D S - Kultur

C O M P U T E R - M A G A Z I N
E
M

* *

P A S C A L

für Kleincomputer

Version 2.1

Kurzdokumentation des PASCAL - Compilers

für KC 85/1-4 , KC 87 und Z-1013

Autoren:

Doz. Dr. sc. Wolfgang Burmeister

Dipl. - Math. Manfred Lehmann

Dr. Klaus Vetters

F e b r u a r 1990

D R E S D E N

1. Laden des Compilers, Speicherkonfiguration

Die Arbeit mit der vorliegenden Version 2.1 des PASCAL-Compilers für Kleincomputer ist mit den KC 85/1-4 und KC 87 in der Grundausstattung möglich. Nur für umfangreichere Programme ist die Verwendung von RAM-Erweiterungsmodulen erforderlich. Das Laden von der Kassette erfolgt durch das Kommando

bei KC 85/1 und KC 87 : PASCAL
bei KC 85/2-4 : LOAD

Nach beendetem Ladevorgang startet der Compiler automatisch. Auf dem Bildschirm erscheint

Top of RAM? Es handelt sich um den RAM-Top für den Compiler. Sie haben jetzt
ENTER oder eine Adresse aus dem aktiven RAM oberhalb von
12400 einzugeben. Die Eingabe von ENTER wirkt wie die Eingabe von 16384. Danach erscheint

Top of RAM for 'T'? Es handelt sich um den RAM-Top für das zu erzeugende Objektcode-Programm. Sie können jetzt
ENTER oder eine Zahl oberhalb von
4000 eingeben. Die Eingabe von ENTER bewirkt die Übernahme des Top of RAM als Top of RAM for 'T'.
Als nächste Frage erscheint

Table Size? Gemeint ist die Länge des internen Arbeitsspeichers, die von der Anzahl der im Programm benutzten Variablen, Prozeduren, Funktionen etc. abhängt. Jetzt kann
ENTER oder eine angemessen große Zahl zwischen
0 und 20000 eingegeben werden. Die Eingabe von ENTER wirkt wie die Eingabe von
(Top of RAM - 12104) DIV 16.

Hinweise

a) Antworten Sie auf alle drei Fragen mit ENTER, so steht Ihnen für Quelltext und Objektcode der Speicher von 12370 bis 16383 zur Verfügung. Der interne Arbeitsspeicher des Compilers ist 266 Byte lang.

b) Auf die Frage Top of RAM? kann in jedem Fall mit ENTER, bei einem bzw. zwei 16k-RAM-Erweiterungsmodulen mit maximal 32768 bzw. 49152 geantwortet werden. Falls beim KC 85/2-4 ein vollständiger 64k-RAM-Bereich zur Verfügung steht, so kann maximal 65535 angegeben werden.

c) Die angegebenen unteren Schranken besitzen nur theoretischen Wert und lassen sich nur bei extrem kurzen Programmen anwenden.

d) Wollen Sie die Größen für Top of RAM, Top of RAM for 'T' oder Table Size ändern, so verlassen Sie den Compiler mit dem Kommando B und gelangen über das Kommando PASNEW wieder in die Initialisierungsphase.

2. Beschreibung des Sprachumfangs

2.1. PASCAL - Symbole

Bezeichner (Identifier) müssen mit einem Buchstaben beginnen und können eine Folge von Klein- und Großbuchstaben und Ziffern enthalten, z.B.

a , B , a1B .

Bezeichner, die sich in den ersten 10 Zeichen nicht unterscheiden, werden als identisch angesehen.

a) Arithmetische und logische Verknüpfungen

Ergibtzeichen	:=			
Real- und Integerzahlen	+	-	*	/
Integerzahlen	DIV		MOD	
Boole'sche Variable	OR		AND	NOT

b) Relationen

IN (für Set-Arithmetik)

c) Klammern, Trennzeichen etc.

Auf der Tastatur vorhandene Zeichen

( ) . , ; : ' # $

Weitere PASCAL-Sonderzeichen

Zeichen	Taste KC 87	Taste KC 85/2-4	Ersatzzeichen
[	SHIFT ü	SHIFT SPACE	(/
]	SHIFT ä	¬	/)
{	SHIFT <		(*
}	SHIFT >		*)
↑	^	^	

d) Programm-Grundstrukturen

PROGRAM BEGIN END FUNCTION PROCEDURE

e) Deklarationen

TYPE	LABEL	ARRAY ... OF ...	PACKED ARRAY ... OF ...
VAR	CONST	SET	RECORD ... END

Das vorliegende KC-PASCAL hat keine Routine zur Kompaktspeicherung von Feldern. Die Angabe von PACKED ist folglich wirkungslos. Mittels CONST können Konstanten vom Typ INTEGER, REAL, CHAR, BOOLEAN und ARRAY ... OF CHAR definiert werden. Mengen dürfen nicht mehr als 256 Elemente enthalten.

f) Schleifen

FOR ... := ... TO ... DO ... WHILE ... DO ...
FOR ... := ... DOWNTO ... DO ... REPEAT ... UNTIL ...

Der Abbruchtest wird bei den FOR-Anweisungen zu Beginn der Schleife überprüft, so daß das Übergehen der FOR-Anweisung möglich ist.

g) Verzweigungen

IF ... THEN ... ELSE ... CASE ... OF ... END
CASE ... OF ... ELSE ...

GOTO Das Sprungziel muß im gleichen Block liegen und mittels LABEL deklariert sein.

h) Records

WITH ... DO ...

i) Pointer (dynamische Variable)

NIL

k) Rekursiv gegenseitig definierte Funktionen und Prozeduren

FORWARD

2.2. Identifikatoren

a) Konstanten

FALSE TRUE MAXINT Es gilt MAXINT = 32767.

b) Type-Konstanten

INTEGER Mittels vorgesetztem # können INTEGER als Hexazahlen programmiert werden. In den WRITE-Anweisungen kann durch die Formatangabe :n:H die Ausgabe in Hexaform erreicht werden (n - Konstante oder Ausdruck vom INTEGER-Typ)

Beispiel: 1210D = 4BAH . Die Anweisung

WRITE(1210:n:H)

ergibt für n=1: A ... niederwertigstes Halbbyte (Nibble)
n=2: BA ... niederwertiges Byte
n=3: 4BA ... dreiziffrige Hexazahl
n≥4: rechtsbündiges Schreiben von 04BA im angegebenen Format.

REAL Die REAL-Zahlen werden in 4-Byte-Gleitkommaarithmetik verknüpft.

CHAR
BOOLEAN

c) Arithmetische, mathematische und Boole'sche Funktionen

ROUND(r)	FRAC(r)	ODD(i)	ENTIER(r)	COS(r)
TRUNC(r)	ABS(r)	SQR(r)	SQRT(r)	SIN(r)
EXP(r)	LN(r)	TAN(r)	ARCTAN(r)	

RANDOM Erzeugung von ganzzahligen Zufallszahlen z mit 0 ≤ z ≤ 255 (durch Abfrage des Refreshregisters).

EOLN Wird TRUE, falls das nächste einzulesende Zeichen das Zeilenende ist, sonst FALSE.

Beispiel: Eingabe einer Zeichenkette variabler Länge

```
i:=1;
REPEAT READ(a[i]); i:=i+1  UNTIL EOLN
```

d) Ordnungsfunktionen für Skalartypen

ORD(a) liefert Position des Skalarwertes a in der durch den Typ von a festgelegten Wertemenge

CHR(i) liefert ASCII-Zeichen, Steuerzeichen oder Graphikzeichen mit Code i

SUCC(a) Nachfolger von a
PRED(a) Vorgänger von a

Beispiele: SUCC(3) = 4, SUCC('B') = 'C',
PRED(TRUE) = FALSE

e) Verwaltung von dynamischem Speicherplatz

NEW(p) Erzeugung einer dynamischen Variablen p^
MARK(q) Merken des Endes der Halde
RELEASE(q) Freigabe der Halde zurück bis q

f) Ein- und Ausgabe

READ(A) READLN(A) WRITE(A) WRITELN(A)

Beispiel zur Formatsteuerung:

```
VAR x: REAL;  i,k: INTEGER;
...    WRITE(x:10:2,i:6,k:6:H) ...
```

PAGE Seitenvorschub am Drucker bzw. Löschen des Bildschirmes

INCH Tastaturabfrage, entspricht INKEY$ bei BASIC

TOUT(n,a,l) Schreiben auf Band ¦ Name n, Anfangsadresse a, ¦ Länge l in Byte

TIN(n,a) Laden vom Band ¦ Name n, Anfangsadresse a

Beispiel: TOUT('Feld v',ADDR(v),SIZE(v))

INP(p) Entspricht den Assembleranweisungen IN und OUT.
OUT(p,c) Die Zahl p ist eine Kanaladresse und c.eine Konstante oder Variable vom Typ CHAR.

Beispiel: Umschaltung auf 20-Zeilen-Format beim KC 87

```
OUT(136,CHR(12))
```

g) Speicherzugriff

SIZE(v) Länge der Variablen v in Byte

ADDR(v) Adresse des ersten Byte der Variablen v

POKE(n,v) Eintragen des Wertes der Variablen v ab Adresse n

PEEK(n,t) Beginnend bei Adresse n wird der Speicherinhalt entsprechend dem angegebenen Typ t ausgewertet. Typ t darf sein: INTEGER, CHAR, REAL oder ARRAY[1..k] OF ...

h) Benutzung von Maschinen-Code

INLINE(F) Einfügen der durch die Parameterfolge F definierten Code-Anweisungen. Als Parameter sind Ausdrücke vom Typ INTEGER oder CHAR zugelassen.

USER(n) Aufruf eines Code-Programms mit Adresse n

Beispiel: Ausgabe der System-Uhrzeit in Std:Min:Sec ab aktueller Kursorposition bei KC 85/1 u. KC 87

```
INLINE(14,17); USER(5);
INLINE(60,89,14,24); USER(5)
```

i) Programm-Abbruch

HALT beendet den Programmlauf mit Ausschrift des Befehlszählers.

3. Kommandos

3.1. Hauptkommandos

B Rückkehr zum Betriebssystem. Die erneute Aktivierung des Compilers erfolgt durch die Kommandos

PASNEW für neue Initialisierung oder
PASOLD für Beibehaltung der RAM-Tops, des Table-Size und des Quelltextes

C n — Übersetzen ab Zeile n bis Programmende. Nach fehlerfreier Übersetzung kann auf die Frage RUN? mit Y oder N geantwortet werden. Bei fehlerhaftem Text werden die Fehler angezeigt (s. Abschn. 5). Aus der Fehleranzeige kann durch E zum Editieren der aktuellen oder durch P zum Editieren der vorhergehenden Zeile übergegangen werden. Anhalten oder Unterbrechen der Übersetzung ist wie beim Kommando L möglich.

D n,m — Löschen der Zeilen n bis m

E n,m,f,s — Übergang zum Editor auf erstem Zeichen der Zeile n. Die gesamte Zeile n wird vorher ausgeschrieben. Weitere Kommandos s. Abschn. 3.3.

F n,m,f,s — Übergang zum Editor auf erstem Zeichen der ersten Zeichenkette f, die zwischen den Zeilen n und m vorkommt. Weitere Kommandos s. Abschnn. 3.3.

G,,f — Laden des mit P n,m,f gesicherten Quelltextes von der Kassette. Ist bereits Quelltext vorhanden, wird der gesamte Quelltext von Zeilennummer 10 an mit Schrittweite 10 neu numeriert.

I n,m — Automatische Erzeugung der Zeilennummern, beginnend bei n mit Schrittweite m. Durch STOP bzw. BRK kann dieser Modus beendet werden.

L n,m — Listen des Quelltextes von Zeile n bis Zeile m. Das Listen kann durch PAUSE angehalten werden, anschließend ist Fortsetzung oder Abbruch mit STOP bzw. BRK möglich.

M n,m — Die Zeile n wird zusätzlich als Zeile m eingefügt.

N n,m — Neunumerierung des gesamten Textes, beginnend mit Zeile n, Schrittweite wird m.

P n,m,f — Sichern des Quelltextes von Zeile n bis m auf der Kassette unter dem Namen f.

R — Starten des zuletzt mit C übersetzten Programms. Durch Syntaxfehler, Erweitern des Textes und das T-Kommando wird R abgeschaltet.

T n,,f — Übersetzen ab Zeile n bis Programmende und Sichern der Übersetzung unter dem Namen f. Nach fehlerfreier Übersetzung kann auf die Frage Ok? mit Y oder N geantwortet werden. Erst auf die Antwort Y erfolgt die Verbindung der Bibliothek (etwa 4 kByte) mit dem Objektprogramm. Es entsteht ein selbständig lauffähiges Codeprogramm, das nach Einlesen automatisch startet. Wiederholter Start ist vom Betriebssystem aus durch G möglich.

V Anzeige des Parameterspeichers in der Form

n m
f
s

X Liefert Anfang- und Endadresse des Quelltextes in Hexaform

3.2 Verkürzte und wiederholte Kommandos

Für die vier Kommandoparameter n,m,f,s existiert ein Speicher, der nach jeder Kommandoeingabe mit den neuen Parametern belegt wird. Im Kommando nicht belegte Parameter bleiben dabei unverändert. Anschließend erfolgt die Ausführung des Kommandos, wobei die Parameter aus dem Parameterspeicher verwendet werden. Nach der Initialisierung des Compilers hat der Parameterspeicher den Zustand

10 10 leere Zeichenkette leere Zeichenkette

Bei diesem Zustand des Parameterspeichers ist das Kommando

I gleichbedeutend mit dem Kommando I 10,10

Beachten Sie dabei: Wenn ein ausgelassener Parameter nicht der letzte ist, so muß das ihm folgende Komma gesetzt werden!

Aus diesen Regeln ergibt sich die Möglichkeit, bei oft verwendeten Kommandos die Parameterangabe abzukürzen.

Beispiel 1: Soll nur die Zeile n editiert werden und dabei keine Unterstützung durch Zeichenketten-Vorwahl erfolgen, so genügt das Kommando

E n

Beispiel 2: An mehreren Prozedurköpfen sollen Änderungen vorgenommen werden.

Kommando: F n,m,PROCEDURE

Nach beendeter Korrektur der so aufgerufenen Zeile kann mit dem Editor-Unterkommando F zum nächsten Prozedurkopf übergegangen werden.

<u>Ausnahmen:</u>

C Übersetzen ab niedrigster Zeilennummer

D n,m Verkürzung nicht möglich

E n weitere Verkürzung nicht möglich

L n,m Weglassen von n bewirkt das Listen ab niedrigster Zeilennummer, Weglassen von m das Listen bis Textende

3.3. Unterkommandos des Editors

Nachdem durch E... oder F... das Editieren eröffnet ist, stehen folgende Unterkommandos zur Verfügung.

Space bewirkt die zeichenweise Übernahme aus der alten in die neue Zeile.

--> übernimmt bis zur nächsten Tabulatorposition.

<-- baut die neue Zeile wieder zeichenweise ab.

C stellt auf Korrigieren um. Der Kursor ist jetzt '+'. Die alten Zeichen werden durch neue Überschrieben.
<-- rückt löschend zurück
ENTER beendet das Unterkommando C

F sucht die nächste Zeichenkette f, der Kursor wird auf die erste Position dieser Zeichenkette gesetzt.

I stellt um auf Zeichen einfügen. Der Kursor ist jetzt '*'. Auf Kursorposition kann ein Zeichen eingefügt werden.
--> Erzeugung von Leerzeichen
<-- rückt löschend zurück
ENTER beendet das Unterkommando I

K löscht das Zeichen auf Kursorposition. Mehrmaliges Drükken von K bewirkt die Löschung der nächsten Zeichen (ohne Reaktion auf dem Bildschirm).

L schreibt die aktuelle Zeile aus, übernimmt dabei bereits getätigte Korrekturen und stellt den Kursor auf das erste Zeichen dieser Zeile.

Q beendet die Korrektur der Zeile, ohne deren Korrekturen zu übernehmen und beendet die Arbeit mit dem Editor.

R wie L, die bereits getätigten Korrekturen werden aber nicht übernommen.

S setzt ab erster Position der zuletzt gefundenen Zeichenkette f die Zeichenkette s ein und geht zum Unterkommando F über.

X stellt um auf Weiterschreiben am Ende der Zeile. Kursor und weitere Funktionen wie bei I

Z löscht ab Kursorposition bis zum Zeilenende.

ENTER beendet die Korrektur der Zeile und die Arbeit mit dem Editor.

4. Kontrollfunktionen des Compilers

Der Compiler hat eine Reihe von Kontrollfunktionen, die abgeschaltet werden können, um die Bearbeitungsgeschwindigkeit zu erhöhen. Sie werden geschaltet durch das entsprechende Funktionszeichen mit vorangestelltem $ zu Beginn eines Kommentars, z.B. {$L-,C-} . Die Kontrollfunktionen A,L,O,C,S sind standardmäßig auf + eingestellt.

Kontrollfunktion A: Im Zustand A+ werden alle Feldindizes auf Einhaltung der Feldgrenzen überprüft, im Zustand A- entfällt diese Kontrolle.

Kontrollfunktion L: Im Zustand L+ wird beim Übersetzen der Quelltext mit den Objektcode-Adressen gelistet. Bei L- werden Zeilen nur dann gelistet, wenn in ihnen Fehler auftreten.

Kontrollfunktion O: Alle Gleitkommaoperationen sowie die Multiplikation und Division bei ganzen Zahlen unterliegen einer ständigen Überlaufkontrolle. Bei O+ unterliegen auch die Addition und Subtraktion von ganzen Zahlen dieser Kontrolle.

Kontrollfunktion C: Im Zustand C+ wird während des Programmlaufs in allen Schleifen, Funktionen und Prozeduren die Tastatur auf Drücken der Tasten PAUSE und STOP bzw. BRK überprüft. Auf diese Weise ist ein zeitweiliges Anhalten des Programms (Taste PAUSE) oder Abbruch möglich. Bei C- wird auf die Tastaturkontrolle verzichtet.

Kontrollfunktion S: Sie kontrolliert die Ausdehnung des Kellerspeichers (Stack). Bei S- unterbleibt diese Kontrolle.

5. Fehlermeldungen des Laufzeitsystems

Halt	Es liegt kein Fehler vor, denn es wurde HALT programmiert oder STOP bzw. BRK gedrückt
Overflow	Arithmetischer Über- oder Unterlauf
Out of RAM	Für die erzeugten Daten reicht der Top of RAM nicht aus
/ by Zero	Division durch Null, hervorgerufen durch die Operationen /, DIV oder MOD
Index too Low	Feldindex zu klein
Index too High	Feldindex zu groß
Maths Call Error	Fehler beim Aufruf einer mathematischen Funktion
Number too large	Zahl ist zu groß
Number expected	Bei READ wurde Zahl erwartet
Exponent expected	Ein Exponent wurde erwartet

6. Fehlermeldungen des Compilers

Verwendete Abkürzungen: w.e.: wird erwartet, n.e.: nicht erlaubt

1 Zahl zu groß
2 ; oder END fehlt
3 nicht deklarierter Name
4 ein Name w.e.
5 := in Konstantendefinition
6 = w.e.
7 dieser Name hier n.e.
8 := w.e.
9) w.e.
10 falscher Typ
11 . w.e.
12 ein Operand w.e.
13 Konstante w.e.
14 Name ist keine Konstante
15 THEN w.e.
16 DO w.e.
17 TO oder DOWNTO w.e.
18 (w.e.
19 dieser Typ nicht schreibbar
20 OF w.e.
21 , w.e.
22 : w.e.
23 PROGRAM w.e.
24 Variable als Parameter w.e.
25 BEGIN w.e.
26 Variable als Parameter von READ w.e.
27 Ausdrücke dieses Typs nicht vergleichbar
28 INTEGER oder REAL w.e.
29 dieser Variablentyp nicht lesbar
30 Name ist kein Typbezeichner
31 Exponent in REAL-Zahl w.e.
32 Ausdruck v. Skalartyp w.e.
33 Leerstring ('') n.e.
34 [oder (/ w.e.
35] oder /) w.e.
36 Feldindex muß skalar sein
37 .. w.e.
38] oder /) oder , in Felddeklaration w.e.
39 untere Grenze größer als obere
40 SET ist zu groß (>256 El.)
41 Typname für Funktionswert w.e.
42 , oder] oder /) in SET w.e.
43 .. oder , oder] in SET w.e.
44 Typname des Parameters w.e.
45 leere Menge als erster Operand n.e.
46 Skalar oder REAL w.e.
47 Skalar w.e.
48 Mengen nicht verträglich
49 < u. > f. Mengenvergl. n.e.
50 FORWARD, LABEL, CONST, VAR, TYPE, BEGIN w.e.
51 Hexazahl w.e.
52 cannot poke sets
53 Feld zu groß
54 END oder ; in Satzdeklaration w.e.
55 Feldname w.e.
56 Variable nach WITH w.e.
57 Variable nach WITH vom Satztyp w.e.
58 zu Satzkomponente fehlt zugehörige WITH-Anweisung
59 Nach LABEL vorzeichenlose ganze Zahl w.e.
60 Nach GOTO vorzeichenlose ganze Zahl w.e.
61 Marke in falschem Niveau
62 Nicht deklarierte Marke
63 Parameter von SIZE muß Variable sein
64 Pointer können nur auf Gleichheit getestet werden
67 H nach zweitem : w.e.
66 Strings dürfen kein CR enthalten
69 Parameter v. NEW, RELEASE, MARK muß v. Pointertyp sein
70 Parameter von ADDR muß Variable sein

Anschrift der Autoren:

Dr. Klaus Vetters
Gartenheimallee 2
Dresden
8021

Ergaenzung fuer Z-1013

Die vorliegende Dokumentation gilt auch fuer den Z-1013 Compiler, wenn folgende Ergaenzungen beachtet werden:

Zu 1) Der Compiler belegt den Speicherbereich von 512-12207 (210-2FAF h) und ist in der 16k Grundvariante nutzbar. Das Laden erfolgt mit HEADERSAVE oder MAINTAPE (Typ C). Der Name lautet:

HC-PASCAL/V2.1

Aus dem Monitor erfolgt das Laden mit L 200 2FAF ab 2. Vorton, Start mit J 200. Der Beginn des Bildspeichers (#EC00) stellt die Obergrenze fuer die RAM-Tops dar.

Zu 2.2.h) Inlines koennen dezimal, hexadez. bzw. gemischt eingefuegt werden (z.B. ld hl,0c309h : #21,9,#c3).

Zu 3.1) Die Kommandos PASNEW bzw. PASOLD werden ersetzt durch:

J 200 (enter) fuer PASNEW
J 203 (enter) fuer PASOLD

Hinweis fuer Insider:
In 200h - 20Eh befindet sich ein Rahmenprogramm
In 210h - 26Fh sind die Systemzellen
In 270h - 0FCFh liegt das Laufzeitsystem (Bibliothek)
In 0F00h - 2FAFh liegt der eigentliche Compiler
Zusaetzlich wird 100h - 183h als Tastaturpuffer genutzt.

Zur Magnetbandarbeit) Die Kommandos G,,f Pn,m,f Tn,,f sowie die Prozeduren TIN und TOUT gelten auch fuer den Z-1013.
Im Compiler ist ein HEADERSAVE-kompatibler Treiber integriert. Dabei werden folgende Typenkennzeichen vorgegeben:

P - PASCAL-Quelltext
D - mit TOUT ausgelagerte Daten(felder)
C - mit dem T-Kommando erzeugte Maschinenprogramme

ACHTUNG: Der Quelltext wird in tokenisierter Form ausgelagert und ist mit ASCII-Leseprogrammen nicht lesbar.

Hinweis zum T-Kommando: Der Compiler wird dabei nicht zerstoert. Nach dem Compilieren wird die Endadresse des Maschinenprogramms angezeigt und das Abspeichern abgefragt (Ok?). Das ausgelagerte Maschinenprogramm kann jederzeit mittels HEADERSAVE gelesen und gestartet werden.
Muss mit dem Monitor geladen werden, ist L 270 endadr zu geben. Das Programm ist mit J FC3 (auch wiederholt) zu starten.

Hinweis zur integrierten Magnetbandschreib/leseroutine:
Bei der Ausgabe wird nach Absetzen des Vorblocks fuer jeden weiteren Block ein Punkt auf dem Bildschirm gesetzt. Nach dem Schreiben meldet sich der Compiler wieder mit seinem Promptzeichen > .
Beim Lesen wird ein Kopf auf dem Band gesucht, das in Typ (und Namen) mit dem im Kommando spezifizierten uebereinstimmt. Jeder andere Kopfblock wird mit 'prg.not found' quittiert. Durch Druecken der STOP-Taste (ctrl-C,S4-K) beim Kopflesen kann die Eingabe verlassen und in den Compiler zurueckgekehrt werden.
Beim Lesen des gesuchten Programms erfolgt kein Bildschirmecho. Lediglich im Fehlerfall erscheinen 'bad rec.' bzw. 'rec.not found'. Diese sind durch (enter) nach Rueckspulen des Bandes zu bestaetigen.