# PRINT USING mit der USR-Funktion

**Haben Sie sich auch schon über die unformatierte Zahlenausgabe Ihres C 64 geärgert? Dann nehmen Sie in Zukunft die folgende USR-Funktion, um für Ihren C 64 ein PRINT USING zu erhalten.**

Wie allgemein bekannt und vielfach bemängelt, bietet das Commodore-Basic keinen PRINT USING-Befehl. Doch gerade bei kommerziellen Problemen kann auf eine Druckaufbereitung von Zahlen nicht verzichtet werden. Eine Rechnung, in der nicht einmal die Dezimalpunkte untereinander stehen, ist eben unübersichtlich und keine Reklame. Für die Druckaufbereitung gibt es verschiedene Lösungen.

Die beste Lösung ist ein Maschinenprogramm. Interessierte Maschinenprogrammierer finden den Quellcode zu dieser Routine in Listing 1. Doch zunächst stellt sich die Frage, wie ein Maschinenprogramm für die Druckaufbereitung aufgerufen werden soll. Offensichtlich ist die USR-Funktion am geeignetsten, da sie sowohl in einer PRINT- als auch in einer PRINT #- oder einer LET-Anweisung verwendet werden kann. Der Aufruf der Funktion erfolgt durch USR(X),L,NK. Dabei ist X die Zahl, die aufbereitet werden soll, L die Gesamtfeldlänge der aufbereiteten Zahl einschließlich Vorzeichen und Dezimalkomma und NK die Anzahl der darzustellenden Nachkommastellen. Die USR-Funktion wandelt zunächst die Zahl X in einen ASCII-String um und berechnet die Stringlänge und die Anzahl der Nachkommastellen. Wenn bei der Wandlung von X der Interpreter die Exponentialform wählt, dann wird die Exponentialdarstellung zunächst in die Fließkommadarstellung umgewandelt. Danach werden die Nachkommastellen aufbereitet. Fehlende Nachkommastellen werden durch angehängte Nullen ergänzt. Müssen Nachkommastellen abgeschnitten werden, dann wird die Zahl gerundet, wenn die erste abgeschnittene Dezimalstelle größer oder gleich 5 ist. Wenn die Anzahl der gewünschten Nachkommastellen null ist, dann wird die Zahl als ganze Zahl (Integer) ohne Dezimalkomma aufbereitet. Wegen der kaufmännischen Anwendung wird der Dezimalpunkt durch ein Dezimalkomma ersetzt. Nach der Aufbereitung der Nachkommastellen wird durch Voranstellen von Leerzeichen der String auf die erforderliche Länge gebracht. Ist der String nach der Aufbereitung der Nachkommastellen schon länger als gewünscht, dann wird er nicht mehr verändert, sondern in voller Länge ausgegeben, um einen Datenverlust zu verhindern.

Das hier vorgestellte Maschinenprogramm verwendet nur relative Sprünge — außer bei den Aufrufen der Betriebssystemroutinen. Daher kann sich jeder Anwender das Programm ohne Änderungen in den Speicherbereich laden, der ihm am geeignetsten erscheint. Als Stringpuffer wird der Bereich ab $100 benutzt. Das hat zur Folge, daß die GETSTR-Routine diesen String nicht in den Stringbereich kopiert. Der Stringbereich wird also nicht unnötig belastet. Eine Wertzuweisung A$=USR(X),L,NK ist dadurch aber auch nicht möglich, da die nächste Stringfunktion den Bereich ab $100 wieder überschreibt. Wenn druckaufbereitete Werte einer Variablen zugewiesen werden sollen, dann muß die Anweisung A$=" "+USR(X),L,NK oder A$=(USR(X),L,NK)+" " lauten, da dann das Ergebnis der Stringverknüpfung in den Stringbereich kopiert wird und der Variablen A$ dauerhaft zugewiesen ist.

Vor dem ersten Aufruf der USR-Funktion muß jetzt noch in Adresse 785 (Low-Byte) und 786 (High-Byte) die Startadresse der USR-Funktion hinterlegt werden. Listing 2 zeigt das Basic-Ladeprogramm für die USR-Funktion. Die Ladeadresse können Sie selbst bestimmen. Das Ladeprogramm setzt die Startadresse der USR-Funktion in den Speicherstellen 785 und 786 entsprechend. Zur Verdeutlichung der Anwendung der USR-Funktion enthält das Ladeprogramm verschiedene Druckaufbereitungen der Zahl $\pi$. Das Ergebnis des Beispiels ist in Bild 1 wiedergegeben.

(Dr. Michael Irskens/ah)

```
           3          3,1          3,14
          31         31,4         31,42
         314        314,2        314,16
        3142       3141,6       3141,59
       31416      31415,9      31415,93
      314159     314159,3     314159,27
     3141593    3141592,7    3141592,65
    31415927   31415926,5   31415926,50
   314159265  314159265,0  314159265,00
  3141592650 3141592650,0 3141592650,00
```

**Bild 1. Beispiele für verschiedene Druckaufbereitungen**

```
1000                ; f o r m a t r o u t i n e
1010                ; -----------------------
1020                ;
1030                ;            dr.m.irskens
1040                ;            leveser allee 13
1050                ;            3061 hespe
1060                ;
1070                ;
1080 space          equ 32    ; ' '
1090 komma          equ 44    ; ','
1100 e              equ 69    ; 'e'
1110 punkt          equ 46    ; '.'
1120 null           equ 48    ; '0'
1130 minus          equ 45    ; '-'
1140 ckcom          equ $aefd ; prueft auf komma
1150 getstr         equ $b487 ; string aus stringpuffer in stringbereich
1160 numtest        equ $ad8d ; ergebnis auf numerisch pruefen
1170 facstr         equ $bddd ; fac in string wandeln
1180 getbyt         equ $b79e ; byte holen
1190 string         equ $100  ; string-puffer
1200 temp           equ $02   ; temporaerer speicher
1210 nk             equ $57   ; fliesskomma-akku#3
1220 flen           equ $58
1230                *equ $c000 ;startadresse
1240 fo10           jsr numtest ;auf numerisch pruefen
1250                jsr facstr ;umwandeln numerisch->alpha
1260                jsr ckcom  ;auf komma pruefen
1270                jsr getbyt ;feldlaenge holen
1280                stx flen
1290                jsr ckcom  ;auf komma pruefen
1300                jsr getbyt ;nachkommastellenzahl holen
1310                stx nk
1320                pla        ; rueecksprungadresse vom stack entfernen,
1330                pla        ; damit kein numtest durchgefuehrt wird
1340                ;
1350                ; pruefen auf darstellung im e-format
1360                ; x-register equ stringlaenge
1c70                ; y-register equ anzahl vorkommastellen
1380                ; y-register equ 0, wenn keine nachkommastellen gefunden
1390 fo19           ldx #$ff
1400                ldy #0
1410 fo20           inx
1420                lda string,x ;zeichen aus puffer laden
1430                beq fo30   ;0->stringende
1440                cmp #e     ;vergleich auf 'e'
1450                beq fo201  ;e-format umwandeln
1460                cmp #punkt  ;vergleich auf '.'
1470                bne fo20
1480                txa
1490                tay        ;anzahl vorkommastellen in y-register
1500                bne fo20
1510                ;
1520                ; umwandeln e-format
1530                ;
1540 fo201          lda string+2
1550                cmp #punkt
1560                bne fo22
1570                ;
1580                ; dezimalpunkt entfernen
1590                ;
1600                dex
1610                ldy #1
1620 fo21           iny
1630                lda string+1,y
1640                sta string,y
1650                bne fo21
```

**Listing 1. PRINT USING-Quellprogramm**

```
1660              ;
1670              ; stellenverschiebung errechnen
1680              ;
1690 fo22         lda string+2,x ; zehnerstelle exponent
1700              and #$0f    ; ziffer errechnen
1710              asl a       ; *2
1720              sta temp
1730              asl a       ; *4
1740              asl a       ; *8
1750              adc temp    ; 8* + 2* equ 10*
1760              adc string+3,x; +einerstelle ascii
1770              sbc #47     ; ascii 0 abziehen
1780              ldy string+1,x ; vorzeichen exponent
1790              cpy #minus
1800              beq fo24
1810              adc #3
1820              stx temp
1830              sbc temp
1840              tay
1850              lda #null
1860 fo23         sta string,x ; string mit nullen erweitern fuer + exponent
1870              inx
1880              dey
1890              bne fo23
1900              lda #0
1910              sta string,x ; neues stringende
1920              beq fo19
1930              ;
1940              ; exponent<0: vornullen ergaenzen
1950              ; string um a-reg.+1 stellen verschieben
1960 fo24         sta temp
1970              lda #0
1980              sta string,x ; "e" durch 0 ersetzen
1990              txa
2000              clc
2010              adc temp
2020              tay
2030 fo25         lda string,x
2040              beq fo26    ; stringende uebertragen
2050              cmp #$30    ; vergleich auf ziffer
2060              bcs fo26    ; ziffer
2070              lda #null
2080              bne fo27
2090 fo26         dex
2100 fo27         sta string,y
2110              dey
2120              bne fo25
2130              lda #punkt
2140              sta string+1
2150              bne fo19
2160              ;
2170              ; dezimalpunkt durch komma ersetzen
2180              ; y-register: anzahl vorkommazeichen
2190              ;
2200 fo30         tya
2210              beq fo50    ;nachkommastellen ergaenzen
2220              lda nk
2230              bne fo45  ; nachkommastellen
2240              tya
2250              tax
2260              lda string+1,x
2270              bne fo901 ; keine nachkommastellen, aber runden
2280 fo45         lda #komma ;'.' gefunden
2290              sta string,y ;durch ',' ersetzen
2300              bne fo71
2310 fo50         cpy nk      ;nachkommastellen ergaenzen
2320              beq fo100
2330              lda #komma
2340              sta string,x
2350              inx
2360              bne fo79
2370 fo71         sty temp    ;anzahl nachkommastellen errechnen
2380              sec
2390              txa
2400              sbc temp
2410              sec
2420              sbc #1      ;1 abziehen fuer dezimalpunkt
2430              cmp nk      ;vergleichen mit sollanzahl
2440              beq fo100 ;anzahl gleich
2450              bcs fo90  ;anzahl > soll
2460              tay         ;anzahl < soll
2470 fo79         lda #null
2480 fo80         sta string,x ;string mit nullen ergaenzen
2490              inx
2500              iny
2510              cpy nk
2520              bne fo80
2530 fo85         lda #0
2540              sta string,x
2550              ;
2560              ; nachkommastellen aufbereitet
2570              ; gesamtfeldlaenge korrigieren
2580              ;
2590              ; x-reg feldlaenge aktuell
2600              ; flen  gesamtlaenge soll
2610 fo100        lda string+1
2620              cmp #$30
2630              bcs fo109     ; 1.zeichen ist eine ziffer
2640              inx
2650              txa
2660              tay
2670 fo105        lda string-1,y
2680              sta string,y
2690              dey
2700              bne fo105
2710              lda #null  ; null vor komma ergaenzen
2720              sta string+1
2730 fo109        cpx flen
2740              bcs fo130 ;feldueberlauf oder feldlaenge gleich
2750              ldy flen
2760 fo110        lda string,x;string verschieben
2770              sta string,y
2780              dey
2790              dex
2800              bpl fo110
2810              lda #space ;leerzeichen
2820 fo120        sta string,y ;rest mit leerzeichen fuellen
2830              dey
2840              bpl fo120
2850 fo130        lda #<string ;startadresse string laden in a/y
2860              ldy #>string
2870              jmp getstr
2880 fo90         sec
2890              sbc nk
2900              sta temp  ;anzahl ueberfluessiger nachkommastellen
2910              txa
2920              sec
2930              sbc temp  ;neue stringlaenge
2940              tax
2950              ;
2960              ; beim abschneiden von nachkommastellen runden
2970              ;
2980              lda string,x
2990 fo901        cmp #$35
3000              bcc fo85
3010              txa
3020              tay         ; index letztes zeichen
3030 fo91         dey
3040              beq fo92  ; string eine 1 voranstellen
3050              lda string,y
3060              cmp #$30    ; vergl < "0"
3070              bcc fo91
3080              clc
3090              adc #1      ; ziffer um eins erhoehen
3100              cmp #$3a    ; ueberlauf abfragen
3110              sta string,y ; veraendertes zeichen speichern
3120              bne fo85    ; kein uebertrag
3130              lda #null  ; null laden
3140              sta string,y
3150              bne fo91    ; naechste ziffer erhoehen
3160 fo92         txa         ; string um eine stelle verschieben fuer "1"
3170              tay
3180 fo94         lda string,y
3190              sta string+1,y
3200              dey
3210              bne fo94
3220              lda #$31    ; "1" laden
3230              sta string+1
3240              inx       ; feldlaenge erhoehen
3250              bne fo85
```

**Listing 1. PRINT USING-Quellprogramm (Schluß)**

```
100 S=0:INPUT "STARTADRESSE";B                                    <182>
110 FOR I=B TO B+335                                              <000>
120 READ A                                                        <160>
130 S=S+A:POKE I,A                                                <062>
140 NEXT I                                                        <224>
150 IF S<>36986 THEN PRINT "{RVSON,SPACE}F
    EHLER IN DEN DATAZEILEN{SPACE,RVOFF}":
    STOP                                                          <119>
160 POKE 785,B-256*INT(B/256):POKE 786,B/2
    56                                                            <040>
170 PRINT " ALLES OK. "                                           <036>
199 REM TESTBEISPIELE                                             <050>
200 FOR E=0 TO 9                                                  <238>
210 F=↑*10↑E                                                      <176>
220 A$=(USR(F),11,0)+""                                           <178>
230 PRINT A$;USR(F),13,1;USR(F),14,2                              <024>
240 NEXT E                                                        <036>
250 END                                                           <252>
299 REM DATAZEILEN                                                <190>
300 DATA 32,141,173,32,221,189,32,253,174,
    32,158,183,134,88,32,253,174,32                               <212>
301 DATA 158,183,134,87,104,104,162,255,16
    0,0,232,189,0,1,240,117,201,69,240                            <057>
302 DATA 8,201,46,208,242,138,168,208,238,
    173,2,1,201,46,208,12,202,160,1                               <250>
303 DATA 200,185,1,1,153,0,1,208,247,189,2
    ,1,41,15,10,133,2,10,10,101,2,125                             <139>
304 DATA 3,1,233,47,188,1,1,192,45,240,23,
    105,3,134,2,229,2,168,169,48,157                              <106>
305 DATA 0,1,232,136,208,249,169,0,157,0,1
    ,240,168,133,2,169,0,157,0,1,138                              <207>
306 DATA 24,101,2,168,189,0,1,240,8,201,48
    ,176,4,169,48,208,1,202,153,0,1                               <048>
307 DATA 136,208,236,169,46,141,1,1,208,12
    9,152,240,18,165,87,208,7,152,170                             <211>
308 DATA 189,1,1,208,119,169,44,153,0,1,20
    8,12,196,87,240,40,169,44,157,0                               <159>
309 DATA 1,232,208,16,132,2,56,138,229,2,5
    6,233,1,197,87,240,19,176,72,168                              <011>
310 DATA 169,48,157,0,1,232,200,196,87,208
    ,247,169,0,157,0,1,173,1,1,201,48                             <115>
311 DATA 176,17,232,138,168,185,255,0,153,
    0,1,136,208,247,169,48,141,1,1,228                            <091>
312 DATA 88,176,20,164,88,189,0,1,153,0,1,
    136,202,16,246,169,32,153,0,1,136                             <163>
313 DATA 16,250,169,0,160,1,76,135,180,56,
    229,87,133,2,138,56,229,2,170,189                             <206>
314 DATA 0,1,201,53,144,179,138,168,136,24
    0,24,185,0,1,201,48,144,246,24,105                            <086>
315 DATA 1,201,58,153,0,1,208,157,169,48,1
    53,0,1,208,229,138,168,185,0,1,153                            <028>
316 DATA 1,1,136,208,247,169,49,141,1,1,23
    2,208,131                                                     <030>
```



**Listing 2. PRINT USING-Basic-Lader. Bitte beachten Sie die Eingabehinweise auf Seite 6**