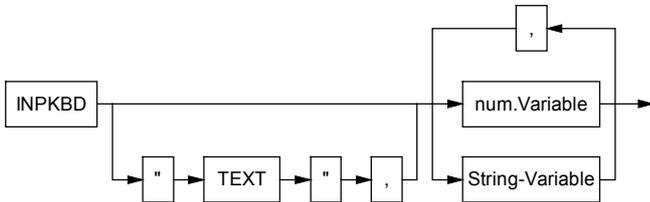
Routine für die Funktionstaste F1

```
1000 DISABLE ONKEY 12
1010
.
.
.
1090 ENABLE ONKEY 12:RETURN xxx
```

Unterprogramm für die Funktionstaste F12

16. Folientastatur-Eingabe

16.1 INPKBD



Funktion:
Diese Anweisung ermöglicht die Zuweisung eines Werts an eine Programm-Variable über die COMTAC-Folientastatur.

Beispiel: INPKBD "Position =",POS
INPKBD \$(32)

Anmerkung:
Einer numerischen Variablen kann nur eine Zahl zugewiesen werden. Bei falscher Eingabe wird der Befehl wiederholt.

Sollen mit einer INPUT-Anweisung mehrere Werte eingelesen werden, dann müssen Sie die numerischen Variablen mit Komma und die Stringvariablen mit der Funktion "Ent" trennen.

Beispiel: INPUT a,b,\$(0),D
Programmablauf
?10,20,Hallo "Ent"
?30

Achtung: Ein String wird mit "Ent" abgeschlossen!

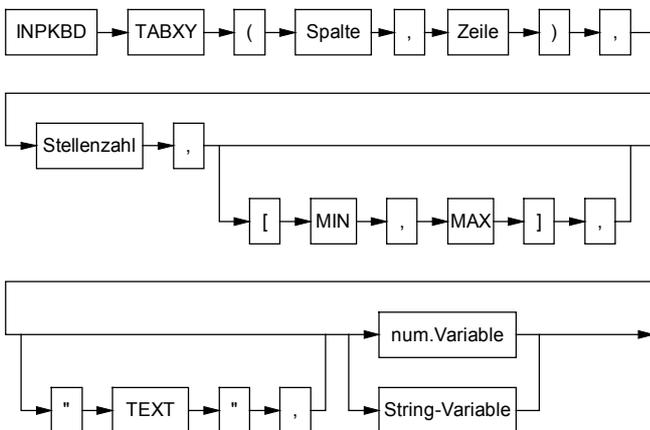
Der aktuelle Wert der zu verändernten Programm-Variable wird an der durch 'TABXY' definierten Stelle des Displays linksbündig angezeigt und kann nun überschrieben werden. Die Eingabe wird begrenzt durch die Stellenzahl. Nach der Quittierung der Eingabe, durch 'CR' wird der Wert auf seine Gültigkeit überprüft (Einhaltung der MIN-, MAX-Angabe), sofern diese definiert ist. Ist die Eingabe korrekt, wird der neue Wert angezeigt und die Funktion beendet. Ist die Eingabe falsch, wird der alte Wert angezeigt und die Funktion wiederholt.

Parameter	Angabe	Bereich	Beschreibung
Spalte	num. Ausdr.	1 - 40	X-Position
Zeile	num. Ausdr.	1 - 4	Y-Position
Stellenzahl	num. Ausdr.	1 - 80	Begrenzung der Eingabe
MIN	num. Ausdr.		minimaler Wert der Eingabe
MAX	num. Ausdr.		maximaler Wert der Eingabe

Beispiel: INPKBD TABXY 1 (10,3),[0,5], "OFFSET=",A

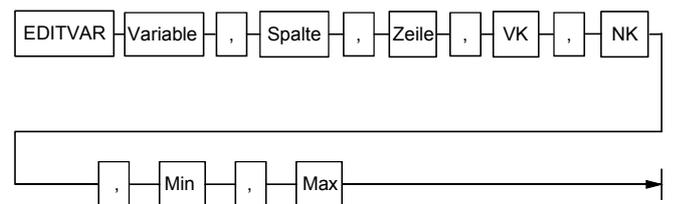
Anmerkung:
Bei der Angabe der Stellenzahl muß das Vorzeichen und der Dezimalpunkt mit berücksichtigt werden.

16.2 INPKBD TABXY [T.]



Funktion:
Diese Anweisung ist eine Erweiterung des INPKBD-Befehls.

16.3 EDITVAR



Funktion:
Folientastatureingabe ohne Programmunterbrechung. Der Befehl wird dauernd ausgeführt, während das Basicprogramm im Hintergrund läuft. Der aktuelle Wert der zu verändernten Programm-Variable wird an der definierten Stelle des Displays linksbündig angezeigt und kann nun überschrieben werden. Nach der Quittierung der Eingabe, durch 'Ent' wird der Wert auf seine Gültigkeit überprüft (Einhaltung der MIN-, MAX-Angabe). Ist die Eingabe korrekt, wird der neue Wert angezeigt. Die Funktion wird mit "Esc" beendet. Ist die Eingabe falsch, wird der alte Wert angezeigt und die Funktion wiederholt.

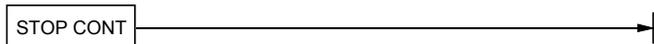
Parameter	Angabe	Bereich	Beschreibung
Spalte	num. Ausdr.	1 - 40	X-Position
Zeile	num. Ausdr.	1 - 4	Y-Position
(VK)		0 - 8	Vorkommastelle
(NK)		0 - 8	Nachkommastelle
Bedingung: VK + NK ≤ 8			
Min	num. Ausdr.		minimaler Wert der Eingabe
Max	num. Ausdr.		maximaler Wert der Eingabe

Beispiel: EDITVAR A, 1, 2, 2, 3, 0, 100

Anmerkung:

- ◆ Sie können maximal 9 Werte gleichzeitig am LCD-Display anzeigen; dabei ist die Anzeige von zyklischen Werten über den Befehl DISPCPXZ und DISPVAR eingeschlossen.
- ◆ Sind mehrere Werte auf dem LCD-Display, dann können die einzelnen Werte mit den Pfeiltasten (↑,↓) angesprungen werden.

16.3.1 STOP DISP



Der Display - Inhalt wird gespeichert. Der Cursor wird ausgeschaltet.

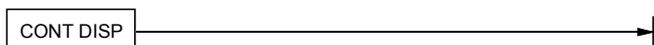
Der Display - Inhalt wird mit CONT DISP wieder zur Anzeige gebracht.

Hinweis

Erfolgt nach STOP DISP ein Display-Befehl (EDITVAR; DISPVAR oder DISPCPXZ) dann wird die vorhergehende Anzeige vollständig gelöscht.

Die neuen Display - Befehle werden erst nach CONT DISP aktiv.

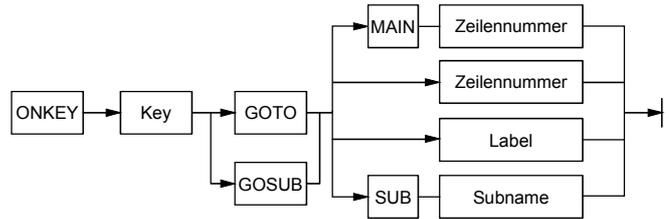
16.3.2 CONT DISP



Nach STOP DISP wird der Display - Inhalt mit CONT DISP wieder zur Anzeige gebracht.

Vor "CONT DISP" muß ein "STOP DISP" ausgeführt werden.

16.4 ONKEY [OK.]



Funktion:

Diese Anweisung definiert eine entsprechende Programmverzweigung, die durchgeführt wird, wenn die entsprechende Funktionstaste betätigt wird und der ONKEY-Interrupt freigegeben ist.

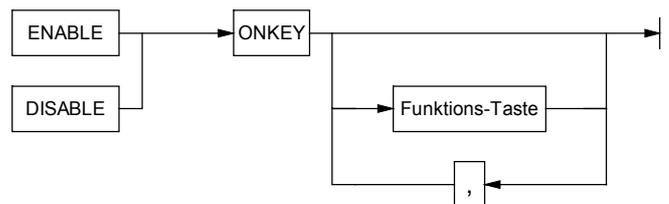
Die COMTAC-Folientastatur hat zwei zusätzliche Funktionstasten:

- ◆ START-Taste = Funktionstaste 17
- ◆ STOP-Taste = Funktionstaste 18.

Parameter	Angabe	Bereich	Beschreibung
Key	num. Ausdr.	1 - 18	Funktionstasten-Nummer
Zeilen-Nr.	Zahl	0-65535	vorhandene Zeilen-Nummer im Programm

Beispiel: ONKEY 3 GOTO 25500
ONKEY 17 GOSUB 8880

16.5 ENABLE [EN./]DISABLE [DI.] ONKEY [OK.]



Funktion:

Freigabe oder Sperren des ONKEY-Interrupts.

Die COMTAC-Folientastatur hat zwei zusätzliche Funktionstasten:

- ◆ START-Taste = Funktionstaste 17
- ◆ STOP-Taste = Funktionstaste 18.

Parameter	Angabe	Bereich	Beschreibung
Key	num. Ausdr.	1 - 18	Funktionstasten-Nummer

Anmerkung:

Wird keine Funktionstaste angegeben, bezieht sich die Anweisung auf alle Funktionstasten die bereits mit der Anweisung ONKEY definiert sind.

16.6 KBDCODE



Funktion:

Diese Anweisung ermöglicht die Abfrage der COMTAC-Foli-
en-Tastatur.

Ist eine Taste gedrückt, wird der entsprechende ASCII-Code
dieser Taste übergeben.

Ist keine Taste gedrückt wird der Wert 0 übergeben.

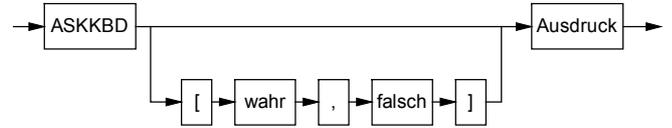
Beispiel: DISP KBDCODE

10 IF KBDCODE = 0 THEN GOTO 10

Tastencode, wird der Variablen KBDCODE zugewiesen.

Taste	Dezimal-Wert	Hex-Wert
Funktionstaste 1	176dez	0b0h
Funktionstaste 2	177dez	0b1h
Funktionstaste 3	178dez	0b2h
Funktionstaste 4	179dez	0b3h
Funktionstaste 5	180dez	0b4h
Funktionstaste 6	181dez	0b5h
Funktionstaste 7	182dez	0b6h
Funktionstaste 8	183dez	0b7h
Funktionstaste 9	184dez	0b8h
Funktionstaste 10	185dez	0b9h
Funktionstaste 11	186dez	0bah
Funktionstaste 12	187dez	0bbh
Funktionstaste 13	188dez	0bch
Funktionstaste 14	189dez	0bdh
Funktionstaste 15	190dez	0beh
Funktionstaste 16	191dez	0bfh
Funktionstaste START	192dez	0c0h
Funktionstaste STOP	193dez	0c1h
Taste ESC (Escape)	026dez	01ah
cursor left	008dez	008h
cursor right	012dez	00ch
cursor up	011dez	00bh
cursor down	022dez	016h
Taste ENT (enter)	013dez	00dh
Taste INS (Einfügen)	002dez	002h
Taste DEL (löschen)	127dez	07fh
Taste 0	048	030h
Taste 1	049	031h
Taste 2	050	032h
Taste 3	051	033h
Taste 4	052	034h
Taste 5	053	035h
Taste 6	054	036h
Taste 7	055	037h
Taste 8	056	038h
Taste 9	057	039h
Taste dezimalpunkt	046	02eh
Taste + / -	045	02dh

16.7 ASKKBD



Funktion:

Diese Anweisung sendet den Ausdruck nach ASKKBD zum
Display und fragt anschließend die COMTAC-Tastatur so-
lange ab, bis die Taste "JA" = Y/y oder die Taste "NEIN" =
N/n gedrückt wird.

Es wird die Information wahr = 65535 übergeben, wenn die
Taste y, Y gedrückt wird.

Es wird die Information falsch = 0 übergeben, wenn die Ta-
ste n, N gedrückt wird.

Jede andere Taste erzeugt ein Beep.

Sollen andere Tasten als Y und N für "Ja" und "Nein" gel-
ten, dann muß dies explizit in eckigen Klammern angegeben
werden.

Parameter	Angabe	Bereich	Beschreibung
<i>wahr</i>	num. Ausdr.	0 - 255	ASCII-Code der ent- sprechenden Taste
<i>falsch</i>	num. Ausdr.	0 - 255	ASCII-Code der ent- sprechenden Taste

Beispiel:

IF ASKKBD ["0","1"] "Ist die Eingabe korrekt?(0/1)" THEN...

Anmerkung:

Der Ausdruck nach ASKKBD bzw. ASKKBD[w,f] wird wie ei-
ne DISP-Anweisung behandelt.

Achtung!

Die ASKKBD-Anweisung kann durch einen Interrupt abge-
brochen werden.

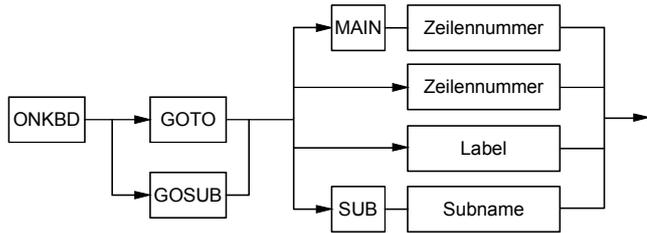
In diesem Fall erhalten Sie den Wert 1 als Abfrageergebnis,
welcher ebenfalls als wahr interpretiert wird.

Um dennoch einen Abfragewert zu erhalten können Sie in
die nächste Zeile das Ergebnis auf 1 abfragen, um dann zur
Eingabeanweisung zurückzuspringen.

Beispiel:

100 A = ASKKBD [49,50]"Wählen Sie Achse 1 oder 2"
110 If a=1 Then 100

16.8 ONKBD



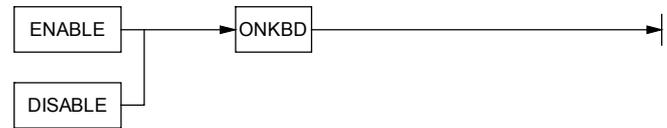
Funktion:

Diese Anweisung definiert eine entsprechende Programmverzweigung die durchgeführt wird, wenn eine Taste der COMTAC-Folien-Tastatur betätigt oder losgelassen wird. Dazu muß der ONKBD-Interrupt freigegeben sein. Die Systemvariable KBDCODE liefert den Tastencode der gedrückten Taste.

Parameter	Angabe	Bereich	Beschreibung
Zeilen-Nr.	Zahl	0-65535	vorhandene Zeilen-Nummer im Programm

Beispiel: ONKBD GOTO 1000

16.9 ENABLE [EN.]/DISABLE [DI.] ONKBD



Funktion:

Freigabe/Sperren des ONKBD-Interrupt

Beispiel: ENABLE ONKBD
DISABLE ONKBD

Anmerkung:

Es ist zu beachten, daß das Loslassen einer Taste auch einen Interrupt auslöst (KBDCODE = 0).

17. Echtzeituhr und Timer

17.1 TIME



Funktion:
Diese Anweisung liefert den formatierten Wert der aktuellen Uhrzeit im Format: HH:MM:SS

Beispiel: PRINT TIME
DISP TIME

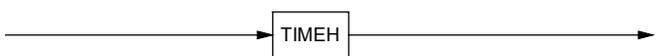
Anmerkung:
Time kann nur in Verbindung mit einem PRINT-, DISP- oder OUTPUT-Befehl verwendet werden.

Ausnahmen:
◆ Zuweisung der Uhrzeit in eine Stringvariable z.B. \$(4) = TIME.
◆ Uhrzeit einstellen: TIME = H,M [S].
Es gilt:

Parameter	Angabe	Bereich	Beschreibung
H	num. Ausdruck	0 - 23	Stunden
M	num. Ausdruck	0 - 59	Minuten
S	num. Ausdruck	0 - 59	Sekunden

Wird kein Wert für Sekunden angegeben, wird dafür der Wert 0 eingesetzt.

17.2 TIMEH



Funktion:
Diese spezielle Systemvariable ermöglicht das Auslesen und/oder Setzen der Stunden der aktuellen Uhrzeit.

Beispiel: TIMEH = 10
IF TIMEH = 12 THEN

Anmerkung:
Wird TIMEH ein Wert zugewiesen muß dieser im Bereich von 0 - 23 liegen.

17.3 TIMEM



Funktion:
Diese spezielle Systemvariable ermöglicht das Auslesen und/oder Setzen der Minuten der aktuellen Uhrzeit.

Beispiel: TIMEM = 33
IF TIMEM = 0 THEN

Anmerkung:
Wird TIMEM ein Wert zugewiesen muß dieser im Bereich von 0 - 59 liegen.

17.4 TIMES

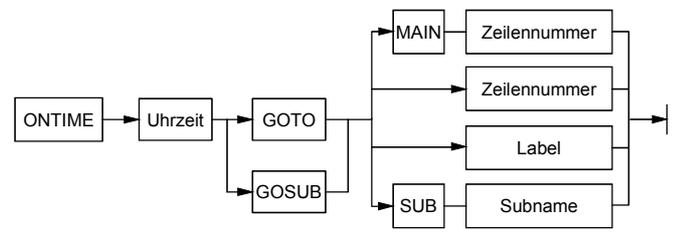


Funktion:
Diese spezielle Systemvariable ermöglicht das Auslesen und/oder Setzen der Sekunden der aktuellen Uhrzeit.

Beispiel: TIMES = 0
IF TIMES = 15 THEN

Anmerkung:
Wird TIMES ein Wert zugewiesen muß dieser im Bereich von 0 - 59 liegen.

17.5 ONTIME

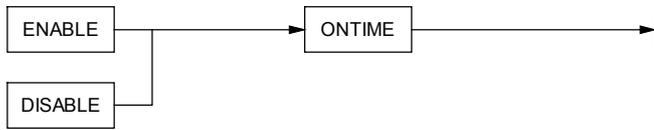


Funktion:
Diese Anweisung definiert eine Programmverzweigung, die durchgeführt wird, wenn die Echtzeit-Uhr die angegebene Uhrzeit erreicht und der ONTIME-Interrupt freigegeben ist. (Siehe dazu ENABLE/DISABLE ONTIME)

Beispiel: ONTIME 12,30,0 GOTO
ONTIME H,M GOSUB 3300

Anmerkung:
Die Uhrzeit muß wie folgt angegeben werden: Stunden, Minuten [,Sekunden] (siehe auch TIME)

17.6 ENABLE [EN.]/DISABLE [DI.] ONTIME



Funktion:

Freigabe / Sperren des Uhrzeit Interrupt.

Beispiel: ENABLE ONTIME / DISABLE ONTIME

17.7 DATE



Funktion:

Diese Anweisung liefert den formatierten Wert des aktuellen Datums im amerikanischen Format: Wochentag MM-DD-XXAA oder im deutschen Format: Wochentag DD.MM.XXAA.

XX = 19 für 1980 - 1999

XX = 20 für 2000 - 2079

Mit dem Parameter 13 kann das entsprechende Format ausgewählt werden.

Beispiel: PRINT DATE
OUTPUT 32; \$(0), DATE

Anmerkung:

Date kann nur in Verbindung mit einem PRINT-, DISP- oder OUTPUT-Befehl verwendet werden.

Ausnahmen:

- ◆ Zuweisung der Datums in eine Stringvariable z. B. \$(7)=DATE

- ◆ Datum einstellen: DATE = W, M, D, A
es gilt:

Parameter	Angabe	Bereich	Beschreibung
W	num. Ausdruck	1-7	Wochentag
M	num. Ausdruck	1-12	Monat
D	num. Ausdruck	1-31	Tag
A	num. Ausdruck	0-99	Jahr

17.8 DATEW



Funktion:

Diese spezielle Systemvariable ermöglicht das Auslesen und/oder Setzen des Wochentags des aktuellen Datums.

Beispiel: DATEW = 1

IF DATEW = 5 THEN

Anmerkung:

Wird DATEW ein Wert zugewiesen muß dieser im Bereich von 1 - 7 liegen.

Es gilt:

Sonntag= 1 Montag= 2 Dienstag= 3 Mittwoch= 4
Donnerstag= 5 Freitag= 6 Samstag= 7

17.9 DATED



Funktion:

Diese spezielle Systemvariable ermöglicht das Auslesen und/oder Setzen des Tages des aktuellen Datums.

Beispiel: DATED = 25
IF DATED = 4 THEN

Anmerkung:

Wird DATED ein Wert zugewiesen muß dieser im Bereich von 1 - 31 liegen.

17.10 DATEM



Funktion:

Diese spezielle Systemvariable ermöglicht das Auslesen und/oder Setzen des Monats des aktuellen Datums.

Beispiel: DATEM = 7
IF DATEM = 12 THEN

Anmerkung:

Wird DATEM ein Wert zugewiesen muß dieser im Bereich von 1 - 12 liegen.

17.11 DATEY



Funktion:

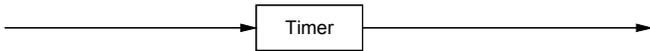
Diese spezielle Systemvariable ermöglicht das Auslesen und/oder Setzen der Jahreszahl des aktuellen Datums.

Beispiel: DATEY = 97
IF DATEY = 99 THEN

Anmerkung:

Wird DATEY ein Wert zugewiesen muß dieser im Bereich von 0 - 99 liegen.

17.12 TIMER



Funktion:

Diese spezielle Funktionsvariable ermöglicht das Auslesen und / oder Setzen des internen Basic-Timers.

Beispiel: PRINT TIMER
X = TIMER

Anmerkung:

Ist der interne Basic-Timer freigegeben, wird TIMER alle 5 Millisekunden inkrementiert bis zum Wert 65 535,995s dann beginnt TIMER wieder bei 0.

Funktion:

Diese Anweisung definiert einen Unterprogramm-Aufruf, der durchgeführt wird, wenn der interne Basic-Timer den Vorgabewert überschreitet.

Dazu muß TIMER freigegeben werden (siehe ENABLE/DISABLE ONTIMER).

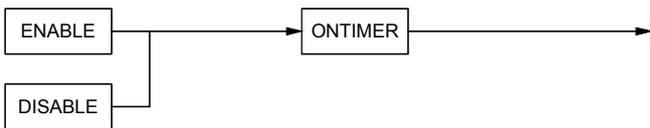
Beispiel: ONTIMER 1 , 200
ONTIMER TIMER+2,1000

Anmerkung:

Ist der Basic-Timer freigegeben, wird dieser alle 5 ms inkrementiert. Ein Interrupt wird erzeugt, wenn der Basic-Timer den Wert des Timer-Compare-Register (TCR) erreicht bzw. überschritten hat. Die kleinste Einheit des TCR ist 1 Sekunde. Im Normalfall wird der Timer Interrupt nur einmalig erzeugt und muß dann wieder neu initialisiert werden. Soll der Timer Interrupt in einem festen Zeitraster erzeugt werden ist dies durch das Aktivieren der Auto-Reload-Funktion möglich. Um diese Funktion zu aktivieren, setzen Sie ein Semikolon anstatt einem Komma. Bei diesem Interrupt ist die Verzweigung mit GOTO nicht möglich. Mit der Initialisierung wird gleichzeitig die Überwachung freigegeben.

ONTIMER kann mit dem Befehl OFFTIMER gelöscht werden.

17.13 ENABLE [EN.]/DISABLE [DI.] ONTIMER [OC.]

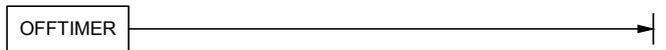


Funktion:

Freigabe/Sperren des internen Basic-Timers.

Beispiel: ENABLE ONTIMER
DISABLE ONTIMER

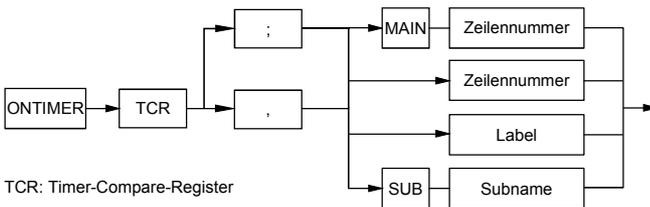
17.15 OFFTIMER



Funktion:

Diese Anweisung löscht den Befehl ONTIMER, sperrt jedoch nicht den Basic - TIMER.

17.14 ONTIMER [OC.]



TCR: Timer-Compare-Register

18. Verarbeitung von Zeichenketten

18.1 Zeichenketten oder "Strings"

Für die Verarbeitung von nicht-numerischen Informationen werden Zeichenketten- oder String-Variable verwendet. COMTAC-Basic benutzt eindimensionale String-Variable, die mit dem Dollar-Zeichen und einem in runden Klammern stehenden Index benannt werden ($\$(\text{Index})$).

Der Index, dessen max. Wert 255 sein darf, kann durch eine Zahl, Variable oder numerischen Ausdruck dargestellt werden.

z. B. $\$(7)$; $\$(A)$; $\$(A-8/4)$

COMTAC-Basic erlaubt für Zeichenketten eine dimensionierte Länge zwischen 1 und 255, während die aktuelle Länge zwischen Null und der dimensionierten Länge liegen darf. Unter der aktuellen Länge einer Zeichenkette versteht man die Anzahl der in ihr gespeicherten Zeichen.

18.2 Speicherung von Zeichenketten

Zeichenketten mit mehr als 16 Zeichen (Standardlänge) müssen vor ihrer Verwendung dimensioniert werden. Z. B.:

DIM\$(5) [100]

reserviert Speicherplatz für einen 100-Zeichen String.

	1	2	3	4	5	6	7	8	9	...	99	100
$\$(5)$												

Die größte zulässige Länge ist 255. Die Zeichenkette kann auch mit weniger als der Standardlänge 16 dimensioniert werden.

18.2.1 Verkettung von Zeichenketten

Im COMTAC-Basic werden Strings mit einem Komma (,) verbunden. Im folgenden Programm werden zwei Strings zu einem String verbunden:

```
10 $(0) = "COMPUTER"
20 $(1) = "SPIELE"
30 $(2) = $(0), $(1)
40 PRINT $(0)
50 PRINT $(1)
60 PRINT $(2)
70 END
```

Programmausgabe:

```
COMPUTER
SPIELE
COMPUTERSPIELE
```

Die Verkettungsoperation in Zeile 30 hängt den zweiten String an das Ende des ersten und weist das Resultat einem dritten String zu.

18.3 Teilzeichenketten

Unter einer Teilzeichenkette versteht man alle Zeichen oder lediglich ein Teil der betrachteten Zeichenkette. Solche Teilzeichenketten werden durch ein, zwei oder drei Indizes in eckigen Klammern bezeichnet, die als Konstante, numerische Ausdrücke oder Variablen in Erscheinung treten können.

Beispiele:

\$(0)[5] bezeichnet einen Teilstring, der mit dem 5. Zeichen des Strings $\$(0)$ beginnt und bis zu dessen Ende reicht.

\$(15)[3,6] bezeichnet einen Teilstring, der mit dem 3. Zeichen des Strings $\$(15)$ beginnt und beim 6. Zeichen endet und 4 Zeichen lang ist.

\$(7)[9,15,13] bezeichnet einen Teilstring, der mit dem 9. Zeichen des Strings $\$(7)$ beginnt und entweder beim 15. Zeichen oder vor dem angegebenen Ende-Zeichen (13) endet.

Dabei gibt der erste Index das erste Zeichen und der zweite Index das letzte Zeichen des Teilstrings an. Der dritte Index ist ein frei wählbares String Endezeichen, durch das der Teilstring ebenfalls begrenzt werden kann.

Die allgemeine Form lautet:

\$(x)[START, ENDE, ENDEZEICHEN] oder

\$(x)[START, ENDEZEICHEN] oder

\$(x)[START]

Teilzeichenketten können auf beiden Seiten von Zuweisungen stehen. z.B.

\$(0) [8,15] = \$(93) [5,9]

Das COMTAC-Basic generiert die Fehlermeldung "**Bad Argument**" wenn:

- ◆ Start- oder Ende-Index kleiner 1 ist.
- ◆ Der Start-Index auf der linken Seite der Zuweisung größer als die aktuelle Länge des Strings +1 ist.
- ◆ Der Start-Index auf der rechten Seite der Zuweisung größer als die aktuelle Länge des Strings ist.
- ◆ Der Start- oder Stop-Index größer als die max. Länge des Strings ist.

18.4 Vordefinierte Zeichenketten

Das COMTAC-Basic kennt die vordefinierten Strings

\$(#0)	RS232/1- Zwischenpuffer
\$(#1)	RS485/1- Zwischenpuffer
\$(#2)	RS232/2- Zwischenpuffer
\$(#3)	RS485/2- Zwischenpuffer
\$(#4)	RS232/3- Zwischenpuffer
UMEM\$	Zugriff auf das DATA-Text-Feld

die sich in ihrer Verwendung und Handhabung von den allgemein gültigen Zeichenketten unterscheiden. Die vordefinierten Zeichenketten sind bereits dimensioniert.

\$(#Schnittstellennummer)

Diese Strings werden verwendet um die über die entsprechende Schnittstelle empfangenen Zeichen aus dem Zwischenpuffer für die weitere Verarbeitung einzulesen; d.h. nach der Ausführung des Befehls "ENTER Schnittstellennummer" steht die eingelesene Information im String \$(# Schnittstellennummer). Diese Zeichenketten haben eine max. Länge von 255 Zeichen und können sonst wie die allgemeingültigen Zeichenketten behandelt werden. Die vordefinierten Zeichenketten sind bereits dimensioniert.

Beispiel: ENTER 0,2
DISP \$(#0)

UMEM\$

wird verwendet um Zeichenketten zeilenorientiert oder absolut adressiert aus dem Data-Text-Feld des COMTAC auszulesen bzw. einzuschreiben.

1. Zeilen orientiert

Jede Zeile im Data-Text-Feld ist ein String mit der Bezeichnung UMEM\$(Zeilen-Nr.). Die max. Länge dieser Zeichenketten ist auch immer die aktuelle Länge und wird durch die Zeilenlänge im Data-Text-Feld bestimmt.

Diese Strings können wie die allgemeingültigen Zeichenketten behandelt werden.

2. absolut adressiert

Der Data-Text-Editor besteht aus beliebig vielen Strings mit der Bezeichnung UMEM\$ (Adresse,Länge). Die Adresse hat Gültigkeit im Bereich von 0 - 4095 und die Länge im Bereich von 0 - 255. Diese Strings können ebenfalls wie allgemeingültige Strings behandelt werden, dürfen aber keine Teilzeichenkette-Ausgabe haben.

Beschreibung siehe Seite 90.

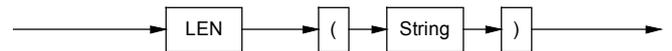
18.5 Vergleich von Zeichenketten

Im COMTAC-Basic sind für Stringausdrücke die beiden vergleichenden Operationen "=" und "<>" zulässig. Sie können in einer IF-Abfrage oder in einer DO-WHILE/DO-UNTIL-Schleife verwendet werden.

```

Beispiel: 1.) IF $(0) = "COMTAC 8000" THEN ...
           2.) DO
           ...
           UNTIL $(10) <> UMEM $(7)[3,8]
           3.) DO
           ...
           WHILE $(#1)[5,5] = "1"
    
```

18.6 LEN\$(x)



Funktion:

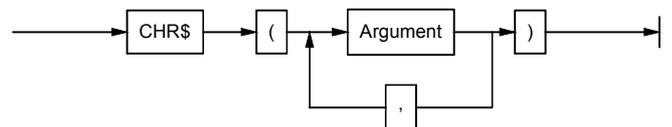
Diese Anweisung liefert die aktuelle Länge eines Strings.

Parameter	Angabe	Bereich	Beschreibung
<i>String</i>	gültige Zeichenkette		String, dessen Länge bestimmt werden soll

Beispiel 1: A = LEN\$(7)

Beispiel 2: PRINT LEN\$(#2)[5,,13])

18.7 CHR\$



Funktion:

Diese Anweisung wandelt einen oder mehrere numerische Werte in ASCII-Zeichen um.

Parameter	Angabe	Bereich	Beschreibung
<i>Argument</i>	num. Ausdr.	0 - 65535	umzuwandelnder Wert

Beispiel 1: \$(B)[4,4] = CHR\$(97)

Beispiel 2: OUTPUT 1,3; CHR\$(13)

Anmerkung:

Verwendet wird das niederwertige Byte des Arguments, das höherwertige Byte wird ignoriert.

18.8 ASC



Funktion:

Diese Anweisung liefert den dezimalen Wert aus dem ASCII-Code für das erste Zeichen des angegebenen String.

Parameter	Angabe	Bereich	Beschreibung
<i>String</i>	gültige Zeichenkette		umzuwandelndes Zeichen

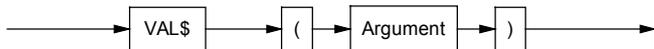
Beispiel 1: V = ASC\$(#2)

Beispiel 2: IF ASC\$(3)[4] <> 17 THEN...

Anmerkung:

Der angegebene String darf kein Leerstring sein.

18.9 VAL\$



Funktion:

Diese Anweisung wandelt den Wert des Arguments in einen String um.

Parameter	Angabe	Bereich	Beschreibung
<i>Argument</i>	num. Ausdr.		umzuwandelnder Wert

Beispiel 1: PRINT VAL\$(A)

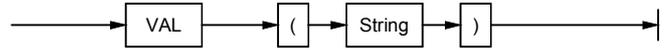
Beispiel 2: UMEM\$(8) = VAL\$(X)

Anmerkung:

Der erhaltene String hat das aktuelle PRINT-Format.

Ausnahme: Bei einer positiven Zahl ist das erste Zeichen kein Leerzeichen.
Mit der Anweisung PRINT USING(..) kann das Format geändert werden.

18.10 VAL



Funktion:

Diese Anweisung wandelt einen String in seinen numerischen Wert um.

Parameter	Angabe	Bereich	Beschreibung
<i>String</i>	gültige Zeichenkette		umzuwandelnder String

Beispiel: Q = VAL\$(13)

IF VAL\$(B)[8] = 137 THEN ...

Anmerkung:

Das erste Zeichen des Strings, das kein Leerzeichen ist, muß eine Ziffer, ein Plus- oder Minus-Zeichen oder ein Dezimalpunkt sein.

Der String wird bis zum ersten nicht-numerischen Zeichen bzw. bis zum Stringende umgewandelt.

19. Digitale Ein/Ausgänge

Digitale Ein/Ausgänge stehen bei COMTAC 2000 jeweils 16 zur Verfügung, bei COMTAC 3000 jeweils 32. Bei Befehlen die alle 32 Ein/Ausgänge von COMTAC 3000 unterstützen ist dies bei der Angabe des Bereichs in Klammern angegeben.

Bei einem COMPAX - Mehrachsverbund der über die Feldbuschnittstelle (siehe ab Seite 95) aufgebaut ist, können die COMPAX Ein/Ausgänge von bis zu 14 COMPAX in den Befehlen IN(x), BCDIN(x), BINOUT (x) = y, BCDOUT(x) = 0, CLROUT x, SETOUT x und REQOUT x,y,t verwendet werden.

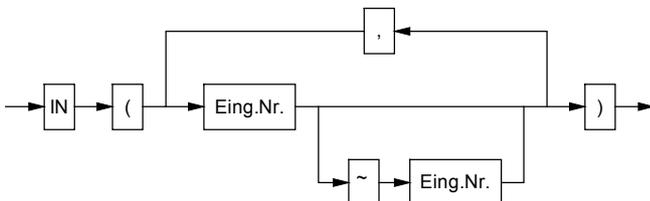
19.1 Feldbus E/A-Belegung

COMTAC- und COMPAX-Ein-/Ausgänge können linear adressiert werden. Maximal sind 256 Ein-/Ausgänge adressierbar. Die Zuordnung wird wie folgt festgelegt:

E/A-Nr	Gerät
01 ... 16	COMTAC Standard E/A's
17 ... 32	COMTAC optionelle E/A's
33 ... 48	COMPAX mit Feldbus-Adresse 1
49 ... 64	COMPAX mit Feldbus Adresse 2
65 ... 80	COMPAX mit Feldbus-Adresse 3
81 ... 96	COMPAX mit Feldbus Adresse 4
97 ... 112	COMPAX mit Feldbus-Adresse 5
113 ... 128	COMPAX mit Feldbus Adresse 6
129 ... 144	COMPAX mit Feldbus-Adresse 7
145 ... 160	COMPAX mit Feldbus Adresse 8
161 ... 176	COMPAX mit Feldbus Adresse 9
177 ... 192	COMPAX mit Feldbus-Adresse 10
193 ... 208	COMPAX mit Feldbus Adresse 11
209 ... 224	COMPAX mit Feldbus Adresse 12
225 ... 240	COMPAX mit Feldbus-Adresse 13
241 ... 256	COMPAX mit Feldbus Adresse 14

➡ Diese lineare Adressierung gilt nur im Feldbusbetrieb!

19.2 IN (digitaler Eingang)



Funktion:

Diese Anweisung bietet dem Anwender die Möglichkeit den logischen Zustand eines digitalen Eingangs oder einer definierten Eingangsgruppe als Zahlenwert einzulesen. COMTAC-Basic geht bei dieser Funktion wie folgt vor: Jedem angegebenen Eingang in der IN-Anweisung wird eine Binärstelle zugewiesen. Die Zuordnung der einzelnen Binärstellen an die angegebenen Eingänge erfolgt von rechts nach links fortlaufend.

Dem letztgenannten Eingang in der IN-Anweisung wird die Binärstelle 2 hoch 0 zugewiesen, dem davor genannten Eingang die Binärstelle 2 hoch 1 usw.

Es können maximal 16 Eingänge in einer IN-Anweisung angegeben werden.

Für eine Eingangsfolge wird der erste und der letzte Eingang getrennt durch das Zeichen ~ angegeben.

Die einzelnen Eingänge oder Eingangsfolgen werden durch ein Komma getrennt.

Parameter	Angabe	Bereich	Beschreibung
Eing.Nr.	num. Ausdr.	1 - 16 (32)	Nummer des dig. Eingangs

Folgende Beispiele sollen die Funktion der IN-Anweisung verdeutlichen:

Beispiel 1: PRINT IN(8~1)

Die logischen Zustände der Eingänge 1 bis 8 werden eingelesen und als dezimale Zahl angezeigt.

Für dieses Beispiel wird angenommen, daß die Eingänge 3,5 und 8 gesetzt sind.

Die Zuweisung der einzelnen Binärstellen an die angegebenen Eingänge sieht wie folgt aus:

Binärstellen	2 ⁷	2 ⁶	2 ⁵	2 ⁴	2 ³	2 ²	2 ¹	2 ⁰
Wertigkeit der Binärstellen	128	64	32	16	8	4	2	1
Zuordnung der Eingänge (Eingangs-Nr)	8	7	6	5	4	3	2	1
log. Zustand der Eingänge	1	0	0	1	0	1	0	0
dez. Wert der angezeigt wird	148 (128+16+4)							

Beispiel 2: PRINT IN(1~3,16,10,13)

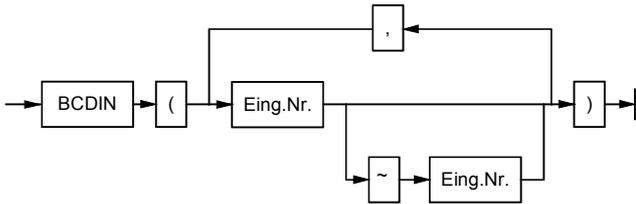
Die logischen Zustände der Eingänge 1 bis 3,10,16 und 13 werden eingelesen und als dezimale Zahl angezeigt.

Für dieses Beispiel wird angenommen, daß die Eingänge 1 und 10 gesetzt sind.

Die Zuweisung der einzelnen Binärstellen an die angegebenen Eingänge sieht wie folgt aus:

Binärstellen	2 ⁷	2 ⁶	2 ⁵	2 ⁴	2 ³	2 ²	2 ¹	2 ⁰
Wertigkeit der Binärstellen	128	64	32	16	8	4	2	1
Zuordnung der Eingänge (Eingangs-Nr)			1	2	3	16	10	13
logischer Zustand der Eingänge			1	0	0	0	1	0
dezimaler Wert der angezeigt wird	34 (32+2)							

19.3 BCDIN



Funktion:

Diese Anweisung bietet die Möglichkeit von einem Eingang bzw. von einer definierten Eingangsgruppe einen Zahlenwert im BCD-Format zu lesen.

COMTAC-Basic geht bei dieser Funktion wie folgt vor:

- ◆ Der zu lesende Wert wird in das BCD-Format (binärcodierte-Dezimalzahl) umgewandelt.
- ◆ Jedem angegebenen Eingang in der BCDIN-Anweisung wird ein Bit dieses BCD-Wertes zugewiesen und dadurch gesetzt bzw. zurückgesetzt.
- ◆ Jede angegebene Eingangs-Vierergruppe bildet eine Dekade.
- ◆ Die Zuordnung der einzelnen Bits und Dekaden an die angegebenen Eingänge erfolgt von rechts nach links fortlaufend.
- ◆ Vom letztgenannten Eingang in der BCDIN-Anweisung wird das niederwertigste Bit der niederwertigsten Dekade gelesen, vom davor genannten Eingang das nächste Bit dieser Dekade usw..
- ◆ Es werden so viele Bits des Werts berücksichtigt wie Eingänge angegeben sind.
- ◆ Es können maximal 16 Eingänge (4 Dekaden) in einer BCDOUT-Anweisung eingelesen werden.
- ◆ Für eine Ausgangsfolge wird der erste und der letzte Eingang getrennt durch das Zeichen ~ angegeben.
- ◆ Die einzelnen Eingänge oder Eingangsfolgen werden durch ein Komma getrennt.

Parameter	Angabe	Bereich	Beschreibung
Eing.Nr.	num. Ausdr.	1 - 16 (32)	Nummer des dig. Eingangs
Wert	num. Ausdr.	0 - 9999	zuzuweisender Zahlenwert

Folgende Beispiele sollen die Funktion der BCDIN-Anweisung verdeutlichen:

Beispiel 1: A=BCDIN(8~1)

Von den Eingängen 1 bis 8 wird die Zahl 35 gelesen. Die Wertigkeit der einzelnen sieht wie folgt aus:

Dekaden	10 ¹				10 ⁰			
Binärstellen	2 ³	2 ²	2 ¹	2 ⁰	2 ³	2 ²	2 ¹	2 ⁰
Wertigkeit der Binärstellen	8	4	2	1	8	4	2	1
Zuordnung der Eingänge (Eing.Nr.)	8	7	6	5	4	3	2	1
Zahlenwert 35 im BCD-Format	0	0	1	1	0	1	0	1

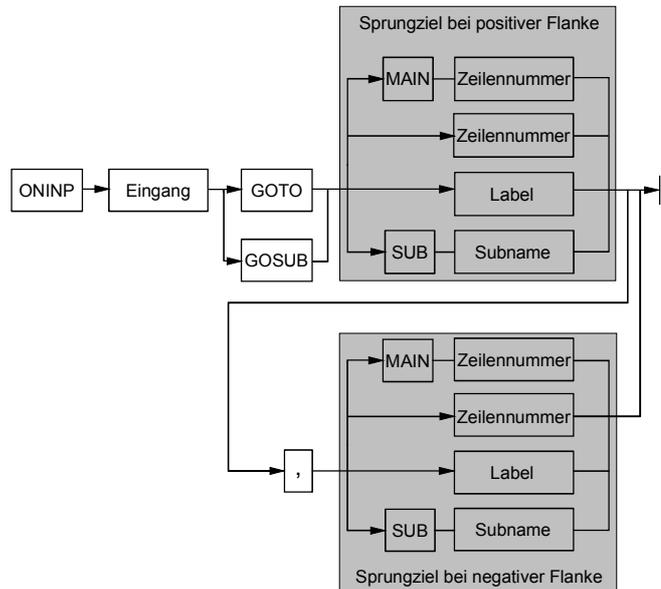
Beispiel 2: A=BCDIN(1~3,16,10,22)

Von den Eingängen 1,2,3,10,16 und 22 wird die Zahl 99 gelesen.

Die Wertigkeit der einzelnen Eingänge sieht wie folgt aus:

Dekaden	10 ¹				10 ⁰			
Binärstellen	2 ³	2 ²	2 ¹	2 ⁰	2 ³	2 ²	2 ¹	2 ⁰
Wertigkeit der Binärstellen	8	4	2	1	8	4	2	1
Zuordnung der Eingänge (Eing.Nr.)			1	2	3	16	10	22
Zahlenwert 99 im BCD-Format	1	0	0	1	1	0	0	1

19.4 ONINP [OP.]



Funktion:

Diese Anweisung definiert eine Programmverzweigung, die durchgeführt wird, wenn der entsprechende Eingang den Interrupt - Zustand angenommen hat und der ONINP-Interrupt freigegeben ist (siehe dazu ENABLE ONINP).

Parameter	Angabe	Bereich	Beschreibung
Eingang	num. Ausdr.	1 - 16	Nr. des Interrupt-Eingangs
Zeilen-Nr. positive (o. negative) ² Flanke	Zahl	0-65535	Ziel-Zeilen-Nr. des Programms bei einer positiven (o. negativen) Flanke am angegebenen Eingang
Zeilen-Nr. negative Flanke	Zahl	0-65535	Ziel-Zeilen-Nr. des Programms bei einer negativen Flanke am angegebenen Eingang

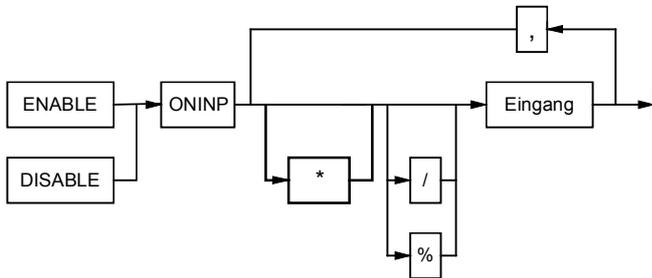
Beispiel: ONINP 2 GOTO 300
ONINP X GOSUB 10000

Anmerkung:

Die Interrupt-Eingänge sind flankengesteuert.

² Gleiches Sprungziel für positive u. negative Flanke nur, wenn die Angabe für die negative Flanke fehlt.

19.5 ENABLE [EN.]/DISABLE [DI.] ONINP [OP.]



Funktion:
Freigabe/Sperren des ONINP-Interrupt.

Parameter	Angabe	Bereich	Beschreibung
Eingang	num. Ausdr.	1 - 16	Nr. des Interrupt-Eingangs

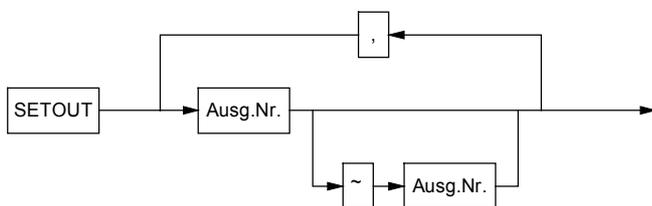
Beispiel 1: ENABLE ONINP/8
Beispiel 2: DISABLE ONINP4

Anmerkung:
Die Interrupt-Eingänge sind flankengesteuert.

Bedeutung der Zeichen

Zeichen	Bedeutung
*	Beim Aktivieren des Interrupt mit "ENABLE ONINP *..." wird bei gefordertem Signalpegel sofort ein Interrupt ausgelöst (ohne daß der entsprechende Eingang eine Flanke aufweist).
/	Der Interrupt wird mit einer negativen Flanke am entsprechenden Eingang ausgelöst.
%	Der Interrupt wird mit einer positiven und negativen Flanke am entsprechenden Eingang ausgelöst (bei jedem Signalwechsel).

19.6 SETOUT

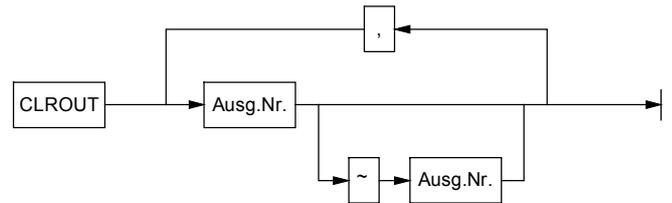


Funktion:
Diese Anweisung setzt einen oder mehrere beliebige digitale Ausgänge.
Für eine Ausgangsfolge wird der erste und der letzte zu setzende Ausgang getrennt durch das Zeichen ~ angegeben.
Die einzelnen Ausgänge oder Ausgangsfolgen werden durch ein Komma getrennt.

Parameter	Angabe	Bereich	Beschreibung
Ausg.Nr.	num. Ausdr.	1 - 16 (32)	Nummer des dig. Ausganges

Beispiel: SETOUT 3,5,18~21
SETOUT X~Y,Z

19.7 CLROUT

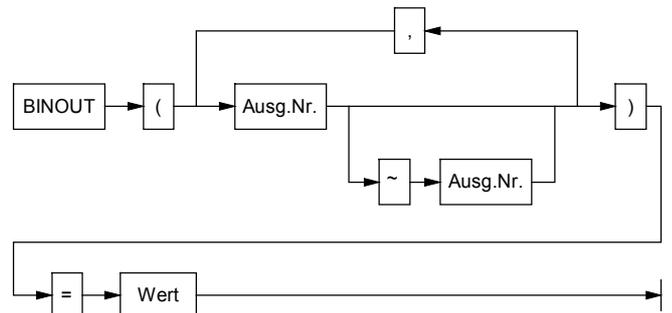


Funktion:
Diese Anweisung setzt einen oder mehrere beliebige digitale Ausgänge zurück.
Für eine Ausgangsfolge wird der erste und der letzte rückzusetzende Ausgang getrennt durch das Zeichen ~ angegeben.
Die einzelnen Ausgänge oder Ausgangsfolgen werden durch ein Komma getrennt.

Parameter	Angabe	Bereich	Beschreibung
Ausg.Nr.	num. Ausdr.	1 - 16 (32)	Nummer des dig. Ausganges

Beispiel: CLROUT 10,4~8,A,B
CLROUT V,8~4

19.8 BINOUT [B.]



Funktion:
Diese Anweisung bietet dem Anwender die Möglichkeit einem Ausgang bzw. einer definierten Ausgangsgruppe einen Zahlenwert zuzuweisen.
COMTAC-Basic geht bei dieser Funktion wie folgt vor:
♦ Der zuzuweisende Wert wird in das binäre Format umgewandelt.
♦ Jedem angegebenen Ausgang in der BINOUT-Anweisung wird eine Binärstelle dieses Wertes zugewiesen und dadurch gesetzt bzw. zurückgesetzt.

- ◆ Die Zuordnung der einzelnen Binärstellen an die angegebenen Ausgänge erfolgt von rechts nach links fortlaufend.
- ◆ Dem letztgenannten Ausgang in der BINOUT-Anweisung wird die Binärstelle 2^0 zugewiesen, dem davor genannten Ausgang die Binärstelle 2^1 usw..
- ◆ Es werden so viele Binärstellen des Werts berücksichtigt wie Ausgänge angegeben sind.
- ◆ Es können maximal 16 Ausgänge in einer BINOUT-Anweisung angegeben werden.
- ◆ Für eine Ausgangsfolge wird der erste und der letzte Ausgang getrennt durch das Zeichen ~ angegeben.
- ◆ Die einzelnen Ausgänge oder Ausgangsfolgen werden durch ein Komma getrennt.

Parameter	Angabe	Bereich	Beschreibung
Ausg.Nr.	num. Ausdr.	1 - 16 (32)	Nummer des dig. Ausganges
Wert	num. Ausdr.	0 - 65535	zuzuweisender Zahlenwert

Folgende Beispiele sollen die Funktion der BINOUT-Anweisung verdeutlichen:

Beispiel 1:

BINOUT(8~1)=35

Den Ausgängen 1 bis 8 wird die Zahl 35 zugewiesen.

Die Zuweisung der einzelnen Binärstellen an die angegebenen Ausgänge sieht wie folgt aus:

Binärstellen	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Wertigkeit der Binärstellen	128	64	32	16	8	4	2	1
Zuordnung der Ausgänge (Ausgangs-Nr)	8	7	6	5	4	3	2	1
Zahlenwert 35 im Binärformat	0	0	1	0	0	0	1	1

Beispiel 2:

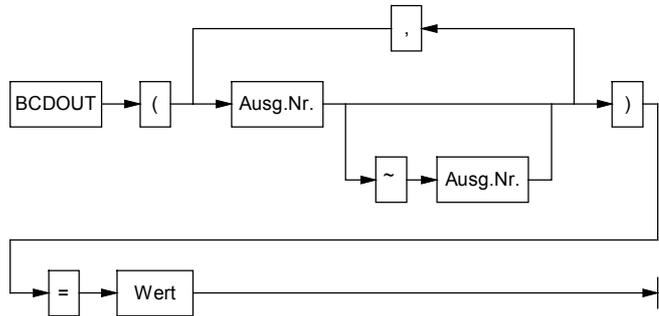
BINOUT(1~3,16,10,13)=99

Den Ausgängen 1,2,3,16,10, und 13 wird die Zahl 99 zugewiesen.

Die Zuweisung der einzelnen Binärstellen an die angegebenen Ausgänge sieht wie folgt aus:

Binärstellen	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Wertigkeit der Binärstellen	128	64	32	16	8	4	2	1
Zuordnung der Ausgänge (Ausgangs-Nr)			1	2	3	16	10	13
Zahlenwert 99 im Binärformat	0	1	1	0	0	0	1	1

19.9 BCDOUT



Funktion:

Diese Anweisung bietet die Möglichkeit einem Ausgang bzw. einer definierten Ausgangsgruppe einen Zahlenwert im BCD-Format zuzuweisen.

COMTAC-Basic geht bei dieser Funktion wie folgt vor:

- ◆ Der zuzuweisende Wert wird in das BCD-Format (binär-codierte-Dezimalzahl) umgewandelt.
- ◆ Jedem angegebenen Ausgang in der BCDOUT-Anweisung wird ein Bit dieses BCD-Wertes zugewiesen und dadurch gesetzt bzw. zurückgesetzt.
- ◆ Jede angegebene Ausgangs-Vierergruppe bildet eine Dekade.
- ◆ Die Zuordnung der einzelnen Bits und Dekaden an die angegebenen Ausgänge erfolgt von rechts nach links fortlaufend.
- ◆ Dem letztgenannten Ausgang in der BCDOUT-Anweisung wird das niederwertigste Bit der niederwertigsten Dekade zugewiesen, dem davor genannten Ausgang das nächste Bit dieser Dekade usw..
- ◆ Es werden so viele Bits des Werts berücksichtigt wie Ausgänge angegeben sind.
- ◆ Es können maximal 16 Ausgänge (4 Dekaden) in einer BCDOUT-Anweisung angegeben werden.
- ◆ Für eine Ausgangsfolge wird der erste und der letzte Ausgang getrennt durch das Zeichen ~ angegeben.
- ◆ Die einzelnen Ausgänge oder Ausgangsfolgen werden durch ein Komma getrennt.

Parameter	Angabe	Bereich	Beschreibung
Ausg.Nr.	num. Ausdr.	1 - 16 (32)	Nummer des dig. Ausganges
Wert	num. Ausdr.	0 - 9999	zuzuweisender Zahlenwert

Folgende Beispiele sollen die Funktion der BCDOUT-Anweisung verdeutlichen:

Beispiel 1: BCDOUT(8~1)=35

Den Ausgängen 1 bis 8 wird die Zahl 35 zugewiesen.

Die Zuweisung der einzelnen Bits und Dekaden an die angegebenen Ausgänge sieht wie folgt aus:

Dekaden	10^1				10^0			
Binärstellen	2^3	2^2	2^1	2^0	2^3	2^2	2^1	2^0
Wertigkeit der Binärstellen	8	4	2	1	8	4	2	1
Zuordnung der Ausgänge (Ausg.Nr.)	8	7	6	5	4	3	2	1
Zahlenwert 35 im BCD-Format	0	0	1	1	0	1	0	1

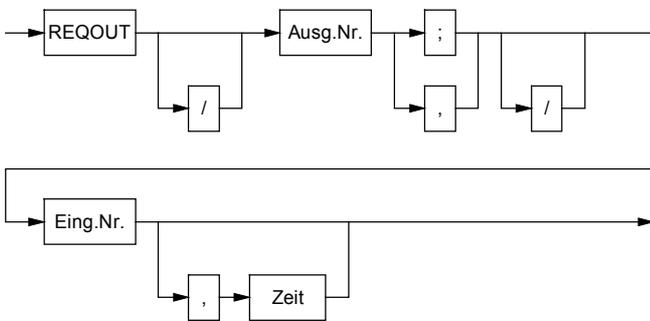
Beispiel 2: BCDOUT(1~3,16,10,22)=99

Den Ausgängen 1,2,3,10,16 und 22 wird die Zahl 99 zugewiesen.

Die Zuweisung der einzelnen Bit's und Dekaden an die angegebenen Ausgänge sieht wie folgt aus:

Dekaden	10 ¹				10 ⁰			
Binärstellen	2 ³	2 ²	2 ¹	2 ⁰	2 ³	2 ²	2 ¹	2 ⁰
Wertigkeit der Binärstellen	8	4	2	1	8	4	2	1
Zuordnung der Ausgänge (Ausg.Nr.)			1	2	3	16	10	22
Zahlenwert 99 im BCD-Format	1	0	0	1	1	0	0	1

19.10 REQOUT



Funktion:

Diese Anweisung setzt einen beliebigen dig. Ausgang und fragt anschließend einen frei wählbaren dig. Eingang ab.

Nimmt der Eingang den vorgegebenen logischen Zustand an, wird die Funktion abgebrochen, der Ausgang zurückgesetzt und die Information wahr = 65535 übergeben.

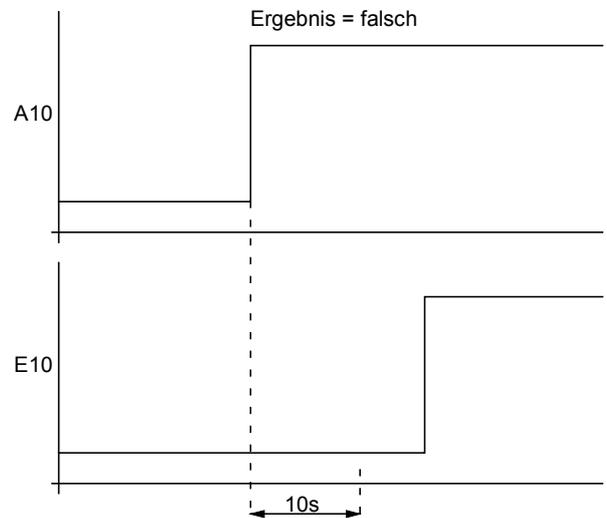
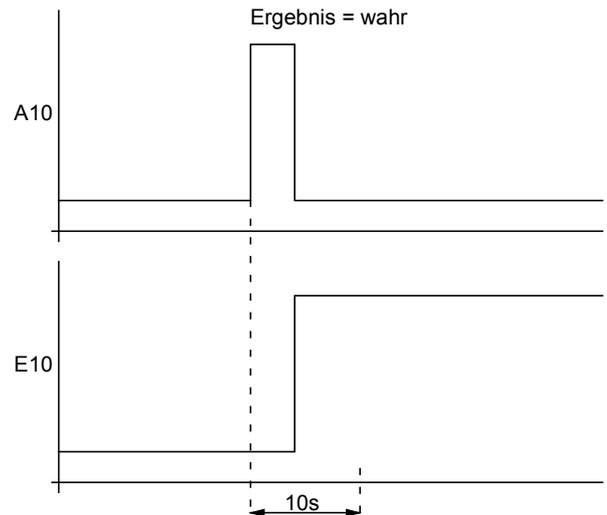
Ist die programmierte Zeit abgelaufen, bevor die Rückmeldung ansteht, wird die Information falsch = 0 übergeben. Der Ausgang wird in diesem Fall nur zurückgesetzt, wenn in der Befehls-Syntax nach Ausg.Nr. das Zeichen ; steht.

Parameter	Angabe	Bereich	Beschreibung
Ausg.Nr.	num. Ausdr.	1 - 16 (32)	Nummer des dig. Ausganges
Eing.Nr.	num. Ausdr.	1 - 16 (32)	Nummer des dig. Eingangs
Zeit	num. Ausdr.	1 - 255	Zeitangabe (1 = 100 ms)

Beispiel 1: IF REQOUT 2;7,30 THEN ...

Beispiel 2: W = REQOUT A,/E

Beispiel 3: W = REQOUT 10,10,100

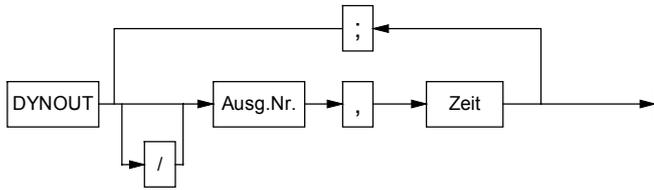


Anmerkung:

Das Zeichen / in der Befehls-Syntax definiert den logischen Zustand 0.

Wird keine Zeitangabe gemacht, setzt COMTAC-Basic als Ersatzwert den Inhalt von P8 ein.

19.11 DYNOUT



Funktion:

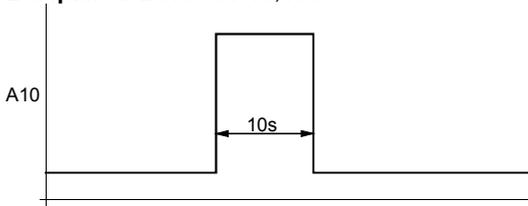
Diese Anweisung setzt einen beliebige digitalen Ausgang, der wieder zurückgesetzt wird, wenn die angegebene Zeit abgelaufen ist.

Parameter	Angabe	Bereich	Beschreibung
Ausg.Nr.	num. Ausdr.	1 - 16	Nummer des dig. Ausgangs
Zeit	num. Ausdr.	1 - 255	1 = 100 ms

Beispiel 1: DYNOUT 7,50

Beispiel 2: DYNOUT A,T1;8,T2;2,100

Beispiel 3: DYNOUT 10,100



Anmerkung:

COMTAC-Basic verweilt **nicht** in dieser Funktion bis die angegebene Zeit abgelaufen ist.

Mit der Anweisung DYNOUT x,0 bzw. DYNOUT /x,0 (x = Ausgangsnummer) kann ein noch aktiver dynamischer Ausgang in den gewünschten statischen Zustand gebracht werden.

19.12 KEYSWITCH



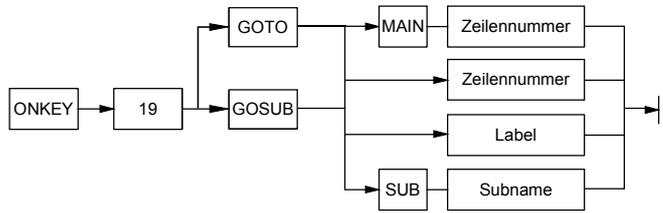
Funktion:

Abfrage des Schüsselschalters (Stecker X7, S1).

Ist S1 gebrückt, dann liefert die Funktion den logischen Zustand wahr.

Beispiel: IF KEYSWITCH THEN GOTO 100

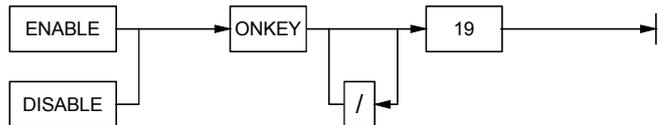
19.13 ONKEY 19 (Schüsselschalter)



Interrupt über Schüsselschalter

Beispiel: ENABLE ONKEY 19

19.14 ENABLE [EN./]DISABLE [DI.] ONKEY 19 (Schüsselschalter)



Interrupt über Schüsselschalter freigeben

Die Nummer 19 steht für den Schüsselschalter. Sie können einen Interrupt beim Öffnen oder beim Schließen des Schüsselschalters aktivieren:

- ◆ Interrupt beim Schließen: ENABLE ONKEY 19
- ◆ Interrupt beim Öffnen: ENABLE ONKEY /19

19.15 EMY_STOP



Funktion:

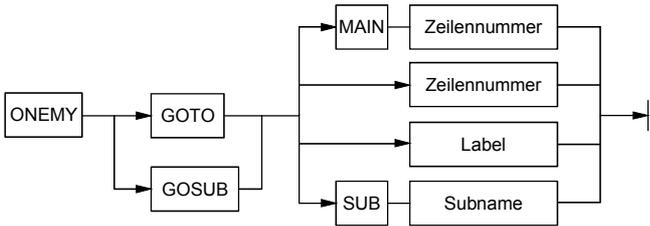
Abfrage des Eingangs "Not Stop" am Stecker X7 (SPS-Eingang).

Eingang $\geq 10V$ und $< 24V \rightarrow$ EMY_STOP = wahr (=65535)

Eingang $< 10V \rightarrow$ EMY_STOP = falsch (=0)

◆ **Beispiel:** IF EMY_STOP THEN GOTO 100

19.16 ONEMY



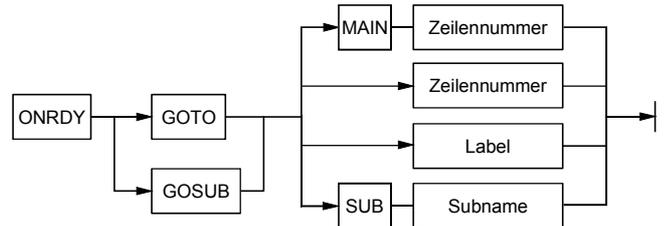
Funktion:

Diese Anweisung definiert eine entsprechende Programmverzweigung, die aufgrund einer Flanke am Not-Stop-Eingangs (X7/5) durchgeführt wird.

Parameter	Angabe	Bereich	Beschreibung
Zeilen-Nr.	Zahl	0-65535	vorhandene Zeilen-Nummer im Programm

Beispiel: ONEMY GOTO 25500
ONEMY GOSUB 8880

19.19 ONRDY



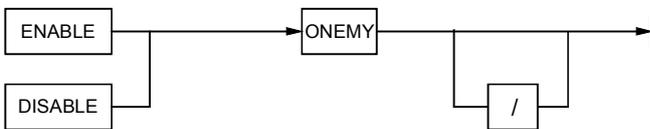
Funktion:

Diese Anweisung definiert eine entsprechende Programmverzweigung, die aufgrund eines Signals am Bereit-Eingangs (X7/6) durchgeführt wird.

Parameter	Angabe	Bereich	Beschreibung
Zeilen-Nr.	Zahl	0-65535	vorhandene Zeilen-Nummer im Programm

Beispiel: ONRDY GOTO 25500
ONRDY GOSUB 8880

19.17 ENABLE [EN.]/DISABLE [DI.] ONEMY

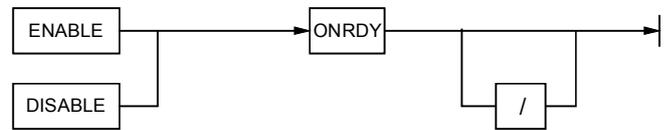


Funktion:

Freigabe oder Sperren des ONEMY-Interrupts.
 ♦ Interrupt bei positiver Flanke: ENABLE ONEMY
 ♦ Interrupt bei negativer Flanke: ENABLE ONEMY /

Beispiel: ENABLE ONEMY
DISABLE ONEMY

19.20 ENABLE [EN.]/DISABLE [DI.] ONRDY



Funktion:

Freigabe oder Sperren des ONRDY-Interrupts.
 ♦ Interrupt bei positiver Flanke: ENABLE ONRDY
 ♦ Interrupt bei negativer Flanke: ENABLE ONRDY /

Beispiel: ENABLE ONRDY
DISABLE ONRDY

19.18 RDY



Funktion:

Abfrage des Eingangs "Bereit" am Stecker X7 (SPS-Eingang).

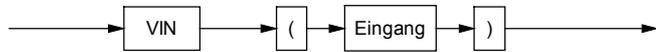
Eingang $\geq 10V$ und $< 24V \rightarrow RDY = \text{wahr}$ (=65535)

Eingang $< 10V \rightarrow RDY = \text{falsch}$ (=0)

Beispiel: IF RDY THEN GOTO 100

20. Analoge Ein-/Ausgänge

20.1 VIN



Funktion:

Diese Anweisung bietet dem Anwender die Möglichkeit die momentan anstehende Spannung an einem der 3 Analog-eingänge abzufragen.

Parameter	Angabe	Bereich	Beschreibung
Eingang	num. Ausdr.	0-2	Nummer des Analog-eingangs

Spannungs- und Wertebereich

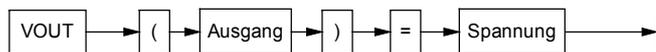
Ausgang	Spannungs-bereich	Zahlenbereich	Pin
Eingang 0	0...5V	0-255	X5/25- X5/23
Ausgang 1	-10V ... +10V	direkt in Volt	X6/12-X6/10
Ausgang 2	0 ...+10V	direkt in Volt	X6/13-X6/10

Beispiel: PRINT;VIN(1)
X=VIN(V)

Anmerkung:

Die Einheit des eingelesenen Wertes ist Volt und liegt im Bereich von ±10,00.

20.2 VOUT



Funktion:

Diese Anweisung bietet die Möglichkeit über einen der 2 Analogausgänge eine Spannung auszugeben.

Parameter	Angabe	Bereich	Beschreibung
Eingang	num. Ausdr.	1-2	Nummer des Analog-ausgangs
Spannung	num. Ausdr.	-10...+10	Ausgangsspg. in Volt

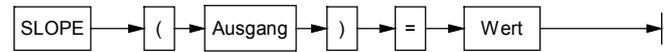
Beispiel: VOUT(1) = -3.4
VOUT(2) = SPG(3)

Anmerkung:

Ein Spannungssprung kann mittels einer programmierbaren Rampenzeit verzögert werden (siehe SLOPE).
Auflösung: 10mV

➡ Die Analogausgänge stehen nur mit der Option D2 zur Verfügung!

20.3 SLOPE



Funktion:

Rampenzeit für einen der 2 Analogausgänge zu programmieren.

Parameter	Angabe	Bereich	Beschreibung
Eingang	num. Ausdr.	1-2	Nummer des Analog-ausgangs
Wert	num. Ausdr.	0...2000	Rampenzeit: Auflösung 0,01s

Beispiel: SLOPE(2)=7 oder SLOPE (A)=X

Anmerkung:

Die Rampenzeit bezieht sich auf einen Spannungssprung von 0 auf 10V.

➡ Die Analogausgänge stehen nur mit der Option D2 zur Verfügung!

21. Interrupt Verarbeitung

21.1 Interrupt Quellen

COMTAC verfügt über 48 verschieden Interrupt-Quellen.

Quelle	Aufruf	Beschreibung
Fehler	ONERR GOTO/GOSUB	COMTAC erkennt einen Fehler
COMPAX-Fehler	ONCPXERR GOTO/GOSUB	Ein am Feldbus angeschlossenes COMPAX meldet an COMTAC Fehler oder Warnung
Not-Stop-Eingang (EMY_STOP)	ONEMY GOTO/GOSUB	Der analoge COMTAC-Eingang "Not Stop" hat den definierten logischen Zustand angenommen.
Timer	ONTIMER Wert,ZNr.	Der Basic-Timer hat den vorgegebenen Wert erreicht, bzw. überschritten.
Folientastatur	ONKBD GOTO/GOSUB	Ein Taste der Folientastatur wurde gedrückt, bzw. wieder losgelassen.
RS232/1-Schnittstelle	ON#0 GOTO/GOSUB	Es wurden entsprechend dem eingestellten Empfangsprotokoll Zeichen empfangen und dadurch das Bit "Input-Rdy" in STSCTR#0 gesetzt.
RS485/1-Schnittstelle	ON#1 GOTO/GOSUB	<p>Kein Feldbusbetrieb: Es wurden entsprechend dem eingestellten Empfangsprotokoll Zeichen empfangen und dadurch das Bit "Input-Rdy" in STSCTR#1 gesetzt.</p> <p>Feldbus Master: Ein Feldbus-Slave Gerät hat eine der folgenden Meldungen erzeugt:</p> <ul style="list-style-type: none"> - Timerout: Gerät ist ausgefallen, Busverbindung unterbrochen - Alarm: Gerät meldet ein Ereignis; Bedienungsanforderung - Fehler: im Gerät ist ein Fehler aufgetreten - Daten: angeforderte Daten können ausgelesen werden - Quit: die Befehlsquittung kann ausgelesen werden
RS232/2-Schnittstelle	ON#2 GOTO/GOSUB	Es wurden entsprechend dem eingestellten Empfangsprotokoll Zeichen empfangen und dadurch das Bit "Input-Rdy" in STSCTR#2 gesetzt.
RS485/2-Schnittstelle (optional - F6)	ON#3 GOTO/GOSUB	Es wurden entsprechend dem eingestellten Empfangsprotokoll Zeichen empfangen und dadurch das Bit "Input-Rdy" in STSCTR#3 gesetzt.
RS232/3-Schnittstelle (optional - F6)	ON#4 GOTO/GOSUB	Es wurden entsprechend dem eingestellten Empfangsprotokoll Zeichen empfangen und dadurch das Bit "Input-Rdy" in STSCTR#4 gesetzt.
RS485/5-Schnittstelle (steht noch nicht zur Verfügung)	ON#5 GOTO/GOSUB	Es wurden entsprechend dem eingestellten Empfangsprotokoll Zeichen empfangen und dadurch das Bit "Input-Rdy" in STSCTR#5 gesetzt.
Echtzeit-Uhr	ONTIME GOTO/GOSUB	Die vorgegebene Uhrzeit wurde erreicht bzw. überschritten.
Bereit-Eingang (RDY)	ONRDY GOTO/GOSUB	Der analoge COMTAC-Eingang "Bereit" hat den definierten logischen Zustand angenommen.
Schlüsselschalter (KEYSWITCH)	ONKEY 19 GOTO/GOSUB	Der Schlüsselschalter wurde geschlossen bzw. geöffnet
Funktionstate 1 ... 18	ONKEY x GOTO/GOSUB	Die angegeben Funktionstaste wurde gedrückt
Digitale Eingänge 1 ... 16	ONINP GOTO/GOSUB	An dem angegebenen Eingang wurde der vorgegebene Flankenwechsel erkannt.

21.1.1 Interrupt Initialisierung

Jeder Interrupt muß zuerst initialisiert werden, bevor COMTAC darauf reagieren kann.

Mit der Anweisung **ONInterrupt GOTO/GOSUB Zeilen-Nr.** werden die Initialisierungsdaten **Zieladresse** und **Programmsteuerung** an COMTAC übergeben.

21.1.2 Interrupt Überwachung

Alle initialisierten und freigegebenen Interrupt-Quellen werden vom Betriebssystem überwacht.

Jede Interrupt-Quelle besitzt einen eigenen Interrupt-Speicher. Das Betriebssystem setzt den Interrupt-Speicher, wenn der entsprechende Interrupt eintrifft, d.h. die entsprechenden Interrupt-Bedingungen erfüllt wurden.

21.1.3 Interrupt Überwachung freigeben

Das Betriebssystem überwacht eine Interrupt-Quelle erst, wenn diese auch explizit freigegeben wurde.

Mit der Anweisung **ENABLE ONInterrupt** wird der Interrupt freigegeben, wenn vorher die entsprechende Interrupt-Initialisierungs-Anweisung (**ON Interrupt GOSUB Zeilen-Nr.**) gemacht wurde. Andernfalls wird die Fehlermeldung 56 ausgegeben.

Außerdem wird mit dieser Anweisung der Interrupt-Speicher gelöscht.

Achtung

Die Anweisung **ONInterrupt GOTO/GOSUB Zeilen-Nr.** muß vor der Anweisung **ENABLE ONInterrupt** stehen.

21.1.4 Interrupt Überwachung sperren

Mit der Anweisung **DISABLE ONInterrupt** beendet das Betriebssystem die Überwachung der angegebenen Interrupt-Quelle. Außerdem wird mit dieser Anweisung ein bereits anstehender Interrupt gelöscht.

21.1.5 Interrupt-Verzweigung

Nach der Abarbeitung jeder BASIC-Zeile wird vom Betriebssystem die Interrupt-Kontroll-Routine aufgerufen. Diese Routine prüft ob Interrupts anstehen und veranlaßt gegebenenfalls zur angegebenen Zieladresse zu verzweigen.

Anmerkung

Ein Interrupt wird immer am Ende einer Programmzeile erkannt. Innerhalb einer Zeile wird kein Interrupt ausgeführt. Es sollte deshalb vermieden werden, mehrere Anweisungen in eine Zeile zu schreiben !

21.1.6 Verzweigung sperren



Mit der Anweisung **DISABLE** wird die Programmverzweigung durch jeden beliebigen anstehenden Interrupt verriegelt (Ausnahme: **ONERR**). Mit der Anweisung **STOP ONInterrupt** wird das Betriebssystem veranlaßt keine Programmverzweigung durchzuführen, wenn der entsprechende Interrupt ansteht.

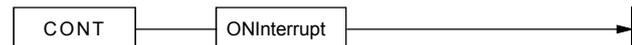
21.1.7 Interrupt-Verzweigung freigeben

Enable



Mit der Anweisung **ENABLE** wird die Verriegelung der Programmverzweigung durch anstehende Interrupts aufgehoben.

CONT ONInterrupt



Mit der Anweisung **CONT ONInterrupt** wird die Verriegelung der Interrupt-Verzweigung für diese Interrupt-Quelle wieder aufgehoben.

21.1.8 Interrupt rücksetzen

CLEARI



Mit der Anweisung **CLEARI** werden alle Interrupts in einen definierten Zustand gebracht; d.h. alle anstehenden Interrupts werden gelöscht, die Interruptüberwachung für alle Interrupts wird gesperrt, die Interruptverriegelung wird für alle Interrupts aufgehoben und die Interrupt-Prioritätsverwaltung rückgesetzt.

Die Interrupt-Initialisierung bleibt erhalten; d.h. bereits angegebene Zieladressen (Interruptsprung-Tabelle), der Initialisierungs-Merker und die Information zur Programmsteuerung werden nicht gelöscht.

CLEAR ONInterrupt



Mit der Anweisung **CLEAR ONInterrupt** wird nur dieser Interrupt in einen definierten Zustand gebracht; d.h. steht dieser Interrupt an, wird er gelöscht, die Interruptüberwachung wird für diesen Interrupt gesperrt und die Interruptverriegelung wird für diesen Interrupt aufgehoben. Ist dieser Interrupt noch in Bearbeitung, d.h. er wurde noch nicht mit **RETI** beendet, wird die Interrupt-Prioritätsstufe die dieser Interrupt belegt freigegeben, und der In_Bearbeitung-Merker gelöscht.

Die Interrupt-Initialisierung bleibt erhalten; d.h. die bereits angegebene Zieladressen (Interruptsprung-Tabelle), der Initialisierungs-Merker und die Information zur Programmsteuerung werden nicht gelöscht.

21.1.9 Interrupt-Rücksprung

Mit der Anweisung **RETI** wird ein Interrupt-Unterprogramm beendet; d.h. die aktuelle Interrupt-Prioritätsstufe wird wieder freigegeben und der In_Bearbeitung-Merker gelöscht.

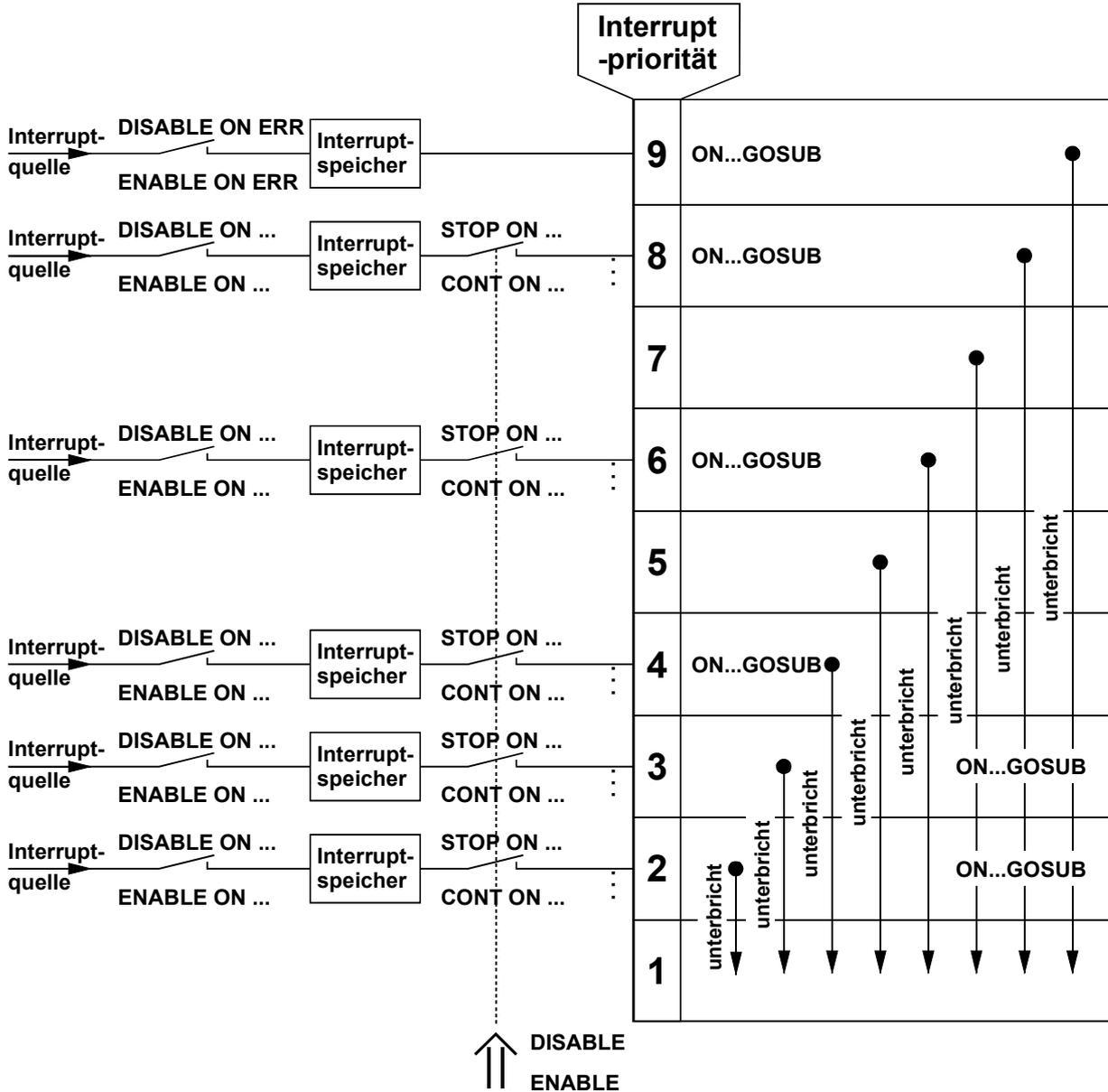
21.2 Interrupt-Priorität

- ◆ Der Fehler-Interrupt (**ONERR GOSUB ...**) hat die höchste Priorität (9) und ist nicht veränderbar. Das Fehler-Interrupt-Unterprogramm kann durch keinen anderen Interrupt unterbrochen werden.
- ◆ Jedem anderen Interrupt kann eine Priorität von 1 (niedrigste) bis 8 (höchste) zugewiesen werden.
- ◆ Ein Interrupt mit höherer Priorität kann ein Interrupt-Unterprogramm mit niedriger Priorität unterbrechen.
- ◆ Das laufende Interrupt-Unterprogramm kann nicht von einem Interrupt mit gleicher Priorität unterbrochen werden.
- ◆ Ein Interrupt-Unterprogramm muß mit der **GOSUB**-Anweisung initialisiert (**ONInterrupt.GOSUB ...**) werden und mit der **RETI**-Anweisung beendet werden.
- ◆ Verzweigt ein Interrupt mit der **GOTO**-Anweisung (**ONInterrupt.GOTO ...**) beachtet auch dieser die Priorität der sich in Bearbeitung befindenden Interrupt-Unterprogramme, aber der Programmteil, welcher durch die **GOTO**-Anweisung dann bearbeitet wird unterliegt keiner Prioritäts-Ordnung.
- ◆ Nach Power-On oder Reset haben alle Interrupt-Quellen (Ausnahme: **ONERR**) die kleinste Priorität (1).

Hinweis zur Interruptbearbeitung nach RETURN

Nach einer **RETURN** - Anweisung wird immer noch mindestens eine weitere **BASIC**-Anweisung ausgeführt, bevor in ein **BASIC**-Interrupt-Unterprogramm verzweigt wird.

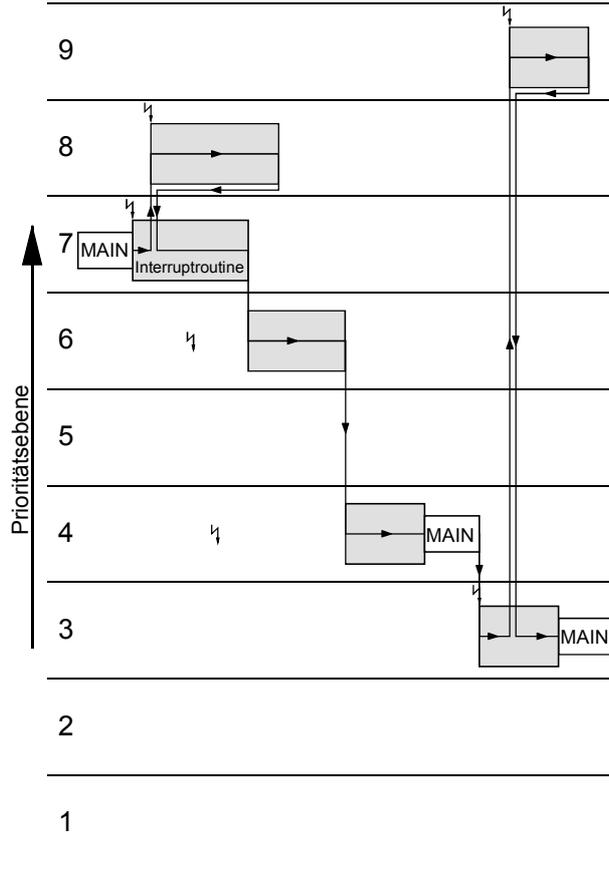
Funktionsweise der Interrupt - Verarbeitung



Beschreibung:

- ◆ Der Fehler-Interrupt "ONERR" hat immer die höchste Priorität (9); .
- ◆ Allen anderen Interrupt - Quellen können beliebige Prioritäten zugewiesen werden. Dabei haben mehrere Interrupt - Quellen gleiche Priorität und nicht jede Priorität muß mit einer Interrupt - Quellen versehen sein.
- ◆ Mit ENABLE ON... wird die Interrupt - Quelle aktiviert, der Interrupt-Speicher gelöscht und das Sperren des Interrupts aufgehoben.
- ◆ Mit DISABLE ON... wird die Interrupt - Quelle deaktiviert und der Interrupt-Speicher gelöscht.
- ◆ Das Interrupt-Signal eines aktivierten Interrupts wird im Interrupt-Speicher gespeichert.
- ◆ Der Aufruf eines Interrupt-Unterprogramms kann über STOP ON... vorübergehend gesperrt werden, ohne daß das Interrupt - Signal verloren geht. Mit CONT ON... wird das Interrupt - Signal wieder weitergeleitet.
- ◆ Mit DISABLE können alle Interrupt - Signale vorübergehend gesperrt werden ohne daß ein Interrupt - Signal verloren geht. Mit ENABLE werden die Interrupt - Signale wieder weitergeleitet.
- ◆ Das Interrupt-Unterprogramm wird automatisch beim Auftreten eines Interrupt - Signale über den Befehl "ON... GOSUB Zeilennummer (oder SUB)" aufgerufen.

Beispiel: Reihenfolge von Interrupt-Unterprogrammen



Die Pfeile (↴) kennzeichnen die Stelle, an der ein Interrupt-Signale der entsprechenden Prioritätssebene auftritt.

Interrupt-Prioritäten innerhalb einer Prioritätssebene bzw. ohne Vergabe von Prioritäten

(1)	ONCPXERR
(2)	ONEMY-Interrupt
(3)	ONTIMER
(4)	ONKBD
(5)	ON#0
(6)	ON#1
(7)	ON#2
(8)	ON#3
(9)	ON#4
(10)	ON#5
(11)	ONTIME
(12)	ONRDY
(13)	ONKEY 19
(14)	ONKEY 17
(15)	ONKEY 18
(16)	ONKEY 1
(17)	ONKEY 2
(18)	...
(19)	ONKEY 16
(20)	ONINP 1
(21)	ONINP 2
(22)	...
(23)	ONINP 16
(24)	ONINP 1
(25)	ONINP 2
(26)	...
(27)	ONINP 16

21.2.1 Fehler Interrupt

Alle Fehlermeldungen des COMTAC können diesen Interrupt auslösen.

Die Fehler-Nr. und die Nummer der Zeile in der dieser Fehler aufgetreten ist wird zwischengespeichert.

Initialisierung:	ONERR GOTO/GOSUB ZeilenNr.
Überwachung freigeben:	ENABLE ONERR
Überwachung sperren:	DISABLE ONERR
Verzweigung sperren:	nicht möglich
Verzweigung freigeben:	nicht möglich
Rücksetzen:	CLEAR ONERR
Priorität setzen/lesen:	nicht möglich
Fehler-Nr.:	ERRSTS
Fehler-Zeile:	ERRSTSL
Fehler-Sub:	ERRSTS#
Befehlsbeschreibung	siehe Seite 92

21.2.2 COMPAX Fehler Interrupt

Ein am Feldbus angeschlossenes COMPAX meldet an COMTAC Fehler oder Warnung. Die Geräteadresse des COMPAX, das den Interrupt ausgelöst hat, wird zwischengespeichert.

Initialisierung:	ONCPXERR GOTO/GOSUB ZeilenNr.
Überwachung freigeben:	ENABLE ONCPXERR
Überwachung sperren:	DISABLE ONCPXERR
Verzweigung sperren:	STOP ONCPXERR
Verzweigung freigeben:	CONT ONCPXERR
Rücksetzen:	CLEAR ONCPXERR
Priorität setzen:	PRIORITY ONCPXERR = Wert
Geräteadresse:	CPXERRADR
Befehlsbeschreibung	siehe Seite 108

21.2.3 Not-Stop Eingang Interrupt

Dieser Interrupt-Eingang ist flankengesteuert und kann wahlweise durch eine positive oder negative Flanke erzeugt werden.

Initialisierung:	ONEMY GOTO/GOSUB ZeilenNr.
Überwachung freigeben:	ENABLE ONEMY bzw. ENABLE ONEMY / (neg. Flanke)
Überwachung sperren:	DISABLE ONEMY
Verzweigung sperren:	STOP ONEMY
Verzweigung freigeben:	CONT ONEMY
Rücksetzen:	CLEAR ONEMY
Priorität setzen:	PRIORITY ONEMY = x
Befehlsbeschreibung	siehe Seite 69

21.2.4 Timer Interrupt:

Initialisierung:	ONTIMER <i>Vergleichswert</i> , <i>ZeilenNr.</i>
mit Auto-Reload-Fkt.:	ONTIMER <i>Vergleichswert</i> ; <i>ZeilenNr.</i>
Timer freigeben:	ENABLE ONTIMER
Timer sperren:	DISABLE ONTIMER
Verzweigung sperren:	STOP ONTIMER
Verzweigung freigeben:	CONT ONTIMER
Rücksetzen:	CLEAR ONTIMER
Priorität setzen:	PRIORITY ONTIMER = x
Befehlsbeschreibung	siehe Seite 60

21.2.5 Folientastatur Interrupt

Alle Tasten der Folientastatur des COMTAC können diesen Interrupt auslösen. Der Tastencode der entsprechenden Taste wird zwischengespeichert. Das Loslassen einer Taste erzeugt ebenfalls einen Interrupt mit dem Tastencode 0.

Initialisierung:	ONKBD GOTO/GOSUB <i>ZeilenNr.</i>
Überwachung freigeben:	ENABLE ONKBD
Überwachung sperren:	DISABLE ONKBD
Verzweigung sperren:	STOP ONKBD
Verzweigung freigeben:	CONT ONKBD
Rücksetzen:	CLEAR ONKBD
Priorität setzen:	PRIORITY ONKBD = <i>Wert</i>
Tastencode:	KBDCODE
Befehlsbeschreibung	siehe Seite 57

21.2.6 Schnittstellen Interrupt

Ein Interrupt wird erzeugt, wenn entsprechend dem eingestellten Empfangsprotokoll der Schnittstelle Zeichen empfangen wurden und dadurch das Bit "Input-Rdy" der Schnittstelle gesetzt wurde.

Initialisierung:	ON#x GOTO/GOSUB <i>ZeilenNr.</i>
Überwachung freigeben:	ENABLE ON#x
Überwachung sperren:	DISABLE ON#x
Verzweigung sperren:	STOP ON#x
Verzweigung freigeben:	CONT ON#x
Rücksetzen:	CLEAR ON#x
Priorität setzen:	PRIORITY ON#x = <i>Wert</i>
Befehlsbeschreibung	siehe Seite 83 und 87

21.2.7 Echtzeit-Uhr Interrupt

Ein Interrupt wird erzeugt, wenn die Echtzeit-Uhr die angegebene Interrupt-Uhrzeit erreicht hat. Dieser Interrupt wird einmal initialisiert, freigegeben und automatisch täglich wiederholt.

Initialisierung:	ONTIME <i>Uhrzeit</i> GOTO/GOSUB <i>ZeilenNr.</i>
Überwachung freigeben:	ENABLE ONTIME
Überwachung sperren:	DISABLE ONTIME
Verzweigung sperren:	STOP ONTIME
Verzweigung freigeben:	CONT ONTIME
Rücksetzen:	CLEAR ONTIME
Priorität setzen:	PRIORITY ONTIME = <i>Wert</i>
Befehlsbeschreibung	siehe Seite 58

21.2.8 Bereit Eingang Interrupt

Dieser Interrupt-Eingang ist flankengesteuert und kann wahlweise durch eine positive oder negative Flanke erzeugt werden.

Initialisierung:	ONRDY GOTO/GOSUB <i>ZeilenNr.</i>
Überwachung freigeben:	ENABLE ONRDY bzw. ENABLE ONRDY / (neg. Flanke)
Überwachung sperren:	DISABLE ONRDY
Verzweigung sperren:	STOP ONRDY
Verzweigung freigeben:	CONT ONRDY
Rücksetzen:	CLEAR ONRDY
Priorität setzen:	PRIORITY ONRDY = x
Befehlsbeschreibung	siehe Seite 70

21.2.9 Schlüsselschalter Interrupt

Dieser Interrupt ist flankengesteuert und kann wahlweise durch eine positive (Schalter schließen) oder negative (Schalter öffnen) Flanke erzeugt werden.

Initialisierung:	ONKEY 19 GOTO/GOSUB <i>ZeilenNr.</i>
Überwachung freigeben:	ENABLE ONKEY 19 bzw. /19 (neg. Flanke)
Überwachung sperren:	DISABLE ONKEY 19
Verzweigung sperren:	STOP ONKEY 19
Verzweigung freigeben:	CONT ONKEY 19
Rücksetzen:	CLEAR ONKEY 19
Priorität setzen:	PRIORITY ONKEY (19) = x
Befehlsbeschreibung	siehe Seite 55

21.2.10 Funktionstasten Interrupt

Alle 16 COMTAC-Funktionstasten, die START (F17) und STOP(F18)-Taste sind interruptfähig. Es kann auf jede dieser Funktionstasten separat im Basicprogramm verzweigt werden.

Initialisierung:	ONKEY x GOTO/GOSUB <i>ZeilenNr.</i>
Überwachung freigeben:	ENABLE ONKEY x
Überwachung sperren:	DISABLE ONKEY x
Verzweigung sperren:	STOP ONKEY x
Verzweigung freigeben:	CONT ONKEY x
Rücksetzen:	CLEAR ONKEY x
Priorität setzen:	PRIORITY ONKEY (x) = x
Befehlsbeschreibung	siehe Seite 65

21.2.11 Dig. Eingang Interrupt

Die COMTAC-Eingänge E1..E16 sind interruptfähig. Es kann auf jeden Eingang separat im Basicprogramm verzweigt werden. Die Interrupt-Eingänge sind flankengesteuert. Der Interrupt kann durch eine positive Flanke, eine negative Flanke oder durch jeden Flankenwechsel erzeugt werden.

Desweiteren kann festgelegt werden (ONINP *), ob ein Interrupt ausgelöst werden soll, wenn zum Zeitpunkt der Interrupt-Freigabe der Interrupt-Pegel bereits ansteht. Für den positiven und negativen Eingangs-Interrupt können eine gemeinsame oder zwei getrennte Spungziele angegeben werden.

Initialisierung:	ONINP x GOTO/GOSUB <i>ZeilenNr.</i>
Überwachung freigeben:	ENABLE ONINP x (pos. Flanke) bzw. ONINP * x
	ENABLE ONINP /x (neg. Flanke) bzw. ONINP * /x
	ENABLE ONINP %x (beide Flanken) bzw. ONINP * %x
Überwachung sperren:	DISABLE ONINP 5
Verzweigung sperren:	STOP ONINP 5
Verzweigung freigeben:	CONT ONINP 6
Rücksetzen:	CLEAR ONINP 7
Priorität setzen:	PRIORITY ONINP (10) = x
Befehlsbeschreibung	siehe Seite 65

21.3 IDLE (Warten auf Interrupt)

IDLE

Funktion:

Diese Anweisung veranlaßt das COMTAC-Basic solange zu warten, bis ein freigegebener Interrupt erkannt wird.

Beispiel: IDLE

Anmerkung:

Der Anwender sollte sich vergewissern, ob ein Interrupt freigegeben ist, bevor diese Anweisung ausgeführt wird.

22. RS485-Schnittstellen

Neben der Standard-RS485-Schnittstelle (RS485/1) können Sie mit der Option F6 (zusätzliche RS485 und RS232) eine weitere RS485-Schnittstelle (RS485/2) einsetzen.

Die Standard-RS485-Schnittstelle (RS485/1) wird mit den Parametern P60 bis P79 eingestellt und kann neben dem ASCII-Protokoll als zyklische, auf COMPAX optimierte Feldbus-Schnittstelle betrieben werden.

Die optionale RS485/2 Schnittstelle wird mit den Parametern P80 bis P89 eingestellt und kann mit ASCII-Protokoll betrieben werden (gleiche Befehle wie RS485/1).

Maximal kann der Schnittstellenverbund aus max. 32 Teilnehmer bestehen.

22.1 Funktionsbeschreibung

Ist die RS485/1-Schnittstelle auf Feldbus-Betrieb eingestellt (Parameter P73=1), gilt die Funktionsbeschreibung im Kapitel Feldbus-Schnittstelle!

➡ Nur die RS485/1-Schnittstelle läßt sich auf Feldbus-Betrieb einstellen.

Die RS485-Schnittstellen ermöglichen die Kommunikation mit einem übergeordneten Rechner (PC) oder auch anderen Geräten die eine RS485-Schnittstelle haben.

Für ein ordnungsgemäßes Arbeiten der Schnittstellen ist eine korrekte Verdrahtung Grundvoraussetzung (Siehe in der Gerätebeschreibung).

Auf die korrekte Einstellung der Schnittstellenparameter (Baudrate, Paritybit, Stopbits ...) ist zu achten.

Nach POWER-ON werden die RS485-Schnittstellen mit den im ZP-RAM (Laufwerk R) gespeicherten Parameter initialisiert.

Beachte:

OUTPUT-Ready=Statusbit 0 der Systemvariable STSCTR#1(3).

INPUT-Ready=Statusbit 1 der Systemvariable STSCTR#1(3).

22.2 Parameter der RS485/1-Schnittstelle (#1)

Nr.	Funktion	Standard	
60	Geräte-Adresse 0 - 255	0	
61	Empfangs-Start-Zeichen 0 - 255	2	
62	Empfangs-Ende-Zeichen 0 - 255	13	
63	Empfangs-Protokoll	2	
	Bit	Funktion	
	0	ProtokollBit 0	0
	1	ProtokollBit 1	1
	2	ProtokollBit 2	0
	3		0
	4	Block-Check-Character; 0 = off, 1 = on	0
	5		1
	6		1
	7	Software-Handshake; 0 = off, 1 = on	0
	Mit den ProtokollBits wird eines der folgenden Empfangsprotokolle ausgewählt:		
	P-Bit	INPUT-Rdy = 1,	
	2 1 0		0 1 0
	0 0 0	nach jedem empfangenen Zeichen	
0 0 1	nach def. Anzahl empfangener Zeichen (P66)		
0 1 0	nach empfangenem Endezeichen (P62)		
0 1 1	nach empf. Start(P61)- und Endezeichen (P62)		
1 0 0	nach empfangenen Startzeichen (P61), Geräteadresse (P60) und Empfangs-Endezeichen (P62)		
64	Baudrate	6	
	0=150 3=1200 6=9600 9=57600		
	1=300 4=2400 7=19200 10=172800		
	2=600 5=4800 8=28800 11=345600		

22.3 Parameter der RS485/2-Schnittstelle (#3) (Option F6)

65	<p>Stopbits/Parity/Hardware</p> <p>Bit 0...2 = Parity/Stopbit-Auswahl 0 ohne Parity, 1 Stopbit 1 ohne Parity, 2 Stopbit 2 mit Parity Even, 1 Stopbit 3 mit Parity Odd, 1 Stopbit</p> <p>Bit 3...5 = ohne Funktion</p> <p>Bit 6 = 4-Draht Master/Slave 0 Slave (TxD = Pin 2 und 7; RxD = Pin 1 und 6) 1 Master (TxD = Pin 1 und 6; RxD = Pin 2 und 7)</p> <p>Bit 7 = 2/4-Draht 0 2-Draht 1 4-Draht</p>	0
66	Anzahl zu empfangender Zeichen 1 - 255	1
67	Sende-Endezeichen 0 - 255	13
68	<p>Sende-Protokoll</p> <p>0: es wird kein Endezeichen automatisch gesendet 1: es wird CR automatisch gesendet 2: es wird CR LF automatisch gesendet 3: es wird das Send-/ Endezeichen (P67) automatisch gesendet 4: es wird das CR LF und das Send-/ Endezeichen (P67) automatisch gesendet 5: es wird das Empfang-Startzeichen (P61) und das Empfang-Endzeichen (P62) automatisch gesendet</p>	1
69	<p>Timeout-Wert der ENTER 1 - Anweisung</p> <p>0 - 255 (1 = 0.1s) 0 = keine Timeout-Überwachung</p>	0

22.3 Parameter der RS485/2-Schnittstelle (#3) (Option F6)

Nr.	Funktion	Standard
80	Geräte-Adresse 0 - 255	0
81	Empfangs-Start-Zeichen 0 - 255	2
82	Empfangs-Ende-Zeichen 0 - 255 (für COMPAX ist der Wert 62 erforderlich)	13
83	Empfangs-Protokoll	2
	Bit Funktion	
	0 ProtokollBit 0	0
	1 ProtokollBit 1	1
	2 ProtokollBit 2	0
	3	0
	4 Block-Check-Character; 0 = off, 1 = on	0
	5	1
	6	1
	7 Software-Handshake; 0 = off, 1 = on	0
	Mit den ProtokollBits wird eines der folgenden Empfangsprotokolle ausgewählt:	
	P-Bit INPUT-Rdy = 1,	
	2 1 0	0 1 0
	0 0 0 nach jedem empfangenen Zeichen	
	0 0 1 nach def. Anzahl empfangener Zeichen (P66)	
	0 1 0 nach empfangenem Endezeichen (P62)	
	0 1 1 nach empfangenem Start(P61)- und Endezeichen (P62)	
	1 0 0 nach empfangenen Startzeichen (P61), Geräteadresse (P60) und Empfangs-Endezeichen (P62)	
84	<p>Baudrate</p> <p>0=150 3=1200 6=9600 9=57600 1=300 4=2400 7=19200 10=76800 2=600 5=4800 8=38400 11=115200</p>	6
85	Parity / Stopbits / Zeichenlänge	0
	Bit 0 Anzahl der Stopbits: 0 = 1 Stopbit 1 = 2 Stopbit	
	Bit 1 Parity Enable: 0 = Disable 1 = Enable	
	Parity Select:	
	Bit 2/ 0/0 = Odd 1/0 = Even	
	Bit 3 0/1 = Mark 1/1 = Space	
	Zeichenlänge:	
	Bit 4/ 0/0 = 8Bit 1/0 = 7Bit	
	Bit 5 0/1 = 6Bit 1/1 = 5Bit	
	Bit 6 ohne Funktion	
	Bit 7 2/4-Draht 0 = 2-Draht 1 = 4-Draht	
86	Anzahl zu empfangender Zeichen 1 - 255	1
87	Sende-Endezeichen 0 - 255	13

88	Sende-Protokoll 0: es wird kein Endezeichen automatisch gesendet 1: es wird CR automatisch gesendet 2: es wird CR LF automatisch gesendet 3: es wird das Sende-/ Endezeichen (P67) automatisch gesendet 4: es wird das CR LF und das Sende-/ Endezeichen (P67) automatisch gesendet 5: es wird das Empfang-Startzeichen (P61) und das Empfang-Endzeichen (P62) automatisch gesendet	1
89	Timeout-Wert der ENTER 1 - Anweisung 0 - 255 (1 = 0.1s) 0 = keine Timeout-Überwachung	0

22.5 Senden

22.4 Empfangen

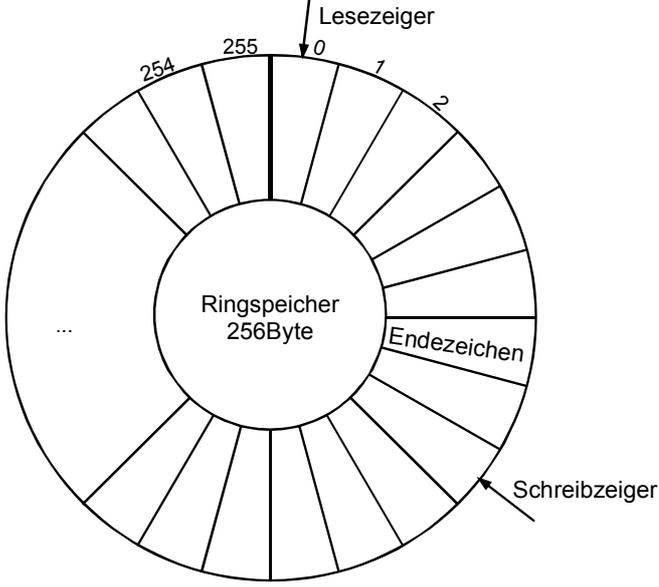
Die Empfänger der RS485-Schnittstellen verfügen über einen 256Byte großen Ringpuffer und je einen Schreib- und Lese-Zeiger.

Alle ankommenden Zeichen werden in Abhängigkeit vom eingestellten Empfangsprotokoll Parameter 63 (83) ausgewertet und gegebenenfalls im Ringpuffer gespeichert.

Wurden alle Zeichen so empfangen, wie es im Empfangsprotokoll definiert ist, wird das Bit INPUT-Ready = 1. Jetzt kann mit der Basic-Anweisung ENTER das Zeichen oder die Zeichenkette aus dem Ringpuffer abgeholt werden. Die ENTER Anweisung kopiert das Zeichen oder die Zeichenkette in den String \$(#1/#3) und kann nun im Basic weiter bearbeitet werden.

Die ENTER Anweisung löscht das Bit INPUT-Ready, wenn kein weiteres Zeichen oder weitere Zeichenketten im Ringpuffer bereit stehen.

Ist INPUT-Ready = 0 wartet die ENTER-Anweisung bis INPUT-Ready = 1 ist oder bricht die ENTER-Anweisung ab, wenn die programmierte Timeout-Zeit in Parameter 69 (89) abgelaufen ist.



Über die Schnittstelle ankommende Daten werden mit dem Schreibzeiger fortlaufend in den Ringspeicher geschrieben. Mit dem Befehl "ENTER Schnittstellenummer" werden die Zeichen ab dem Lesezeiger bis zum nächsten Endezeichen (oder der vereinbarten Anzahl zu empfangener Zeichen) ausgelesen.

Werden die Daten nicht mit ENTER ausgelesen, dann werden die Daten nach 256 Zeichen überschrieben.

COMTAC-BASIC leitet den durch die Basic-Anweisung OUTPUT definierten Ausgabestring in den 256 Byte großen Ausgabepuffer.

Eine im Hintergrund ablaufende Ausgaberroutine sendet Zeichen für Zeichen über die Schnittstelle an den Empfänger.

Mit Parameter 68 (88) kann festgelegt werden, ob nach dem Senden einer Zeichenkette automatisch noch Endezeichen gesendet werden.

Das Bit OUTPUT-Ready (STSCTR#1/#3) bleibt solange 0, bis der String vollständig ausgegeben ist; d.h. der Ausgabepuffer ist leer, es kann der nächste String ausgegeben werden.

22.6 BCC RS485

Der Block-Check-Character dient zur Verbesserung der Übertragungssicherheit und kann wahlweise über ein Bit Parameter 63 (83) ein/ausgeschaltet werden. Der BCC wird immer als letztes Zeichen eines Datenblocks gesendet; also nach dem Endezeichen (wenn vorhanden).

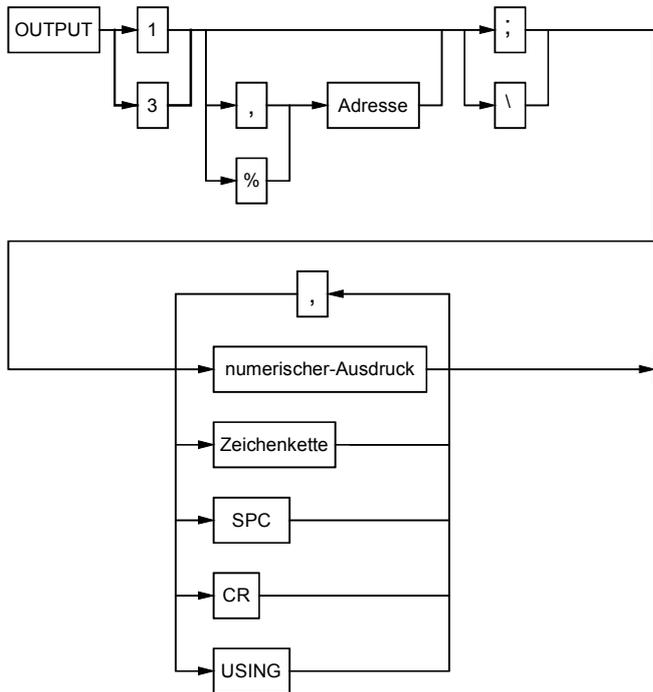
Der Block-Check-Character wird gebildet aus einer XOR-Verknüpfung (bytwweise) aller gesendeten bzw. empfangenen Zeichen eines Datenblocks exklusiv des letzten Endezeichens.

Ist BCC-Überwachung eingeschaltet ermittelt COMTAC beim Empfang von Zeichen automatisch den BCC und vergleicht diesen mit dem empfangenen BCC.

Ist der Vergleich negativ wird eine Fehlermeldung ausgegeben.

Beim Senden bildet COMTAC ebenfalls automatisch den BCC und gibt diesen als letztes Zeichen aus.

22.7 OUTPUT [O.] RS485



Funktion:

Diese Anweisung gibt Zeichen, Zeichenketten und/oder den Wert eines numerischen Ausdrucks über die RS485-Schnittstelle aus. Optimal kann sofort der zu erwartenden Antwort eingelesen werden in den String \$(#1/#3). Dafür muß in der Befehls-Syntax der OUTPUT-Anweisung das Semikolon durch den Backslash ersetzt werden.

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	0 - 255	Geräteadr. des Empfängers

Optional ist es möglich eine Geräteadresse anzugeben. Durch die Befehlssyntax wird selektiert, in welchem Format die Adresse ausgegeben wird:

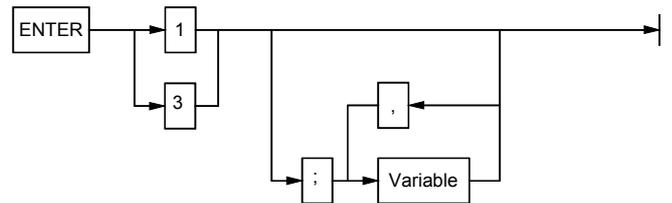
- ◆ Komma vor der Adresse = ASCII-Format
- ◆ % vor der Adresse = Binär-Format

Beispiel: OUTPUT 1;"WINKEL=",W
OUTPUT 1,5;"PA",XPOS,CHR\$(13)

Anmerkung:

Das Betriebssystem prüft nun selbständig das OUTPUT-Ready-Flag. Wie bei der ENTER-Anweisung wird nach Ablauf der Timeout-Zeit der Befehl abgebrochen. Nur dann ist der Ausgabepuffer der entsprechenden RS232-Schnittstelle leer. Mit Parameter 68 (88) kann festgelegt werden, ob nach dem Senden einer Zeichenkette automatisch noch Endezeichen gesendet werden. Die Funktion der Anweisungen SPC, CR, USING finden Sie im Kapitel 14. Ausgabeformatbefehle.

22.8 ENTER [E.] (RS485)



Funktion:

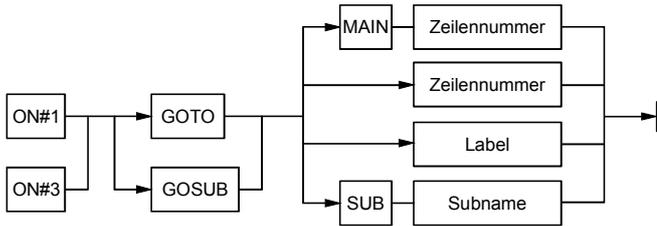
Diese Anweisung übergibt ein bereitstehendes Zeichen oder eine Zeichenkette aus dem Ringpuffer der RS485-Schnittstelle in den String \$(#1/#3) zur weiteren Verarbeitung. Enthält die einzulesende Zeichenkette eine oder mehrere Zahlen, können diese Zahlen mit der ENTER-Anweisung direkt Variablen zugewiesen werden, indem an die ENTER-Anweisung ein Variablenname bzw. eine Variablennamen-Liste angehängt wird. COMTAC-BASIC weist die erste Zahl die sich in der Zeichenkette befindet der ersten Variablen der Variablen-Name-Liste zu, die zweite Zahl der zweiten Variablen und so weiter. Die Anzahl der Zahlen in der Zeichenkette und die Anzahl der Variable- Namen muß nicht identisch sein.

Beispiel: Bereitstehender String = "10V200X-30"
ENTER 1,adresse; A,B,C,D
Nach Ausführung der Anweisung ist \$(#1) = "10V200X-30"
A = 10
B = 200
C = -30 und D unverändert.

Anmerkung:

Befindet sich keine bereitstehende Zeichenkette im Ringpuffer der seriellen Schnittstelle (INPUT-Ready = 0) dann wartet COMTAC-BASIC bis eine Zeichenkette bereitsteht bzw. bis die Timeout-Zeit Parameter 69 (89) abgelaufen ist. Wird die Timeout-Zeit überschritten generiert COMTAC eine Fehlermeldung.

22.9 ON#1/#3



Funktion:

Diese Anweisung definiert eine Programmverzweigung, die durchgeführt wird, wenn ein String im Zwischenpuffer der RS485-Schnittstelle für die weitere Verarbeitung bereitsteht (INPUT-Ready = 1) und dieser Interrupt freigegeben ist. (Siehe dazu ENABLE/DISABLE ON#1/#3).

Parameter	Angabe	Bereich	Beschreibung
Zeilen-Nr.	Zahl	0 - 65535	vorhandene Zeilen-Nummer im Programm

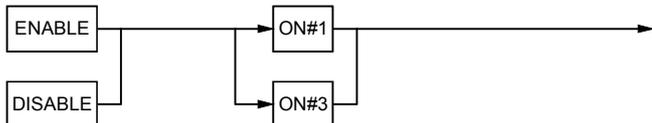
Beispiel: ON#1 GOTO ...

ON#1 GOSUB ...

Anmerkung:

Mit Parameter 63 (83) wird festgelegt, wann ein String im Zwischenpuffer zur Übernahme bereitsteht.

22.10 ENABLE [EN.] / DISABLE [DI.] (RS485)

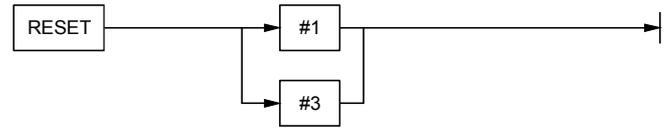


Funktion:

Freigabe/Sperren des RS485 - Interrupt.

Beispiel: ENABLE ON#1
DISABLE ON#1

22.11 RESET [RS.] RS485

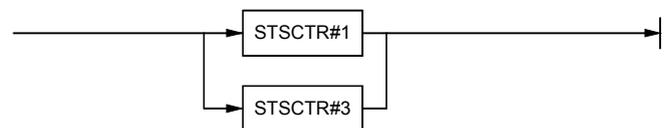


Funktion:

Diese Anweisung setzt die RS485-Schnittstelle in den Power-On Zustand zurück.

Beispiel: RESET #1

22.12 STSCTR#1/#3



Funktion:

Diese speziellen Systemvariablen liefern die Statusinformation der RS485-Schnittstellen. Die einzelnen Bits haben folgende Bedeutung:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
BCC Error	Parity Error	Overrun Error	0	0 oder 1	0 oder 1	INPUT Ready	OUTPUT Ready
128	64	32	16	8	4	2	1

- ◆ Bit 0: der Ausgabepuffer ist leer; es kann ein String ausgegeben werden
- ◆ Bit 1: im Zwischenpuffer steht ein String für die weitere Verarbeitung bereit und kann mit ENTER eingelesen werden
- ◆ Bit 2: reserviert
- ◆ Bit 3: reserviert
- ◆ Bit 4: reserviert
- ◆ Bit 5: Eingangspuffer überlaufen
- ◆ Bit 6: es ist ein Parity-Fehler festgestellt worden
- ◆ Bit 7: es ist ein BCC-Fehler festgestellt worden

Beispiel: PB.STSCTR#1

IF (STSCTR#1 'AND' 10B) = 0 THEN

Anmerkung:

- ◆ Die Fehlerbits (BCC-, Parity-, Timeout-Error) werden durch das Lesen von STSCTR#1/#3 automatisch gelöscht.
- ◆ Unter STSCTR#3 stehen die Statusinformationen der RS485/2 Schnittstelle.

23. RS232-Schnittstelle

23.1 Funktionsbeschreibung

Die RS232-Schnittstelle ermöglicht die Kommunikation mit einem übergeordneten Rechner (PC) oder auch anderen Geräten die eine RS232-Schnittstelle haben.

Für ein ordnungsgemäßes Arbeiten der Schnittstelle ist eine korrekte Verdrahtung Grundvoraussetzung. (Siehe Gerätebeschreibung) Auf die korrekte Einstellung der Schnittstellenparameter (Baudrate, Paritybit, Stopbits ...) ist zu achten. Nach POWER-ON werden die RS232-Schnittstellen mit den im ZP-RAM (Laufwerk R) gespeicherten Parameter initialisiert.

RS232/1 mit Parameter 50...59 und

RS232/2 mit Parameter 90...99.

RS232/3 mit Parameter 40...49 (Option F6).

Parameter 100 definiert welche Schnittstelle als Terminal-schnittstelle verwendet wird (standard ist die RS232/1).

Beachte:

RTS Hardware-Handshake-Leitung der RS232-Schnittstelle (Ausgang);(1 = Empfänger bereit, 0 = Empfänger nicht bereit).

CTS Hardware-Handshake-Leitung der RS232-Schnittstelle (Eingang);(1 = Sender freigeben, 0 = Sender sperren).

OUTPUT-Ready Statusbit 0 der Systemvariable STSCTR#0 bzw. STSCTR#2 bzw. STSCTR#4

INPUT-Ready Statusbit 1 der Systemvariable STSCTR#0 bzw. STSCTR#2 bzw. STSCTR#4

➡ Die RS232/3 kann neben dem ASCII-Protokoll auch mit dem Protokoll 3964 (R) betrieben werden (siehe P41, P42, P43).

23.2 Parameter der RS232/1-Schnittstelle (#0)

Nr.	Funktion	Standard	
50	Geräte-Adresse 0 - 255	0	
51	Empfangs-Start-Zeichen 0 - 255	2	
52	Empfangs-Ende-Zeichen 0 - 255 (für COMPAX ist der Wert 62 erforderlich)	13	
53	Empfangs-Protokoll	34	
	Bit	Funktion	
	0	ProtokollBit 0	0
	1	ProtokollBit 0	1
	2	ProtokollBit 0	0
	3	-	0
	4	Block-Check-Character; 0=off, 1=on	0
	5	Auto-RTS; 0 = off, 1 = on	1
	6	Hardware-Handshake; 0 = off, 1 = on	0
	7	Software-Handshake; 0 = off, 1 = on	0
	Mit den ProtokollBits wird eines der folgenden Empfangsprotokolle ausgewählt:		
	P-Bit	INPUT-Rdy=1,	
	2 1 0		0 1 0
	0 0 0	nach jedem empfangenen Zeichen	
	0 0 1	nach def. Anzahl empf. Zeichen (P56)	
0 1 0	nach empfangenem Endezeichen (52)		
0 1 1	nach empf. Start(P51)- u. Endezeichen (P52)		
1 0 0	nach empfangenen Startzeichen(P51), Moduladresse(P50) und Empfangs-Ende-Zeichen(P52)		
54	Baudrate 0=150 3=1200 6=9600 9=57600* 1=300 4=2400 7=19200 10=76800* 2=600 5=4800 8=38400 11=115200*	8	
55	Parity und Stopbits 0 = ohne Parity, 1 Stopbit 1 = ohne Parity, 2 Stopbit 2 = mit Parity EVEN, 1 Stopbit 3 = mit Parity ODD, 1 Stopbit	0	
56	Anzahl zu empfangender Zeichen 1 - 255	1	
57	Sende-Endezeichen 0 - 255	13	
58	Sende-Protokoll	1	
	0: es wird kein Endezeichen automatisch gesendet		
	1: es wird CR automatisch gesendet		
	2: es wird CR LF automatisch gesendet		
	3: es wird das Sende-/ Endez.(P57) autom. gesendet		
	4: es wird das CR LF und das Sende-/ Endez.(P57) autom. gesendet		
5: es wird das Empfang-Startz. (P51) und das Empfang-Endez. (P52) autom. gesendet			
59	Timeout-Wert 0 - 255 (1 = 0.1s) 0 = keine Timeout-Überwachung	0	

* Die Einstellungen 9, 10, 11 der Parameter P54 (RS232/1) und P94 (RS232/1) gelten für eine Geräte-Taktfrequenz von 29,4912MHz.
Für ältere Geräte mit geringerer Taktfrequenz gelten folgende Baudrate:

	P54/P94=9	P54/P94=10	P54/P94=11
11,0592MHz			
RS232/1	57600	57600	115200
RS232/2	57600	57600	57600
24,5760MHz			
RS232/1	38400	76800	76800
RS232/2	38400	76800	76800

Die Geräte-Taktfrequenz kann mit dem Befehl XTAL ermittelt werden.

23.3 Parameter der RS232/2-Schnittstelle (#2)

Nr.	Funktion	Standard
90	Geräte-Adresse 0 - 255	0
91	Empfangs-Start-Zeichen 0 - 255	2
92	Empfangs-Ende-Zeichen 0 - 255	13
93	Empfangs-Protokoll	34
	Bit Funktion	
	0 ProtokollBit 0	0
	1 ProtokollBit 1	1
	2 ProtokollBit 2	0
	3 -	0
	4 Block-Check-Character; 0 = off, 1 = on	0
	5 Auto-RTS; 0 = off, 1 = on	1
	6 Hardware-Handshake; 0 = off, 1 = on	0
	7 Software-Handshake; 0 = off, 1 = on	0
	Mit den ProtokollBits wir eines der folgenden Empfangsprotokolle ausgewählt.	
	P-Bit INPUT-Rdy = 1,	
	2 1 0	0 1 0
	0 0 0 nach jedem empfangenen Zeichen	
	0 0 1 nach def. Anzahl empfangener Zeichen (P96)	
	0 1 0 nach empfangenem Endezeichen (P92)	
	0 1 1 nach empf. Start(P91)- und Endezeichen (P92)	
	1 0 0 nach empfangenen Startzeichen(P91), Moduladresse(P90) und Empfangs-Endezeichen(P92)	
94	Baudrate 3=1200 6=9600 9=57600* 4=2400 7=19200 10=76800* 5=4800 8=38400 11=115200*	8
95	Parity und Stopbits 0 = ohne Parity, 1 Stopbit 1 = ohne Parity, 2 Stopbit 2 = mit Parity EVEN, 1 Stopbit 3 = mit Parity ODD, 1 Stopbit	0
96	Anzahl zu empfangender Zeichen 1 - 255	1
97	Sende-Endezeichen 0 - 255	13
98	Sende-Protokoll 0: es wird kein Endezeichen automatisch gesendet 1: es wird CR automatisch gesendet 2: es wird CR LF automatisch gesendet 3: es wird das Sende-/ Endez.(P97) autom. gesendet 4: es wird das CR LF und das Sende-/ Endez.(P97) autom. gesendet 5: es wird das Empfang-Startz. (P91) und das Empfang-Endez. (P92) autom. gesendet	1
99	Timeout-Wert 0 - 255 (1 = 0.1s) 0 = keine Timeout-Überwachung	0
100	Terminal-Schnittstellenzuordnung 0 = RS232/1 ist die Terminalschnittstelle 1 = RS485/1 ist die Terminalschnittstelle 2 = RS232/2 ist die Terminalschnittstelle	0

* siehe Seite 83.

23.4 Parameter der RS232/3-Schnittstelle (#4) (Option F6)

Nr.	Funktion	Standard
40	Geräte-Adresse 0 - 255	0
41	Empfangs-Start-Zeichen 0 - 255 bzw. Zeichenverzugszeit bei 3964-Prozedur (1↔50ms)	2
42	Empfangs-Ende-Zeichen 0 - 255 bzw. Quittungsverzugszeit bei 3964-Prozedur (1↔50ms)	13
43	Empfangs-Protokoll	34
	Bit Funktion	
	0 ProtokollBit 0	0
	1 ProtokollBit 1	1
	2 ProtokollBit 2	0
	3 -	0
	4 Block-Check-Character; 0=off, 1=on	0
	5 Auto-RTS; 0=off, 1=on	1
	6 Hardware-Handshake; 0 = off, 1 = on	0
	7 Software-Handshake; 0 = off, 1 = on	0
	Mit den ProtokollBits wir eines der folgenden Empfangsprotokolle ausgewählt.	
	P-Bit INPUT-Rdy = 1,	
	2 1 0	0 1 0
	0 0 0 nach jedem empfangenen Zeichen	
	0 0 1 nach def. Anzahl empfangener Zeichen (P46)	
	0 1 0 nach empfangenem Endezeichen (P42)	
	0 1 1 nach empf. Start (P41)- und Endezeichen (P42)	
	1 0 0 nach empfangenen Startzeichen(P41), Moduladresse(P40) und Empfangs-Endezeichen(P42)	
	1 0 1 3964-Prozedur	
44	Baudrate 0=150 3=1200 6=9600 9=57600 1=300 4=2400 7=19200 10=76800 2=600 5=4800 8=38400 11=115200	6
45	Parity / Stopbits / Zeichenlänge	0
	Bit 0 Anzahl der Stopbits: 0 = 1 Stopbit 1 = 2 Stopbit	
	Bit 1 Parity Enable: 0 = Disable 1 = Enable	
	Parity Select: Bit 2/ 0/0 = Odd 1/0 = Even Bit 3 0/1 = Mark 1/1 = Space	
	Zeichenlänge: Bit 4/ 0/0 = 8Bit 1/0 = 7Bit Bit 5 0/1 = 6Bit 1/1 = 5Bit	
	Bit 6 ohne Funktion	
	Bit 7 ohne Funktion	
46	Anzahl zu empfangender Zeichen 1 - 255	1
47	Sende-Endezeichen 0 - 255 bzw. Wiederholfaktor bei 3964-Prozedur	13

48	Sende-Protokoll 0: es wird kein Endezeichen automatisch gesendet 1: es wird CR automatisch gesendet 2: es wird CR LF automatisch gesendet 3: es wird das Sende-/ Endezeichen (P97) automatisch gesendet 4: es wird das CR LF und das Sende-/ Endezeichen (P97) automatisch gesendet 5: es wird das Empfang-Startzeichen (P91) und das Empfang-Endzeichen (P92) automatisch gesendet	1
49	Timeout-Wert 0 - 255 (1 = 0.1s) 0 = keine Timeout-Überwachung	0

23.5 Empfangen

Die Empfänger der beiden RS232-Schnittstellen verfügen über je einen 256 Byte großen Ringpuffer und je einen Schreib- und Lese-Zeiger.

Alle ankommenden Zeichen werden in Abhängigkeit vom eingestellten Empfangsprotokoll Parameter 53 (bzw. P93/P43) ausgewertet und gegebenenfalls im Ringpuffer gespeichert.

Wurden alle Zeichen so empfangen, wie es im Empfangsprotokoll definiert ist, wird das Bit INPUT-Ready = 1. Jetzt kann mit der Basic-Anweisung ENTER das Zeichen oder die Zeichenkette aus dem Ringpuffer abgeholt werden. Die ENTER Anweisung kopiert das Zeichen oder die Zeichenkette in den String \$(#0) \$(#2) bzw. \$(#4) und kann nun im Basic weiter bearbeitet werden.

Die ENTER Anweisung löscht das Bit INPUT-Ready, wenn kein weiteres Zeichen oder eine weitere Zeichenkette im Ringpuffer bereit steht.

Ist INPUT-Ready = 0 wartet die ENTER-Anweisung bis INPUT-Ready = 1 ist oder bricht die ENTER-Anweisung ab, wenn die programmierte Timeout-Zeit in Parameter 59 (bzw. P998/P49) abgelaufen ist.

23.6 Senden

COMTAC-BASIC leitet den durch die Basic-Anweisung OUTPUT definierten Ausgabestring in den 256 Byte großen Ausgabepuffer.

Eine im Hintergrund ablaufende Ausgaberroutine sendet Zeichen für Zeichen über die Schnittstelle an den Empfänger.

Mit Parameter 58 (bzw. P98/P48) kann festgelegt werden, ob nach dem Senden einer Zeichenkette automatisch noch Endezeichen gesendet werden.

Das Bit OUTPUT-Ready (STSCTR#0, STSCTR#2 bzw. STSCTR#4) bleibt solange 0, bis der String vollständig ausgegeben ist; d.h. der Ausgabepuffer ist leer, es kann der nächste String ausgegeben werden.

Der Sendevorgang kann durch Hard- und Software-Handshake angehalten und wieder fortgesetzt werden.

Hardware-Handshake

CTS = 0 Sender ist gesperrt

CTS = 1 Sender ist frei

Software-Handshake

Empfänger sendet X-OFF (13H) Sender wird gesperrt.

Empfänger sendet X-ON (11H) Sender wird freigegeben.

23.7 BCC (RS232)

Der Block-Check-Character dient zur Verbesserung der Übertragungssicherheit und kann wahlweise über ein Bit Parameter 53 (bzw. P93/P43) ein-/ausgeschaltet werden. Der BCC wird immer als letztes Zeichen eines Datenblocks gesendet; also nach dem Endezeichen (wenn vorhanden).

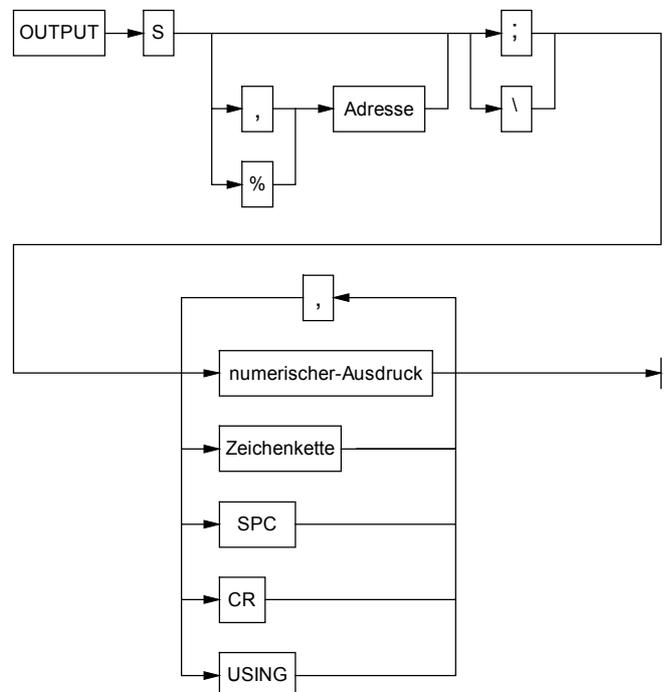
Der Block-Check-Character wird gebildet aus einer XOR-Verknüpfung (byteweise) aller gesendeten bzw. empfangenen Zeichen eines Datenblocks exklusiv des/der Endezeichen.

Ist BCC-Überwachung eingeschaltet ermittelt COMTAC beim Empfang von Zeichen automatisch den BCC und vergleicht diesen mit dem empfangenen BCC.

Ist der Vergleich negativ wird eine Fehlermeldung ausgegeben.

Beim Senden bildet COMTAC ebenfalls automatisch den BCC und gibt diesen als letztes Zeichen aus.

23.8 OUTPUT [O.] (RS232)



Funktion:

Diese Anweisung gibt Zeichen, Zeichenketten und/oder den Wert eines numerischen Ausdrucks über die angegebene RS232-Schnittstelle aus.

OUTPUT 0... Ausgabe über RS232/1-Schnittstelle

OUTPUT 2... Ausgabe über RS232/2-Schnittstelle

OUTPUT 4... Ausgabe über RS232/3-Schnittstelle

Optional kann sofort die zu erwartenden Antwort in den String \$(#0) eingelesen werden. Dafür muß in der Befehls-Syntax der OUTPUT-Anweisung das Semikolon (;) durch den Backslash (\) ersetzt werden.

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	0 - 255	Geräteadr. des Empfängers
S	Zahl	0,2 oder 4	0 RS232/1-Schnittstelle 2 RS232/2-Schnittstelle 4 RS232/3-Schnittstelle

Optional ist es möglich eine Geräteadresse anzugeben. Durch den Befehlssyntax wird selektiert, in welcher Form die Adresse ausgegeben wird:

- ◆ Komma vor der Adresse: ASCII-Format
- ◆ Punkt vor der Adresse: Binär-Format

Beispiel: OUTPUT 0; "WINKEL=",W
OUTPUT 2,5; "SPA",XPOS,CHR\$(10)

Anmerkung:

Das Betriebssystem prüft nun selbständig das OUTPUT-Ready-Flag.

Wie bei der ENTER-Anweisung wird nach Ablauf der Timeout-Zeit der Befehl abgebrochen.

Nur dann ist der Ausgabepuffer der entsprechenden RS232-Schnittstelle leer.

Mit Parameter 58 (bzw. P98/P48) kann festgelegt werden, ob nach dem Senden einer Zeichenkette automatisch noch Endezeichen gesendet werden.

Die Funktion der Anweisungen SPC, CR, USING finden Sie im Kapitel Terminal/LCD-Ausgabe.

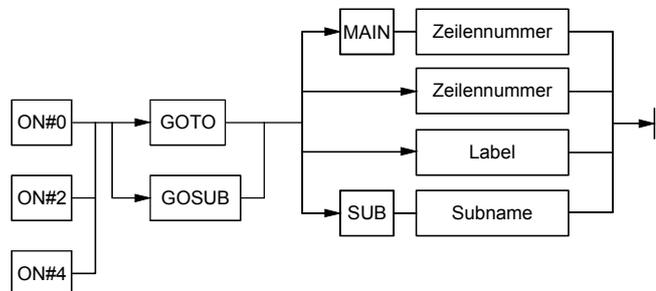
Nach Ausführung der Anweisung ist
\$(#0) = "10V200X-30"
A = 10
B = 200
C = -30 und D unverändert.

Anmerkung:

Befindet sich keine bereitstehende Zeichenkette im Ringpuffer der seriellen Schnittstelle (INPUT-Ready = 0) dann wartet COMTAC-BASIC bis eine Zeichenkette bereitsteht bzw. bis die Timeout-Zeit Parameter 59 (bzw. P99/P49) abgelaufen ist.

Wird die Timeout-Zeit überschritten generiert COMTAC eine Fehlermeldung.

23.10 ON#0 / #2 /#4



Funktion:

Diese Anweisung definiert eine Programmverzweigung, die durchgeführt wird, wenn ein String im Zwischenpuffer der angegebenen RS232-Schnittstelle für die weitere Verarbeitung bereitsteht (INPUT-Ready = 1) und dieser Interrupt freigegeben ist. (Siehe dazu ENABLE/DISABLE ON#0 bzw. ON#2).

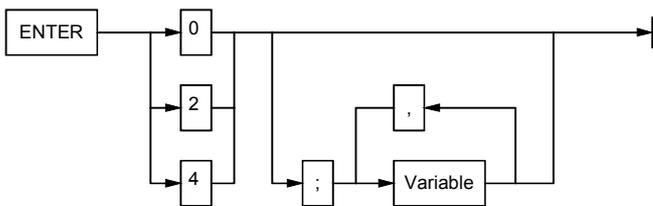
Parameter	Angabe	Bereich	Beschreibung
Zeilen-Nr.	Zahl	0 - 65535	vorhandene Zeilennummer im Programm

Beispiel: ON#0 GOTO 100
ON#2 GOSUB 150

Anmerkung:

Mit Parameter 53 (bzw. P93/P43) wird festgelegt, wann ein String im Zwischenpuffer zur Übernahme bereitsteht.

23.9 ENTER [E.] (RS232)



Funktion:

Diese Anweisung übergibt ein bereitstehendes Zeichen oder eine Zeichenkette aus dem Ringpuffer der RS232-Schnittstelle in den String \$(#0), \$(#2) bzw. \$(#4) zur weiteren Verarbeitung.

Enthält die einzulesende Zeichenkette eine oder mehrere Zahlen, können diese Zahlen mit der ENTER-Anweisung direkt Variablen zugewiesen werden, indem an die ENTER-Anweisung ein Variablenname bzw. eine Variablennamen Liste angehängt wird.

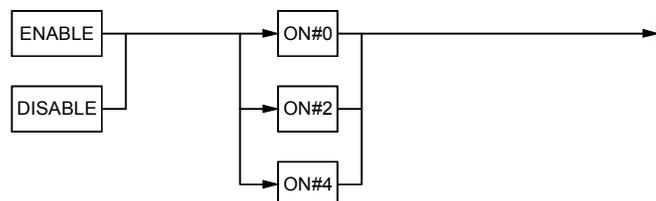
COMTAC-BASIC weist die erste Zahl die sich in der Zeichenkette befindet der ersten Variablen der Variablennamen-Liste zu, die zweite Zahl der zweiten Variable und so weiter.

Die Anzahl der Zahlen in der Zeichenkette und die Anzahl der Variablennamen muß nicht identisch sein.

Beispiel:

Bereitstehender String = "10V200X-30"
ENTER 0; A,B,C,D

23.11 ENABLE [EN.]/DISABLE [DI.] RS232



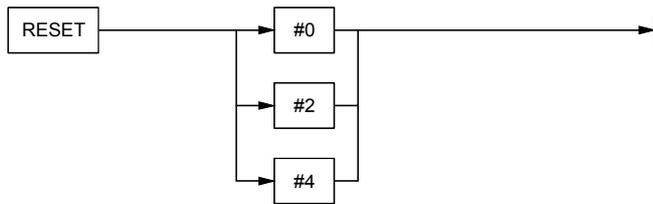
Funktion:

Freigabe/Sperren des RS232/1, RS232/1 bzw. RS232/2 - Interrupt.

Beispiel:

ENABLE ON#0 (gibt den RS232/1-Interrupt frei)
 DISABLE ON#2 (sperrt den RS232/2 - Interrupt)

23.12 RESET [RS.] RS232

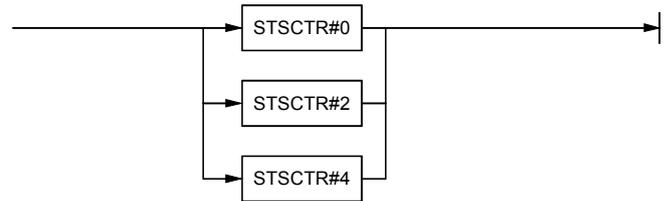


Funktion:

Diese Anweisung setzt die RS232/1 bzw. /2 oder /4-Schnittstelle in den Power-On Zustand zurück.

Beispiel: RESET #0
 RESET #2

23.13 STSCTR#0 /#2 /#4



Funktion:

Diese speziellen Systemvariablen liefern die Statusinformation der RS232-Schnittstellen. Die einzelnen Bits haben folgende Bedeutung:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
BCC Error	Parity Error	Overrun Error	Framing Error	CTS	RTS	INPUT Ready	OUTPUT Ready
128	64	32	16	8	4	2	1

- Bit 0: der Ausgabepuffer ist leer; es kann ein String ausgegeben werden.
- Bit 1: im Zwischenpuffer steht ein String für die weitere Verarbeitung bereit und kann mit ENTER eingelesen werden.
- Bit 2: Logischer Zustand der Handshake-Leitung RTS.
- Bit 3: Logischer Zustand der Handshake-Leitung CTS.
- Bit 4: Framing Fehler festgestellt
- Bit 5: Eingangspuffer überlaufen
- Bit 6: es ist ein Parity-Fehler festgestellt worden.
- Bit 7: es ist ein BCC-Fehler festgestellt worden.

Beispiel: PB.STSCTR#0

IF (STSCTR#2 'AND' 10B) = 0 THEN

STSCTR#2 = STSCTR#2 'OR' 4 : REM RTS setzen

Anmerkung:

Mit einer entsprechenden Zuweisung (STSCTR#0 = x bzw. STSCTR#2 = x) kann der Zustand der Handshake-Leitung RTS verändert werden. Die Fehlerbits (BCC-, Parity-, Timeout-Error) werden durch das Lesen von STSCTR#0 /#2 /#4 automatisch gelöscht.

24. DATA-TEXT-FELD

24.1 Allgemeines

Zum Ablegen von Daten und/oder Text hat COMTAC-Basic im Arbeitsspeicher einen 4096 Byte großen Bereich reserviert.

Der Anwender kann hier z.B. Text für die Bedienungsführung speichern und/oder Werkstückdaten oder Werkzeugparameter.

Mit dem Data-Text-Feld können Daten an andere Programme übergeben werden.

Durch die Anweisungen NEW, CLEAR oder RUN wird das Data-Text-Feld nicht gelöscht.

Das Data-Text-Feld kann als Datenfile im ZP-RAM (Laufwerk R) oder auf Diskette abgespeichert und ebenso wieder geladen werden.

Zum Beschreiben und Lesen des Daten-Text-Feldes bietet COMTAC-Basic einen speziellen Editor und eine Vielzahl von Anweisungen, die im folgenden beschrieben werden.

24.2 Data-Text-Editor: DTFEDIT



Funktion:

Data-Text-Editor aufrufen.

Der Editor kann im Command-Mode auch durch Drücken der Taste Ctrl+x aufgerufen werden.

Der Editor prüft, ob eine Zeileneinteilung des Data-Text-Feldes bereits gemacht wurde.

Ist dies der Fall, wird der aktuelle Inhalt des Data-Text-Feldes angezeigt und kann editiert werden.

Alle Zeichen die das Terminal nicht darstellen kann erscheinen auf dem Bildschirm als ~ Zeichen.

Fehlt die Zeileneinteilung, muß die gewünschte Zeilenlänge eingegeben werden und der Editor initialisiert das Data-Text-Feld wie folgt: Die ersten 4000 Byte werden durch die Zeilenlänge geteilt und so die Anzahl der möglichen Zeilen ermittelt.

Achtung!

Die Zeilen werden anschließend mit Leerzeichen gefüllt, der vorhandene Inhalt des Data-Text-Feldes geht verloren.

Die folgenden Angaben sind für den Anwender wichtig, der sowohl Text als auch Daten im Data-Text-Feld speichern möchte.

Der Speicherplatz der ersten Zeile beginnt mit dem Byte 0 des Data Text-Feldes.

Die eingegebene Zeilenlänge wird im Byte 4000, die Anzahl der Zeilen in den Bytes 4001 und 4002 (H-L) gespeichert.

Die Bytes 4003-4015 werden vom Editor für Berechnungen während dem Editieren verwendet.

Die Kopfzeile oder Headline wird in den Bytes 4016-4095 abgelegt.

24.3 DTFLLEN



Funktion:

Mit dieser Systemvariablen kann die Zeilenlänge (Anzahl der Zeichen/Zeile) des Data-Text-Feldes geändert bzw. ausgelesen werden.

Beispiel: DISP DTFLLEN
DTFLLEN = 60

24.4 DTFLCNT



Funktion:

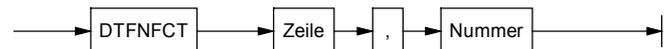
Mit dieser Systemvariablen kann die Zeilenanzahl des Data-Text-Feldes ausgelesen werden.

Beispiel: DISP DTFLCNT

Anmerkung:

Die Anzahl der Zeilen des Data-Text-Feldes ist nur indirekt über DTFLLEN (Anzahl Zeichen/Zeile) veränderbar.

24.5 DTFNFCT



Funktion:

Auslesen der n-ten Zahl aus der angegebenen Zeile.

Wird in der angegebenen Zeile keine Zahl gefunden, übergibt die Funktion den Wert 0.

Als Trennzeichen zwischen 2 Zahlen gilt jedes ASCII-Zeichen, das keine Ziffer, Komma oder Dezimalpunkt ist (Komma = Dezimalpunkt).

Parameter	Angabe	Bereich	Beschreibung
Zeile	num. Ausdr.	1...	Zeile des Data-Text-Feldes
Nummer	num. Ausdr.		n-te Zahl in der Zeile

Beispiel: DISP DTFNFCT 10,3
A = DTFNFCT Y,N

Mit dem Text "3000;80;5480;4700;988" in der Zeile 8 des Data-Text-Feldes, Y = 8 und N = 2 wird der Variablen A der Wert 80 zugewiesen.

24.6 DTFVFCT



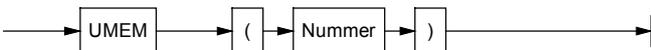
Funktion:
Weist der in der angegebenen Zeile des Data-Text-Felds stehenden Variablen den unmittelbar folgenden Zahlenwert zu.

Parameter	Angabe	Bereich	Beschreibung
Zeile	num. Ausdr.	1...	Zeile des Data-Text-Feldes

Beispiel: DTFVFCT 15
DTFNFCT Y
Mit dem Text "B-3413 X88 Z250 Q9000" in der Zeile 15 des Data-Text-Feldes, und Y = 15 wird der Variablen B die Zahl - 3413, der Variablen X die Zahl 88, der Variablen Z die Zahl 250 und der Variablen Q die Zahl 9000 zugewiesen.

Anmerkung:
Diese Funktion ist nur mit Variablenamen möglich die aus einem großen Buchstaben bestehen.

24.7 UMEM(x) [;(x)]



Funktion:
Die indizierte Variable UMEM(x) benutzt das Data-Text-Feld als Speicherplatz. Jedes Elemente dieser Matrize benötigt 6 Byte zum Speichern einer Fließkommazahl. Demzufolge stehen dem Anwender $4096/6 = 682$ Elemente der Variable UMEM zur Verfügung, UMEM(0) bis UMEM(681).

Parameter	Angabe	Bereich	Beschreibung
Nummer	num. Ausdr.	0 - 681	Nummer des Elements (Index)

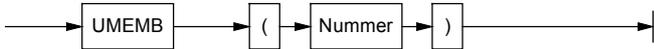
Die allgemeine Formel zur Ermittlung der Adresse **a** des ersten Bytes des Elements UMEM(x) im Data-Text-Feld lautet:
 $a = x * 6$

Beispiel: UMEM(x)=w
UMEM(A)=B
Mit A = 3 und B = 5233 werden im Data-Text-Feld die Bytes $x*6 = 18$ bis 23 mit dem Wert 5233 im Fließkommaformat beschrieben.

Anmerkung:
Datenfiles die im ZP-RAM (Laufwerk R) abgelegt sind, können mit der Anweisung UMEM Dateiname (Index) direkt verändert und gelesen werden.
Durch das Einfügen des Dateinamen nach UMEM wird das entsprechende Datenfile angewählt.
Der Dateiname wird direkt in 'Hochkomma' oder indirekt durch ';(x)' angegeben.
Die Namensweiterung ".DAT" wird nicht mit angegeben.

Beispiel: DISP UMEM "TEST" (105)
UMEM ;\$(1) (333) = 7000
A = UMEM "W_DATEN" (200)

24.8 UMEMB(x) [;B(x)]



Funktion:
Die indizierte Byte-Variable UMEMB(x) benutzt das Data-Text-Feld als Speicherplatz. Demzufolge stehen dem Anwender 4096 Elemente der Variable UMEMB(x) zur Verfügung, UMEMB(0) bis UMEMB(4095). UMEMB(0) belegt Byte 0 des Data-Text-Feldes, UMEMB(1) das Byte 1 usw.

Parameter	Angabe	Bereich	Beschreibung
Nummer	num. Ausdr.	0 - 4095	Nummer des Elements (Index)

Beispiel: UMEMB(x)=w
UMEMB(SP) = SPEED
Mit SP = 8 und SPEED = 123 wird im Data-Text-Feld das Byte 8 mit der Zahl 123 belegt.

Anmerkung:
Datenfiles die im ZP-RAM (Laufwerk R) abgelegt sind, können mit der Anweisung UMEMB Dateiname (Index) direkt verändert und gelesen werden.
Durch das Einfügen des Dateinamen nach UMEMB wird das entsprechende Datenfile angewählt.
Der Dateiname wird direkt in 'Hochkomma' oder indirekt durch ';(x)' angegeben.
Die Namensweiterung ".DAT" wird nicht mit angegeben.

Beispiel: PRINT UMEMB "DATEN" (5)
UMEMB ;\$(8) (33) = 10
Z = UMEM "ZEICHEN" (4002)

24.9 UMEM\$(Zeile) [;(Zeile)]



Funktion:
Die String-Variable UMEM\$(Zeile) gibt dem Anwender die Möglichkeit jede Zeile des Data-Text-Feldes wie einen String zu behandeln; d.h. alle COMTAC-Basic Befehle für die Stringverarbeitung sind anwendbar.
Die aktuelle Länge des UMEM\$(Zeile)-Strings ist durch die Zeilenlänge definiert und wird durch eine Stringzuweisung nicht verändert.

Parameter	Angabe	Bereich	Beschreibung
Nummer	num. Ausdr.	1 - 4000	Nummer des Elements (Index)

Beispiel: UMEM\$(y)=String
UMEM\$(Y) = \$(8)
Mit Y = 15 und \$(8) = "HAUSER" werden im Data-Text-Feld die ersten 6 Byte in der Zeile 15 mit dem Text "HAUSER" überschrieben.
Ist die Zeile länger als 6 Zeichen, bleiben die restlichen Bytes der Zeile unverändert.

Anmerkung:

Datenfiles die im ZP-RAM (Laufwerk R) abgelegt sind, können mit der Anweisung UMEM\$ Dateiname (Zeile) direkt verändert und gelesen werden.

Durch das Einfügen des Dateinamen nach UMEM\$ wird das entsprechende Datenfile angewählt.

Der Dateiname wird direkt in 'Hochkomma' oder indirekt durch '\$(x)' angegeben.

Die Namensweiterung ".DAT" wird nicht mit angegeben.

Beispiel: DISP UMEM\$ "TEXT" (7)
 UMEM\$;\$(13) (33) = "ACHTUNG"

**24.10 UMEM\$(Adresse,Länge)
 [;\$(Adresse,Länge)]**



Funktion:

Die String-Variable UMEM\$(Adresse,Länge) gibt dem Anwender die Möglichkeit das Data-Text-Feld als individuellen Stringspeicher zu verwenden.

Durch die Angabe der Adresse und Länge im Data-Text-Feld ist der String definiert und kann als solcher behandelt werden; d.h. alle COMTAC-Basic Befehle für die Stringverarbeitung sind anwendbar.

Die aktuelle Länge des UMEM\$(Adresse,Länge)-Strings ist durch die Angabe der Länge definiert und wird durch eine Stringzuweisung nicht verändert.

Beispiel:

UMEM\$(a,l)=String

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	0 - 4095	Adresse im Data-Text-Feld
Länge	num. Ausdr.	1 - 255	Länge des Strings

Beispiel: DISP UMEM\$(10,4)
 UMEM\$(A,L)=\$(4)

Mit A = 30, L = 20 und \$(4) = "COMTAC 2000" werden im Data-Text-Feld die Bytes 30-40 durch den zugewiesenen Text COMTAC 2000 überschrieben, die restlichen 9 Bytes (41-49) des Strings UMEM\$(30,20) bleiben unverändert.

Anmerkung:

Datenfiles die im ZP-RAM (Laufwerk R) abgelegt sind, können mit der Anweisung UMEM\$ Dateiname (Zeile) direkt verändert und gelesen werden.

Durch das Einfügen des Dateinamen nach UMEM\$ wird das entsprechende Datenfile angewählt.

Der Dateiname wird direkt in 'Hochkomma' oder indirekt durch '\$(x)' angegeben.

Die Namensweiterung ".DAT" wird nicht mit angegeben.

Beispiel: DISP UMEM\$ "TXT001" (0,10)
 UMEM\$;\$(25) (10,10) = "BETRIEB"

25. Fehlerbehandlung

25.1 ERRSTS [ES.]



Funktion:
Diese spezielle Systemvariable liefert die Fehler-Nummer des zuletzt aufgetretenen Fehlers.
Bedeutung der Fehlernummern: siehe nächste Seite.

Beispiel: PRINT ERRSTS
A = ERRSTS

Anmerkung:
ERRSTS kann kein Wert zugewiesen werden.

25.2 ERRSTS\$



Funktion:
Diese spezielle Systemvariable liefert die Fehler-Meldung des zuletzt aufgetretenen Fehlers.

Beispiel: PRINT ERRSTS\$
OUTPUT 2,4;ERRSTS\$

Anmerkung:
ERRSTS\$ kann nur in Verbindung mit einem PRINT-, DISP-, oder OUTPUT-Befehl verwendet werden.

25.3 ERRSTSL



Funktion:
Diese spezielle Systemvariable liefert die Zeilen-Nummer der Basic-Zeile, in der ein Fehler festgestellt wurde.

Beispiel: PRINT ERRSTSL
A = ERRSTSL

Anmerkung:
ERRSTSL kann kein Wert zugewiesen werden.

25.4 ERRSTS#



Funktion:

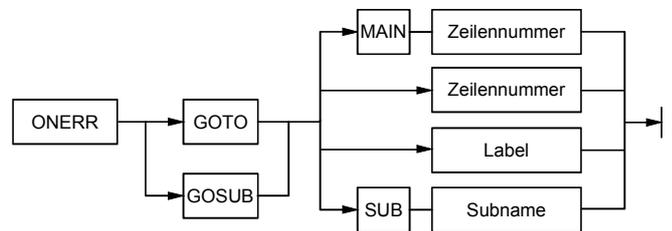
Diese spezielle Systemvariable liefert den Namen des SUBs, in dem ein Fehler festgestellt wurde.

Beispiel: PRINT ERRSTS#
A = ERRSTS#

Anmerkung:

ERRSTS# kann kein Wert zugewiesen werden.

25.5 ONERR [OE.]



Funktion:

Diese Anweisung definiert eine Programmverzweigung, die durchgeführt wird, wenn ein Fehler während der Programmarbeitung erkannt wird und der ONERR-Interrupt freigegeben ist.

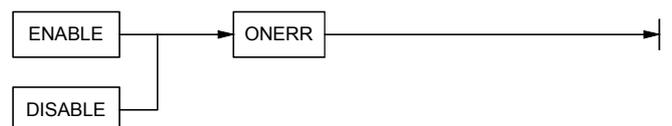
Parameter	Angabe	Bereich	Beschreibung
Zeilen-Nr.	Zahl	0 - 65535	vorhandene Zeilen-Nr im Programm

Beispiel: ONERR GOTO 200
ONERR GOSUB 700

Anmerkung:

ONERR verzweigt unmittelbar nach dem Erkennen eines Fehlers zur angegebenen Zeilen-Nummer.

25.6 ENABLE [EN.]/DISABLE [DI.] ONERR [OE.]



Funktion:

Freigabe/Sperren des ONERR-Interrupt.

Beispiel: ENABLE ONERR
DISABLE ONERR

25.7 Fehlerbeschreibung

1 Bad Syntax

Befehlssyntax ist nicht korrekt.

2 Renumber not possible

Der Editor kann die angegebene Neunumerierung der Zeilen nicht durchführen.

3 Copy not possible

Der Editor kann die angegebenen Zeilen nicht an die spezifizierte Stelle kopieren.

6 No Data

Es wurde eine READ-Anweisung gegeben aber keine DATA-Anweisung gefunden oder alle DATA-Anweisungen wurden bereits gelesen und keine RESTORE-Anweisung wurde gegeben.

7 Can't continue

Das Programm kann nicht fortgesetzt werden wenn ein Fehler erzeugt wurde oder das Programm editiert wurde.

8 Memory Allocation

Der Programm- oder Datenspeicher ist zu klein.

9 A-Stack

Es wurden zu viele Werte auf den Argument-Stack gelegt (PUSH) oder es ist kein Wert auf dem Argument-Stack (POP).

10 C-Stack

Diese Fehlermeldung wird generiert, wenn

- ◆ eine RETURN-Anweisung bearbeitet werden soll
- ◆ bevor eine GOSUB-Anweisung bearbeitet wurde.
- ◆ eine WHILE- oder UNTIL-Anweisung bearbeitet werden soll bevor eine DO-Anweisung bearbeitet wurde.
- ◆ eine NEXT-Anweisung bearbeitet werden soll bevor eine FOR-Anweisung bearbeitet wurde.
- ◆ keine freies Byte mehr auf dem C-Stack vorhanden ist. Der C-Stack hat 254 Byte Speicherplatz. Jede FOR...NEXT-Schleife benötigt 18 Byte des C-Stack, jede DO...WHILE-, DO...UNTIL-Schleife benötigt 4 Byte des C-Stack und jede GOSUB-Anweisung benötigt ebenfalls 4 Byte des C-Stack. COMTAC-Basic kann demnach max. 14 verschachtelte FOR...NEXT-Schleifen bearbeiten.

11 I-Stack

Betriebssystem ist ausgefallen

12 Array Size

Es wurde ein Element einer indizierten Variable angesprochen, das nicht dimensioniert ist.

13 Invalid Line Number

Es wurde eine Programmverzweigung an einer nicht existierenden Zeilen-Nummer durchgeführt.

14 Transter Parity. Fehler beim Senden/Empfangen von Floppy-Daten.

15 Floppy-interface not defined. Die Floppy-Schnittstelle ist nicht definiert (Parameter 10).

16 Floppy report. Fehlermeldung vom Floppylaufwerk.

18 No F-Bus Master. Comtac ist nicht als Feldbus-Master konfiguriert (Parameter 73).

19 This is no COMPAX

Es wurde eine CPX-Anweisung angewendet für ein Feldbus-Gerät das COMTAC nicht als COMPAX erkannt hat.

20 F_Bus address missing

Das angesprochene Feldbusgerät befindet sich nicht am Bus.

21 F_Bus device alert

Das angesprochene Feldbusgerät meldet ALARM.

22 F_Bus device missing

Das angesprochene Feldbusgerät ist ausgefallen oder die Verbindung zu diesem Gerät ist unterbrochen.

23 COMTAC-Slave hardware failure

Das Slave-Prozessorsystem des COMTAC hat einen Systemfehler.

24 FBUS timeout

Das angesprochene Feldbusgerät hat die eingestellte Timeout-Zeit überschritten.

25 F_Bus device no data

Das angesprochene Feldbusgerät hat keine Daten bereitgestellt, die mit der ENTER-Anweisung übernommen werden können.

26 F_Bus device no command

Das angesprochene Feldbusgerät ist nicht bereit für den Empfang von Daten.

27 F_Bus device error

Das angesprochene Feldbusgerät meldet FEHLER.

28 Parenthesis missing

Es wurde bei der Eingabe eine öffnende oder schließende Klammer vergessen.

29 Quotation mark missing

Es wurde bei der Eingabe ein Anführungszeichen vergessen.

30 Bad Argument

Es wurde ein unzulässiger Wert in einer Anweisung verwendet oder einer Variablen ein unzulässiger Wert zugewiesen.

31 Divide by Zero

Es trat eine Division durch 0 auf.

32 Arith. underflow

Das Ergebnis einer arith. Operation ist kleiner als $\pm 1 \cdot 10^{-127}$.

33 Arith. overflow

Das Ergebnis einer arith. Operation ist größer als $\pm .99999999 \cdot 10^{127}$.

34 File already exists. Delete

Das abzuspeichernde Daten/Programm-File ist unter dem angegebenen Namen bereits auf dem Speichermedium vorhanden.

35 File not found

Das angegebene Daten/Programm-File ist auf dem Speichermedium nicht vorhanden.

36 Insufficient space

Für das angegebene Daten/Programm-File ist nicht mehr ausreichend Platz auf dem Speichermedium.

37 No disk in drive

Es ist keine Diskette im Laufwerk

38 Write protected

Die Diskette ist schreibgeschützt.

39 Read/Write

Es wurde ein Lese- oder Schreibfehler auf der Diskette erkannt.

40 RS232 timeout

Es wurde die eingestellte Timeout-Zeit der RS232-Schnittstelle überschritten.

41 RS485/1 timeout

Es wurde die eingestellte Timeout-Zeit der RS485/1-Schnittstelle überschritten.

42 Not formatted

Die Diskette ist nicht formatiert.

43 No Data-File. Die Extension .DAT fehlt im Filenamem.

44 Floppy not present

Das Diskettenlaufwerk ist nicht angeschlossen.

45 RS232/1 timeout

Es wurde die eingestellte Timeout-Zeit der RS232/1-Schnittstelle überschritten.

46 PC Link timeout. Fehler beim UP/Download.

47 Checksum Error in System CWs

Es wurde ein Fehler im Speicher für die System-Parameter erkannt.

48 Checksum Error in User CWs

Es wurde ein Fehler im Speicher für die Benutzer-Parameter erkannt.

49 Checksum Error in Program

Es wurde ein Fehler im Speicher für die Programme erkannt.

50 Array already defined

Diese Fehlermeldung wird generiert, wenn

- ◆ eine DIM-Anweisung gegeben wird für eine Variable die bereits dimensioniert ist.
- ◆ eine DIM-Anweisung gegeben wird für einen String der bereits dimensioniert ist.

51 Array not found

Es soll ein Array (Matrix) im ZP-RAM (Laufwerk R) gelöscht werden, das nicht existiert.

52 Output 1-8 is overloaded. Einer der Ausgänge 1-8 ist überlastet.

53 Output 9-16 is overloaded.

54 RS485/2 timeout.

55 RS232/3 timeout.

56 Jump address missing

Fehlermeldung für fehlende Interrupt-Zieladresse

57 No connection. Fehler beim 3964(R) Protokoll der RS232/3 Schnittstelle.

58 Invalid LABEL. Das angegebene LABEL konnte nicht gefunden werden.

59 Invalid SUB. Das angegebene SUB konnte nicht gefunden werden.

60 Output 17-24 overloaded.

61 Output 25-32 overloaded.

62 Checksum Error in Data. Es wurde ein Fehler in einem Datenfile oder Array (Matrix) erkannt. Der Name des fehlerhaften Arrays (Matrizen) oder Datenfiles steht in \$(#0).

63 F-Bus device no response. Das Feldbusgerät quittiert den gesendeten Befehl nicht.

64 Add file not possible. Das gewünschte Programmfile kann nicht an das Bestehende angehängt werden, weil es kompiliert ist.

65 CPX Response negative. Die COMPAX-Fehlernummer kann über ASC\$(#1) ermittelt werden.

Ausfall eines Feldbusgeräts

Der Ausfall eines Feldbusgeräts wird sofort mit der Fehlermeldung "F_Bus device fail" gemeldet.

Die Fehlernummer wird gebildet aus:

Fehlernummer = Feldbus-Geräteadresse + 100.

26. Feldbus-Schnittstelle

26.1 Funktionsbeschreibung

Die Feldbus-Schnittstelle ist die RS485-Schnittstelle mit dem speziellen COMTAC-Feldbus-Protokoll!

Der Feldbus arbeitet nach dem Master/Slave Prinzip; im gesamten System existiert nur ein Master und ein oder mehrere Slaves.

Mit Parameter 74 wird die Übertragungsrate (Baudrate) eingestellt .

Es können max. 31 Geräte an den Bus angeschlossen werden. Jedes angekoppelte Gerät besitzt eine eigene Busadresse (1 - 99); die Adresse 0 ist für den Master reserviert.

In der Initialisierungsphase (Power-On, Reset) überprüft der Master welche Feldbus-Geräte am Bus angekoppelt sind.

Mit Parameter 70 kann eingestellt werden bis zu welcher Geräte-Adresse der Master prüfen soll.

Von jeder Unterstation wird

- ◆ der Typ (Klassifizierung),
 - ◆ die Adresse,
 - ◆ die Anzahl der zyklischen Eingang-Bytes und
 - ◆ die Anzahl der zyklischen Ausgang-Bytes
- notiert.

Danach wird der zyklische Update gestartet.

In einem zyklischen Update werden nacheinander von jeder Unterstation die aktuelle Statusinformation und die Eingangs-Daten abgefragt; die neuen Ausgabe-Daten werden übergeben.

Sind alle angeschlossenen Geräte durch den zyklischen Update bedient worden, kann ein azyklischer Auftrag (Parameter schreiben/lesen, String ausgeben/einlesen) vom Master ausgeführt werden (OUTPUT, ENTER, FBUS+P, FBUS+D).

Ist die Zykluszeit noch nicht abgelaufen, kann ein weiterer azyklischer Auftrag ausgeführt werden; bevor der nächste zyklische Update durchgeführt wird.

COMTAC macht eine automatische Befehlswiederholung bei Übertragungsfehlern oder wenn das angesprochene Gerät noch nicht bereit ist.

Mit Parameter 75 kann eingestellt werden, wievielmals ein Befehl wiederholt werden soll, bevor COMTAC eine Fehlermeldung generiert.

Mit Parameter 76 kann die max. Übertragungsfehler-Quote eingestellt werden; d.h. wieviele Fehlübertragungen hintereinander an einen Slave erlaubt sind, bevor COMTAC eine Fehlermeldung generiert.

26.2 Parameter der Feldbus - Schnittstelle

Nr.	Funktion	Standard
70	Maximaladresse des Feldbusverbands. In der Initialisierungsphase prüft COMTAC ab der Adresse 1 bis zur Maximaladresse ob die Geräte am Bus angeschlossen sind.	10
71	Feldbus-Interrupt-Maske Bit 0 = device timeout 0=off, 1=on Bit 1 = device event 0=off, 1=on Bit 2 = device error 0=off, 1=on Bit 3 = device data 0=off, 1=on Bit 4 = device write response 0=off, 1=on Bit 5 = disable error message 21 Bit 6 = disable error message 27 Bit 7 = disable auto device write response check	0
72	Feldbus-Zykluszeit 0...255 (1 = 1ms) 0: Baudrateunabhängige Standardwerte	0
73	Feldbusprotokoll 0: kein Feldbus-Betrieb 1: Feldbus Master	1
74	Feldbus-Baudrate 0=150 3=1200 6=9600 9=57600 1=300 4=2400 7=19200 10=172800 2=600 5=4800 8=28800 11=345600	11
75	Befehlswiederholung (Feldbusprotokoll Master) 0...250 COMTAC wiederholt automatisch eine Übertragung an ein Slave-Gerät soviel mal wie hier angegeben, bevor eine Fehlermeldung ausgegeben wird.	10
76	max. Übertragungsfehlerquote (Feldbusprot. Master) Erlaubte Anzahl von Fehlübertragungen an einen Slave. Treten hintereinander mehr Fehlübertragung auf, als hier angegeben, wird eine Fehlermeldung ausgegeben.	10
77	Ein/Ausschalten der Feldbus-Initialisierungsanzeige = 0 --> Anzeige ON = 1 --> Anzeige OFF	0
78	reserviert	0
79	0: keine Time-out Überwachung 1-255: Time-out 0,1 - 25,5 sec.	

26.3 Zykluszeiten

Die max. erreichbare Update-Zeit von E/A - Informationen der Feldbusperipherie ist abhängig von der eingestellten Übertragungsgeschwindigkeit:

Baudrate [KBaud]	Zykluszeit [msec]	Zeichenzeit [µsec]
345.60	7	32
172.80	10	64
57.60	15	192
28.80	30	384

Diese Zeit wird niemals unterschritten, um den Unterstationen Zeit für ein Minimum an Vorverarbeitung zu lassen. Die Zykluszeit erhöht sich, wenn viele Unterstationen an COMTAC angeschlossen sind.

Für jede Feldbuskonfiguration läßt sich die tatsächliche Zykluszeit berechnen. Es gilt immer das Maximum von Standardzeit und berechneter Zeit als tatsächliche Zykluszeit.

Die tatsächliche Zykluszeit(Updatezeit) berechnet sich nach der Formel:

$$T_{zyk} = ((5 * n_{ST} + n_{EA}) * T_{char}) + (n_{ST} * T_{verw})$$

T_{zyk} = Tatsächliche Zykluszeit
 n_{ST} = Anzahl der Stationen
 n_{EA} = Anzahl der Ein/Ausgangsbytes
 T_{char} = Zeichenzeit (z.B. 32 µs bei 345,6 kBaud)
 T_{verw} = Verwaltungszeit (= 100 µs)

Ein COMPAX hat 10 Byte Eingangsdaten und 6 Byte Ausgangsdaten.

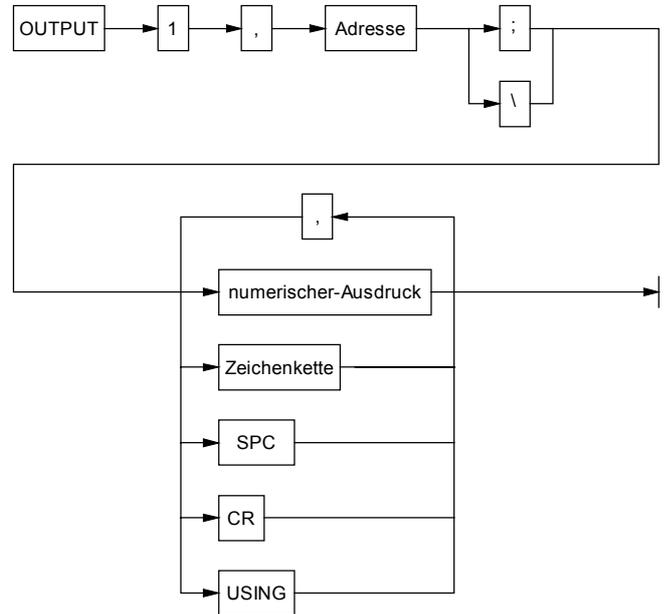
COMTAC ist so ausgelegt, daß es max. 31 COMPAX (= 496 E/A-Daten) über den Feldbus verwalten kann.

Die Zykluszeit für n COMPAX-Geräte bei einer Baudrate von 345.6 kBaud:

$$T_{zyk}(n) = (((5 * n) + (16 * n)) * 32 \mu s) + (100 \mu s * n)$$

$T_{zyk}(n) = 772 \mu s * n$
 $T_{zyk}(1) = 0.772 \text{ ms}$
 $T_{zyk}(2) = 1.544 \text{ ms}$
 $T_{zyk}(3) = 2.316 \text{ ms}$
 $T_{zyk}(4) = 3.088 \text{ ms}$
 $T_{zyk}(5) = 3.860 \text{ ms}$
 $T_{zyk}(6) = 4.632 \text{ ms}$
 $T_{zyk}(31) = 23.93 \text{ ms}$

26.4 OUTPUT [O.](Feldbus)



Funktion:

Diese Anweisung gibt Zeichen, Zeichenketten und/oder den Wert eines numerischen Ausdrucks über die Feldbus-Schnittstelle aus.

Optional kann sofort die zu erwartende Antwort in den String \$(#1) eingelesen werden. Dafür muß in der Befehls-Syntax der OUTPUT-Anweisung das Semikolon durch den Backslash ersetzt werden.

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	0 - 255	Geräteadr. des Empfängers

Beispiel: OUTPUT 1,7;"WINKEL=",W
OUTPUT 1,255;"PA",XPOS,CHR\$(13)

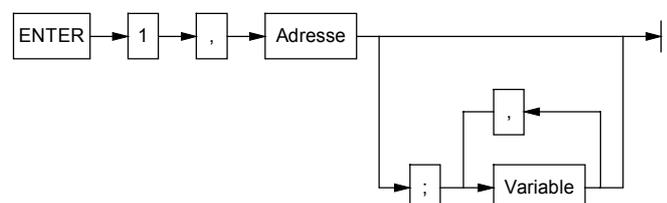
Anmerkung:

Mit dem Wert 255 als Adressen-Angabe (Broadcast-Adresse) wird der zu sendende String von allen angeschlossenen Feldbusgeräten übernommen.

Die Funktion der Anweisungen SPC, CR, USING finden Sie im Kapitel Terminal/LCD-Ausgabe.

➡ Beachten Sie den Hinweis bezüglich "negativer Befehlsquittierung" den Hinweis auf Seite 99!

26.5 ENTER [E.] (Feldbus)



Funktion:

Diese Anweisung liest Zeichen oder eine Zeichenkette von einem Feldbus-Gerät in den String \$(#1) zur weiteren Verarbeitung.

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	0 - 255	Geräteadr. des Senders

Enthält die einzulesende Zeichenkette eine oder mehrere Zahlen, können diese Zahlen mit der ENTER-Anweisung direkt Variablen zugewiesen werden, indem an die ENTER-Anweisung ein Variablenname bzw. eine Variablenname-Liste angehängt wird.

COMTAC-BASIC weist die erste Zahl die sich in der Zeichenkette befindet der ersten Variablen der Variablenname-Liste zu, die zweite Zahl der zweiten Variablen und so weiter.

Die Anzahl der Zahlen in der Zeichenkette und die Anzahl der Variablennamen muß nicht identisch sein.

Beispiel: Bereitstehender String = "10V200X-30"

ENTER 1; A,B,C,D

Nach Ausführung der Anweisung ist

\$(#1) = "10V200X-30"

A = 10

B = 200

C = -30 und D unverändert.

Anmerkung:

Das Feldbus-Gerät zeigt über seinen 'allgemeinen Status' (STG7) an, ob ein Zeichen oder eine Zeichenkette bereitsteht.

➡ Beachten Sie den Hinweis bezüglich "negativer Befehlsquittierung" den Hinweis auf Seite 99!

STG = Allgemeiner Status des Feldbus-Gerätes

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
STG7	STG6	STG5	STG4	STG3	STG2	STG1	STG0
128	64	32	16	8	4	2	1

STG0: gibt Auskunft über die Klasse des Feldbus-Gerätes

0: Azyklische Klasse mit zyklischer Statusmeldung

1: zyklische Klasse mit stetigem I/O Abbild

STG1: ohne Funktion

STG2: ohne Funktion

STG3: ohne Funktion

STG4: ist die Anzeige, daß im Feldbus-Gerät die Befehlsquittierung bereitsteht

0: keine Befehlsquittung

1: Befehlsquittung vorhanden

STG5: wird verwendet, um eine Alarmprozedur anzustoßen

0: kein Alarm (Event)

1: Alarm (Event) aufgetreten

STG6: ist die Anzeige eines Defektes im Feldbus-Gerät

0: kein Defekt

1: Defekt oder Funktionseinschränkung

STG7: ist die Anzeige, daß im Feldbus-Gerät Daten zum Lesen bereitstehen

0: keine Daten für Leseauftrag

1: Daten für Leseauftrag vorhanden

TYP = Klassifizierungsbyte des Feldbus-Gerätes

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TYP7	TYP6	TYP5	TYP4	TYP3	TYP2	TYP1	TYP0
128	64	32	16	8	4	2	1

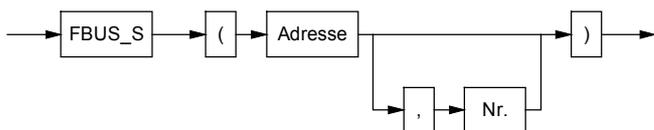
Die Bits 5 - 7 sind für die Grobklassifizierung vorgesehen

TYP7	TYP6	beschreibung
0	0	dieses Gerät erlaubt nur zyklischen Datenverkehr
0	1	dieses Gerät erlaubt nur azyklischen Datenverkehr
1	0	dieses Gerät erlaubt zyklischen und azyklischen Datenverkehr
1	1	dieses Gerät erlaubt nur ein speziell definiertes Protokoll

TYP5	beschreibung
0	dieses Gerät hat Byte-Struktur
1	dieses Gerät hat Wort-Struktur

Folgende Nummernkreise für Geräteklassen sind dadurch gegeben:

26.6 FBUS_S [FS.]



Funktion:

Diese spezielle Funktionsvariable liefert die durch Nummer selektierte Statusinformation des adressierten Feldbus-Geräts.

Wird nur die Adresse angegeben, hat die Funktionsvariable den Wert 65536 = wahr, wenn ein Gerät mit dieser Adresse am Feldbus angeschlossen ist; sonst ist der Wert 0 = falsch.

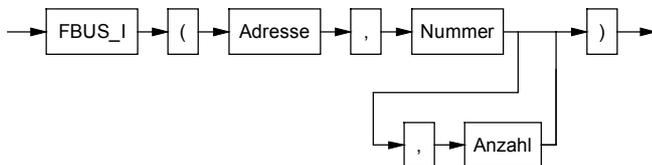
Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	01 - 99	Feldbus Geräte-Adresse
Nummer	num. Ausdr.	0 - 3	Status Index

Status-Index	Beschreibung
0	STG allgemeiner Geräte Status
1	STI Byteanzahl der zyklischen Eingangsdaten
2	STO Byteanzahl der zyklischen Ausgangsdaten
3	TYP Gerätetype (Klassifizierung)

Beispiel: PRINT FBUS_S(4,2)
S = FBUS_S(3,0)

1 - 31	Gerät byteweise strukturiert mit zyklischem Datenverkehr
33 - 63	Gerät wortweise strukturiert mit zyklischem Datenverkehr
65 - 95	Gerät byteweise strukturiert mit azyklischem Datenverkehr
97 - 127	Gerät wortweise strukturiert mit azyklischem Datenverkehr
129-159	Gerät byteweise strukturiert mit zykl. und azykl. Datenverkehr
161-191	Gerät wortweise strukturiert mit zykl. und azykl. Datenverkehr
193-255	Gerät mit speziellen Eigenschaften

26.7 FBUS_I [FI.]



Funktion:

Diese Funktionsvariable liest die durch Nummer und Anzahl selektierten zyklischen Eingangs-Daten des adressierten Feldbus-Geräts.

Das mit 'Nummer' ausgewählte Byte ist das LSB.

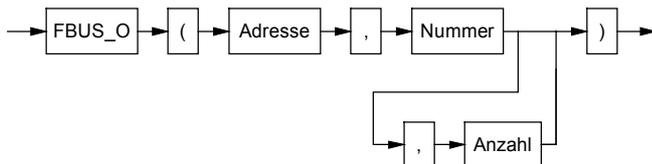
Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	01 - 99	Feldbus Geräte-Adresse
Nummer	num. Ausdr.	1 - STI	Nummer des E-Daten-Byte ab dem gelesen werden soll
Anzahl	num. Ausdr.	1,2 o. 4	Anzahl zu lesender Byts

Beispiel: PRINT FBUS_I(7,1,4)
IF FBUS_I(1,0)@4 THEN ...

Anmerkung:

COMTAC-Basic liest immer 2 Byte Eingangs-Daten, wenn keine Anzahl-Angabe gemacht wird.

26.8 FBUS_O [FO.]



Funktion:

Diese Funktionsvariable ermöglicht das Beschreiben der durch Nummer und Anzahl selektierten zyklischen Ausgangs-Daten des adressierten Feldbus-Geräts mit einem neuen Wert.

Das mit 'Nummer' ausgewählte Byte ist das LSB.

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	01 - 99	Feldbus Geräte-Adresse
Nummer	num. Ausdr.	1 - STO	Nummer des A-Daten-Byte ab dem geschrieben werden soll
Anzahl	num. Ausdr.	1,2 o. 4	Anzahl zu schreibender Byts

Beispiel: FBUS_O(3,1,1) = 01000100B
FBUS_O(5,2) = AUSGABE

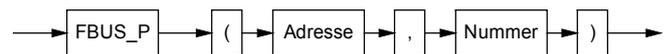
Anmerkung:

COMTAC-Basic beschreibt immer 2 Byte Ausgangs-Daten, wenn keine Anzahl-Angabe gemacht wird.

Die aktuelle zyklische Ausgangsinformation kann gelesen werden.

Mit der Broadcast-Adresse (Adresse = 255) werden alle am Feldbus angeschlossenen Geräte angesprochen.

26.9 FBUS_P [FP.]



Funktion:

Diese Funktionsvariable ermöglicht das Lesen und Beschreiben des durch Nummer selektierten Parameters in dem durch Adresse definierten Feldbus-Gerät.

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	01 - 99	Feldbus Geräte-Adresse
Nummer	num. Ausdr.	1 - 255	Parameter Nummer

Beispiel: PRINT FBUS_P(70,2)
FBUS_P(33,10) = 3000

Anmerkung:

Ein Parameter hat den Wertebereich von 0 - 65535.

Mit der Broadcast-Adresse (Adresse = 255) werden alle am Feldbus angeschlossenen Geräte angesprochen.

26.10 FBUS_D [FD.]



Funktion:

Diese spezielle Funktionsvariable ermöglicht das Lesen und Beschreiben der durch Nummer und Nummer+1 selektierten Parameter in dem durch Adresse definierten Feldbus-Gerät. Die beiden Parameter werden zu einen Doppel-Parameter zusammengefügt und wie folgt behandelt:

- ◆ Parameter(n) = niederwertiger Anteil
- ◆ Parameter(n+1) = höherwertiger Anteil + Vorzeichen

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	01 - 99	Feldbus Geräte-Adresse
Nummer	num. Ausdr.	1 - 255	Parameter Nummer

Beispiel: PRINT FBUS_D(55,128)
 FBUS_D(13,24) = -250000

Anmerkung:

Ein Doppel-Parameter hat den Wertebereich von - 2 147 483 648 bis + 2 147 483 647.
 Es ist zu beachten, daß beim Beschreiben eines Doppel-Parameters der Parameter(n) und der Parameter(n+1) verändert werden.
 Mit der Broadcast-Adresse (Adresse = 255) werden alle am Feldbus angeschlossenen Geräte angesprochen.

Mehrfachzuweisung

COMTAC-Basic bietet die Möglichkeit mit einer Anweisung aufeinander folgenden Parametern Werte zuzuweisen, indem die Werte durch Komma oder Semikolon getrennt, nach dem Gleichheitszeichen aufgelistet werden.

Die beiden Trennzeichen haben folgende Bedeutung:

- ◆ Komma - beschreibe Parameter
- ◆ Semikolon - beschreibe Doppel-Parameter

Beispiel: FBUS_P(3,0)=18;100000,10,555

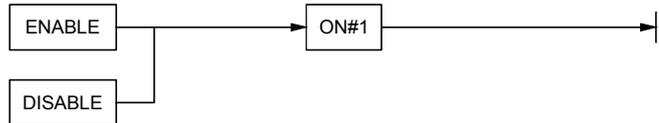
Nach Ausführung dieser Anweisung wird in dem Gerät mit der Feldbusadresse 3 dem

- ◆ Parameter 0 = 18
- ◆ Parameter 1,2 = 100000
- ◆ Parameter 3 = 10 und
- ◆ Parameter 4 = 555 zugewiesen.

Anmerkung:

Mit Parameter 71 können die Interrupt-Meldungen maskiert werden.

26.12 ENABLE [EN.]/DISABLE [DI.] Feldbus



Funktion:

Freigabe/Sperren des Feldbus - Interrupts.

Beispiel: ENABLE ON#1
 DISABLE ON#1

26.13 FBINTADR

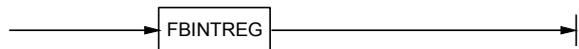


Funktion:

Diese Funktionsvariable liefert die Adresse des Feldbus-Gerätes, das den ON#1-Interrupt ausgelöst hat.

Beispiel: ADR = FBINTADR
 DISP FBINTADR

26.14 FBINTREG



Funktion:

Diese Funktionsvariable liefert die Interrupt Ursache, die den ON#1-Interrupt ausgelöst hat.

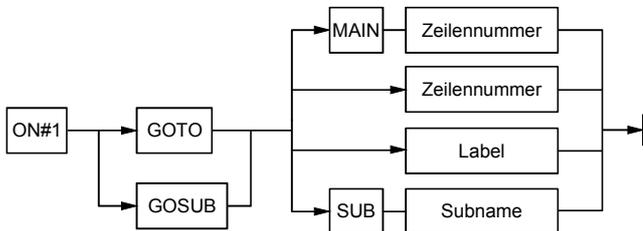
- 0 = Timeout
- 1 = Alarm (Event)
- 2 = Fehler
- 3 = Daten
- 4 = Quit

Beispiel: ON FBINTREG GOSUB ...
 IF FBINTREG = 0 THEN ...

26.15 FBDSRQ(x)



26.11 On#1



Funktion:

Diese Anweisung definiert eine Programmverzweigung, die durchgeführt wird, wenn von einem Feldbusgerät eine der folgenden Meldungen erzeugt wird:

- Timeout
Gerät ist ausgefallen, Busverbindung unterbrochen
- Alarm (Event)
Gerät meldet ein Ereignis, Bedienungsanforderung
- Fehler
im Gerät ist ein Fehler aufgetreten
- Daten
Angeforderte Daten können ausgelesen werden
- Quit
die Befehlsquittung kann ausgelesen werden

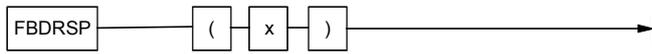
Parameter	Angabe	Bereich	Beschreibung
Zeilen-Nr.	Zahl	0 - 65535	vorhandene Zeilen-Nummer im Programm

Beispiel: ON#1 GOTO ...
 ON#1 GOSUB ...

Funktion:

Alarm (Event) - Daten eines Feldbus-Slave Gerätes einlesen. Antwort steht in \$(#1).

Beispiel: FBDSRQ(2)

26.16 FBDRSP(x)**Funktion:**

Quit - Daten eines Feldbus-Slave Gerätes einlesen. Antwort steht in \$(#1)[1].

ASC\$(#1) = 0 --> Befehl ausgeführt

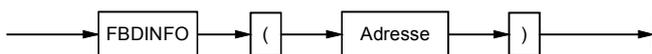
ASC\$(#1) <> 0 --> Befehl wurde nicht ausgeführt, weil ... (siehe Gerätefehlerliste)

Beispiel: FBDRSP(2)

26.17 RESET [RS.] Feldbus**Funktion:**

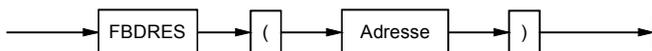
Diese Anweisung setzt die Feldbus-Schnittstelle in den Power-On Zustand zurück.

Beispiel: RESET #1

26.18 FBDINFO(x)**Funktion:**

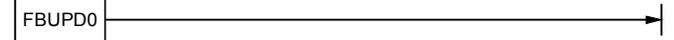
Klartext-Info über einen Feldbus-Slave-Gerät einlesen. Antwort steht in \$(#1).

Beispiel: FBDINFO(2)

26.19 FBDRES(x)**Funktion:**

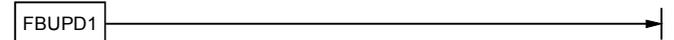
Feldbus-Slave-Geräte zurücksetzen.

Beispiel: FBDRES(4)

26.20 FBUPD0**Funktion:**

Zyklischer Update des Feldbus-Master (COMTAC) abschalten.

Beispiel: FBUPD0

26.21 FBUPD1**Funktion:**

Zyklischer Update des Feldbus-Master (COMTAC) einschalten.

Beispiel: FBUPD1

Reaktion auf negative Befehlsquittierung von COMPAX

Wird eine Feldbus-Anweisung an ein COMPAX von diesem negativ quittiert, weil an COMPAX ein Fehler vorliegt (z. B. E55), prüft das Betriebssystem ob bereits ein ONCPXERR-Interrupt ansteht. Ist dies der Fall, wird der BASIC-Text Zeiger an den Anfang der Feldbus-Anweisung gestellt und verzweigt zur:

- ◆ ONCPXERR-Interrupt Routine (sofern dies aufgrund der vergebenen Interrupt-Prioritäten möglich ist),
- oder
- ◆ zu jeder anderen anstehenden Interrupt Routine mit höherer Priorität.

Nach dem Rücksprung aus dem (ONCPXERR-) Interrupt-Unterprogramm (in welchem der COMPAX-Fehler behoben wurde) wird nun automatisch die Feldbus-Anweisung wiederholt.

Steht kein ONCPXERR-Interrupt an (z. B. nicht freigegeben), wird die Feldbus-Anweisung entsprechend dem Wert in P75 x Mal wiederholt. Danach wird ebenfalls der BASIC-Text Zeiger an den Anfang der Feldbus-Anweisung gestellt und die COMTAC Fehlerverwaltungs-Routine mit der Fehler Nr 65 aufgerufen.

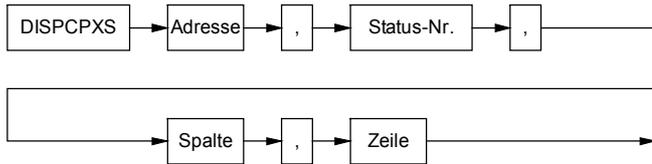
Ist der ONERR-Interrupt freigegeben, können in der ONERR-Interrupt Routine die COMTAC-Fehlernummer (ERRSTS) und die COMPAX-Fehlernummer (ASC\$(#1)) ausgewertet werden und gegebenenfalls dann der COMPAX-Fehler behoben werden. Nach dem Rücksprung aus dem ONERR-Interrupt-Unterprogramm wird nun automatisch die Feldbus-Anweisung wiederholt.

27. COMPAX - Befehle

➡ Die COMPAX - Befehle sind nur im Feldbus-Betrieb möglich!

Beachten Sie den Hinweis bezüglich "negativer Befehlsquittierung" den Hinweis auf Seite 99!

27.1 DISPCPXS



Funktion:
Einmaliges Lesen eines COMPAX-Status und Anzeige im Display. Wahlweise mit zugehörigem Text.

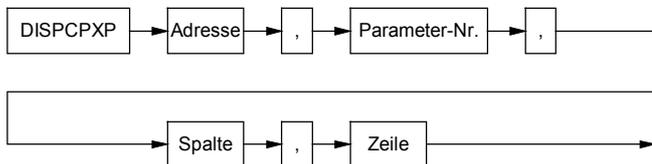
➡ Nur im Feldbus-Betrieb möglich!

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	1-99	Geräteadr. des COMPAX
S-Nr.	num. Ausdr.	1-250	COMPAX Status-Nummer
Spalte	num. Ausdr.	0-40	Display-Spalte
Zeile	num. Ausdr.	1-4	Display-Zeile

Beispiel: DISPCPXS 1,1,1,1
DISPCPXS 5,1,0,20

Anmerkung:
Ist der Wert für die Spalte = 0, wird die gesamte Zeile beschrieben mit Achsnummer, Statusbeschreibung und Statuswert.

27.2 DISPCPXP



Funktion:
Einmaliges Lesen eines COMPAX-Parameter und Anzeige im Display. Wahlweise mit zugehörigem Text.

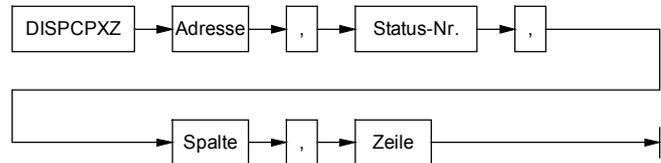
➡ Nur im Feldbus-Betrieb möglich!

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	1-99	Geräteadr. des COMPAX
Parameter-Nr.	num. Ausdr.	1-250	COMPAX Parameter-Nummer
Spalte	num. Ausdr.	0-40	Display-Spalte
Zeile	num. Ausdr.	1-4	Display-Zeile

Beispiel: DISPCPXP 31,4,0,1
DISPCPXP 10,100,X,Y

Anmerkung:
Ist der Wert für die Spalte = 0, wird die gesamte Zeile beschrieben mit Achsnummer, Parameterbeschreibung und Parameterwert.

27.3 DISPCPXZ



Funktion:
Der einmal angewiesene Befehl bewirkt das zyklische Lesen und Anzeigen eines Statuswerts von COMPAX M.
Die Anzeige kann mit STOP DISP zwischengespeichert und mit CONT DISP wieder angezeigt werden (siehe Seite 54)

➡ Nur im Feldbus-Betrieb möglich!

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	1 - 99	Geräteadresse von COMPAX
Statusnr.	num. Ausdr.	0...	Statusnr. von COMPAX mit 0 wird die Anzeige abgeschaltet.
Spalte	num. Ausdr.	0...40	Spalte im COMTAC-Display 0: Anzeige von Achs-nr., Bedeutung und Wert. >0: Nur der Wert wird angezeigt.
Zeile	num. Ausdr.	1...4	Display-Zeile

Beispiel: DISPCPXZ 1,1,1,3
DISPCPXZ 1,0,1,3

Anmerkung:

Ist der Wert für die Spalte = 0, wird die gesamte Zeile beschrieben mit Achsnummer, Statusbeschreibung und Statuswert.

Ist der Wert für die COMPAX Status-Nummer = 0 wird die zyklische Anzeige an der angegebenen Stelle im Display (Spalte, Zeile) beendet. Es können max. 9 Werte gleichzeitig zyklisch angezeigt werden.

27.4 CPXSTEXT



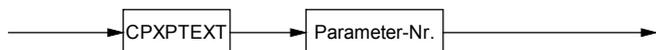
Funktion:

Diese Funktion wird in der DISP-Anweisung verwendet um die aktuelle Beschreibung für einen COMPAX-Status anzuzeigen.

Parameter	Angabe	Bereich	Beschreibung
Statusnr.	num. Ausdr.	1...250	Statusnr. von COMPAX

Beispiel: DISP CPXSTEXT 5
DISP TABXY(10,3),CPXSTEXT 1

27.5 CPXPTEXT



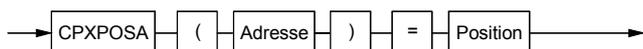
Funktion:

Diese Funktion wird in der DISP-Anweisung verwendet um die aktuelle Beschreibung für einen COMPAX-Parameter anzuzeigen.

Parameter	Angabe	Bereich	Beschreibung
Parameternummer	num. Ausdr.	1...250	Parameternr. von COMPAX

Beispiel: DISP CPXPTEXT 94
DISP TABXY(10,3),CPXPTEXT 100

27.6 CPXPOSA [PA.]



Funktion:

Absolutpositionierung.

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	1 - 99	Geräteadresse von COMPAX
Wert	num. Ausdr.		gültige Position

Beispiel: CPXPOSA(5) = 1000

27.7 CPXPOSR [PR.]



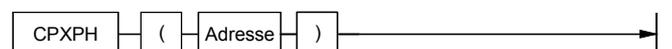
Funktion:

Relativpositionierung.

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	1 - 99	Geräteadresse von COMPAX
Wert	num. Ausdr.		gültige Position

Beispiel: CPXPOSR(AX) = 500

27.8 CPXPH [ZP.]



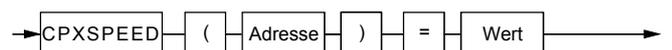
Funktion:

Maschinennullpunkt anfahren.

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	1 - 99	Geräteadresse von COMPAX

Beispiel: CPXPH(7)

27.9 CPXSPEED [SD.]



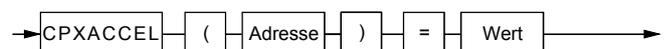
Funktion:

Geschwindigkeitsvorgabe.

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	1 - 99	Geräteadresse von COMPAX
Wert	num. Ausdr.		Geschwindigkeit

Beispiel: CPXSPEED(1) = 80

27.10 CPXACCEL [AL.]



Funktion:

Beschleunigungsvorgabe.

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	1 - 99	Geräteadresse von COMPAX
Wert	num. Ausdr.		Beschleunigung

Beispiel: CPXACCEL(8) = 2500

27.14 CPXSP [SP.]

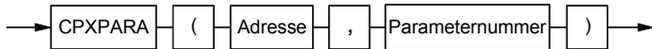


Funktion:
Programm/Positionierung anhalten.

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	1 - 99	Geräteadresse von COMPAX

Beispiel: CPXSP(1)

27.11 CPXPARA [PAR.]



Funktion:
Lesen eines COMPAX - Parameters.

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	1 - 99	Geräteadresse von COMPAX
Parameternummer.	num. Ausdr.	1...250	Parameternr. von COMPAX

Beispiel: A = CPXPARA (1,1)

27.15 CPXBK [BK.]



Funktion:
Positionierung/Programmschritt abbrechen.

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	1 - 99	Geräteadresse von COMPAX

Beispiel: CPXBK(10)

27.12 CPXSTA [STA.]



Funktion:
Lesen eines COMPAX - Statuswerts.

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	1 - 99	Geräteadresse von COMPAX
Statusnr.	num. Ausdr.	1...250	Statusnr. von COMPAX

Beispiel: A = CPXSTA (1,1)

27.16 CPXQT [QT.]

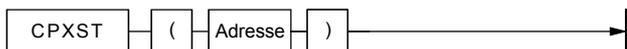


Funktion:
Fehler quittieren.

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	1 - 99	Geräteadresse von COMPAX

Beispiel: CPXQT(9)

27.13 CPXST [ST.]



Funktion:
Programm starten, Positionierung fortsetzen.

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	1 - 99	Geräteadresse von COMPAX

Beispiel: CPXST(8)

27.17 CPXON [P1.]



Funktion:
Antrieb unter Moment bei geöffneter Bremse.

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	1 - 99	Geräteadresse von COMPAX

Beispiel: CPXON(1)

27.18 CPXOFF [P0.]



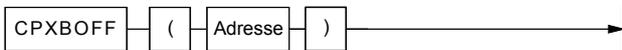
Funktion:

Antrieb stromlos bei geöffneter Bremse.

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	1 - 99	Geräteadresse von COMPAX

Beispiel: CPXOFF(6)

27.19 CPXBOFF [P0B.]



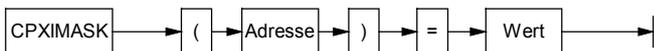
Funktion:

Antrieb stromlos bei geschlossener Bremse.

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	1 - 99	Geräteadresse von COMPAX

Beispiel: CPXBOFF(7)

27.20 CPXIMASK [CIM.]



Funktion:

Maske für die COMPAX-Eingangs-Funktionen E1 bis E6. Standardmäßig sind den COMPAX-Eingängen E1 bis E6 Funktionen zugewiesen. Wird das entsprechende Bit der Maske gesetzt, ist der Zugriff auf diese COMPAX-Eingangs-Funktion über die CPXCTR-Anweisung freigegeben, gleichzeitig verliert der entsprechende COMPAX-Eingang diese Funktion und steht zur freien Verfügung. Bit 0 = Maske für E1, Bit 1 = Maske für E2, ... Bit 6 = Maske für E6 (entspricht dem COMPAX-Parameter P221).

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	0...99	Geräteadresse des COMPAX
Wert	num. Ausdr.	0..65535	zuzuweisender Zahlenwert

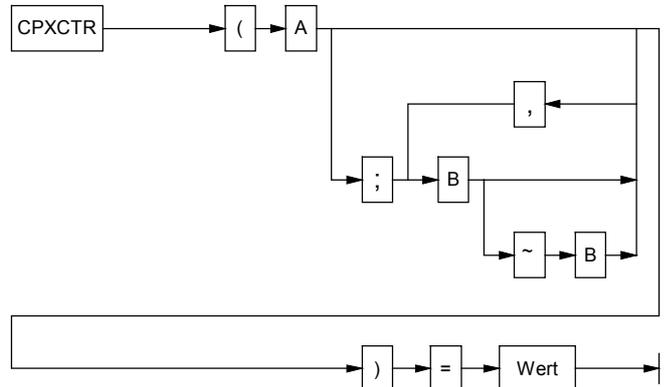
Beispiel: CPXIMASK (12) = 0003H

In diesem Beispiel werden die Eingangs-Funktionen 1 und 2 des COMPAX mit der Geräteadresse 12 maskiert und können nun mit der CPXCTR-Anweisung aktiviert oder deaktiviert werden.

Anmerkung:

Mit der Broadcast-Adresse (Adresse = 255) können am Feldbus alle angeschlossenen Geräte angesprochen werden.

27.21 CPXCTR [CC.]



Funktion:

Mit dieser Anweisung können Sie die Funktionen der COMPAX-Eingänge (ab COMPAX Software V2.10 die virtuellen Eingänge von COMPAX) über das COMPAX-Steuerwort auslösen, wenn der Zugriff vorher mit der CPXIMASK-Anweisung freigegeben wurde.

Es ist zu beachten, daß die Nummerierung der Bits nicht wie üblich mit 0 sondern mit 1 beginnt!

Das CPXCTR (COMPAX-Steuerwort BIT 1–Bit 16) kann über CPXIMASK mit den selben Funktionen belegt werden wie die COMPAX-Eingänge (E1 - E16)!

COMTAC-Basic geht bei dieser Funktion wie folgt vor:

- ◆ Der zuzuweisende Wert wird binär interpretiert.
- ◆ Dem letztgenannten Bit in der CPXCTR-Anweisung wird die Binärstelle 2 hoch 0 zugewiesen, dem davor genannten Bit die Binärstelle 2 hoch 1 usw.
- ◆ Es werden sovielen Binärstellen des Wertes berücksichtigt wie Bit angegeben sind.
- ◆ Ist kein Bit angegeben werden alle 16 Bit des Steuerwortes durch die Wertzuweisung beeinflusst (Bit 16 = MSB, Bit 1 = LSB). Bsp.: CPXCTR(1)=3 → E17=E18="1"
- ◆ Für eine Bitfolge wird das erste und das letzte Bit getrennt durch das Zeichen ~ angegeben.
- ◆ Einzelne Bit oder Bitfolgen werden durch ein Komma getrennt.

Parameter	Angabe	Bereich	Beschreibung
A	num. Ausdr.	0...99	Geräteadresse des COMPAX
B	num. Ausdr.	1...16	Bit-Nummer virtuelle Eingänge 17-32
Wert	num. Ausdr.	0..65535	zuzuweisender Zahlenwert

Folgende Beispiele sollen die Funktion der CPXCTR-Anweisung verdeutlichen:

Beispiel 1: CPXCTR(1;3~1)=3

Den Bits 1 bis 3 des Steuerwortes vom COMPAX mit der Geräteadresse 1 wird die Zahl 3 zugewiesen (E17=E18=1).

Die Zuweisung der einzelnen Binärstellen an die angegebenen Bits sieht wie folgt aus:

Binärstellen	2 ⁷	2 ⁶	2 ⁵	2 ⁴	2 ³	2 ²	2 ¹	2 ⁰
Wertigkeit der Binärstellen	128	64	32	16	8	4	2	1
Zuordnung der Bit (Bit-Nr.)						3	2	1
Zahlenw. 3 im Binärformat						0	1	1

Beispiel 2: CPXCTR(5)=2

Das Bit 2 des Steuerwortes vom COMPAX mit der Geräteadresse 5 wird gesetzt; alle anderen Bits werden rückgesetzt.

Anmerkung:

CPXCTR kann auch gelesen werden und zeigt immer den Zustand aller 16 Bit an (Bit 16 = MSB, Bit 1 = LSB). Mit der Broadcast-Adresse (Adresse = 255) können am Feldbus alle angeschlossenen Geräte angesprochen werden.

Anmerkung:

CPXOMASK kann nicht gelesen werden. Mit der Broadcast-Adresse (Adresse = 255) können am Feldbus alle angeschlossenen Geräte angesprochen werden.

27.22 CPXSTS [CS.]



Funktion:

Diese Anweisung bietet dem Anwender die Möglichkeit das COMPAX-Ausgangswort (A1 - A16) als Zahlenwert einzulesen.

Achtung: Bitzählweise beginnt mit 0 (Bit 0 = A1).

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	0...99	Geräteadresse des COMPAX

Beispiel: DISP CPXSTS(4)
IF CPXSTS(1)@5 THEN ...

27.23 CPXOMASK [COM.]



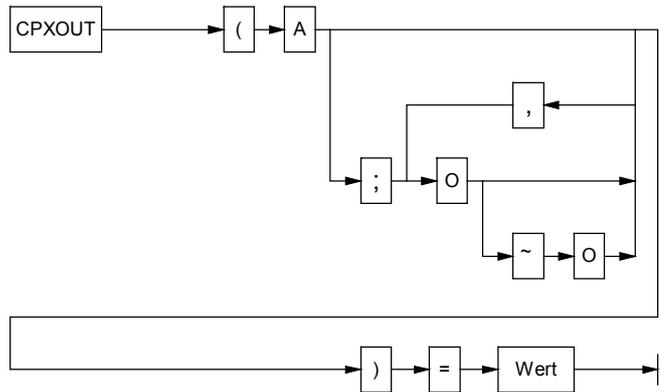
Funktion:

Maske für diegänge A1 bis A16 von COMPAX. Wird das entsprechende Bit der Maske gesetzt, ist der Zugriff auf diesen COMPAX-Ausgang über die CPXOUT-Anweisung freigegeben und gleichzeitig jeder andere Zugriff auf diesen Ausgang gesperrt (Statusanzeige, Bedienung über den Satzspeicher oder CPX-Befehle die auf den Ausgang wirken). Bit 0 = Maske für A1, Bit 1 = Maske für A2, ... Bit 5 = Maske für A6.

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	0...99	Geräteadresse des COMPAX
Wert	num. Ausdr.	0..65535	zuzuweisender Zahlenwert

Beispiel: CPXOMASK(15) = 0007h
In diesem Beispiel werden diegänge 1-3 (Bit 0 - 2) des COMPAX mit der Geräteadresse 15 maskiert und können nun mit der CPXOUT-Anweisung gesetzt oder rückgesetzt werden.

27.24 CPXOUT [CO.]



Funktion:

Diese Anweisung bietet dem Anwender die Möglichkeit den COMPAX-Ausgängen oder einem definierten COMPAX-Ausgang bzw. Ausgangsgruppe einen Zahlenwert zuzuweisen, wenn der Zugriff vorher mit der CPXOMASK-Anweisung freigegeben wurde. COMTAC-Basic geht bei dieser Funktion wie folgt vor: Der zuzuweisende Wert wird in das binäre Format umgewandelt. Jedem angegebenen Ausgang in der CPXOUT-Anweisung wird eine Binärstelle dieses Wertes zugewiesen und dadurch gesetzt bzw. zurückgesetzt. Die Zuordnung der einzelnen Binärstellen an die angegebenengänge erfolgt von rechts nach links fortlaufend. Dem letztgenannten Ausgang in der CPXOUT-Anweisung wird die Binärstelle 2 hoch 0 zugewiesen, dem davor genannten Ausgang die Binärstelle 2 hoch 1 usw. Es werden soviele Binärstellen des Wertes berücksichtigt wiegänge angegeben sind. Ist kein Ausgang angegeben, werden alle 16 COMPAX-Ausgänge durch die Wertzuweisung beeinflusst (Ausgang 16 = MSB, Ausgang 1 = LSB). Es können maximal 16gänge in einer CPXOUT-Anweisung angegeben werden. Für eine Ausgangsfolge wird der erste und der letzte Ausgang getrennt durch das Zeichen ~ angegeben. Die einzelnengänge oder Ausgangsfolgen werden durch ein Komma getrennt.

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	0...99	Geräteadresse des COMPAX
O	num. Ausdr.	1..16	Ausgangs-Nummer
Wert	num. Ausdr.	0..65535	zuzuweisender Zahlenwert

Folgende Beispiele sollen die Funktion der CPXOUT-Anweisung verdeutlichen:

Beispiel 1: CPXOUT(1;8~1)=11

Den Ausgängen 1 bis 8 vom COMPAX mit der Geräteadresse 1 wird die Zahl 11 zugewiesen.
Die Zuweisung der einzelnen Binärstellen an die angegebenen Ausgänge sieht wie folgt aus:

Binärstellen	2 ⁷	2 ⁶	2 ⁵	2 ⁴	2 ³	2 ²	2 ¹	2 ⁰
Wertigkeit der Binärstellen	128	64	32	16	8	4	2	1
Zuordnung der Ausgänge (Ausgangs-Nr.)	8	7	6	5	4	3	2	1
11 im Binärformat	0	0	0	0	1	0	1	1

Beispiel 2: CPXOUT(5;1~3,8,10,15)=60

Den Ausgängen 1,2,3,8,10 und 15 vom COMPAX mit der Geräteadresse 5 wird die Zahl 60 zugewiesen.
Die Zuweisung der einzelnen Binärstellen an die angegebenen Ausgänge sieht wie folgt aus:

Binärstellen	2 ⁷	2 ⁶	2 ⁵	2 ⁴	2 ³	2 ²	2 ¹	2 ⁰
Wertigkeit der Binärstellen	128	64	32	16	8	4	2	1
Zuordnung der Ausgänge (Ausgangs-Nr.)	-	-	1	2	3	8	10	15
60 im Binärformat	0	0	1	1	1	1	0	0

Anmerkung:

CPXOUT kann auch gelesen werden und zeigt immer den Zustand aller 16 Ausgänge an (Ausgang 16 = MSB, Ausgang 1 = LSB).
Mit der Broadcast-Adresse (Adresse = 255) können am Feldbus alle angeschlossenen Geräte angesprochen werden.

27.25 CPXINP [CI.]



Funktion:

Einlesen der Eingänge E16 bis E1 von einem COMPAX in eine Variable; E16 ist MSB, E1 ist LSB.

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	0...99	Geräteadresse des COMPAX

Beispiel: DISP CPXINP(8)
IF CPXINP(2)@3 THEN ...

27.26 CPXOVR [CV.]



Funktion:

Einstellen der COMPAX-Drehzahlreduzierung.

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	0...99	Geräteadresse des COMPAX
Wert	num. Ausdr.	0..255	zuzuweisender Zahlenwert

Beispiel: CPXOVR (7) = 100

Anmerkung:

CPXOVR kann auch gelesen werden.
Mit der Broadcast-Adresse (Adresse = 255) können am Feldbus alle angeschlossenen Geräte angesprochen werden.

27.27 CPXPOS [CP.]



Funktion:

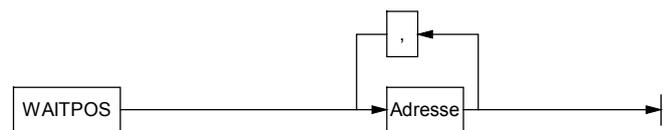
Einlesen der COMPAX-Istposition.

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	0...99	Geräteadresse des COMPAX

Beispiel: DISP CPXPOS(8)

27.28 WAITPOS

27.28.1 Warten auf "Position erreicht"



Funktion:

Warten auf 'Position erreicht' der spezifizierten COMPAX-Geräte.
Sind alle angegebenen Achsen in Position, ist die WAITPOS-Anweisung abgeschlossen.

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	0...99	Geräteadresse des COMPAX

Beispiel: WAITPOS 1,2
WAITPOS A1,A2,A3

Anmerkung:

Die WAITPOS-Funktion kann durch einen Interrupt unterbrochen werden.
Verzweigt das Programm durch den Interrupt in ein Unterprogramm, wird nach dem Rücksprung die WAITPOS-Funktion wiederholt.

27.28.2 Abfrage auf "Position erreicht"



Funktion:

'Position erreicht' - Abfrage von COMPAX-Achsen.
Diese spezielle Funktionsvariable hat den Wert 65535 = wahr, wenn die angegebenen Achsen in Position sind oder den Wert 0 = falsch, wenn eine der Achsen nicht in Position ist.

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	0...99	Geräteadresse des COMPAX

Beispiel: IF WAITPOS 1,2,3 THEN ...
DO
...
...
UNTIL WAITPOS A1,A2,A3,A4

Anmerkung:

27.29.2 Abfrage auf "Maschinennull erreicht"



Funktion:

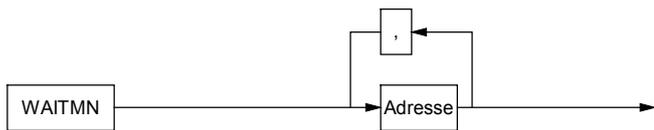
'Maschinen-Null angefahren'-Abfrage von COMPAX-Achsen.
Diese spezielle Funktionsvariable hat den Wert 65535 = wahr, wenn alle angegebenen Achsen Maschinen-Null angefahren haben; oder den Wert 0 = falsch, wenn eine der Achsen den Maschinen-Null noch nicht angefahren hat.

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	0...99	Geräteadresse des COMPAX

Beispiel: IF WAITMN 1,2,3 THEN ...
DO
...
...
UNTIL WAITMN A1,A2,A3,A4

27.29 WAITMN

27.29.1 Warten auf "Maschinennull erreicht"



Funktion:

Warten auf 'Maschinen-Null angefahren' der spezifizierten COMPAX-Geräte.
Haben alle angegebenen Achsen Maschinen-Null angefahren, ist die WAITMN-Anweisung abgeschlossen.

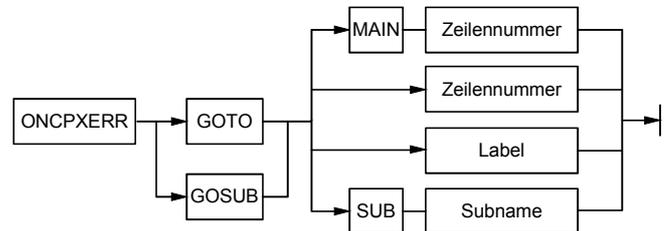
Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	0...99	Geräteadresse des COMPAX

Beispiel: WAITMN 1,2
WAITMN A1,A2,A3

Anmerkung:

Die WAITMN-Funktion kann durch einen Interrupt unterbrochen werden.
Verzweigt das Programm durch den Interrupt in ein Unterprogramm, wird nach dem Rücksprung die WAITMN-Funktion wiederholt.

27.30 ONCPXERR



Funktion:

Diese Anweisung definiert eine Programmverzweigung, die durchgeführt wird, wenn ein COMPAX einen Fehler meldet und der ONCPXERR-Interrupt freigegeben ist.

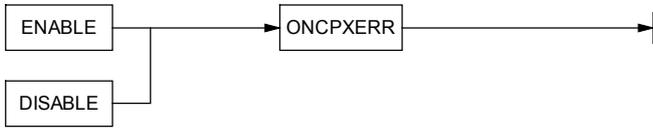
Achtung:

Dieser Interrupt muß quittiert werden.
Durch Lesen der COMPAX - Adresse z. B. mit A=CPXERRADR wird der Interrupt wieder freigegeben; ansonsten werden weitere COMPAX - Fehler nicht erkannt.

Parameter	Angabe	Bereich	Beschreibung
Zeilen-Nr.	Zahl	0..65535	vorhandene Zeilen-Nr im Programm

Beispiel: ONCPXERR GOTO ...
ONCPXERR GOSUB ...

27.31 DISABLE [DI.] / ENABLE [EN.] ONCPXERR



Funktion:

Freigabe/Sperren des ONCPXERR-Interrupt.

Beispiel: ENABLE ONCPXERR
DISABLE ONCPXERR

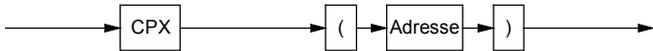
Funktion:

Diese Funktionsvariable hat den Wert 65536 = wahr, wenn ein COMPAX mit dieser Adresse am Bus angeschlossen ist; sonst ist der Wert 0 = falsch.

Parameter	Angabe	Bereich	Beschreibung
Adresse	num. Ausdr.	1...99	Feldbus Geräteadresse des COMPAX

Beispiel: IF CPX(3) THEN ...
DISP CPX(5)

27.32 CPX



27.33 CPXERRADR



Funktion:

Diese Funktionsvariable liefert die Adresse des COMPAX, das den ONCPXERR-Interrupt ausgelöst hat.

Beispiel: IF CPXERRADR=5 THEN ...
DISP CPXERRADR

28. Beispielprogramm

Ziel:

- ◆ Referenzfahrt am angeschlossenen COMPAX.
- ◆ Positionierung.
- ◆ Zyklische Statusanzeige eines COMPAX-Status am CONTAC-Display.

Aufgabe:

- ◆ Maschinennull anfahren und warten bis der Maschinennull erreicht ist.
- ◆ Zielposition am COMTAC abfragen.
- ◆ Aktuelle Position in Zeile 2, aktuelle Geschwindigkeit in Zeile 3 anzeigen.

Programm

```

100 CLEAR D ..... ! Display loeschen
110 A=1 ..... ! A = Adresse des COMPAX
130 DISP TABXY(5,1),"Maschinen-Null wird angefahren!"
140 OUTPUT 1,A;"PH" ..... ! Befehl fuer MN-anfahren an CPX ausgeben
150 WAIT 100 ..... ! Kleine Wartezeit vor der Statusabfrage
160 IF CPXSTS(A)@2 THEN 170 ELSE 160 ..... ! MN angefahren ?
170 OUTPUT 1,A;"SD10" ..... ! Geschwindigkeit = 10%
180 CLEAR D
190 DISP TABXY(1,2),"Istposition: "
200 DISP TABXY(1,3),"Geschwindigkeit: "
220 INPKBD "Bitte Zielposition eingeben!",POS
230 OUTPUT 1,A;"PA",POS
240 WAIT 100 ..... ! Kleine Wartezeit vor der Statusabfrage
250 DISP TABXY(18,2),CPXPOS(A) ..... ! Istpositionswert anzeigen
260 OUTPUT 1,A;"S4" ..... ! Geschwindigkeitswert anfordern
270 WAIT 100 ..... ! 0.1 s warten bis COMPAX Antwort bereitgestellt hat
280 ENTER 1,A ..... ! COMPAX Antwort einlesen
290 DISP TABXY(18,3),$(#1)[6] ..... ! Status Geschwindigkeit ab der 6. Stelle anzeigen (S004:....)
300 IF NOT(CPXSTS(A)@4) THEN 250
310 DISP TABXY(18,2),CPXPOS(A) ..... ! Istpositionswert anzeigen
320 OUTPUT 1,A;"S4" ..... ! Geschwindigkeitswert anfordern
330 WAIT 100 ..... ! 0.1 s warten bis COMPAX Antwort bereitgestellt hat
340 ENTER 1,A ..... ! COMPAX Antwort einlesen
350 DISP TABXY(18,3),$(#1)[6] ..... ! Status Geschwindigkeit ab der 6. Stelle anzeigen (S004:....)
360 WAIT 1000
370 DISP TABXY(10,4),"Position erreicht!"
380 WAIT 1000
390 CLEAR D,4
400 CLEAR D,1
410 GOTO 190

```

29. Parameter, nach Nummern sortiert

Parameter - Übersicht

Nr.	Verwendung
00...09	COMTAC System-Parameter
10...19	COMTAC - Systemparameter
20...29	reserviert
30...39	reserviert
40...49	RS232/3 - Schnittstellen-Parameter
50...59	RS232/1 - Schnittstellen-Parameter
60...69	RS485/1 - Schnittstellen-Parameter
70...79	RS485/1 - Schnittstellen-Parameter (Feldbus)
80...89	RS485/2 - Schnittstellen-Parameter
90...99	RS232/2 - Schnittstellen-Parameter
100	Terminal-Schnittstellenzuordnung
101...200	Anwender spezifisch

System-Parameter

Nr.	Funktion	Standard
00	COMTAC-Typ und Softwareversion; wird bei "Power on" gesetzt Beispiel: Mit der Software Version 2.53 ist bei einem COMTAC 2000 CT.(0) = 2253 und bei einem COMTAC 3000 Ct.(0)=3253.	2000/ 3000
01	Datum der Betriebssoftware	-
02	Terminal-Typ 0 = TV 905 3 = VT100	0
03	Fabrikations-Nummer	-
04	Prüfdatum	-
05	Power-ON-Wert von A1 ... A16 Nach Power-ON oder Ctrl_R erfolgt die automatische Zuweisung: BINOUT(16~1)-=Parameter 5	0
06	Power-ON-Wert von A17 ... A32 Nach Power-ON oder Ctrl_R erfolgt die automatische Zuweisung: BINOUT(32~17)-=Parameter 6	0
07	Einschaltverzögerung 1 - 255 1=100ms	50
08	REQOUT Time-Out 1 - 255 1 = 100 ms	10
09	Repeat-Zeit für COMTAC-Tastatur 1 - 255 1 = 100 ms	1
10	Schnittstellenzuordnung zum Diskettenlaufwerk 0 = RS232/1 ist das Diskettenlaufwerk 1 = RS485/1 ist das Diskettenlaufwerk 2 = RS232/2 ist das Diskettenlaufwerk 3 = RS485/2 ist das Diskettenlaufwerk 4 =RS232/3 ist das Diskettenlaufwerk 99 = kein Diskettenlaufwerk	99
11	Floppy-Timeout-Wert 0 - 255 (1 = 0.1s) 0 = keine Timeout-Überwachung	10

12	Floppy-Baudrate 4 = 2400 5 = 4800 6 = 9600 7 = 19200 8 = 38400	7
13	System-Flags mit folgender Bedeutung:	0
	Bit Funktion	
0	Parallel-Eingabe (Terminal / Folientastatur) 0 = off, 1 = on	0
1	Listing-Option; 0 = Normal, 1 =Befehlskürzel	0
2	Tastatur-Repeat; 0 = off, 1 = on	1
3	F8-Funktion 0=Stop 1:ohne Funktion	0
4	Checksumme-Daten 0=off 1=on	0
5	Checksumme-User Param. 0=auto 1=Befehl	0
6	Checksumme-User Param. 0=off 1=on	0
7	Autostartfunktion ZP-RAM; 0=auto 1=Befehl	0
8	Autostartfunktion ZP-RAM; 0 = off, 1 = on	1
9	Autostartfunktion Diskette; 0 = off, 1 = on	0
10	Autostartfunktion NV-Ram 0 = off, 1 = on	0
11	Checksumme-Programm und Systemparameter; 0 = off, 1 = on	0
12	NV-Ram 0 = off, 1 = on	0
13	Datumsausgabe; 0 =engl. 1 = dt	0
14	Ready-Beep; 0 = off, 1 = on	0
14	Positionswert für die automatische Anzeige der Uhrzeit im LCD 0 = Anzeige gesperrt 1...160 = Position des 1. Zeichens im LCD	33
15	reserviert	-
16	Positionswert für die automatische Zeilenr.-Anzeige im LCD bei der DEBUG-Funktion 0: Anzeige gesperrt 1...160: Position des 1. Zeichens im LCD.	0
17	Überwachungszeit (0 ≡ 0,5s)	0
18	Passwortschutz	-
19	Erweiterter Passwortschutz	-

⇒ Mit den Parametern P10-12 ist die Schnittstelle zum HFM2 (Doppeldiskettenlaufwerk) eingestellt. Die Einstellung weiterer Schnittstellendaten der verwendeten Schnittstelle übernimmt COMTAC selbst.

⇒ Durch Drücken von F12 während dem Einschalten von COMTAC werden die COMTAC - Parameter auf Standardwerte gesetzt.

Parameter der RS232/3-Schnittstelle (#4) (Option F6)

Nr.	Funktion	Standard
40	Geräte-Adresse 0 - 255	0
41	Empfangs-Start-Zeichen 0 - 255 bzw. Zeichenverzugszeit bei 3964-Prozedur (1↔50ms)	3
42	Empfangs-Ende-Zeichen 0 - 255 bzw. Quittungsverzugszeit bei 3964-Prozedur (1↔50ms)	62
43	Empfangs-Protokoll	34
	Bit Funktion	
	0 ProtokollBit 0	0
	1 ProtokollBit 1	1
	2 ProtokollBit 2	0
	3 -	0
	4 Block-Check-Character; 0=off, 1=on	0
	5 Auto-RTS; 0=off, 1=on	1
	6 Hardware-Handshake; 0 = off, 1 = on	0
	7 Software-Handshake; 0 = off, 1 = on	0
	Mit den ProtokollBits wir eines der folgenden Empfangsprotokolle ausgewählt.	
	P-Bit INPUT-Rdy = 1,	
	2 1 0	0 1 0
	0 0 0 nach jedem empfangenen Zeichen	
	0 0 1 nach def. Anzahl empfangener Zeichen (P46)	
	0 1 0 nach empfangenem Endezeichen (P42)	
	0 1 1 nach empf. Start (P41)- und Endezeichen (P42)	
	1 0 0 nach empfangenen Startzeichen(P41), Moduladresse(P40) und Empfangs-Endezeichen(P42)	
	1 0 1 3964-Prozedur	
44	Baudrate 0=150 3=1200 6=9600 9=57600 1=300 4=2400 7=19200 10=76800 2=600 5=4800 8=38400 11=115200	6
45	Parity / Stopbits / Zeichenlänge	0
	Bit 0 Anzahl der Stopbits: 0 = 1 Stopbit 1 = 2 Stopbit	
	Bit 1 Parity Enable: 0 = Disable 1 = Enable	
	Parity Select: 0/0 = Odd 1/0 = Even	
	Bit 2/ Bit 3 0/1 = Mark 1/1 = Space	
	Zeichenlänge: Bit 4/ 0/0 = 8Bit 1/0 = 7Bit	
	Bit 5 0/1 = 6Bit 1/1 = 5Bit	
	Bit 6 ohne Funktion	
	Bit 7 ohne Funktion	
46	Anzahl zu empfangender Zeichen 1 - 255	1
47	Sende-Endezeichen 0 - 255 bzw. Wiederholfaktor bei 3964-Prozedur	13

48	Sende-Protokoll 0: es wird kein Endezeichen automatisch gesendet 1: es wird CR automatisch gesendet 2: es wird CR LF automatisch gesendet 3: es wird das Send-/ Endezeichen (P97) automatisch gesendet 4: es wird das CR LF und das Send-/ Endezeichen (P97) automatisch gesendet 5: es wird das Empfang-Startzeichen (P91) und das Empfang-Endezeichen (P92) automatisch gesendet	1
49	Timeout-Wert 0 - 255 (1 = 0.1s) 0 = keine Timeout-Überwachung	0

Parameter der RS232/1-Schnittstelle (#0)

Nr.	Funktion	Standard
50	Geräte-Adresse 0 - 255	0
51	Empfangs-Start-Zeichen 0 - 255	2
52	Empfangs-Ende-Zeichen 0 - 255 (für COMPAX ist der Wert 62 erforderlich)	13
53	Empfangs-Protokoll	34
	Bit Funktion	
	0 ProtokollBit 0	0
	1 ProtokollBit 0	1
	2 ProtokollBit 0	0
	3 -	0
	4 Block-Check-Character; 0=off, 1=on	0
	5 Auto-RTS; 0 = off, 1 = on	1
	6 Hardware-Handshake; 0 = off, 1 = on	0
	7 Software-Handshake; 0 = off, 1 = on	0
	Mit den ProtokollBits wird eines der folgenden Empfangsprotokolle ausgewählt:	
	P-Bit INPUT-Rdy=1,	
	2 1 0	0 1 0
	0 0 0 nach jedem empfangenen Zeichen	
	0 0 1 nach def. Anzahl empfangener Zeichen (P56)	
	0 1 0 nach empfangenem Endezeichen (52)	
	0 1 1 nach empf. Start (P51)- u. Endezeichen (P52)	
	1 0 0 nach empfangenen Startzeichen(P51), Moduladresse(P50) und Empfangs-Endezeichen(P52)	
54	Baudrate 0=150 3=1200 6=9600 9=57600* 1=300 4=2400 7=19200 10=76800* 2=600 5=4800 8=38400 11=115200*	8
55	Parity und Stopbits 0 = ohne Parity, 1 Stopbit 1 = ohne Parity, 2 Stopbit 2 = mit Parity EVEN, 1 Stopbit 3 = mit Parity ODD, 1 Stopbit	0
56	Anzahl zu empfangender Zeichen 1 - 255	1
57	Sende-Endezeichen 0 - 255	13

58	Sende-Protokoll 0: es wird kein Endezeichen automatisch gesendet 1: es wird CR automatisch gesendet 2: es wird CR LF automatisch gesendet 3: es wird das Sende-/ Endezeichen (P57) automatisch gesendet 4: es wird das CR LF und das Sende-/ Endezeichen (P57) automatisch gesendet 5: es wird das Empfang-Startzeichen (P51) und das Empfang-Endzeichen (P52) automatisch gesendet	1
59	Timeout-Wert 0 - 255 (1 = 0.1s) 0 = keine Timeout-Überwachung	0

* siehe Seite 83.

Parameter der RS485/1-Schnittstelle (#1)

Nr.	Funktion	Standard	
60	Geräte-Adresse 0 - 255	0	
61	Empfangs-Start-Zeichen 0 - 255	2	
62	Empfangs-Ende-Zeichen 0 - 255	13	
63	Empfangs-Protokoll	2	
	Bit	Funktion	
	0	ProtokollBit 0	0
	1	ProtokollBit 1	1
	2	ProtokollBit 2	0
	3		0
	4	Block-Check-Character; 0 = off, 1 = on	0
	5		1
	6		1
	7	Software-Handshake; 0 = off, 1 = on	0
Mit den Protokollbits wird eines der folgenden Empfangsprotokolle ausgewählt:			
P-Bit	INPUT-Rdy = 1,		
2 1 0		0 1 0	
0 0 0	nach jedem empfangenen Zeichen		
0 0 1	nach def. Anzahl empfangener Zeichen (P66)		
0 1 0	nach empfangenem Endezeichen (P62)		
0 1 1	nach empfangenem Start(P61)- und Endezeichen (P62)		
1 0 0	nach empfangenen Startzeichen (P61), Geräteadresse (P60) und Empfangs-Endzeichen (P62)		
64	Baudrate	6	
	0=150 3=1200 6=9600 9=57600		
	1=300 4=2400 7=19200 10=172800		
	2=600 5=4800 8=28800 11=345600		

65	Stopbits/Parity/Hardware Bit 0...2 = Parity/Stopbit-Auswahl 0 ohne Parity, 1 Stopbit 1 ohne Parity, 2 Stopbit 2 mit Parity Even, 1 Stopbit 3 mit Parity Odd, 1 Stopbit Bit 3...5 = ohne Funktion Bit 6 = 4-Draht Master/Slave 0 Slave (TxD = Pin 2 und 7; RxD = Pin 1 und 6) 1 Master (TxD = Pin 1 und 6; RxD = Pin 2 und 7) Bit 7 = 2/4-Draht 0 2-Draht 1 4-Draht	0	
66	Anzahl zu empfangender Zeichen 1 - 255	1	
67	Sende-Endezeichen 0 - 255	13	
68	Sende-Protokoll 0: es wird kein Endezeichen automatisch gesendet 1: es wird CR automatisch gesendet 2: es wird CR LF automatisch gesendet 3: es wird das Sende-/ Endezeichen (P67) automatisch gesendet 4: es wird das CR LF und das Sende-/ Endezeichen (P67) automatisch gesendet 5: es wird das Empfang-Startzeichen (P61) und das Empfang-Endzeichen (P62) automatisch gesendet	1	
	69	Timeout-Wert der ENTER 1 - Anweisung 0 - 255 (1 = 0.1s) 0 = keine Timeout-Überwachung	0

Parameter der Feldbus - Schnittstelle

Nr.	Funktion	Standard	
70	Maximaladresse des Feldbusverbunds. In der Initialisierungsphase prüft COMTAC ab der Adresse 1 bis zur Maximaladresse ob die Geräte am Bus angeschlossen sind.	10	
71	Feldbus-Interrupt-Maske Bit 0 = device timeout 0=off, 1=on Bit 1 = device event 0=off, 1=on Bit 2 = device error 0=off, 1=on Bit 3 = device data 0=off, 1=on Bit 4 = device write response 0=off, 1=on Bit 5 = disable error message 21 Bit 6 = disable error message 27 Bit 7 = disable auto device write response check	0	
	72	Feldbus-Zykluszeit 0...255(1 = 1ms) 0: Baudrateunabhängige Standardwerte	0
	73	Feldbusprotokoll 0: kein Feldbus-Betrieb 1: Feldbus Master	1
	74	Feldbus-Baudrate 0=150 3=1200 6=9600 9=57600 1=300 4=2400 7=19200 10=172800 2=600 5=4800 8=28800 11=345600	11

75	Befehlswiederholung (Feldbusprotokoll Master) 0...250 COMTAC wiederholt automatisch eine Übertragung an ein Slave-Gerät soviel mal wie hier angegeben, bevor eine Fehlermeldung ausgegeben wird.	10
76	max. Übertragungsfehlerquote (Feldbusprot. Master) Erlaubte Anzahl von Fehlübertragungen an einen Slave. Treten hintereinander mehr Fehlübertragung auf, als hier angegeben, wird eine Fehlermeldung ausgegeben.	10
77	Ein/Ausschalten der Feldbus-Initialisierungsanzeige = 0 --> Anzeige ON = 1 --> Anzeige OFF	0
78	reserviert	0
79	0: keine Time-out Überwachung 1-255: Time-out 0,1 - 25,5 sec.	

84	Baudrate 0=150 3=1200 6=9600 9=57600 1=300 4=2400 7=19200 10=76800 2=600 5=4800 8=38400 11=115200	6		
85	Parity / Stopbits / Zeichenlänge	0		
	Bit 0 Anzahl der Stopbits: 0 = 1 Stopbit 1 = 2 Stopbit			
	Bit 1 Parity Enable: 0 = Disable 1 = Enable			
	Parity Select: 0/0 = Odd 1/0 = Even 0/1 = Mark 1/1 = Space			
	Zeichenlänge: Bit 4/ 0/0 = 8Bit 1/0 = 7Bit Bit 5 0/1 = 6Bit 1/1 = 5Bit			
	Bit 6 ohne Funktion			
	Bit 7 2/4-Draht 0 = 2-Draht 1 = 4-Draht			
	86		Anzahl zu empfangender Zeichen 1 - 255	1
87	Sende-Endezeichen 0 - 255	13		
88	Sende-Protokoll 0: es wird kein Endezeichen automatisch gesendet 1: es wird CR automatisch gesendet 2: es wird CR LF automatisch gesendet 3: es wird das Send-/ Endez.(P67) autom. gesendet 4: es wird das CR LF und das Send-/ Endez.(P67) autom. gesendet 5: es wird das Empfang-Startz. (P61) und das Empfang-Endez. (P62) autom. gesendet	1		
	89		Timeout-Wert der ENTER 1 - Anweisung 0 - 255 (1 = 0.1s) 0 = keine Timeout-Überwachung	0

22.3 Parameter der RS485/2-Schnittstelle (#3) (Option F6)

Nr.	Funktion	Standard	
80	Geräte-Adresse 0 - 255	0	
81	Empfangs-Start-Zeichen 0 - 255	2	
82	Empfangs-Ende-Zeichen 0 - 255 (für COMPAX ist der Wert 62 erforderlich)	13	
83	Empfangs-Protokoll	2	
	Bit Funktion		
	0 ProtokollBit 0	0	
	1 ProtokollBit 1	1	
	2 ProtokollBit 2	0	
	3	0	
	4 Block-Check-Character; 0 = off, 1 = on	0	
	5	1	
	6	1	
	7 Software-Handshake; 0 = off, 1 = on	0	
	Mit den ProtokollBits wird eines der folgenden Empfangsprotokolle ausgewählt:		
	P-Bit INPUT-Rdy = 1,		
	2 1 0	0 1 0	
	0 0 0 nach jedem empfangenen Zeichen		
0 0 1 nach def. Anzahl empfangener Zeichen (P66)			
0 1 0 nach empfangenem Endezeichen (P62)			
0 1 1 nach empf. Start(P61)- und Endezeichen (P62)			
1 0 0 nach empfangenen Startzeichen (P61),Geräteadresse (P60) und Empfangs-Endezeichen (P62)			

23.3 Parameter der RS232/2-Schnittstelle (#2)

Nr.	Funktion	Standard	
90	Geräte-Adresse 0 - 255	0	
91	Empfangs-Start-Zeichen 0 - 255	2	
92	Empfangs-Ende-Zeichen 0 - 255	13	
93	Empfangs-Protokoll	34	
	Bit Funktion		
	0 ProtokollBit 0	0	
	1 ProtokollBit 1	1	
	2 ProtokollBit 2	0	
	3 -	0	
	4 Block-Check-Character; 0 = off, 1 = on	0	
	5 Auto-RTS; 0 = off, 1 = on	1	
	6 Hardware-Handshake; 0 = off, 1 = on	0	
	7 Software-Handshake; 0 = off, 1 = on	0	
	Mit den ProtokollBits wird eines der folgenden Empfangsprotokolle ausgewählt.		
	P-Bit INPUT-Rdy = 1,		
	2 1 0	0 1 0	

	0 0 0	nach jedem empfangenen Zeichen	
	0 0 1	nach def. Anzahl empfangener Zeichen (P96)	
	0 1 0	nach empfangenem Endezeichen (P92)	
	0 1 1	nach empf. Start(P91)- und Endezeichen (P92)	
	1 0 0	nach empfangenen Startzeichen(P91), Moduladresse(P90) und Empfangs-Endezeichen(P92)	
94	Baudrate	3=1200 6=9600 9=57600* 4=2400 7=19200 10=76800* 5=4800 8=38400 11=115200*	8
95	Parity und Stopbits	0 = ohne Parity, 1 Stopbit 1 = ohne Parity, 2 Stopbit 2 = mit Parity EVEN, 1 Stopbit 3 = mit Parity ODD, 1 Stopbit	0
96	Anzahl zu empfangender Zeichen	1 - 255	1
97	Sende-Endezeichen	0 - 255	13
98	Sende-Protokoll	0: es wird kein Endezeichen automatisch gesendet 1: es wird CR automatisch gesendet 2: es wird CR LF automatisch gesendet 3: es wird das Sende-/ Endez.(P97) autom. gesendet 4: es wird das CR LF und das Sende-/ Endez.(P97) autom. gesendet 5: es wird das Empfang-Startz. (P91) und das Empfang-Endez. (P92) autom. gesendet	1
99	Timeout-Wert	0 - 255 (1 = 0.1s) 0 = keine Timeout-Überwachung	0
100	Terminal-Schnittstellenzuordnung	0 = RS232/1 ist die Terminalschnittstelle 1 = RS485/1 ist die Terminalschnittstelle 2 = RS232/2 ist die Terminalschnittstelle	0

* siehe Seite 83.

30. Sachwortverzeichnis

Abfrage		CPXSPEED.....	101
auf "Maschinennull erreicht".....	106	CPXST.....	102
auf "Position erreicht".....	106	CPXSTA.....	102
der COMTAC-Folien-Tastatur.....	55	CPXSTEXT.....	101
der Terminal-Tastatur.....	51	CPXSTS.....	104
ABS.....	19	CR.....	48
Abspeichern und Laden von		Cursor	
Programmen und Daten.....	30	Darstellung.....	48
Addition.....	19	CURSOR.....	45
Adresse des COMPAX, das den		Cursor Bildschirm	
ONCPXERR-Interrupt ausgelöst hat		Darstellung.....	45
.....	107	Position.....	45
aktuelle Uhrzeit.....	57	Cursor Display	
aktuelles Datum.....	58	Darstellung.....	47
Allgemeine Anweisungen.....	36	Position.....	47
Analogausgänge beschreiben.....	70	Cursor positionieren.....	48
Analoge Ein-/Ausgänge.....	70	Cursor verschieben.....	48
Analoge Eingänge abfragen.....	70	CURSOR_D.....	47
AND.....	20	DATA.....	38
Änderungen.....	5	Data-Text-Editor.....	88
Arbeitsspeicher Belegung.....	28	Data-Text-Feld	
Arithmetische Operatoren /		Zahlenwert zuweisen.....	89
Funktionen.....	19	Zeile auslesen.....	88
ASC.....	61	Zeilenlänge ändern.....	88
ASK.....	52	Zeilenlänge lesen.....	88
ASKKBD.....	55	DATA-TEXT-FELD.....	88
ATN(x).....	20	DATE.....	58
Ausgabe		DATED.....	58
im COMTAC-Display.....	46	Dateiname verändern.....	32
über die Feldbus-Schnittstelle.....	95	Dateinamen.....	31
über RS232.....	85	DATEM.....	58
über RS485.....	81	Datenfeld anlegen.....	38
Ausgabeformate.....	48	Datenfeld lesen.....	38
Auslesen des Data-Text-Felds.....	89	Daten-File.....	31
AUTO.....	25	Daten-Format und Genauigkeit....	17
Autostartfunktion.....	35, 109	DATEW.....	58
AUTO-START-Funktion.....	33	DATEY.....	58
Balkengrafik.....	44	Datum lesen/setzen.....	58
BARGRAPH.....	44	Datumsausgabe.....	35, 109
Basic-Zeilen		DB.....	46
kopieren.....	25	DB.@.....	43
löschen.....	26	DEBUG-Funktion.....	33
neu numerieren.....	27	DEL Basic-Zeilen.....	26
BCC		DEL Datei.....	32
RS232.....	85	DEL DIM.....	37
RS485.....	80	DEL Variable.....	37
BCDIN.....	64	DH.....	46
BCDOUT.....	66	digitale Ausgänge setzen.....	65
bedingte Programmverzweigung.....	40	digitale Ausgänge zurücksetzen..	65
BEEP.....	44	Digitale Ein/Ausgänge.....	63
Befehlskürzel.....	19	digitalen Ausgang zeitgesteuert	
Befehlsquittierung von COMPAX.....	99	beschreiben.....	68
Befehlssyntax.....	6	digitaler Eingang einlesen.....	63
Befehlsübersicht.....	7	digitales Ausgangswort schreiben	65
Begriffe.....	18	DIM.....	36
Beispielprogramm.....	108	DIR.....	31
Bereit-Eingang abfragen.....	69	DIRDIM.....	37
Betrag.....	19	DISABLE / ENABLE ONCPXERR	
Betriebsarten.....	17	107
Bildschirm löschen.....	43	Disketten formatieren.....	31
Bildschirmaufbau.....	23	DISP.....	46
Bildschirmausgabe in der Zeile 22		DISPCPXP.....	100
.....	43	DISPCPXS.....	100
Binäre Terminalausgabe.....	44	DISPCPXZ.....	100
BINOUT.....	65	Display löschen.....	46
Bit-Abfrage.....	20	Display-Zeile.....	24
Block-Check-Character.....	85		
Carriage Return ausgeben.....	48		
CHAR DELETE.....	25		
CHAR INSERT.....	25		
Checksumme.....	35, 109		
Checksummenbildung.....	30		
CHR\$.....	61		
CLEAR.....	36		
CLEAR ONInterrupt.....	73		
CLEAR SPACE.....	25		
CLEARD.....	46		
CLEARI.....	36, 72		
CLEAR\$.....	36		
CLEART.....	43		
CLRLD.....	47		
CLROUT.....	65		
COMPAX			
Ausgänge beschreiben.....	104		
Ausgänge maskieren.....	104		
Ausgangswort (A1..A16) lesen.....	104		
Befehle.....	100		
Eingänge lesen.....	105		
Eingangs-Funktionen E1 bis E16 von			
COMTAC auslösbar machen.....	103		
Eingangs-Funktionen E1 bis E16 von			
COMTAC auslösen.....	103		
Istposition einlesen.....	105		
Parameter			
Beschreibung lesen.....	101		
Parameter lesen.....	100		
Status			
Beschreibung lesen.....	101		
Status lesen.....	100		
Status zyklisch lesen.....	100		
COMTAC-Basic Schlüsselwörter.....	18		
CONT.....	28		
CONT DISP.....	54		
CONT ONInterrupt.....	72		
CONTROL.....	34		
Controlwörter.....	5		
COPY Basic-Zeilen.....	25		
COPY Datei.....	32		
COPY Disk.....	32		
COPYDEL.....	26		
COS(x).....	20		
CPX.....	107		
CPXACCEL.....	101		
CPXBK.....	102		
CPXBOFF.....	103		
CPXCTR.....	103		
CPXERRADR.....	107		
CPXIMASK.....	103		
CPXINP.....	105		
CPXOFF.....	103		
CPXOMASK.....	104		
CPXON.....	102		
CPXOUT.....	104		
CPXOVR.....	105		
CPXPARA.....	102		
CPXPH.....	101		
CPXPOS.....	105		
CPXPOSA.....	101		
CPXPOSR.....	101		
CPXPTEXT.....	101		
CPXQT.....	102		
CPXSP.....	102		

DISPVAR.....	46	Parameter lesen / beschreiben.....	97	KEYSWITCH.....	68
DIV.....	19	Schnittstelle.....	94	Kommentar-Zeilen.....	39
Division.....	19	Status.....	96	Labels.....	22
Divisionsrest.....	19	zyklische Ausgangsdaten lesen /		LED des COMTAC einschalten ..	47
DO ... UNTIL.....	41	beschreiben.....	97	Leerzeichen ausgeben.....	48
DO ... WHILE.....	42	zyklische Eingangs-Daten lesen ..	97	LEN.....	61
Drehzahlreduzierung bei COMPAX		Feldbus E/A-Belegung.....	63	LEN().....	29
.....	105	Floppy-Baudrate.....	34, 109	LINE DELETE.....	25
Druckerausgabe.....	43	FOR ... NEXT.....	42	LINE ERASE.....	25
DTFEDIT.....	88	Format.....	31	LINE INSERT.....	25
DTFLCNT.....	88	Formatbefehle.....	48	Linien zeichnen.....	44
DTFLLEN.....	88	Formatwandlung.....	61	LIST.....	26
DTFNFCT.....	88	FREE.....	28	LOAD.....	32
DTFVFCT.....	89	Freigabe/Sperren		LOG.....	19
DYNOUT.....	68	des ONCPXERR-Interrupt.....	107	Logische Operatoren.....	20
Echtzeituhr und Zähler.....	57	des RS485 - Interrupt.....	82	Maßstab für die Balkengrafik-	
EDITVAR.....	53	des Tastatur-Interrupt.....	56	Funktion ausgeben.....	44
Einerkomplement.....	19	des Uhrzeit Interrupt.....	58	Matrizen dimensionieren.....	36
Eingabe über die COMTAC-		Funktionstasten		MOD.....	19
Folientastatur.....	53	Anwendung der.....	52	MTOP.....	29
Eingabe-Bereich.....	24	Funktionstasten-Belegung.....	24	Multiplikation.....	19
Empfangen RS485.....	80	ganzzahliger Anteil.....	19	natürlicher Logarithmus.....	19
EMY_STOP.....	68	ganzzahliger Teil einer Division ..	19	NEW.....	25
Enable.....	72	GCHR.....	45	NOT.....	19
ENABLE [EN.]/DISABLE [DI.]		Gerätezuordnung.....	5	Not Stop Eingang abfragen.....	68
ONRDY.....	69	GET.....	51	Not-Stop Eingang.....	75
ENABLE/DISABLE		GOSUB.....	40	Numerische Variablen.....	17
Feldbus.....	98	GOTO.....	28	NV-Ram.....	35
ONERR.....	91	GOTO Zeilennummer.....	40	ODER-Verknüpfung.....	20
ONINP.....	65	Grafikzeichen ausgeben.....	45	OFFTIMER.....	59
ONKBD.....	56	Hexadezimale Terminalausgabe.....	43	ON.....	40
ONKEY.....	54, 68, 69	HLINE.....	45	ON#0 / #2.....	86
ONTIME.....	58	HOME.....	25	On#1 Feldbus.....	98
ONTIMER.....	59	IDLE.....	77	ON#1/#3 RS485.....	82
RS232.....	86	IF - THEN - (ELSE).....	41	ONCPXERR.....	106
RS485.....	82	IN (digitaler Eingang).....	63	ONEMY.....	69
STOP.....	28	Indizierte Variable.....	17	ONERR.....	91
ENTER (RS232).....	86	Info-Bereich.....	24	ONINP.....	64
ENTER Feldbus.....	95	Inhaltsverzeichnis (Directory) des		ON-Interrupt.....	40
ENTER RS485.....	81	Speichermediums.....	31	ONKBD.....	56
ERRSTS.....	91	INPKBD.....	53	ONKEY.....	54
ERRSTS#.....	91	INPKBD TABXY.....	53	ONKEY 19.....	68
ERRSTS\$.....	91	INPUT (Tastatureingabe).....	51	ONRDY.....	69
ERRSTSL.....	91	INPUT-TABXY.....	51	ONTIME.....	57
EXKLUSIV-ODER-Verknüpfung..	20	INT.....	19	ONTIMER.....	59
EXP.....	19	Interner Zähler freigeben/sperren	59	OR.....	20
Exponentialfunktion.....	19	Interner Zähler lesen/setzen.....	59	OUTPUT (RS232).....	85
FBDINFO.....	99	Interrupt		Output Feldbus.....	95
FBDRES.....	99	Bereit Eingang.....	76	Output RS485.....	81
FBDRSP.....	99	COMPAX Fehler.....	75	PAGE ERASE.....	25
FBDSRQ.....	98	Dig. Eingang.....	77	Parameter.....	34
FBINTADR Feldbus.....	98	Echtzeit-Uhr.....	76	Parameter - Übersicht.....	34, 109
FBINTREG Feldbus.....	98	Fehler.....	75	Parameter der Feldbus -	
FBUPD0.....	99	Funktionstasten.....	77	Schnittstelle.....	94, 111
FBUPD1.....	99	Schnittstellen.....	76	Parameter der Feldbus-Schnittstelle	
FBUS_D.....	97	Timer.....	76	78, 111
FBUS_I.....	97	Interrupt		Parameter der RS232/2-	
FBUS_O.....	97	löschen.....	36	Schnittstelle.....	84, 112
FBUS_P.....	97	Verarbeitung.....	71	Parameter der RS232/3-	
FBUS_S.....	96	Interrupt Initialisierung.....	72	Schnittstelle.....	84, 110
Fehler		Interrupt Quellen.....	71	Parameter der RS232-Schnittstellen	
Behandlung.....	91	Interrupt rücksetzen.....	72	83, 110
Beschreibung.....	92	Interrupt Überwachung.....	72	Parameter der RS485/2.....	79, 112
Nummer des zuletzt aufgetretenen		Interrupt Ursache die den ON#1-		Parameterübergabe an ein	
Fehlers.....	91	Interrupt ausgelöst hat.....	98	Unterprogramm.....	38
Zeilen-Nummer der Basic-Zeile mit		Interrupt:.....	76	Passwortschutz.....	35
Fehler.....	91	Interrupt-Priorität.....	73	PB.....	44
Feldbus		Interrupt-Prioritäten.....	75	PFEILE.....	25
Adresse des Feldbus-Gerätes, das		Interrupt-Rücksprung.....	73	PH.....	43
den ON#1-Interrupt ausgelöst hat ..	98	Interrupt-Verzweigung.....	72	PH.@.....	44
Freigabe/Sperren des Feldbus -		Interrupt-Verzweigung freigeben ..	72		
Interrupts.....	98	KBDCODE.....	55		

PI	20	RETI	41, 73	TIMEH	57
POP	38	RETURN	41	TIMEM	57
Potenzfunktion	19	RND	19	TIMER	59
Power-ON-Wert der Ausgänge34, 109		RS232	83	TIMES	57
Power-ON-Wert von A1 ... A1634, 109		RS485	78	Trigonometrische Funktionen.....	20
PRINT	43	RS485-Parameter	78, 111	Überwachungszeit.....	35
PRINT@	43	RS485-Schnittstelle	78	Uhranzeige	25
Programm		RUN	27	UMEM	89
kopieren	32	SCALE	44	UMEM\$	61
laden	32	Schlüsselschalter abfragen	68	UMEM\$(Adresse,Länge).....	90
löschen	32	Schnittstellenzuordnung		UMEM\$(Zeile)	89
schrittweise abarbeiten	28	Diskettenlaufwerk	34, 109	UMEMB	89
speichern	31	Schlüsselschalter	68	UND-Verknüpfung.....	20
Start	27	Senden über RS485	80	unterbrochenes Programm	
Programmerstellung	23	SETLED	47	fortsetzen	28
Programm-File	31	SETOUT	65	Unterprogramm aufrufen.....	40
Programmiersprache	17	Setzen der aktuellen Uhrzeit	57	Unterprogramme	21
Programmschleife.....	41	SGN	19	USING [U.] (B)	49
Programmsprung.....	40	SIN(x)	20	USING(#.#)	49
Programmverschachtelung.....	22	SLOPE	70	USING(0)	49
Programmverzweigung.....	41	Softwareversionen.....	5	USING(Fx).....	49
bei COMPAX-Fehler	106	SPC	48	USING-Befehle speichern	48
bei Fehler	91	Speicheraufteilung.....	30	VAL	62
durch Interrupt.....	41	Speichermedien/Dateinamen	30	VAL\$	62
über den Bereit-Eingang	69	Speicherung von Zeichenketten..	60	Variablen auf 0 setzen	36
über den internen Zähler	59	SQR.....	19	Variablenamen	17
über die Tastatur.....	56	Stack zurücksetzen	36	Verarbeitung von Zeichenketten ..	60
über einen digitalen Eingang.....	64	Status über Feldbus lesen.....	96	Vergleich von Zeichenketten.....	61
über Funktionstasten	54	Statusinformation der RS232-		Vergleichende und logische	
über Interrupt-Meldung eines		Schnittstellen	87	Operatoren	20
Feldbusgerät	98	Statusinformation der RS485	82	Verkettung von Zeichenketten	60
über Not-Stop-Eingang	69	STOP	39	Verzweigung sperren	72
wenn ein String im Zwischenpuffer der		STOP DISP	54	VIN	70
RS485-Schnittstelle bereitsteht	82	STORE	31	VLINE	44
Zeitgesteuert.....	57	Stringlänge	61	Vordefinierte Zeichenketten	61
Programmverzweigungen und		STSCTR#0 /#2/#4	87	Vorzeichen	19
Programmschleifen.....	40	STSCTR#1	82	VOUT	70
PSTEP	28	STSCTR#1/#3	82	WAIT	39
PUSH	38	SUB	21	WAITMN.....	106
Quadratwurzel	19	Subtraktion	19	WAITPOS	105, 106
RAD	20	Systemvariable	28	Warten auf "Maschinennull erreicht"	
Rangfolge der Operatoren und		TAB	48	106
Funktionen.....	20	TABXY	48	Warten auf "Position erreicht"	105
RDY	69	TAN(x)	20	Warten auf Interrupt.....	77
READ	38	Tastatureingabe	51	Warten auf 'Maschinen-Null	
Rechteck zeichnen	45	Tastatureingabe ohne		angefahren'	106
RECTANGLE.....	45	Programmunterbrechung	53	Wartezeit programmieren	39
REM	39	Tastencode.....	55	XOR	20
REN	27	Teilzeichenketten	60	XTAL	29
RENAME	32	Terminal Tastatur-Beep.....	44	Zahlenformat definieren	49
Repeat-Zeit für COMTAC-Tastatur		Terminal/Folientastatur-Eingabe .	53	Zeichenketten oder "Strings".....	60
.....	34, 109	Terminalausgabe.....	43	ZP-RAM formatieren	31
REQOUT	67	Terminal-Ausgabe	46	Zufallszahl	19
RESET Feldbus	99	Terminal-Eingabe	51	Zwischenspeichern von Variablen38	
RESET RS232.....	87	Terminal-Schnittstellenzuordnung		Zykluszeiten Feldbus-Schnittstelle	
RESET RS485.....	82	84, 113	95
RESTORE	38	Terminal-Typ	34, 109		
		TIME	57		