

Persönliche Geschichten über Videospiele https://www.videospielgeschichten.de



# Anleitung: Einbau eines Ersatz-ROM in einen C64 oder VIC20

Thilo Niewöhner am Samstag, dem 6. Oktober 2018

Nach einigen Experimenten mit ROMs für den C64 schreibe ich hier eine kurze Anleitung, wie man die alten 2364 ROMs, die sich in vielen 8-bit-Rechnern finden, durch neue ersetzt.

Das Hauptproblem mit diesen ROMs ist, dass es sie praktisch nicht mehr als Ersatzteile gibt. Wenn sie ausfallen – die Frage nach dem "Falls" stellt sich fast nicht mehr –, ist es sehr schwer, wenn nicht unmöglich, sie mit Originalen zu ersetzen. Es gibt bereits viele Anleitungen, wie man (E)EPROM für den Einsatz auf einem C64-Board anpasst. Ein (sehr gutes) Beispiel findet sich bei Tynemouth Software.

## Schritt 1 – Guter Draht zur Hardware

Das erste Problem, das sich stellt: Die Anzahl der Pins und die Belegung des 2364 und des 27C128 unterscheiden sich gravierend. Der einfache direkte Austausch klappt also schon mal nicht.

Die Hardwareanpassung ist also einer der notwendigen Schritte, für den es aber zum

Glück viele verschiedene Lösungen gibt.

Eine, die ich für mich selber erstellt habe, findet sich auf meinem Github Repository. Dieses kleine Board ist die Grundlage für den Einbau eines EPROM aus der 27C-Serie (27C64, 27C128 usw.) auf den vorhandenen Sockel im C64, etwa als BASIC oder KERNAL ROM. Wer das möchte, kann übrigens auch EEPROM verwenden, also elektrisch löschbare ROMs. Diese stammen dann aus der 28C-Serie, sind ansonsten aber genauso einsetzbar.



KiCad-Schaltplan für den "2364 to 27512 adapter", abgeleitet von Tynemouth Softwares Entwurf. (Bild: Thilo Niewöhner)



Entwurf der Platine in KiCad: "2364 to 27512 adapter". (Bild: Thilo Niewöhner)

Wichtig hier: Die Adressleitungen A13-A15 werden fest auf +5V verbunden. Dies hat damit zu tun, dass bei EPROMs auf diesen Pins zum Teil Steuerleitungen liegen, die LOW-aktiv sind. Erst ab einer Größe von 128 kbit sind dort zusätzliche Adressleitungen zu finden.

Diese Pins auf +5V zu legen, verhindert also Fehlfunktionen. Allerdings ist damit auch nur ein 64-Kilobit-Block (8 kByte) adressierbar. Sollen mehrere ROMs über Schalter oder externe Logik ausgewählt werden, ist hier mehr Aufwand in der Beschaltung erforderlich. In diesem Beitrag geht es aber erstmal nur um einen 1:1-Ersatz bestehender EPROMs.

Nachdem nun das EPROM erfolgreich mit dem alten Sockel verbunden werden kann, ist der nächste Schritt, das EPROM mit den richtigen Daten zu füllen, also das EPROM zu "brennen".

Hierbei gibt es eine Stolperfalle: Wird ein EPROM mit mehr als 64 kbit (wie die originalen 2364) verwendet, müssen die 64-kbit-ROMS des C64 im Adreßraum des größeren EPROM richtig plaziert werden. Wie aus dem Schaltplan zu sehen ist, sind die Adreßleitungen A13-A15 auf HIGH gelegt. Dies dient zum einen der Kompatibilität mit kleineren ROMs, bei denen hier keine Adreßleitungen liegen, sondern LOW-aktive Steuerleitungen. Wären diese Leitungen LOW, würde das EPROM nicht wie erwartet funktionieren.

Zum anderen legt es den adressierbaren Bereich auf den untersten Speicherblock (16-bit-Adresse: 000x xxxx xxxx xxxxb) fest.

Um sicherzugehen, dass der Computer auf jeden Fall ein gültiges ROM-Abbild vorfindet

und in seinen Speicher einbinden kann, auch wenn man wegen der Adreßleitungen unsicher ist, ist der einfachste Ansatz, das ROM in mehreren Kopien im EPROM abzulegen. Für ROM-Module, die umschaltbare ROMs bekommen sollen, müssen diese dann aber wiederum anders angelegt werden.

# Schritt 2 – Die richtigen Bits an die richtige Stelle

Um nun das EPROM mit Daten zu füllen, brauchen wir ein EPROM-Programmiergerät bzw. einen EPROM-Brenner.

Das Standardgerät unter Elektronikhobbyisten ist mittlerweile der TL866, hergestellt von AutoElectric in China. In der Maker-Community ist dieses Gerät weitverbreitet und stellt momentan die günstigste Lösung dar, selbst EPROM und EEPROM zu programmieren.

In der Software ist nun als erstes das richtige EPROM auszuwählen. In meinem Beispiel ein AM27C128, also ein EPROM mit 128 kbit bzw. 16 kByte.

Select Device						
Search Device	- Manufactory		Device			
	ACE	~	AM27C010	@DIP32		~
1	ACT		AM27C010	@PLCC32		
Type	ACTRANS		AM27C020	@DIP32		
-	ADVANCE GROUP		AM27C020	@PLCC32		
C ALL	AiT Semi		AM27C040	@DIP32		
ROM/FLASH/NVRAM	ALI (Acer)		AM27C040	@PLCC32		
C MCU/MPU	ALLIANCE		AM27C080	@DIP32		
C PLP CLL CPLP	AMD		AM27C080	@PLCC32		
O PLD/GAL/CPLD	AMIC		AM27C128	@DIP28		
C SRAM/DRAM	ANACHIP		AM27C128	@PLCC32		
C Logic IC	APLUS FLASH		AM27C256	@DIP28		
	ARTSCHIP		AM27C256	@PLCC32		
	ASAHI KASEI (AKM)		AM27C512	@DIP28		
	ASD		AM27C512	@PLCC32		
	ASI		AM27C1024	@DIP40		
	ATC		AM27C2048	@DIP40		
	ATMEL		AM27C4096	@DIP40		
	BRITING	~	AM27LV010	@DTP32		× .
Advanced Micro Devices Inc.	IC Total: 14	177	Select		Cancel	

Auswahl EPROM AMD AM27C128. (Bild: Thilo Niewöhner)

Danach zeigt das Fenster, dass der Puffer zunächst mit "FF" gefüllt wird. Das repräsentiert leeren Speicher, also den Urzustand des EPROM, gibt aber keine Information über den tatsächlichen Zustand des EPROMs. Dieses muss natürlich ebenfalls leer sein, was über die "Blank Check"-Funktion des TL866 überprüft werden kann.

Jetzt laden wir die .bin-Datei für das ROM, das wir brennen wollen. Hier ist es das Original Kernal-ROM.

Nach der Auswahl der bin-Datei und einem Klick auf "Öffnen" erscheint folgendes Bild:

· 🖬 🛱	. F			<b>AD</b>	Æ	25	ġ	FIL	5 <b>R</b>	)  1	8			P	B	TEST				
elect IC — AM27C12	8 @	DIP28		] -	Chip IC	Infor pType: Size:	rmati EEP Ox	on (No ROM 4000 (	Projo Chl Bytes	ect og Sum:(	pened 0x0031	) F COOI	)						nformat	tion
roduct Iden ChipID:	tific	ation		<sup>S</sup>	et Int 40P	erfac adapte	er C	ICSP	port	Γ	ICSP	_VCC	Enabl	e <b>B</b>	uff se	lect –			Code	Memo
Address	0 FF	1 FF	2 Fi	le loai	d Opti	ons		-								×	II			Ξ.
000010:	FF	FF	F	-File l	ormat BINAR	v			oad mo	ode						-				•
000020:	FF	FF	F	C	INTEL	HEX			No	rmal					-					•
000030:	FF	FF	F											-		-				(E)
000040:	FF	FF	F							From	n File S	itart A	ddr(He	x):	00000					•
000050:	FF	FF	-							TO B	offer S	trat Ad	H-CHE	xo. 🗖	00000	1				<u>e)</u>
000000:	FF	FF	F F	- To R	egion										00000	-				•
000070.	EE.	FF	5	G C	ode Me	mory		- C	lear Bi	offer w	hen lo	ading t	he file				1.111			•
000000.	FF	FF	E						Class	1		4.6								•
888888	FF	FF	Ē						Clear	r burren	with	Geradit			_					
0000B0:	FF	FF	Ē																	
000000:	FF	FF	Ē							OK		1	(	ancel	1					
0000D0:	FF	FF	F													1				
0000E0:	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF .				-
0000F0:	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF .				
ntions —								Conf	ie In	forms	ton-			New York		010111000				
Erase befo	2.1		1	V Che	eck II	ţ –														
Verify aft	er		1	- Aut	o SN_	NUM			V	PP V	/olt	age:	13.	50V .	-	VD	D Writ	e: 6.	25V •	•
Skip OxFF	Se	tRang	e: 🔎	ALL	CS	ect									-			. i-		-
	0.	. 000	00000	5 → D	000031	FFF				VCC	Ver	ify:	5.0	ov .	-	Pul	s Dela	Y: 10	Ous .	·

MiniPro ROM laden. (Bild: Thilo Niewöhner)

Lasst alle Parameter unverändert und klickt "OK". Damit wird die .bin-Datei ab der Adresse 00000h in den Puffer geladen und der Rest des Puffers mit FFh gefüllt. (FFh entspricht leeren Speicherplätzen)

Da die normale ROM-Größe des C64 8 kByte ist, könnt Ihr nachschauen und feststellen, dass ab 02000h der Puffer nur noch "FF" zeigt.

Ka Mi	iniPro v6.60	)																	<u> </u>		Х
File( <u>F</u> )	Select IC	C( <u>S</u> )	Proje	ct( <u>P</u> )	Devi	ce( <u>D</u> )	Too	ls(V)	Help	»( <u>Н</u> )	Lang	uage(	<u>L</u> )								
2		; 🖥			٩D	Æ	25	ð,	FIL	۲ ( <b>R</b>		?			P		}   <del>1</del>	छा 🛄			
Sel	Lect IC — AM27C12	80	DIP28			Chi IC	Info: pType Size:	rmati : EEF Ox	on (No 'ROM 4000	Proj Chl Bytes	ect o «Sum:	pened) 0x0021	) — 7 A70,	A					🏴 📜 In	formatio	n
-Pro Ch	duct Iden ipID:	tific	ation		Se I	t Int 40P	erfa adapt	er O	ICSP	port		ICSP	VCC :	Enabl	e	Buff	seled	;t		Code M	íemo
Ad	ldress	0	1	2	3	4	5	6	7	8	9	A	В	С	D	E	F	ASCII			
0	01F90:	40	18	FE	40	B9	ED	40	C7	ED	40	25	FE	4C	34	FE	4C	LL.	.LL%	6.L4.L	
0	01FA0:	87	EA	40	21	FE	40	13	EE	40	DD	ED	40	EF	ED	40	FE			L	
0	01FB0:	ED	40	OC.	ED	40	09	ED	40	07	FE	40	00	FE	40	F9	FD	-LL	ايداي		
9	01FC0:	60	10	03	60	10	03	60	1E	03	60	20	03	60	22	03	60	11.	.11	·1".1	
6	01FD0:	24	03	00	20	03	40	9E	F4	40	DD	15	40	E4	10	40	DD	\$.16.		L.L.	
0	01560:	F0 60	80	28	03 hC	00	ZH CC	03 59	50	26	03	40	2E 2B	F0	46	65	E5 EE	.1(.1)	*.1,.L		
6	02000.	40 FF	FF	FF	40 FF	FF	FF	FF	FF	42 FF	FF	FF	FE	FF	FF	40 FF	FF		. nnbru		-
6	02000.	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF				
9	02020:	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF				
G	02030:	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF				
6	02040:	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF				
6	02 05 0 :	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF				
0	02 06 0 :	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF				
6	02070:	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF				
9	02080:	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF				
_0⊳t	tions Erase befo	r		ŀ	🗸 Che	ck II	I		Conf	ig Ir	form	aton —									
	Verify aft	er		Γ	Aut	o SN_	NUM			7	PP 1	Volt	age:	13.	50V	•		VDD Wri	te: 6.2	25V -	
	Skip OxFF Blank Chec	Se k	tRang 000	;e: 💽	ALL ->	O S	ect FFF				VCC	Ver	ify:	5.0	0V	•	P	uls Del	ay: 100	Dus 🔻	
Ready								F	lard	ware	Int	erfa	ce 1	/er:	No	Devi	ice!		0	0000 0000	

MiniPro ROM geladen ab Adress \$2000. (Bild: Thilo Niewöhner)

Das ist sehr gut, da wir ab dieser Adresse dasselbe ROM noch einmal einfügen wollen. Das machen wir um sicherzugehen, dass wir mit der voreingestellten Adresse (siehe "A13-A15 auf 5V") auf jeden Fall ein gültiges ROM-Abbild erwischen. Im späteren Ausbau lassen sich verschiedene ROMs anlegen, die dann von außen umschaltbar sein können.

Sollten hier aber Daten vorhanden sein, prüft bitte, ob Ihr nicht die falsche Datei erwischt habt. Die richtigen .bin-Dateien haben genau 8 kByte.

Dazu wiederholen wir den oben beschrieben Prozeß, verändern aber beim zweiten Dialog die Parameter.

Lasst den Parameter "From File Start Address" auf 0000h. Das bedeutet, dass die .bin-Datei von Anfang an geladen und nichts ausgelassen wird, ganz wie vorhin.

Aber den anderen Parameter "To Buffer Start Address" müssen wir auf 2000h einstellen und "Clear buffer when loading the file" auf "Disabled".

Damit wird die zweite Kopie überschneidungsfrei nach der Ersten in den Puffer eingelesen, ohne den Rest des Puffers wieder zu leeren (und damit unsere erste Kopie zu löschen).

🐞 MiniPro v6.60		- 🗆 🗙
File( <u>F</u> ) Select IC( <u>S</u> ) Project( <u>P</u> )	Device( <u>D</u> ) Tools( <u>V</u> ) Help( <u>H</u> ) Language( <u>L</u> )	
🗳 🖬 🛸 🌄 🖪	耶 跎 🌮 识 🔛 🕱 💡 👘 🖡	<b>#</b>
Select IC	IC Information(No Project opened)	
AM27C128 @DIP28	ChipType: EEFROM ChkSum: 0x001F 8E14 IC Size: 0x4000 Bytes	Ti Information
Product Identification	Set Interface	Code Memo
Address U I Z		
001F90: 40 18 F	-File Format	
001FB0: ED 4C 0	BINARY     Normal	
001FC0: 6C 1A 0	C INTEL HEX	.111 .1".1
001FD0: 24 03 6	From File Start Addr(Hex): 00000	1&.LLL.
001FE0: F6 6C 2		(.1*.1,.LL
001FF0: 4C 0A E	TO Buffer Strat Addr(HEX): 02000	.LRRBYCH.
002000: 85 56 2		a
002010: BC A5 0	Code Memory     Clear Buffer when loading the file	i8Hi
002020: B4 61 9	Disable	.a.iV.p S.
002030: 20 B4 B		Yoh
002040: B9 BA 6		.q.r (.
002050: 5D EO A	OK Cancel	.WL(q.r
002060: B1 71 8_		.g.qr.q.r
002070: 20 28 BA	A5 71 A4 72 18 69 05 90 01 C8 85 71 84 (	q.r.iq.
002080: 72 20 67	B8 A9 5C A0 00 C6 67 D0 E4 60 98 35 44 r	q\q5D 👻
Options Erase befor	✓ Check ID	
🔽 Verify after	Auto SN_NUM VPP Voltage: 13.50V - VDD	Write: 6.25V -
Skip OxFF SetRange: @	ALL C Sect	
Blank Check0x00000000	VCC Verify: 5.00V Vels	Delay: 100us
Ready	Hardware Interface Ver: No Device!	0000 0000

MiniPro ROM Kopie laden ab 2000h. (Bild: Thilo Niewöhner)

Schauen wir in den Puffer, sehen wir, dass jetzt ab 2000h statt "FF" ebenfalls Daten vorhanden sind. Ihr werdet auch sehen, dass sich hier die Daten von ganz oben wiederholen.

Ihr solltet auch überprüfen, dass der Puffer bis zum Ende beschrieben wurde.

🜆 Min	iPro v6.60																		<u> </u>		×
File( <u>F</u> )	Select IC	C( <u>S</u> )	Projec	ct(P)	Devi	ce( <u>D</u> )	Too	ls(⊻)	Help	р <u>(Н</u> )	Lang	juage(	<u>_)</u>								
<b></b>		j 🖥			٩D	Ð	25	ġ,	FIL	<mark>۲</mark>		8			P		ן <u>ד</u>	छा 🛄			
Sele	ect IC —					$\Box^{IC}$	Info	rmati	on(No	Proj	ect o	pened)	I ———								
	AM27C12	80	DIP28			Chi; IC	pType Size:	EEF Ox	'ROM 4000	Chl Bytes	kSum∶∣	0x001F	8E14	1					"ü In	formatio	n 📕
Prod	luct Iden	tific	ation		_Se	t Int	erfac	e								Buff	selec	,t —			
Chi	pID:				$\bullet$	40P	adapt	er O	ICSP	port		ICSP	VCC I	Enabl	e					Code M	emo
Add	ress	0	1	2	3	4	5	6	7	8	9	Å	В	С	D	E	F	ASCII			
00	)3F00:	DD	8D	07	DD	40	59	EF	AD	95	02	8D	06	DD	AD	96	02	LY			
00	03F10:	8D	07	DD	A9	11	8D	ØF	DD	A9	12	4D	A1	02	8D	A1	02		M	1	
00	3F20:	A9	FF	8D	06	DD	8D	07	DD	AE	98	02	86	A8	60	AA	AD				_
00	13F30:	90	02	28	84	88	09	60	80	99	02	98	09	00	80	98	02	*		1	-
00	J3F40: 19E50+	00	20	10	08 E 0	00	29 60	16	48	48	8н 1 Л	48	20	48	DH EC	6D	12	A			-
00	3F60.	na	27 DØ	FR	AD	10	na	20	01	80	86	62	20 4C	nn	ED	A9	81	- /	)		-
00	3F70:	8D	ØD	DC	AD	ØE	DC	29	80	09	11	8D	ØE	DC	40	8E	EE		Ś		
00	3F80:	03	40	5B	FF	40	A3	FD	40	50	FD	40	15	FD	40	18	FD	.L[.L	.LP.L	L	
00	3F90:	40	18	FE	40	B9	ED	40	C7	ED	40	25	FE	40	34	FE	40	LL.	.LL%	.L4.L	
00	)3FA0:	87	EA	40	21	FE	40	13	EE	40	DD	ED	40	EF	ED	40	FE	1.1.1.1.1		LL.	
00	)3FB0:	ED	4C	0C	ED	40	09	ED	40	07	FE	4C	00	FE	40	F9	FD				
00	)3FC0:	6C	18	03	6C	10	03	6C	1E	03	6C	20	03	6C	22	03	60	11.	.11	.1".1	
00	3FD0:	24	03	60	26	03	40	9E	F4	40	DD	F5	40	E4	Fó	40	DD	\$.1&.1		L	
00	3FE0:	F6	60	28	03	60	28	03	60	20	03	40	9B	Fő	40	05	E5	.1(.1	*.1,.L		
1 96	13FF0:	40	UA	£5	4C	មម	£5	52	52	42	59	43	FE	E2	FC	48	FF	L	RRBYC	· · · · H ·	
-Opti E:	ons rase befo	r		F	🗸 Che	ck II	1	$   ^{\mathbf{IC}}$	Conf	ig Ir	forms	aton —									
🛛 🔽 V.	erify aft	er		Г	Aut	o SN_	NUM			7	PP 1	Volta	age:	13.	50V	•		VDD Wri	te: 6.2	5V 🔻	
∏ SÌ ∏ BÌ	kip OxFF lank Chec	Se sk	tRang 000	;e: 💽		C S	ect FFF				VCC	Veri	ify:	5.0	ov	•	P	uls Del	ay: 100	us 🔻	
Ready								Н	lard	ware	Int	erfa	ce I	/er:	No	Devi	.ce!		0	000 0000	

MiniPro ROM geladen Puffer gefüllt bis 3FF0h. (Bild: Thilo Niewöhner)

Schaut Euch die Zeilen 001FF0h und 003FF0h (und gerne auch weitere solche Pärchen) an. Diese Zeilen sind die jeweils letzten Bytes der ROM-Datei und müssen identisch sein.

Sind sie das, brennen wir jetzt das EPROM.

Anmerkung: Nicht immer sind die Parameter für das Programmieren korrekt.

In der Regel ist die Programmierspannung bei 12,5 V, die Lesespannung 5 V. Prüft das Datenblatt Eurer (E)EPROMs und korrigiert die Werte. Ansonsten kann es zu Fehlern kommen.

Eile(E) Select IC(S) P	roject(D)	Device(D)	) Tools()/		о( <u>Ц</u> )	Language	(1)							- <u>-</u>	-		×
			) 25) ř	FIL	L [ <b>2</b>		.(L)				TÉ	ST	itttiin.				
-Select IC			Toformat	ion(No	Proje	r • •	i)										
AW27C128 @D	TP28		ipType: E	EPROM	Chk	Sum:0x001	1F 8E14	4						<b></b>	Infor	matio	n l
Product Identifice	tion		Size: (	0x4000	Bytes				F	tt -		+		<u></u>	mo	matro	للس
ChipID:		• 40P	adapter (	) ICSP	port	🔲 ICSI	P_VCC :	Enable	e	Juli S	erec.					Code M	emo
Address 0	1 2	3 4	56	7	8	9 A	B	С	D	E	F	ASC	II				
001FB0: ED	40 00	ED 40	09 El	) 40	07 02	FE 40	00	FE	40	F9	FD	.L 1	L.	ųĿ.		L	
001FD0: 24	03 6C	26 03	4C 9	5 F4	4C	DD F5	40	60 E4	F6	4C	DD	\$.	18.1				
001FE0: F6	6C 28	03 60	2A 0	3 6C	20	03 40	9B	F6	40	05	E5	.1	(,1)	•.1,	.L	L	
00100: 85	0H E5	4C 00 0F BC	A5 6	2 52 1 C9	42 88	59 43 90 03	FE 20	E2 D4	BA	48 20	FF CC			. кке . а	YG		
002010: BC	A5 07	18 69	81 F	9 F3	38	E9 01	48	A2	05	B5	69		i	8	н.	i	
002020: B4	61 95 Dh DC	61 94	69 CI	a 10 - 20	F5	A5 56	85	70 95	20	53	B8	. a	.a.i	i	.V.p	S.	
002040: B9	BA 60	85 71	84 72	2 20	CA	BB A9	57	20	28	BA	20		`.q.	r		(.	
002050: 5D	EØ A9	57 A0	00 4	28	BA	85 71	84	72	20	67	BB	1.	.w.	.L(.	. q . r		
002060: B1 002070: 20	71 85 28 BA	67 A4 A5 71	71 C	3 98 2 18	D0 69	02 E6 05 90	72 01	85 C8	71	A4 71	72	• 9	.g.o	l r.i	r.	q.r	
002080: 72	20 67	B8 A9	5C A	00 00	C6	67 D0	E4	60	98	35	44	r	g	\	g	.5D	
002090: 7A	00 68	28 B1	46 0	9 20	2B	BC 30	37	D0	20	20	F3	z.	h(.,	F	. 07 .		
0020H0: FF	80 22	64 23	нө ө	а ві ТС С(	~	65 02	60	ы	22	65	04					. u	-
Erase befor	ŀ	Check I	D	IC COM		TOLMATOR		_						_			
Verify after	] مىرىيە	Auto SN	_NUM		v	'PP Volt	age:	12.9	50V _	•	1	VDD	Wri	te:	5.000	•	
Skip UxFF Set	00000000		3FFF			VCC Ver	rify:	5.00	vo	-	Pu	uls	Del	ay:	100us	-	
Blank Check	,	,															
Ready				Hardu	ware	Interf	ace 1	Ver:	No	Devi	ce!				0000	0000	
MiniPro ROM Sr		/-															
	Dannur	ngen. (E	Blld: Th	nilo N	liewä	öhner)											
MiniPro v6.60	bannur	igen. (ł	Blld: Tr	ilo N	liewö	öhner)								-	-		$\times$
MiniPro v6.60 File(F) Select IC(S) P	roject(P)	Device(D)	BIId: Th	nilo N ) Help	liewċ ₀(H)	öhner) Language	e(L)							-	-		×
Image: Select IC(S)       File(F)       Select IC(S)       P       P       P	roject(P)	Device(D)	311d: Th Tools(V ) 25 d	nilo N ) Help R E	liewć •(H) <b>5 (R</b>	öhner) Language	:(L)		P		TĚ	ST		-			×
MiniPro v6.60 File(F) Select IC(S) P	roject(P)	Device(D)	311d: Th Tools(V 25 d Informat	Dilo N ) Help P EL Si on (No EPROM	o(H)	öhner) Language () ? () ? () ? () ? () ?	:(L) 1) 1F 8E1:	4	P		TĚ	ŝt		-	-		×
MiniPro v6.60 File(F) Select IC(S) P File(F) Select IC Select IC AM27C128 m Chip	roject(P)	Device(D)	311d: Tr Tools(V ) 25 Č Informat IpType: E	nilo N ) Help R Ell i on (No EPROM	o(H) Proje Chk	öhner) Language () ? () ? () ? () ? () ? () ? () ? () ?	:(L) 1) 1F 8E1-	4	P		TE	ŝŢ			Infor	matio	×
MiniProv6.60 File(F) Select IC(S) P File(F) Select IC(S) P Select IC Select IC AM27C128 an Chip ChipTh	roject(P)	Device(D)	311d: Tr Tools(V 25 d Informat pType: E	nilo N ) Help P FIL (ion (No EPROM	liewć p(H) b R Proje Chk	öhner) Language () ? ect openec Sum: 0x001	:(L) 1) 1F 8E14	4	P		TE	ŝt		<b>F</b>	Infor	matio	×
MiniPro v6.60 File(F) Select IC(S) P Select IC Select IC AM27C128 Product Identi ChipID: Product Identi	roject(P)	Device(D)	Bild: Th Tools(V ) 25 d Informat	hilo N ) Help P FIL i on (No EPROM AM27	(H) (H) Proje Chk	Danguage         Language         P         P         P         Sum: 0x001	:(L) 1)	4	P		cation	ST I	coket -	<b>**</b> **	Infor	<b>matio</b> Code M	
MiniPro v6.60 File(F) Select IC(S) P Select IC Select IC AM27C128 Product Identi ChipID: Address 001FR0:	roject(P) 1 E	Device(D)	Bild: Th Tools(V ) 25 E Informat	hilo N ) Help P FIL ion (No EPROM AM27	(H) Proje Chk	Description         Language         Language         P         R	:(L) 1) 1F 8E1-	4	P		catior	sīt	coket -	<b>**</b>	Infor	matio Code M	
MiniPro v6.60 File(F) Select IC(S) P File(F) Select IC(S) P Select IC AM27C128 an Chip Product Identi ChipID: 901FB0: 901FC0: St	roject(P) 1 E 1 Program Program Ranj ✓ CODE M art Adr:	Device(D)	Bild: Th Tools(V ) 25 d Informat (pType: E	hilo N ) Help E EL ion(No EPROM	(H) Proje Chk	Description         Language         Image: Constraint of the section opened         Sum: Ox000         8	:(L) 1) IF 8E14	4	P		cation	ST	coket -	·····	Infor	matio Code M L	
MiniPro v6.60 File(F) Select IC(S) P Select IC Select IC AM27C128 Chip Product Identi ChipID: Address 001FB0: 001FB0: 001FF0:	roject(P) Program rogram Ranp CODE M art Adr: End Adr:	Device(D)	Bild: Th Tools(V ) 25 č Informat apType: E	hilo N ) Help P P ion(No EPROM AM27	(H) Proje Chk	Description         Language         Language         P	:(L) 1) 1F 8E14	4	P		cation	ST   1 in S	coket		- Infor	matio Code M	×
MiniPro v6.60 File(F) Select IC(S) P Select IC Select IC AM27C128 Product Identi ChipID: Address 001FB0: 001FB0: 001FE0: 001FF0: 001FF0:	roject(P) roject(P) roga Program Program Rogram Ranj CODE M art Adr: End Adr:	Device(D)	Bild: Th Tools(V) 25 E Informat pType: E	hilo N ) Help ? [1] :ion(No EPROM	(H) Proje Chk	Description         Language         Language         Image: Comparison of the system         Sum: 0x001         8	:(L) 1)	4				st l			Infor L L L	matio Code M L  L L	
MiniPro v6.60 File(F) Select IC(S) P Select IC AM27C128 on ChipID: Product Identi ChipID: 901FB0: 901FE	roject(P) 1 E 1 Program Program Ranj ✓ CODE M art Adr: End Adr:	Device(D) Device(D) Device(D) Device(D) Device(D) Chi Chi ge Memory 0000000 Plug IC in	Bild: Th Tools(V ) 25 d Informat pType: E 0 7 to 40PIN so	hilo N ) Help E FIL i on (No EPROM AM27 ocket,Cliv	o(H) Chk Chk 7C12	Display to the sector opened Sum : 0x001 8 00000000000000000000000000000000	:(L) 1) 1F 8E1-	4				st l			Infor	matio Code M L  L L	×
MiniPro v6.60 File(F) Select IC(S) P Select IC AM27C128 on Product Identi ChipID: Address 001FE0: 001FE0: 001FE0: 001FE0: 001FE0: 001FE0: 001FE0: 002610: 00200: 00200:	roject(P) 1 E	Device(D)	Bild: Th Tools(V) 25 d Informat Informat Informat	hilo N ) Help P FIL i on (No EPROM AM27 ocket,Cliv	o(H) Proje Chk	Description         Language         Language         Image: Control operation         Sum : 0x001         8         ogram>-But	:(L) 1) 1F 8E1-	4				st			L 	matio Code M 1 .L .H i	
MiniPro v6.60 File(F) Select IC(S) P Select IC Select IC AM27C128 on Product Identi ChipID: Address 001FE0: 001FE0: 001FE0: 001FE0: 001FE0: 001FF0: 001FF0: 001FF0: 001FF0: 002000: 002010: 0020	roject(P) Program rogram Ranı CODE M art Adr: End Adr:	Device(D)	Bild: Th Tools(V 25 E Informat pType: E 0 TF to 40PIN sc	hilo N ) Help P FIL i on (No EPROM AM27 scket,Cliv	o(H) Proje Chk	Description         Language         Language         Image: Content of the second	ton	4				st			L 1 .L  	matio Code M L ".1 .L. L .H. i S. oh	
MiniPro v6.60 File(F) Select IC(S) P Select IC AM27C128 and Product Identi ChipID: Address 001FB0: 001FB0: 001FE0: 001FF0: 002010: 002010: 002010: 002030: 002040: 002000: 002000: 002000: 002000: 002000: 002000: 0000000000	roject(P) 1 E   TP28 Program rogram Ranı ✓ CODE M art Adr: End Adr:	Device(D)	Bild: Th Tools(V 25 d Informat pType: E 10 10 10 10 10 10 10 10 10 10	hilo N ) Help ) Help	o(H) Proje Chk	Description         Language         Language         Image: Comparison of the second sec	ton	4				sin S			L 1 L C     	matio Code M L 	
MiniPro v6.60 File(F) Select IC(S) P Select IC AM27C128 an Product Identi ChipID: Address 001FB0: 001FE0: 001FE0: 001FF0: 002010: 002010: 002010: 002010: 002010: 002050: 002050: 002050: 002060:	roject(P) 1 E 1 Program Program Ranj ✓ CODE M art Adr: End Adr:	Device(D)	Bild: Th Tools(V 25 d Informat pType: E to 40PIN sc	hilo N ) Help E FIL i on (No EPROM AM27 ocket, Cliv	o(H) Proje Chk	Description         Language         Language         Language         Sum: 0x001         Sum: 0x001         8         ogram>-But	ton	4				sī   1 in \$			L 1 1 1 1 1 1 1	matio Code M L  L      	
MiniPro v6.60 File(F) Select IC(S) P Select IC AM27C128 on Product Identi ChipID: Address 001FF0: 001FF0: 001FF0: 001FF0: 002000: 002010: 002050: 002050: 002050: 002050: 002070: 002070:	roject(P) TP28 Program rogram Rang CODE N art Adr: End Adr:	Device(D)	Bild: Tr Tools(V ) 25 C : Informat :pType: E : to 40PIN ac	hilo N ) Help P FIL i on (No EPROM AM27 ocket,Cliv	o(H) Proje Chk	Description         Language         Language         Image: Content of the second	:(L) 1) 1F 8E1 ton	4				n in S			L 1 1 1 1 1 1 1	matio Code M L i i i i i i i	
MiniPro v6.60 File(F) Select IC(S) P Select IC AM27C128 MiniPro v6.60 File(F) Select IC(S) P Select IC AM27C128 AM27C1	roject(P) Program rogram Ranı CODE M art Adr: End Adr:	Device(D)	Bild: Tr Tools(V 25 E Informat pType: E 10 TF to 40PIN sc	hilo N ) Help ) Help ) Fill :ion (No EPROM AM27 ocket, Cliv	o(H) Proje Chk	Description         Language         Language         Image: Comparison of the second sec	ton	4				sin \$			L 11.L. 12 14 1	matio Code M L        	
MiniPro v6.60 File(F) Select IC(S) P Select IC AM27C128 an Product Identi ChipID: Address 001FB0: 001FB0: 001FE0: 001FE0: 002060: 002016: 002040: 0020000: 0020000: 0020000: 00200000: 00200000: 0020000	roject(P) 1 E 1 Program Program Ranj ✓ CODE M art Adr: End Adr:	Device(D)	Bild: Th Tools(V 25 d Informat pType: E to 40PIN sc Program	hilo N ) Help I FIL i on (No EPROM AM27 ocket, Cliv	cz	Diphner)	ton	4				in \$			L	matio Code M L       	
MiniPro v6.60 File(F) Select IC(S) P Select IC AM27C128 and Product Identi ChipID: Address 001FF0: 001FF0: 001FF0: 001FF0: 001FF0: 001FF0: 002000: 002010: 002050: 0020	roject(P)	Device(D)	Bild: Tr Tools(V 25 d Informat pType: E 0 7 to 40PIN sc Program	hilo N ) Help P FIL i on (No EPROM AM27 ocket,Cliv	ek-Pro	Display and a second se	ton	4				s, T			L 1 L C V.p 4 C V.p 4 C V.p 5 C V.p C V.p C V.p C V.p C V.p C V.p C V.p C V.p C V.p C V.p C V.p C V.p C V.p C V.p C V.p V.p V.p V.p V.p V.p V.p V.p V.p	matio Code M L ".1 .L. L L  S. oh (.  q.r  5D	
MiniPro v6.60 File(F) Select IC(S) P Select IC AM27C128 om Product Identi ChipID: Address 001FE 0: 001FE 0: 002 09 0: 002 000 0: 002	roject(P) 1 E 1 Program Program Ranı ▼ CODE M art Adr: 2nd Adr: 2nd Adr: 5nd	Device(D)	Bild: Tr Tools(V 25 E Informat pType: E to 40PIN sc Program D NUM	Nilo N Help File i on (No EPROM AM27 ocket, Cliv	o(H) Proje Chk C12 Cc Cc Cc Cc Cc Cc Cc Cc Cc Cc	Dinner) Language Tot openec Sum: 0x000 8 ogram>-But tormaton- PP Volt	ton	4				a in S	coket	<b>E</b>	L 1.L 1.L 1.L 1.L 1.C 1.L	matio Code M L        	
MiniPro v6.60 File(F) Select IC(S) P Select IC AM27C128 and Product Identi ChipID: Address 001FE0: 001FE0: 001FE0: 001FE0: 001FE0: 001FE0: 001FE0: 002020: 002080: 0020	roject(P) 1 E 1 Program Program Ranı ✓ CODE N art Adr: End Adr: End Adr: F Range: •	Device(D)	Bild: Tr ) Tools(V ) 25 E : Informat .pType: E to 40PIN sc Program D_NUM Sect	hilo N ) Help P FIL :ion(No EPROM AM27 ocket,Cliv IC Cont	ck- <pro< td=""><td>Dinner) Language Tot opened Sum: 0x000 8 ogram&gt;-But formaton PP Volt</td><td>:(L) I) IF 8E14</td><td>4</td><td><b>5</b>0V</td><td></td><td></td><td>in \$</td><td>coket</td><td></td><td>L 1 L C V.p  W.q  87. b</td><td>matio</td><td></td></pro<>	Dinner) Language Tot opened Sum: 0x000 8 ogram>-But formaton PP Volt	:(L) I) IF 8E14	4	<b>5</b> 0V			in \$	coket		L 1 L C V.p  W.q  87. b	matio	
MiniPro v6.60 File(F) Select IC(S) P Select IC AM27C128 and Product Identi ChipID: Address 001FE 0: 001FE 0: 001FE 0: 001FE 0: 001FE 0: 001FE 0: 001FE 0: 001FE 0: 002 010: 002 010: 002 010: 002 03 0: 002 03 0: 002 05 0: 002 05 0: 002 05 0: 002 08 0: 002 09 0: 002 00 0: 002	roject(P) 1 E TP28 Program Program Range: (● 100000000	Device(D)	Bild: Tr Tools(V 25 d Informat pType: E to 40PIN sc Program Program	Nilo N Help I Fill i on (No EPROM AM27 ocket, Cliv	ck- <pro< td=""><td>Deprive the second seco</td><td>ton</td><td>4</td><td>50V .</td><td></td><td></td><td>vin \$</td><td>wri Del.</td><td><b>te:</b>[]</td><td>L L C      </td><td>matio</td><td></td></pro<>	Deprive the second seco	ton	4	50V .			vin \$	wri Del.	<b>te:</b> []	L L C      	matio	

MiniPro ROM programmieren. (Bild: Thilo Niewöhner)

MiniPro Programmer wird in der Standardkonfiguration eine abschließende Überprüfung

des EPROM-Inhaltes vornehmen und Euch eine Rückmeldung geben.

# Schritt 3 – Test am lebenden Objekt

Hat alles geklappt, könnt Ihr das EPROM jetzt im C64 ausprobieren.

Achtung bei der Einbaulage!

Achtet auf die kleine Kerbe nahe Pin 1, damit Ihr es nicht verkehrtherum einbaut.

Hat etwas nicht geklappt, fangt von vorne an. Bei EPROMs (das sind die mit dem kleinen Fenster) ist es wahrscheinlich notwendig sie zu löschen. Dafür braucht Ihr aber ein UV-Löschgerät. EEPROMs dagegen werden vom TL866 automatisch gelöscht.

Viel Erfolg bei Euren Reparaturen und haltet die C64, VIC20 und wie sie alle heißen in Betrieb.

Es wäre schade drum!

Schaut auch auf Twitter vorbei und lest die Twitter-Threads und die verlinkten Blogbeiträge der Retro Community.

Es wird immer wieder Lesenswertes und Interessantes gepostet.

Ach ja: Falls Ihr noch einen Fehler findet, lasst es mich wissen.

## Ergänzungen/Anmerkungen

#### EPROM/EEPROM/OTP-ROM

Ich schreibe hier zur Übersichtlichkeit immer von EPROM. Allerdings habe ich mehrere dieser Umbauten auch mit EEPROMs durchgeführt.

"EPROM" schließt also sowohl die UV-löschbaren EPROMs als auch die elektrisch löschbaren EEPROMs ein. Auch sog. OTP-ROM ("One-Time Programmable") sind hier prinzipiell einsetzbar, wenn auch in vielen Fällen unwirtschaftlich, da sie nur ein einziges Mal beschrieben werden können. Für Dinge wie den Ersatz des PLA kann es aber notwendig sein, auf schnellere OTP-ROM mit <45ns zurückzugreifen.

#### TL866

Die Varianten TL866A und TL866CS unterscheiden sich in einer zusätzlichen 6-poligen Schnittstelle und geänderter Software, die das einfachere Programmieren von Mikrocontrollern erlaubt. Dabei hat der TL866A diese Schnittstelle bereits eingebaut, beim CS ist ein späterer Umbau möglich, erfordert aber einige Arbeit.

Wer noch kein Programmiergerät hat, sollte die A-Version bestellen, da die Preisunterschiede meist marginal, die zusätzlichen Möglichkeiten aber wenigstens interessant sind.

Zum aktuellen TL866 II kann ich noch nichts sagen. Wenn Ihr da mehr wisst, sagt Bescheid! ?

#### No Device!

Ja, das ist korrekt. Als ich die Screenshots erstellt habe, hatte ich den TL866 nicht zur Hand.?

Für diesen Zweck ist das nicht kritisch. Wenn Ihr aber selbst ROMs brennen wollt, achtet darauf, dass Euer Brenner korrekt erkannt wird.

### Weiterführende Links

- GitHub profile von VintageProject (Thilo Niewöhner)
- Commodore VIC20 Repair bei Tynemouth Software

Dieser Beitrag wurde publiziert am Samstag, dem 6. Oktober 2018 um 10:00 Uhr in der Kategorie: Hardware. Kommentare können über den Kommentar (RSS) Feed verfolgt werden. Du kannst zum Ende springen und ein Kommentar abgeben. Pingen ist momentan nicht erlaubt.



#### Über Videospielgeschichten

Videospielgeschichten ist eine offene Plattform für Hobbyautoren und Journalisten. Die Webseite wurde 2009 gegründet, um es jedem Menschen, unabhängig von seiner Profession, zu ermöglichen, persönlich, authentisch und unabhängig über Videospiele zu schreiben

https://www.videospielgeschichten.de