

# Vier Pseudo-VICs mit 32 Sprites

**Sie wollen mit 32 Sprites und vier Bildschirmbereichen gleichzeitig arbeiten? Nichts leichter als das. Mit Provic 64 können simultan Grafik, Text oder veränderte Zeichensätze dargestellt werden.**

Die Autoren (Jürgen, 21, Physikstudent, und Stefan, 18, Schüler) haben sich Mitte 1983 einen Commodore 64 angeschafft. Schon nach kurzer Zeit stellte sich der bei den C 64-Fans übliche Frust über die schlechte Dokumentation und die schwierige Informationsbeschaffung ein, besonders wenn es um die speziellen Grafikfähigkeiten dieses Computers geht. So sitzen wir oft stundenlang vor dem Bildschirm, der nur undefinierbare Zeichen zeigt, weil wir bei dem Versuch, die Geheimnisse des C 64 zu enträtseln, in irgendeinen unbekanntem Darstellungsmodus geraten sind.

Dabei entdeckten wir, daß der C 64 nicht nur acht, sondern auch 16, 24, 32 oder noch mehr Sprites gleichzeitig zeigen kann. Zusätzlich ergibt sich die Möglichkeit, mehrere Bildschirmmodi zu mischen.

Nun haben wir uns entschlossen, den Kunstgriff, der dies ermöglicht, anderen C 64-Fans nicht vorzuenthalten. Also entwickelten wir ein Programm in Maschinensprache und dazu ein kleines Demonstrationsprogramm in Basic.

## Zum Programm

Durch Aufruf der Initialisierungsroutine wird der Interruptmechanismus des C 64 verändert. Nicht mehr der Timer der CIA 1, sondern der VIC 6567 löst jetzt den Interrupt aus, und zwar synchron zum Takt des Bildschirmsignals. Außerdem werden vier sogenannte Pseudo-VICs eingerichtet. Alle POKEs, von Spritebewegung über Bildschirmfarbe bis zur Grafikkonfiguration, müssen ab jetzt in diese Pseudo-VICs erfolgen. Jeder dieser Pseudo-VICs ist für einen der vier Bildschirmbereiche zuständig:

Der Bildschirm wird in vier waagerechte Bereiche aufgeteilt, deren Grenzlinien fast beliebig nach oben oder unten verschoben werden können. Jeder einzelne Bereich kann acht Sprites darstellen und eine eigene Farb- und Grafikkonfiguration aufweisen. So können zum Beispiel Normalschrift, HiRes-Grafik, Multicolor-Grafik und eventuell ein selbstdefinierter Zeichensatz gleichzeitig auf dem Bildschirm gezeigt werden.

Provic 64 kann selbstverständlich wieder abgeschaltet werden (bei Kassetten- oder Diskettenoperationen nötig).

Für Maschinensprachefreaks nun eine kurze Funktionsbeschreibung der Interruptroutine:

Bei Aufruf der Einschaltoutine (SYS 52544) wird der IRQ-Vektor auf die Hauptroutine des Provic 64 gestellt und der bisherige Interrupt durch den Timer A der CIA 1 verboten, während der Raster-IRQ des VIC 6567 erlaubt wird.

Sobald der Bildschirmstrahl die eingestellte Rasterzeile erreicht, wird ein Interrupt ausgelöst und der Prozessor bearbeitet die Hauptroutine des Provic 64. In dieser wird zunächst an-

hand eines Zählers (\$ CFFF) festgestellt, welcher Bildschirmbereich an der Reihe ist. Dann wird die Rasterzeile, die das Ende dieses Bildschirms kennzeichnet, eingestellt.

Anschließend werden, falls ein entsprechendes Flag gesetzt ist, die Sprite- und andere Bildschirmparameter in den VIC 6567 übertragen. Nach dem Weiterzählen des IR-Zählers (\$ CFFF) wird entweder der Interrupt beendet, oder zur IRQ-Routine des Betriebssystems gesprungen (nach jedem vierten Interrupt). So werden in der Sekunde 200 Interrupts (vier pro Fernsehbild) ausgelöst und 50 mal in der Sekunde (einmal pro Bild) die normale IRQ-Routine abgearbeitet. Dadurch zählt die interne Uhr TI in 50stel Sekunden und TI\$ wird unbrauchbar.

Beim Aufruf der Ausschaltoutine wird der Raster-IRQ des VIC 6567 unterbunden, der Interrupt des Timers A in CIA 1 erlaubt und der IRQ-Vektor auf die IRQ-Routine des Betriebssystems eingestellt.

## Handhabung der Pseudo-VICs

Im Grunde ist jeder der vier Pseudo-VICs wie der echte VIC zu behandeln. Ausnahmen sind hier nur die Register 30 (Sprite-Sprite-Kollision) und 31 (Sprite-Hintergrund-Kollision), die sich auf den jeweils vorausgegangenen Bildschirmbereich beziehen. Die Register 19 und 20 (Lightpenkoordinaten), sowie 25 und 26 (IRQ-Flags und -maske) werden nicht behandelt, da diese Funktionen nur direkt über den VIC 6567 sinnvoll gehandhabt werden können. Außerdem hat jeder Pseudo-VIC noch zusätzliche Register für zwei Flags (REG 47 und REG 57), acht Sprite-Pointer (REG 48 bis REG 55), Videomatrix-Anfangsadresse Highbyte (REG 56) und die CIA 2, REG 0, Bits 0 und 1 (REG 58) (Adreßbits 14 und 15 des VIC 6567).



So werden die Fähigkeiten von »Provic 64« demonstriert

Die vier Basisadressen der PVICs sind:

PVIC 1	52992 (\$ CF00)	= REG 0
PVIC 2	53056 (\$ CF40)	= REG 0
PVIC 3	53120 (\$ CF80)	= REG 0
PVIC 4	53184 (\$ CFC0)	= REG 0

Da die Pseudo-VICs praktisch gleichberechtigt sind, hier die Registerbeschreibung eines Pseudo-VICs:

<b>REG 0:</b>	X-Koordinate des Sprite 0
<b>REG 1:</b>	Y-Koordinate des Sprite 0 Beachte: Die Y-Koordinaten sollten im Bereich des zugehörigen Bildschirmbereichs liegen, sonst ist der Sprite nicht zu sehen. Näheres siehe unten.
<b>REG 2 bis 15:</b>	Wie REG 0 und 1, aber für Sprites 1 — 7
<b>REG 16:</b>	MSB (höchstes Bit) der X-Koordinaten
<b>REG 17:</b>	Bits 0 bis 2: Y-Abstand der Zeichen vom oberen Bildrand in Rasterzeilen (Softscrolling!) Bit 3: Umschaltung 24/25-Zeilendarstellung Bit 4: Bild an/aus: es hat keinen Sinn, das Bild teilweise ausschalten zu wollen, da der VIC dieses Bit nur einmal pro Fernsehbild prüft (also entweder das ganze Bild an oder aus) Bit 5: HiRes-Grafik-Modus an Bit 6: Hintergrundmehrfarb-Modus an Bit 7: Nummer der Interrupt-Rasterzeile Bit 8; es hat wenig Sinn, dieses Bit zu setzen, da so nur Rasterzeilen angesprochen werden, die unterhalb des Bildschirms liegen. Ist in irgendeinem Pseudo-VIC die 9-Bit-Zahl für die Rasterzeile größer als 311, so wird überhaupt kein IRQ mehr ausgelöst.
<b>REG 18:</b>	Nummer der Rasterzeile Bits 0 — 7; hier ist anzugeben, wann der nächste Interrupt ausgelöst werden soll, das heißt wo der Bildschirmbereich dieses PVICs zu Ende sein soll. Dabei sollte folgende Reihenfolge eingehalten werden: REG 18: PVIC 1 < PVIC 2 < PVIC 3 < PVIC 4 (Zyklische Vertauschungen möglich!)
<b>REG 19 und 20:</b>	nicht verwendet (siehe oben)
<b>REG 21:</b>	Sprite enable (einschalten)
<b>REG 22:</b>	Bits 0 bis 2: Softscrolling in X-Richtung Bit 3: Umschaltungen 38/40-Spaltendarstellung Bit 4: Multicolor-Modus ein Bit 5 bis 7: unbenutzt
<b>REG 23:</b>	Sprite vergrößern in Y-Richtung
<b>REG 24:</b>	Bits 1 bis 3: Adresse Zeichengenerator (Bits 11 bis 13) Bits 4 bis 7: Adresse Video-RAM (Bits 10 bis 13)

### Übergang eines Sprites zwischen zwei Bildschirmbereichen:

Soll ein Sprite zwischen zwei Bildschirmbereichen wechseln, muß in beiden Bereichen derselbe Sprite (also zum Beispiel beidesmal Sprite 4) die gleiche Position besitzen, und zwar so lange, wie der Sprite die Trennlinie zwischen den Bereichen überdeckt. Wird dies nicht beachtet, werden die entsprechenden Sprites zerschnitten und verschoben.

Aktivieren von Provic 64: Von Basic aus mit SYS 52544 und von Maschinensprache aus mit JSR \$CD40.

Ausschalten von Provic 64: Von Basic aus mit SYS 52970 und von Maschinensprache aus mit JSR \$CEEA.

### Der Basic-Lader:

Der Lader erzeugt Provic 64 aus den DATA-Zeilen und falls kein Prüfsummenfehler vorliegt, wird Provic 64 sofort als Maschinenprogramm auf Floppy oder Datasette (Zeile 400 ent-

<b>REG 25 und 26:</b>	nicht verwendet (siehe oben)
<b>REG 27:</b>	Sprite-Priorität vor Hintergrund
<b>REG 28:</b>	Flags für Multicolor-Sprites
<b>REG 29:</b>	Sprite vergrößern in X-Richtung
<b>REG 30:</b>	Sprite-Sprite-Kollision
<b>REG 31:</b>	Sprite-Hintergrund-Kollision Achtung: Geben die Kollisionen des vorangegangenen Bildschirmbereichs an: Findet im Bereich von PVIC 3 eine Kollision statt, wird dies im PVIC 4 registriert. Kollisionen im Bereich von PVIC 4 werden im PVIC 1 registriert. Dieses Register muß gelöscht werden, um neue Kollisionen anzeigen zu können!
<b>REG 32:</b>	Rahmenfarbe
<b>REG 33 bis 36:</b>	Hintergrundfarben 0 bis 3
<b>REG 37 und 38:</b>	Multicolor-Sprite-Farben 0 und 1
<b>REG 39 bis 46:</b>	Farben für Sprites 0 bis 7
<b>REG 47:</b>	Flag für Spritebehandlung; nur wenn der Inhalt dieses Registers nicht Null ist, werden die Register, die etwas mit Sprites zu tun haben, vom PVIC in den VIC 6567 übertragen. Das sind REG 0 bis REG 16, REG 21, REG 23, REG 27 bis REG 31, REG 37 bis 46 sowie REG 48 bis 55. Ist der Inhalt Null, gelten für die Sprites die Werte des vorherigen PVICs, während die Kollisionen erst im nächsten PVIC, in dem REG 48 ungleich Null ist, angezeigt werden.
<b>REG 48 bis 55:</b>	Sprite-Pointer für Sprites 0 bis 7; Die Pointer auf die Bitmuster der Sprites werden nicht mehr in die Speicherzellen 2040 bis 2047 geschrieben, sondern in diese Register des PVICs.
<b>REG 56:</b>	In diesem Register muß das Highbyte der Video-RAM-Anfangsadresse plus 3 stehen; normalerweise also $4 + 3 = 7$ (da der Bildschirm nach dem Einschalten des Computers bei 1024 beginnt, $1024 = \$ 0400$ ). Bei Verlegung des Video-RAMs ist also der Inhalt dieses Registers zu korrigieren.
<b>REG 57:</b>	Flag für Bildschirmparameter-Behandlung; nur wenn der Inhalt dieses Registers nicht Null ist, werden die REG 17, 22, 24, sowie 32 bis 36 und REG 58 in den VIC 6567 übertragen.
<b>REG 58:</b>	Bits 0 und 1: Adressbits 14 und 15 des VIC 6557; werden nach CIA 2 REG 0 Bits 0 und 1 übertragen. Mit diesen Bits kann Video-RAM, Charaktergenerator, Grafik-Bitmap in 16-KByte-Schritten verschoben werden. Da die Bits low-aktiv sind, sind sie beim Einschalten gesetzt (also REG 58 = 3). Bits 2 bis 7: unbenutzt, immer 0.

sprechend ändern!) abgespeichert. Dieses Maschinenprogramm enthält auch gleich die Standardwerte der Pseudo-VICs.

Laden von Provic 64: Im Programm am besten mit der Zeile IF PEEK(52544) > < 120 THEN LOAD "PROVIC 64"; Gerätenummer, 1 die am Anfang des Basic-Programms stehen sollte.

Das Demonstrationsprogramm zeigt einige der Vorzüge von Provic 64. Es ist nur als Anregung gedacht, deshalb verzichten wir hier auf eine nähere Beschreibung.

Provic 64 ist nicht nur für Basic-Programmierer, sondern vor allem auch für Maschinensprache-Freaks gedacht, da erst durch schnelle Maschinenprogramme die Möglichkeiten von Provic 64 voll ausgeschöpft werden können.

(Jürgen und Stefan Haas/rg)

## Tabellarische Übersicht zu Provic

Belegter Adreßraum:

\$CD40	Einschaltroutine
\$CD58	Interruptroutine
\$CEEA	Ausschaltroutine

\$CF00	Pseudo-VIC 1
\$CF40	Pseudo-VIC 2
\$CF80	Pseudo-VIC 3
\$CFC0	Pseudo-VIC 4
\$CFFF	Interruptzähler

Provic 64 einschalten:

in Basic: SYS 52544

in Maschinensprache: JSR \$CD40

Provic 64 ausschalten:

in Basic: SYS 52970

in Maschinensprache JSR \$CEEA

Benutzte RAM-Adressen:

in der Zero-Page: 187 (\$BB)

188 (\$BC)

Rechenzeitwuchs bei aktiviertem Provic 64:

alle Spriteflags (REG 47) und Bildschirmparameterflags (REG 57) gelöscht:

für jedes gesetzte Spriteflag (REG 47): + 2,5 %

zirka

+ 2,4 %

zirka

+ 0,5 %

zirka

+ 15,0 %

Falls der Rechner abstürzt rettet Run-Stop/Restore!

Die Zeitvariable TI zählt bei aktiviertem Provic 64 in 50stel Sekunden

(statt 60stel);

TI\$ wird somit unbrauchbar.

Zeiger für Interruptaussprung von PVIC 3 bis PVIC 4: \$CEE5

Zeiger für Interruptaussprung von PVIC 1: \$CEE8

## Basic-Lader »Provic 64«

10	REM INTERRUPT-ROUTINE ZUR ERZEUGUNG	<118>
11	REM	<154>
12	REM EINES MAXIMAL VIERTEILIGEN	<159>
13	REM	<156>
14	REM BILDSCHIRMES UND 32 SPRITES	<140>
15	REM	<158>
16	REM AUF EINEM COMMODORE 64 COMPUTER	<103>
17	REM	<160>
18	REM 1984 BY GEBR. HAAS	<125>
19	REM	<162>
100	REM PRUEFSUMMEN-KONTROLLE	<049>
101	REM	<244>
110	RESTORE:PS=0	<185>
120	FOR A=0 TO 511	<087>
130	READ WERT	<075>
140	PS=PS+WERT	<112>
150	NEXT A	<089>
160	IF PS<>60913 THEN PRINT"DATA-PRUEFSUMME	
	[SPACE]FALSCH[SPACE]!":END:*	<123>
199	REM	<086>
200	REM MASCHINENCODE UEBERTRAGEN	<060>
201	REM	<088>
210	RESTORE	<094>
220	FOR A=0 TO 447	<195>
230	READ WERT	<175>
240	POKE 52544+A,WERT	<228>
250	NEXT A	<189>
260	FOR A=0 TO 63	<182>
265	READ WERT	<211>
270	FOR B=0 TO 3	<139>
280	POKE 52992+B*64+A,WERT	<022>

290	POKE 52992+B*64+18,95+50*B	<113>
295	NEXT B,A	<089>
299	REM	<187>
300	REM PROVIC 64 ABSPEICHERN	<029>
301	REM	<189>
310	N\$="PROVIC[SPACE]64"	<220>
320	FOR A=1 TO LEN(N\$)	<016>
330	POKE 49151+A,ASC(MID\$(N\$,A,1))	<107>
340	NEXT A	<024>
350	POKE 183,LEN(N\$)	<068>
360	POKE 187,0:POKE 188,192	<054>
370	POKE 185,1	<005>
380	POKE 193,64:POKE 194,205	<121>
390	POKE 174,0:POKE 175,208	<074>
400	POKE 186,8: REM FUER FLOPPY	
	: 8	FUER DATASETTE: 1
		<174>

410	SYS 62957: REM SAVE ROUTINE	<100>
420	END	<037>
999	REM	<121>
1000	REM PROVIC 64 MASCHINENCODE	<104>
1001	REM	<123>
1002	DATA 120,169,88,162,205,141,20,3,142,21,3,	
	169,1,141,13,220,141,26,208	<170>
1003	DATA 141,255,207,88,96,169,1,141,25,208,	
	172,255,207,240,15,136,240,9,136	<105>
1004	DATA 240,3,162,192,44,162,128,44,162,64,44,	
	162,0,189,18,207,141,18,208	<253>
1005	DATA 189,47,207,208,3,76,141,206,173,30,	
	208,29,30,207,157,30,207,173,31	<048>
1006	DATA 208,29,31,207,157,31,207,189,21,207,	
	141,21,208,189,23,207,141,23	<200>
1007	DATA 208,189,29,207,141,29,208,189,0,207,	
	141,0,208,189,1,207,141,1,208	<000>
1008	DATA 189,2,207,141,2,208,189,3,207,141,3,	
	208,189,4,207,141,4,208,189,5	<004>
1009	DATA 207,141,5,208,189,6,207,141,6,208,189,	
	7,207,141,7,208,189,8,207,141	<114>
1010	DATA 8,208,189,9,207,141,9,208,189,10,207,	
	141,10,208,189,11,207,141,11	<001>
1011	DATA 208,189,12,207,141,12,208,189,13,207,	
	141,13,208,189,14,207,141,14	<254>
1012	DATA 208,189,15,207,141,15,208,189,16,207,	
	141,16,208,189,27,207,141,27	<019>
1013	DATA 208,189,28,207,141,28,208,189,37,207,	
	141,37,208,189,38,207,141,38	<038>
1014	DATA 208,189,39,207,141,39,208,189,40,207,	
	141,40,208,189,41,207,141,41	<019>
1015	DATA 208,189,42,207,141,42,208,189,43,207,	
	141,43,208,189,44,207,141,44	<020>
1016	DATA 208,189,45,207,141,45,208,189,46,207,	
	141,46,208,189,56,207,133,188	<094>
1017	DATA 169,248,133,187,160,0,189,48,207,145,	
	187,200,189,49,207,145,187,200	<142>
1018	DATA 189,50,207,145,187,200,189,51,207,145,	
	187,200,189,52,207,145,187	<245>
1019	DATA 200,189,53,207,145,187,200,189,54,207,	
	145,187,200,189,55,207,145	<241>
1020	DATA 187,189,57,207,240,67,189,58,207,41,3,	
	157,58,207,173,0,221,41,252	<037>
1021	DATA 29,58,207,141,0,221,189,17,207,141,17,	
	208,189,22,207,141,22,208,189	<119>
1022	DATA 24,207,141,24,208,189,32,207,141,32,	
	208,189,33,207,141,33,208,189	<019>
1023	DATA 34,207,141,34,208,189,35,207,141,35,	
	208,189,36,207,141,36,208,224	<024>
1024	DATA 192,208,5,169,255,141,255,207,238,255,	
	207,138,240,3,76,129,234,76	<043>
1025	DATA 49,234,120,169,49,162,234,141,20,3,	
	142,21,3,169,129,141,13,220,169	<058>
1026	DATA 0,76,80,205,234,0,0,0,0,0,0,0,0,0,0,	
	0,0,0,0,0,0,27,95,0,0,0,200	<139>
1027	DATA 0,21,121,240,0,0,0,0,14,6,1,2,3,4,0,	
	1,2,3,4,5,6,7,8,0,0,0,0,0	<140>
1028	DATA 0,0,0,7,1,3,0,0,0,0,0	<094>

