```
;********** XBASIC-Basisroutinen **********
init    org   &a000
        ld    a,&c3         ;JP-Befehl
        ld    (&ac16),a     ;Patch für Umwandlung
        ld    hl,incode     ;in den Interpretercode
        ld    (&ac17),hl
        ld    (&ac19),a     ;Patch für
        ld    hl,list       ;Zeile listen
        ld    (&ac1a),hl
        ld    (&ac07),a     ;Patch für
        ld    hl,komman     ;Kommando ausfuehren
        ld    (&ac08),hl
        ld    (&ac0a),a     ;Patch für
        ld    hl,fnber      ;Funktionsberechnung
        ld    (&ac0b),hl
        ret
;**************** Umwandlung in Interpretercode
incode  ld    de,beftab
        call  &e327         ;Wort in Tabelle suchen
        ret   nc            ;nicht gefunden
        ld    a,(hl)        ;nächstes Zeichen
        call  &ff7b         ;Buchstabe oder Zahl?
        ret   c             ;ja, kein neues Befehlswort
        pop   af
        pop   af
        ld    a,(de)        ;Token aus Tabelle
        pop   de
        pop   bc
        push  af
        rlca                ;Funktionstoken?
        ld    a,&e0
        jr    c,settok      ;nein, &E0
        ld    a,&ff         ;ja, &FF
settok  call  &df25         ;in den Puffer schreiben
        pop   af            ;Token aus Tabelle
        call  &df25         ;in den Puffer schreiben
        jp    &df86         ;zurück zum Interpreter
;***************** Token als Befehlswort listen
 list   ld    a,c           ;Token aus Programm:
        rlca                ;Funktionstoken?
        jr    nc,findto     ;ja
        ld    a,&e0         ;Extended-Kommando?
        cp    c
        ret   nz            ;nein, Error
        pop   de            ;Zeiger in Programmzeile
        pop   bc            ;aus dem Stapel holen
        pop   ix
```

```
        pop  hl
        ld   a,(hl)    ;nächstes Token holen
        inc  hl
        push hl        ;Stapel restaurieren
        push ix
        push bc
        push de
        ld   c,a
findto  ld   hl,beftab
        call &e313     ;Token in Tabelle suchen
        ret  nc        ;nicht gefunden, Error
        ld   a,(hl)    ;1. Buchstabe
        inc  hl        ;Zeiger auf 2. Buchstaben
        pop  bc
        pop  bc        ;Error-Rücksprung vom Stack
        ret            ;Token listen
;************************ Kommando ausführen
komman  cp   &c0       ;&E0*2=&C0
        ret  nz         ;nicht &E0, Error
        inc  hl         ;Programmzeiger erhöhen
        ld   a,(hl)    ;Token holen
        add  a,a       ;*2
        ret  nc        ;Error, falls < &80
        pop  bc        ;Error-Rücksprung vom Stack
        ld   c,a
        ld   b,0
        ex   de,hl
        ld   hl,komadr;Zeiger in die Tabelle
        add  hl,bc     ;der Kommando-Adressen
        jp   &ddbb     ;Adr. holen, Sprung auf Befehl
;************************ Funktion berechnen
fnber   sub  &1e       ;Akku-&1E=Funktionstoken*2
        cp   &80       ; >= &80 ?
        ret  nc        ;ja, Error
        sub  &40       ; < &40 ?
        ret  c         ;ja, Error
        pop  bc        ;Error-Rücksprung vom Stack
        ld   c,a       ;Funktionstoken*2
        ld   b,0       ;nach bc
        push hl
        ld   hl,fnadr ;Zeiger in die Tabelle
        jp   &d0b4     ;Adr. holen, Sprung auf Funktion
;************************** Tabellen
beftab  ds   1         ;Tabelle der Befehlswörter
komadr  ds   1         ;Sprungadressen Kommandos
fnadr   ds   1         ;Sprungadressen Funktionen
;****************************************
```