C64 BASIC V2.0: Befehle, Schlüsselwörter und Token

geschrieben von Andreas Potthoff | 23. April 2023

Beim Commodore 64 (auch VC-20) werden bei der internen Verarbeitung der **BASIC-Befehle (Schlüsselwörter)** eines BASIC-Programms *für eine schnellere Verarbeitung und Speicherplatzersparnis* sogenannte **Token (Schlüsselzeichen)** statt der Schlüsselwörter verwendet.

Die folgende Tabelle zeigt die BASIC-Schlüsselwörter, den dazugehörigen Token (dezimal / hexadezimal), die ROM-Einsprungadresse (hexadezimal), die Abkürzung und den Typ des Befehls.

Nach der Tabelle erhalten Sie weiterführende Informationen zur Tokenisation der Befehle.

Tabelle C64 Token

BASIC Schlüsselwort	Token dez	Token hex	ROM Exec dez	ROM Exec hex	Abk.	Typ
END	128	$80	43057	$A831	eN	Anweisung/Befehl
FOR	129	$81	42818	$A742	fO	Anweisung/Befehl
NEXT	130	$82	44318	$AD1E	nE	Anweisung/Befehl
DATA	131	$83	43256	$A8F8	dA	Anweisung/Befehl
INPUT#	132	$84	43941	$ABA5	iN	Anweisung/Befehl
INPUT	133	$85	43967	$ABBF	-	Anweisung/Befehl
DIM	134	$86	45185	$B081	dI	Anweisung/Befehl
READ	135	$87	44038	$AC06	rE	Anweisung/Befehl
LET	136	$88	43429	$A9A5	lE	Anweisung/Befehl

BASIC Schlüsselwort	Token dez	Token hex	ROM Exec dez	ROM Exec hex	Abk.	Typ
GOTO	137	$89	43168	$A8A0	gO	Anweisung/Befehl
RUN	138	$8A	43121	$A871	rU	Anweisung/Befehl
IF	139	$8B	43304	$A928	-	Anweisung/Befehl
RESTORE	140	$8C	43037	$A81D	reS	Anweisung/Befehl
GOSUB	141	$8D	43139	$A883	goS	Anweisung/Befehl
RETURN	142	$8E	43218	$A8D2	reT	Anweisung/Befehl
REM	143	$8F	43323	$A93B	-	Anweisung/Befehl
STOP	144	$90	43055	$A82F	sT	Anweisung/Befehl
ON	145	$91	43339	$A94B	-	Anweisung/Befehl
WAIT	146	$92	47149	$B82D	wA	Anweisung/Befehl
LOAD	147	$93	57704	$E168	lO	Anweisung/Befehl
SAVE	148	$94	57686	$E156	sA	Anweisung/Befehl
VERIFY	149	$95	57701	$E165	vE	Anweisung/Befehl
DEF	150	$96	46003	$B3B3	dE	Anweisung/Befehl
POKE	151	$97	47140	$B824	pO	Anweisung/Befehl
PRINT#	152	$98	43648	$AA80	pR	Anweisung/Befehl
PRINT	153	$99	43680	$AAA0	?	Anweisung/Befehl
CONT	154	$9A	43095	$A857	cO	Anweisung/Befehl
LIST	155	$9B	42652	$A69C	lI	Anweisung/Befehl
CLR	156	$9C	42590	$A65E	cL	Anweisung/Befehl
CMD	157	$9D	43654	$AA86	cM	Anweisung/Befehl
SYS	158	$9E	57642	$E12A	sY	Anweisung/Befehl
OPEN	159	$9F	57790	$E1BE	oP	Anweisung/Befehl
CLOSE	160	$A0	57799	$E1C7	clO	Anweisung/Befehl
GET	161	$A1	43899	$AB7B	gE	Anweisung/Befehl
NEW	162	$A2	42562	$A642	-	Anweisung/Befehl

BASIC Schlüsselwort	Token dez	Token hex	ROM Exec dez	ROM Exec hex	Abk.	Typ
TAB(	163	$A3	43752	$AAE8	tA	Anweisung/Befehl, Spezial
TO	164	$A4	42861	$A76D	-	Anweisung/Befehl, Spezial
FN	165	$A5	46068	$B3F4	-	Anweisung/Befehl, Spezial
SPC(	166	$A6	43769	$AAF9	sP	Anweisung/Befehl, Spezial
THEN	167	$A7	43314	$A932	tH	Anweisung/Befehl, Spezial
NOT	168	$A8	44756	$AED4	nO	Anweisung/Befehl, Spezial
STEP	169	$A9	42905	$A799	stE	Anweisung/Befehl, Spezial
+	170	$AA	47210	$B86A	-	Operator, numerisch/string
-	171	$AB	47187	$B853	-	Operator, numerisch
*	172	$AC	47659	$BA2B	-	Operator, numerisch
/	173	$AD	47890	$BB12	-	Operator, numerisch
^	174	$AE	49019	$BF7B	-	Operator, numerisch
AND	175	$AF	45033	$AFE9	aN	Operator, logisch
OR	176	$B0	45030	$AFE6	-	Operator, logisch
>	177	$B1	49076	$BFB4	-	Operator, logisch
=	178	$B2	44756	$AED4	-	Operator, logisch
<	179	$B3	45078	$B016	-	Operator, logisch

BASIC Schlüsselwort	Token dez	Token hex	ROM Exec dez	ROM Exec hex	Abk.	Typ
SGN	180	$B4	48185	$BC39	sG	Funktion, numerisch
INT	181	$B5	48332	$BCCC	-	Funktion, numerisch
ABS	182	$B6	48216	$BC58	abS	Funktion, numerisch
USR	183	$B7	784	$0310	uS	Funktion, numerisch/string
FRE	184	$B8	45949	$B37D	fR	Funktion, numerisch, Spezial
POS	185	$B9	45982	$B39E	-	Funktion, numerisch, Spezial
SQR	186	$BA	49009	$BF71	sQ	Funktion, numerisch
RND	187	$BB	57495	$E097	rN	Funktion, numerisch
LOG	188	$BC	45794	$B9EA	-	Funktion, numerisch
EXP	189	$BD	49133	$BFED	eX	Funktion, numerisch
COS	190	$BE	57956	$E264	-	Funktion, numerisch
SIN	191	$BF	57963	$E26B	sI	Funktion, numerisch
TAN	192	$C0	58036	$E2B4	-	Funktion, numerisch
ATN	193	$C1	58128	$E30E	aT	Funktion, numerisch

BASIC Schlüsselwort	Token dez	Token hex	ROM Exec dez	ROM Exec hex	Abk.	Typ
PEEK	194	$C2	47117	$B80E	pE	Funktion, numerisch
LEN	195	$C3	46972	$B77C	-	Funktion, numerisch
STR$	196	$C4	46181	$B465	stR	Funktion, string
VAL	197	$C5	47021	$B7AD	vA	Funktion, numerisch
ASC	198	$C6	46987	$B78B	aS	Funktion, numerisch
CHR$	199	$C7	46828	$B6EC	cH	Funktion, string
LEFT$	200	$C8	46848	$B700	leF	Funktion, string
RIGHT$	201	$C9	46892	$B72C	rI	Funktion, string
MID$	202	$CA	46903	$B737	mI	Funktion, string
GO	203	$CB	43026	$A812	-	Anweisung/Befehl, Spezial
π	255	$FF	44702	$AE9E	-	Funktion, numerisch, Konstante
ST (STATUS)	-	-	65463	$FFB7	-	Systemvariable
TI (TIME)	-	-	?	?	-	Systemvariable
TI$ (TIME$)	-	-	43488	$A9E0	-	Systemvariable

Tokenisation

Hier werden die Schlüsselwörter (Befehle) in ein Single-Byte-Wert (Token) umgewandelt, wenn sie in einem Programm gespeichert sind. Das geschieht entweder durch einen

Programmstart mit RUN oder im Direktmodus (Konsole) durch das Drücken der Taste RETURN, wenn dort Kommandos eingegeben worden sind. Der BASIC-Interpreter arbeitet die Token der Reihenfolge nach ab.

De-Tokenisation

Hier werden die Token (Bytes) in lesbare BASIC-Befehle (Schlüsselwörter) umgewandelt, was eigentlich *nur für das LIST-Kommando* als Ausgabe und für gute menschliche Lesbarkeit zutrifft. Ansonsten wird die De-Tokenisation nicht angewandt.

Single-Byte-Token

Ein Token kann beim Commodore BASIC V2 (ab 4.0) einen Single-Byte-Wert zwischen 128-255 ($80-$FF) haben und belegt nur 1 Byte Arbeitsspeicher. Token-Codes sind immer größer oder gleich 128 ($80); d.h. das höchstwertige Bit in einem Byte, das einen Token repräsentiert, ist immer gesetzt und somit werden keine PETSCII-Zeichen (<128/<$80) als Token-Code benutzt.

Two-Bytes-Token

Ab dem Commodore BASIC 7.0 (C 128) werden wegen des umfangreichen BASIC-Befehlssatzes zwei Bytes für ein Token benötigt.

Ausführen von Token

Jedes Token hat eine sogenannte Ausführungsadresse (EXEC) im ROM, wo dann der entsprechende Code für den jeweiligen Token bzw. BASIC-Befehl ausgeführt wird.

Schlüsselwörter

BASIC V2 enthält 76 Schlüsselwörter, 8 Operatoren, 1 Konstante und 3 Systemvariablen, die in verschiedenen Gruppen gegliedert

sind:

- 128-162 ($80-$A2): *Befehle*
- 163-169 ($A3-$A9): “*Bywords*“, die Teil der Syntax der vorherigen Befehle sind
- 170-179 ($AA-$B3): *Arithmetische und logische Operatoren*
- 180-202 ($B4-$CA): *Funktionen*
- 203 ($CB): *Befehl* – GO (der hier als Ausnahme hinter den Funktionen liegt)
- 204-254 ($CC-$FE): Ein freier Bereich für 51 zusätzliche Token, z.B. der für BASIC-Erweiterungen von Drittanbietern genutzt wird
- 255 ($FF): *Konstante* – Pi
- *Systemvariablen*: ST (STATUS), TI (TIME), TI$ (TIME$). Im BASIC V2 ROM werden die Systemvariablen als Ausnahmen in den Routinen zur Behandlung *normaler* Variablen behandelt.

Abkürzungen

Die meisten BASIC-Schlüsselwörter kann man bei der Eingabe abkürzen. Abgekürzte Schlüsselwörter werden i.d.R. gebildet, indem man die ersten (manchmal bis zu drei) Zeichen eintippt und das nächste Zeichen mit SHIFT eingibt.

Auch hier gibt es wieder einige Ausnahmen. Einige Schlüsselwörter (CLOSE, GOSUB, LEFT$, RESTORE, RETURN, STEP, STR$) benötigen gekürzt 3 statt 2 Zeichen. Der BASIC-Befehl PRINT wird nur mit einem Zeichen, dem ? abgekürzt. Es gibt auch einige BASIC-Befehle (INPUT, COS, FN, TO, IF, INT, LEN, LOG, NEW, ON, OR, POS, REM, TAN) die nicht abkürzbar sind.

Speichern

Es kann schon mal vorkommen, dass *eine Programmzeile* 80 Zeichen überschreitet, also dafür mehr als 2 Zeilen auf dem Bildschirm angezeigt werden. Dies geschieht dadurch, dass die

Ausgabe der Token beim LIST-Befehl eben ungekürzt passiert und somit Programmzeilen mit mehr als 80 Zeichen auftreten können.

Wenn sie eine solche Programmzeile ändern wollen, müssen sie die Abkürzungen erneut eingeben bevor sie das Programm speichern. Achten sie dann darauf, dass sie nicht mehr als 80 Zeichen für eine Programmzeile insgesamt verwenden. Alle zusätzlichen Zeichen danach werden nach dem Drücken von RETURN automatisch abgeschnitten. Beim Speichern eines Programms auf einen Datenträger werden die Token und nicht die Schlüsselwörter benutzt.

Befehle	CLOSE, CLR, CMD, CONT, DATA, DEF, DIM, END, FOR, GET, GET#, GOSUB, GOTO, IF, INPUT, INPUT#, LET, LIST, LOAD, NEW, NEXT, ON, OPEN, POKE, PRINT, PRINT#, READ, REM, RESTORE, RETURN, RUN, SAVE, STOP, SYS, VERIFY, WAIT
Befehle Spezial (Bywords)	FN, GO, NOT, SPC(, TAB(, THEN, TO, STEP
Arithmetische und logische Operatoren	+, –, *, /, ^, >, =, <, AND, OR
Funktionen	ABS, ASC, ATN, CHR$, COS, EXP, FRE, INT, LEFT$, LEN, LOG, MID$, PEEK, POS, RIGHT$, RND, SGN, SIN, SQR, STR$, TAN, USR, VAL
Konstanten und Systemvariablen	Pi, ST, TI, TI$