Stardos cartridge reverse engineering

by Jens Schönfeld, february 2008

On (todo:Datum), a request came up on Lemon64, someone was looking for the Stardos software. The cart was in Dave's collection, and he agreed to send it to me for reverse-engineering.

At first, the cart did not do anything – the Eprom was not seated properly in the socket. I removed it and put it back in, then the computer started with the Stardos startup-screen. I tried a few commands without ever reading the manual, then quickly decided to start taking the thing apart.

For full reverse-engineering, all chips have to be removed in order to see all printed circuit traces. With the amount of easter eggs and greetings on the board, it was no surprise to find more easter eggs under the the chips – see pictures at the end of this document.

The circuit consists of four TTL chips and one Eprom. Please reference the schematics and pictures at the end of this document for the locations.

U2: 74LS00 (quad NAND gate)
U3: 7407 (six open-collector drivers)
U4: 74LS157 (quad 2-input multiplexers)
U5: 74LS30 (8-input NAND gate)
U6: 27128 (16KByte Eprom)

I'll start describing the circuit at the 74LS30, which is wired up to decode $e000-$ffff for CPU-read accesses (phi2 clock high). The DMA line going into the chip suggests that DMA cycles to the Eprom are not desired, but that doesn't really make sense to me. DMA was only used on the REU (was it available back in 1986??), and you cannot really stop anyone from reading out the Eprom with such a weak protection.

The BA line is used to detect CPU cycles only, so when the VIC is accessing the Kernal area during Badlines, the Eprom does not overlay the computer's memory. There's an unused input routed to an external connector, which is not described in the manual. It could be connected to the Hiram line of the CPU, so the 8K ram under the Kernal would be available to the CPU – this would increase compatibility to a level of an internal Kernal rom replacement. It's not really a non-intrusive method, but a clever method for reducing the amount of soldering work to a minimum. I called the output of the LS30 "KernalSelect", because that's what it really is after the open pin has been wired to Hiram.

The Eprom wiring is fairly straight-forward, only two data lines (bits 1 and 2) are swapped. The binary the I retrieved with an Eprom programmer cannot be disassembled directly, the file needs to be converted with a small tool that swaps data bits 1 and 2. The Eprom is selected either when the KernalSelect line goes low, or when the RomL line goes low with the right register setting. Address line A13 is generated by the LS00:

Gates U2a and U2b are wired as an RS-Flipflop. The inputs of the RS are wired to the $de00 and $df00 select-lines, with a weird analog circuit in between: A 47uF tantalum capacitor is always pulled to +5V through a 3k3 resistor, and a 7407 open-collector driver pulls the positive terminal of the capacitor to GND whenever an address in $dexx (for reset) or $dfxx (for set) is accessed.

The capacitors on the R and S inputs form a fairly extreme low-pass filter. If you only make one access to an IO register, the RS flipflop will most probably not react. It will only do something if you make lots of accesses in a very short time, which explains the loops that do 255 accesses to a random IO address. At actually does not matter which address is accessed, as long as it's the right page.

TheRS flipflop does not have a known state after power-up or after a reset. On a reset, it just keeps it's state. On power-up, I'd expect a short oscillation of the outputs until one of the two capacitors is charged, so the power-up state of the circuit is totally random. This will have to be caught by software.

The 74ls157 is wired as a pass-through multiplexer. Yes, the Stardos cartridge is prepared to have a pass-through port! Four lines for Exrom, Game, RomL and RomH select are reserved on solderpads on the end of the cartridge.
The multiplexer will give control to the pass-through port whenever the Stardos cart does not want to map something into the C64's memory.

Memory map(s)

There are only two memory maps on the Stardos cart: One with only the Kernal Rom replaced, and another with the Kernal Rom replaced and an additional 8K bank mapped in at $8000. The $8000 rom is switched on with lots of accesses to $dexx, and it is switched off with lots of accesses to $dfxx.

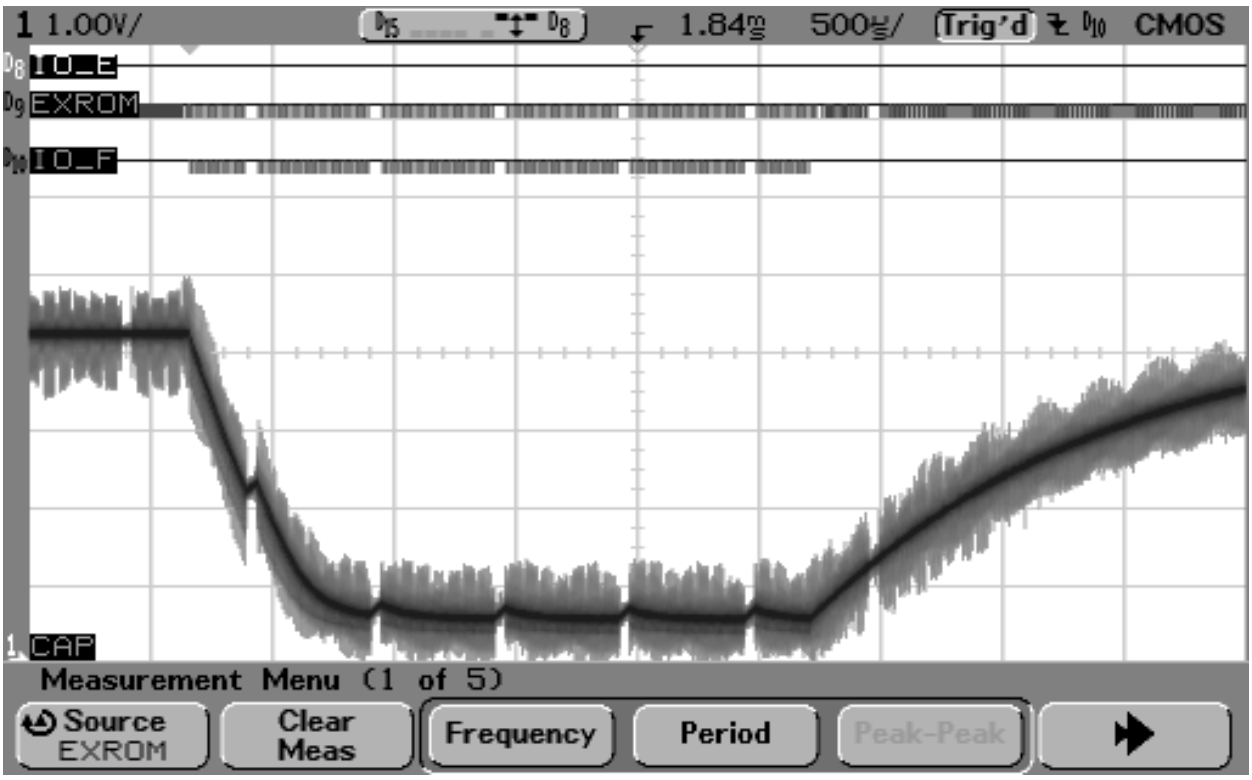
Without the Hiram line connected to the small black connector, memory at $e000 cannot be accessed, so a lot of programs won't work. With Hiram connected, all known banking with processor register $0001 will work as expected.

Circuit flaws

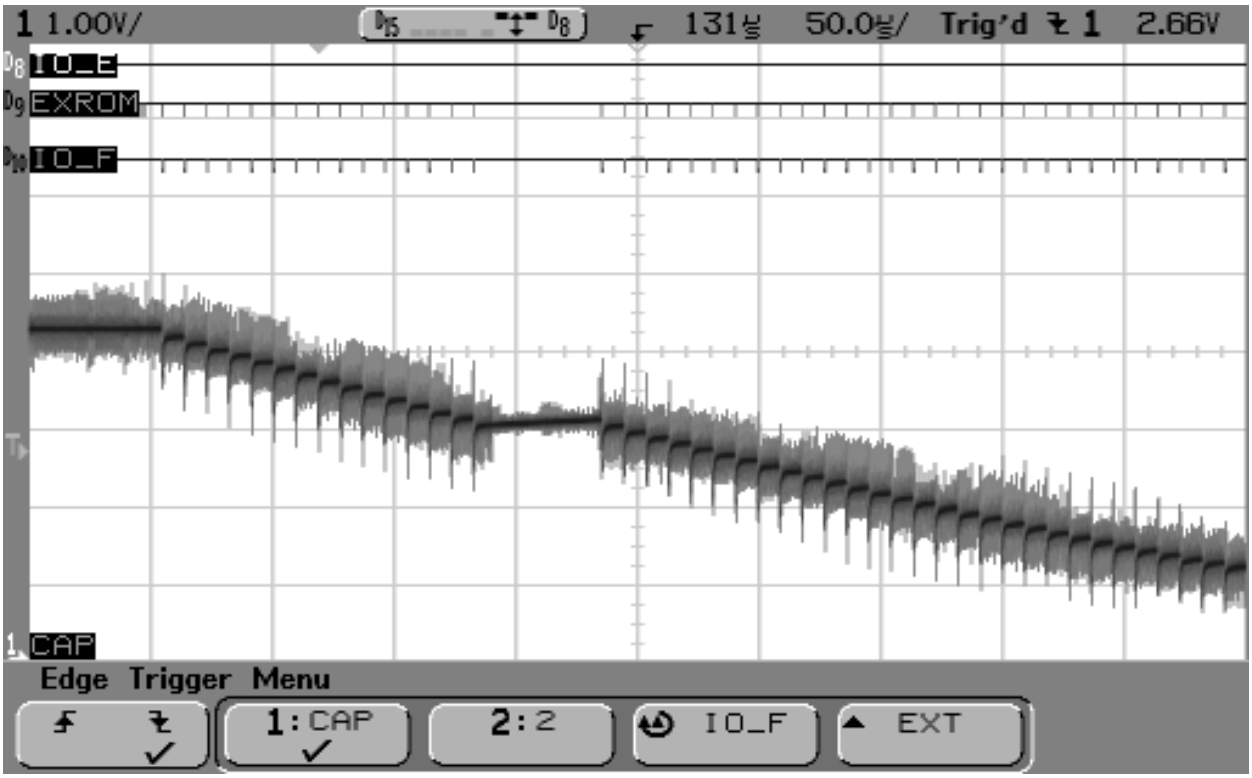
Apart from the fact that the low-pass-filters on the RS inputs of the flipflop are a wild hack, there's only one small flaw in the circuit: The unused Hiram connector does not have a pull-up resistor. This is really just a small flaw, as LS-type TTL chips see an open line as logic-high most of the time, and nobody will seriously operate the cartridge without the line connected.

Diagrams

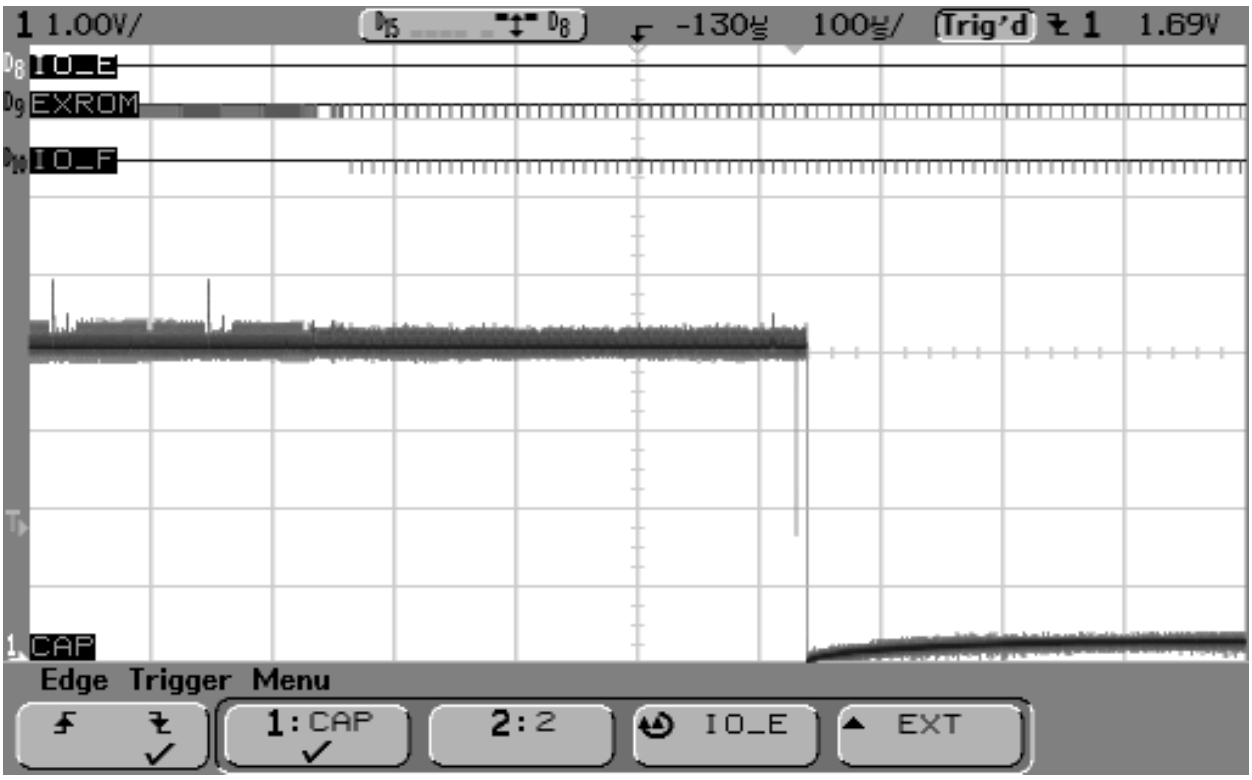
With the wild hack of big capacitors and hundreds of accesses to a register, it is most interesting how the circuit behaves in reality. Here's a few diagrams of the capacitor voltage versus the accesses to a register:



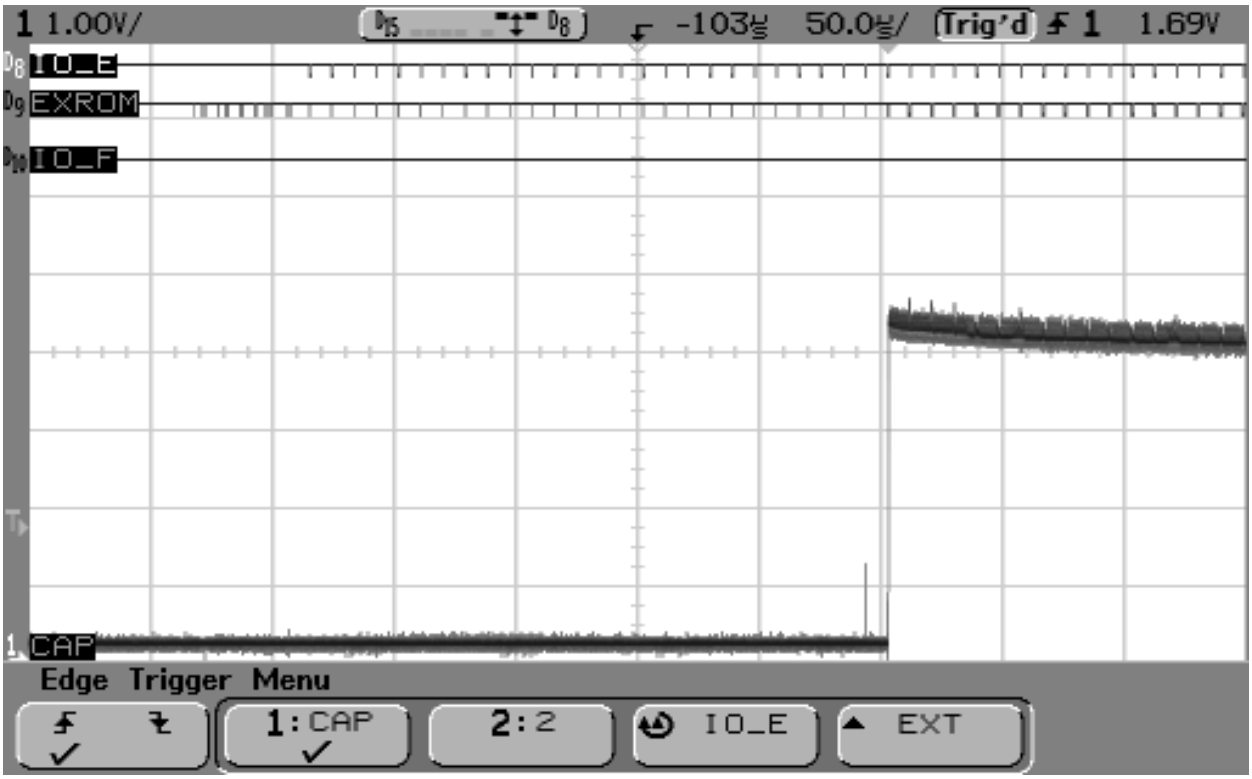
When switching on, the computer turns off the $8000 rom. The capacitor voltage drops while addresses in $dfxx are accessed, and it rises when these registers are left alone. Note that a badline interrupts the loop, as this enlarged diagram shows:



Enlarged view of a badline during the access loop.

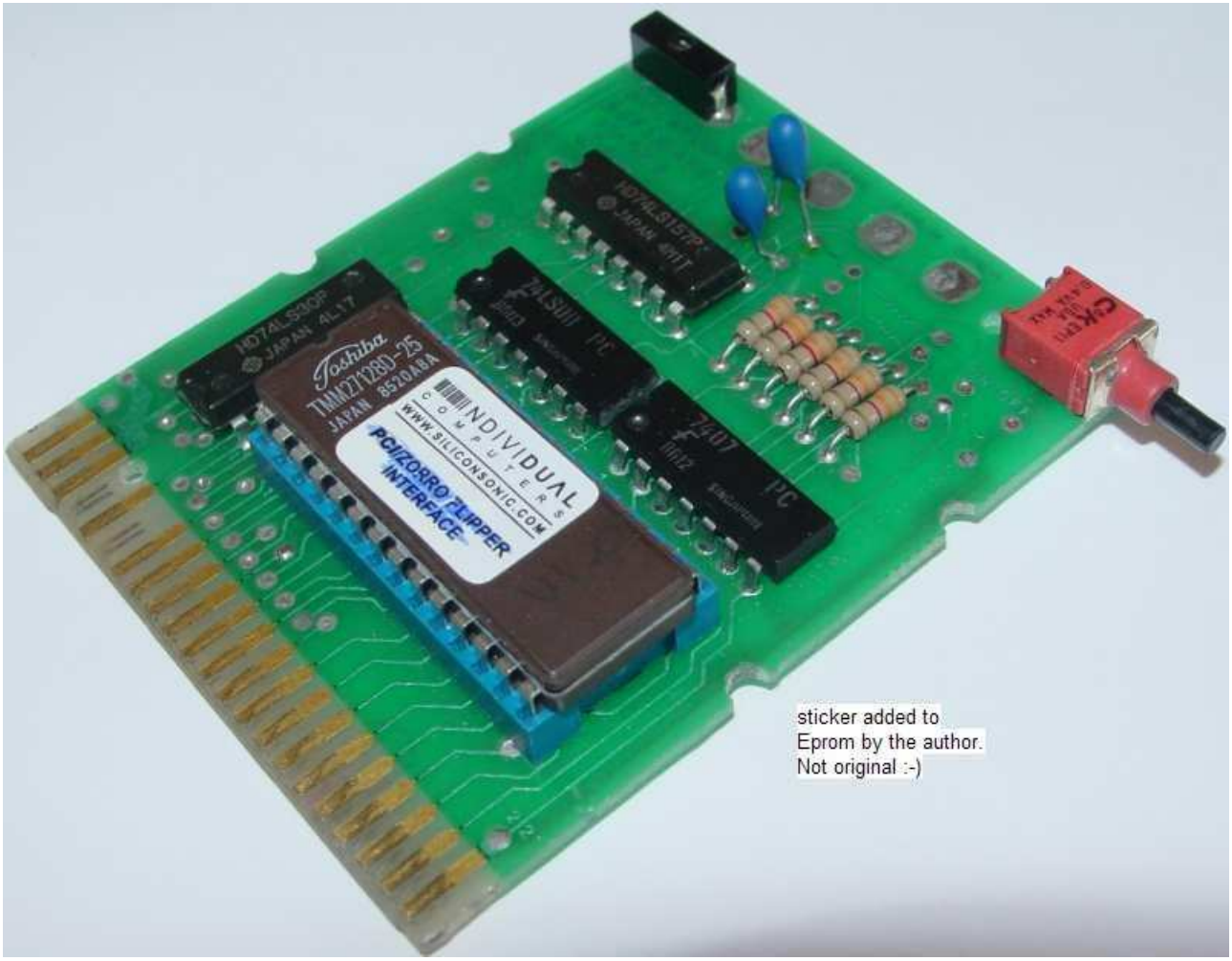


The analog channel in this diagram is on pin 6 (nQ output of the RS flipflop). Funny: Exactly 42 accesses are necessary to make the flipflop switch. Now we have the question to the answer that Deep Thought gave us :-) Note that this is a statistical process. It will take more accesses with a badline, and it'll take more accesses on a new board, as IO accesses are about 50-100ns shorter.



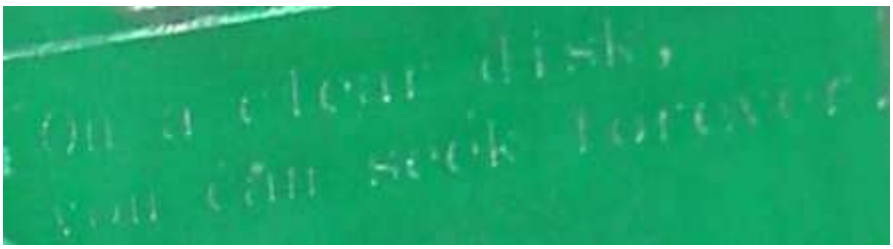
Same setup, other direction: Switching on the $8000 rom takes 27 cycles.

Pictures



Easter eggs:

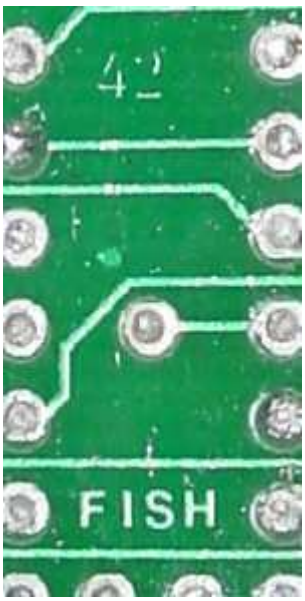
Reference to Douglas Adams' Hitchhikers guide to the galaxy – Bugblatter Bait!



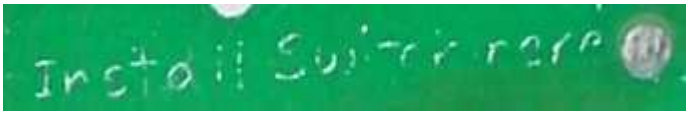
On a clear disk, you can seek forever.

Under the 7407 chip: Smily and the jackpot!

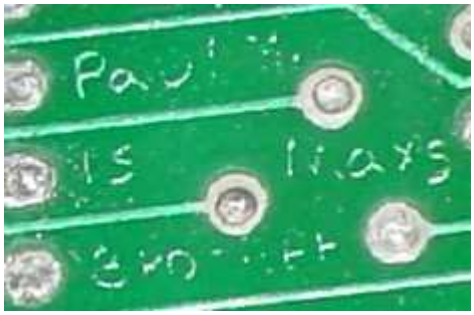
Under the 74ls157: Flush (and maybe something else above – unreadable)



Under the 74ls00: More references not to panic, because 42 is mostly harmless, as the fish in your ear translates.



"install switch here".



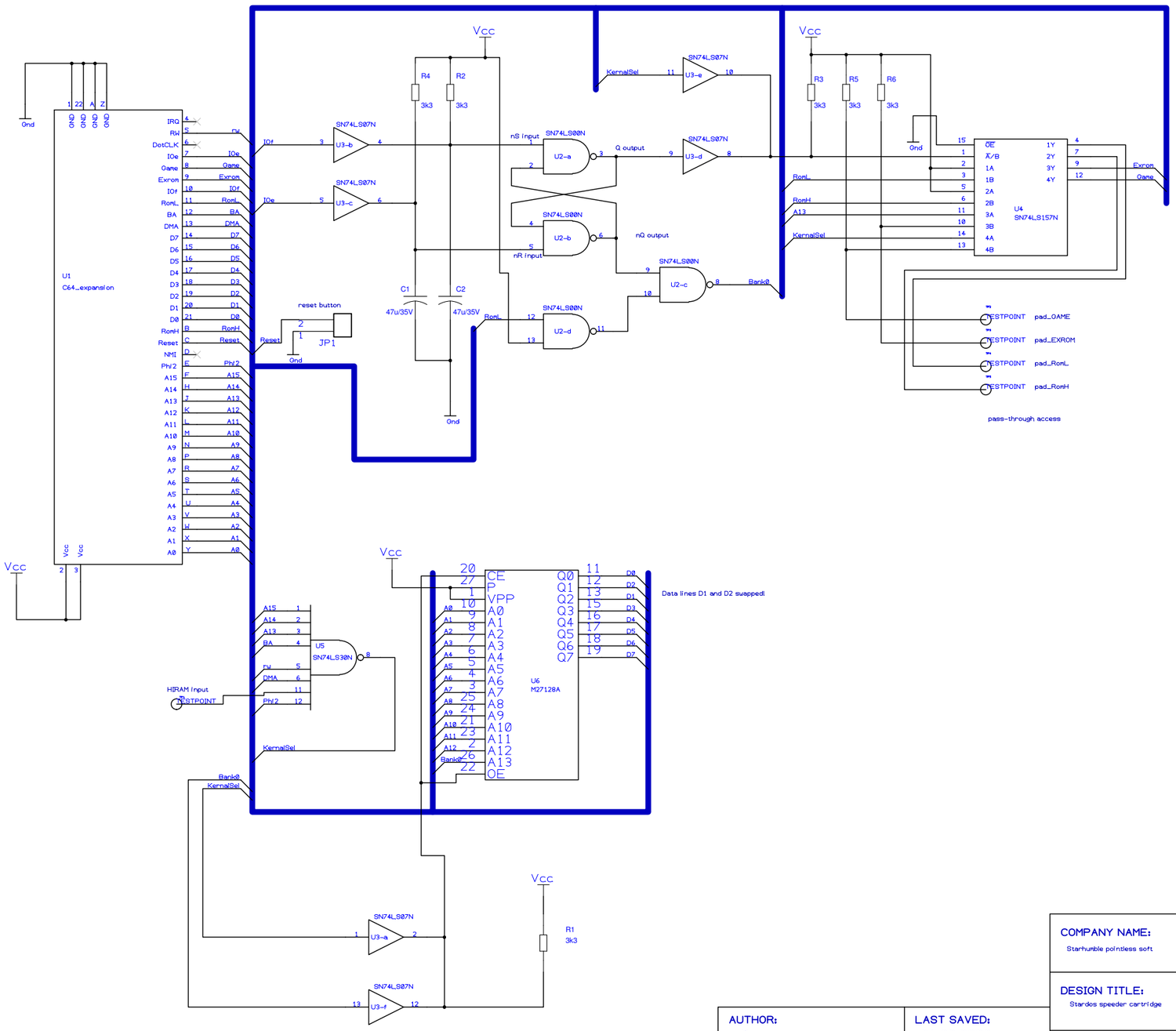
Under the Eprom: Unreadable and readable things...

Thanks to Groepaz for writing a VICE emulation pretty quick. He forced an error in his emulation that I experienced after putting the cartridge back together, so I had a very precise idea where to look for the fault and repair Dave's cartridge. It will now be returned to the owner with a big thank-you from the whole C64 community.

If it's reversed, it's forever.

Jens Schönfeld

Schematics



AUTHOR: Jens	LAST SAVED: 01.03.2008
CHECKED:	DATE:
ISSUED:	DATE:

COMPANY NAME:
Starhumble pointless soft

DESIGN TITLE:
Stardos speeder cartridge

PAGE: Page1	DRAWING NO.	REVISION:
SCALE:	SIZE: A3	SHEET 1 OF 1