Erweiterung

MPBASIC kann um neue Prozeduren/Funktionen erweitert werden. s. MPBASIC, Prozedurtabelle. 2021 habe ich mich mit dieser Thematik beschäftigt (Tiny, BASIC-Erweiterung).

Erstaunlicherweise sind in der Literatur jedoch keine Beispiele zu finden. Auch Informationen, welche Register nutzbar sind, fehlen. Teilweise stehen auch falsche oder fehlerhafte Angaben in der Literatur.

Hat die Funktion nur einen Eingabeparameter, wird dieser in RR4 übergeben.
Hat die Funktion nur einen Ausgabeparameter, wird dieser in RR2 zurückgegeben.

Prozeduren mit mehr Parametern s. Beispiel2.

Beispiel1:

simple Prozedur CLS, simple Funktion LET (nicht mit dem Kommando LET verwechseln!)

```
      17/    8000 : 03 43 4C 53         tab_prc:    db    3,"CLS"
      18/    8004 : 80 0D                           dw    p_cls
      19/    8006 : 03 4C 45 54                     db    3,"LET"
      20/    800A : 80 13                           dw    f_let
      21/    800C : FF                              db    0FFh        ;
Listenende
      28/    800D :
      29/    800D : 5C 0C               p_cls:      ld    r5, #12    ; cls
      30/    800F : D6 08 18                        call    putch
      31/    8012 : AF                              ret
      32/    8013 :
      40/    8013 : 28 E4               f_let:      ld    R2, R4        ; Y
:= X
      41/    8015 : 38 E5                           ld    R3, R5
      42/    8017 : AF                              ret
```

Der Hex-Code ist im Speicher an %8000 abzulegen.

Nutzung in BASIC

```
1 PROC SETRR[8,%8000];REM Erweiterung einbinden
10 PROC CLS
20 LET A=41,B=42
30 PRINT A,B
40 PROC [A]=LET[B];PRINT LET[A]
50 PRINT A,B

ES4.0:
1OSETRR[8,%8000]
10OCLS
20LA=41,B=42
30PA,B
40O[A]=LET[B]
```

```
50PLET[A]
60PA,B
9999E
```

in Zeile 1 wird die Erweiterung aktiviert (Reg8+9 mit Adresse der externen Prozedurtabelle laden). Zeile 10 ruft die Prozedur „CLS“ auf. Zeile 40 demonstriert den Aufruf der Funktion „LET“ via PROC. Es passiert dasselbe wie bei LET A=B. Zeile 50 kopiert den Wert aus A in eine temporäre Variable, und diese wird ausgegeben. Es passiert also dasselbe wie bei PRINT A.

Beispiel2: mehr Parameter

Drei Ein- und drei Ausgabeparameter. Die Eingabeparameter werden den Ausgabeparametern zugewiesen (3faches LET).

```
     109/    801F :                      ;-----------------------------------
-----------------------------------------
     110/    801F :                      ; Beispiel 3 Ein- und
Ausgabeparameter PROC [Y1,Y2,Y3] = para [X1,X2,X3]
     111/    801F :                      ; in  X1, X2 auf Stack, RR4 = X3
     112/    801F :                      ; out Y1, Y2 auf Stack, RR2 = Y3
     113/    801F :                      ;   SP            returnadr.
     114/    801F :                      ;   SP+2             Parameter X2
     115/    801F :                      ;   SP+4             Parameter X1
     116/    801F :                      ;   SP+6            (intern f.
interpreter)
     117/    801F :                      ;   SP+8            Platz f.
Ergebnisvariable Y1
     118/    801F :                      ;   SP+10            Platz f.
Ergebnisvariable Y2
     119/    801F :                      ; Beim Ende muss der SP auf (intern
f. interpreter) zeigen
     121/    801F :                      ;-----------------------------------
-----------------------------------------
     122/    801F :
     123/    801F : 50 E2                f_let3:     pop    R2        ; RR2
Rückkehradresse zum Interpreter
     124/    8021 : 50 E3                           pop    R3
     125/    8023 : 50 E6                           pop    R6        ; RR6
Adresse von X2
     126/    8025 : 50 E7                           pop    R7
     127/    8027 : 50 E8                           pop    R8        ; RR8
Adresse von X1
     128/    8029 : 50 E9                           pop    R9
     129/    802B :                                 ;...
     130/    802B : A8 FE                           ld    R10,gpr        ; stack
     131/    802D : B8 FF                           ld    R11,gpr+1
     132/    802F : 70 E3                           push    R3        ; Ret-Adr.
wieder auf Stack
     133/    8031 : 70 E2                           push    R2
     134/    8033 : A0 EA                           incw    RR10        ;
(intern) übergehen
```

```
     135/    8035 : A0 EA                        incw    RR10
     136/    8037 : 92 8A                        lde    @RR10,R8    ; Y1 :=
X1
     137/    8039 : A0 EA                        incw    RR10
     138/    803B : 92 9A                        lde    @RR10,R9
     139/    803D : A0 EA                        incw    RR10
     140/    803F : 92 6A                        lde    @RR10,R6    ; Y2 :=
X2
     141/    8041 : A0 EA                        incw    RR10
     142/    8043 : 92 7A                        lde    @RR10,R7
     143/    8045 : 28 E4                        ld    R2,R4        ; Y3 :=
X3
     144/    8047 : 38 E5                        ld    R3,R5
     145/    8049 : AF                           ret
     1
```

Nutzung in BASIC

```
1 PROC SETRR[8,%8000];REM Erweiterung einbinden
90 PROC[A,B,C]=LET3[%111,%222,%333]
100 PRINTHEX A,B,C

ES4.0:
1OSETRR[8,%8000]
90O[A,B,C]=LET3[%111,%222,%333]
100HA,B,C
9999E
```

Parameterübergabe s. MPBASIC, Übergabe der Parameter.