The project is about an application which is a combination of emulation, dissassembler and debugger for the VCS 2600.

I have written cpu- and emu-core completely from scratch. Therefore there are still a lot of bugs and problems to fix at the moment, but things are looking promissing even at this early stage. Restrictions in the current version:

- Support for 2 and 4k roms only (bank-switching will be supported in future versions).
- No sound support yet.
- While some roms are already playable (frogger, dodge em) others dont even show up propper video output (eg asteroids).
- Requires Windows 95,98 or NT with DirectX 3.0 or better with Video-Resoltion of 1024 by 768 or higher.

The emulation-core generates fast video-output in windowed mode by directly manipulating memory of the video-hardware. This feature requests DirectX-support (V3.0 or better).

On an 800MHz machine the emulation may run between 26 and 34 FPS (unoptimized emucore yet, so still a lot of optimization potential)

The roms disassambled binary-data apears here as assambly-code.

The instruction where the programmcounter points to and which will be executed next, ist highlighted.

Breakpoints can simply be toggled on and off, by mouseclick.(number of breakpoints limmited by number of instructions only).

While in trace-mode, the emudebugger dynamicly changes the focused/highlighted line. All importand registers, such as PIA, TIA, 6502 and some emuinternals are shown while emulation is freezed (space-bar) or in break-mode (pause/break-key).

While in single-step-mode, a red exclamationmark indicates registerchanges since the last step.

	Adress Inst	Oper	Byt0	Byt1	Byt2		Adr	Mnomic	Val	Adr	Mnomic	Val	
	Oxf96a sta	\$02	0x85	0x02			0x280	swcha	Oxff	0x00	vsync	0x00	
36	Oxf96c lda	(\$eb),y	Oxbl	Oxeb			0x281	swacnt	0x00	0x01	vblank	0x40	
	Oxf96e sta	\$1b	0x85	Oxlb			0x282	swchb	0x0b	0x02	wsync	0x00	
	🛑 0xf970 lda	(\$ed),y	Oxbl	Oxed			0x283	swbcnt	0x00	0x03	rsync	0x00	
	0xf972 sta	\$1c	0x85	0x1c	1.		0x284	intim	0x49	0x04	nusizO	0x01	
	0xf974 dey		0x88				0x294	timlt	0x00	0x05	nusizl	0x01	
	0xf975 bne	\$£3	0xd0	0xf3			0x295	tim8t	0x00	0x06	colup0	0x22	
	Oxf977 sta	\$02	0x85	0x02			0x296	t1m64t	0100	0x07	colupi	Ux22	
kana kana kana	0xf979 sty	\$1b	0x84	0x1b			0x297	C1024C	000	0x08	colupi	0xd4	
	0xf97b sty	\$1c	0x84	Oxic			Intern	a1	Value	0x09	colube	0x90	
••••	Uxf97d Idy	şea	Uxa4	Uxea			DDV	a.	true	0x0a	refn0	0x03	
	Oxf9/f dey	¢11	0x88	0-11			Been 5	7	106	0x0c	refnl	0x00	
	0x1980 Bm1	711 #¢04	0x30	Ox11			Beam >	c	38	b0x0	nfO	0x30	
	Oxf984 bne	\$0a	0vd0	0x04			D-enak	51	0x00	0x0e	pfl	0x00	
	0xf986 ldv	¥\$∩∩	OxaO	0x00			D-grp0	0	0x7e	0x0f	pf2	0x00	
	Oxf988 sta	\$02	0x85	0x02		-	D-grp1	-	0x7e	0x10	respO	0x00	
es es es	1						X-p0		176	0x11	respl	0x00	
н н н	Register	Value	Adress	10	Byt0	-	X-pl		116	0x12	resmO	0x00	
	? ProgCounter	0xf972	Oxffa	ad ###	####		X-m0		85	0x13	resml	0x00	
	StackPointer	OxOlff	Oxffa	ae ###	*****		X-ml		136	0x14	resbl	0x00	-
	Accumulator	Oxff	Oxffa	af .##	#####.		X-bl		187	0x15	audc0	0x05	
	X-Register	0x00	Oxffb	#	###		Frames	5	0	0x16	audcl	0x0c	
- Λv	Y-Register	0x05	Oxffb	ol	#.#		Tia-cl	lk 33	978866	0x17	audfO	Oxla	
	Negative-Flag	g true	Oxffb	o2#	####		Cpu-cl	lk 6	613826	0x18	audfl	0x91	
	Overflow-Fla	y false	Oxffb	o3 .##	#####.		FPS		0.000	0x19	audv0	0x00	
	Break-Flag	false	Oxffb	04 ###	******					0x1a	audvl	0x00	
	Vecimal-Flag	false	Uxffb	os ###	I IIIII					0x1b	grpU	Uxff Ow7	
, 🔽 Step 🔽 catch Framedone	Interupt-Flag	g crue	Outfb	06 ###	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,					Owle	grpi	0x7e	
OK 🛛 🔽 Irace 🗖 catch Timerevent	Corru-Flag	faise	Over	.0 .##	*####. ****					Owla	enamu	0x00	
	Carry-Flag	crue	Ovffh	9 ###	****					0v1f	endmi enabl	0x00	
Abbrechen <u>L</u> oad			Oxffb	<i>###</i>	****					0x20	hmn0	0x29	
			Oxffb	nh ##						0x21	hmpl	0x26	
			Oxffh	oc ###	1##.##					0x22	hmmO	0x00	
				- """		•							-

Step-Flag: single-step mode, emu breaks automaticly after each emulated cpu-instruction.

Trace-Flag: animates the assembly-code list-control while emulation ist running (of course slows down emulation dramaticaly).

Catch Framedone-Flag: Breaks after completion of an individual frame

Catch Timerevent: Breaks if timer is activated and INTIM reaches 0

Visual output of the internal "rasterobjects-buffer"

For each of the visible objects (p0, p1, m0, m1, bl, pf) the emulation internaly computes an independent raster buffer, depending on such registers as COLUPx, GRPx, NUSIZx etc.

The order of the symbolic-names on the left side indicates the order of the buffer-lines. The complete memoryspace is shown in a special form of binary-mode (dot indicates 0, # indicates 1). This is for easier recignition of graphical information.

A red arrow always indicates the last adress of cpu's memory-access.

Constructive comments are welcome under stefan.burger@uumail.de