Project Documentation

Extern Kernal 8 – REX 9628

Project number: 127

Revision: 0

Date: 20.05.2019

Extern Kernal 8 – REX9628 Rev. 0

Module Description

Purpose & Introduction

This project is the reverse engineering of The Extern Kernal 8 – REX Datentechnik 9628 cartridge. It serves educational purposes only. The use of this information is on own risk. It is advised not to reproduce the Extern Kernal 8 board due to possible copyright infringements.

This board is an extern kernel cartridge for the Commodore C64. It allows to switch between up do 8 Kernals (operating systems) stores in two 27C256 EPROMs.

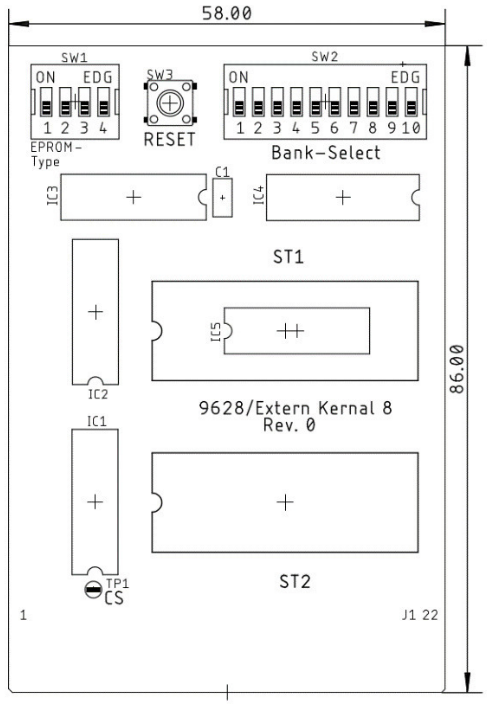


Figure 1: Layout of the Extern Kernal 8 including measures

It is possible to combine different types of EPROMs (27C256, 27C128 and 27C64), which have different memory sizes. This memory size determines the number of kernals, that are stored in such an EPROM.

Extern Kernal 8 is not suitable for storing (game) firmware, which is intended to be used with a generic 8k or 16k cartridge, like games etc.

The kernals are mapped to the C64 address space between 0xE000 and 0xFFFF. One signal, that is used to access the RAM in the same area ($\overline{\text{HIRAM}}$) is not mapped to the expansion port, so software, that is requiring this part of the RAM will conflict with the Kernal cartridge. A possible solution is to connect this internal signal to the solder pad TP1 ("CS").

Configuration

The purpose of DIP-switch **SW1** to set the combination of EPROM types.

1	2	3	4	ST1	ST2
on	off	on	off	2764	2764
on	off	on	off	27128	28128
off	on	off	on	27256	27256
on	off	off	on	2764/27128	27256
off	on	on	off	27256	2764/27128

DIP-switch **SW2** selects the kernal

1	2	3	4	5	6	7	8	EPROM	Addr. Offset
on	off	off	off	off	off	off	off	ST1	0x0000
x	on	off	off	off	off	off	off	ST1	0x2000
x	x	on	off	off	off	off	off	ST1	0x4000
x	x	x	on	off	off	off	off	ST1	0x6000
x	x	x	x	on	off	off	off	ST2	0x0000
x	x	x	x	x	on	off	off	ST2	0x2000
x	x	x	x	x	x	on	off	ST2	0x4000
x	x	x	x	x	x	x	on	ST2	0x6000

x: do not care

A 27C128 can hold up to two kernals, a 27C64 is only capable of one.

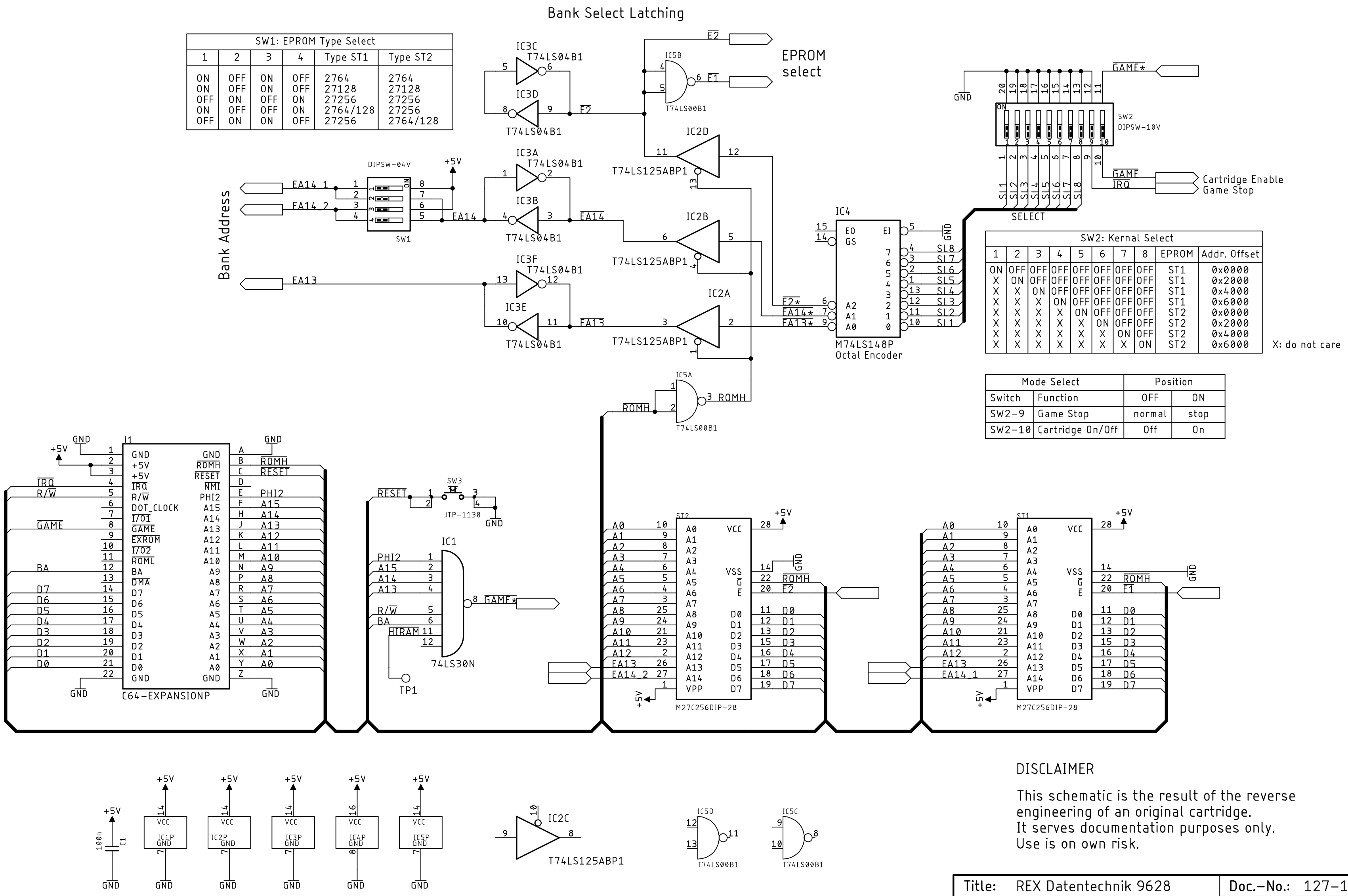
When inserting the EPROMs into the socket, make sure, that the notch of the socket and of the IC are adjusted and that all pins of the IC insert properly into the socket.

Switch SW2-9 provides a game stop function. SW2-10 can activate or deactivate the cartridge.

Mode Select		Position	
Switch	Function	off	on
SW2-9	Game Stop	normal	stop
SW2-10	Cartridge on/off	off	on

It is strictly advised to configure the cartridge while power off. Changing the settings while the C64 is running will usually cause system crashes.

SW3 acts as a reset switch.



SW1: EPROM Type Select					
1	2	3	4	Type ST1	Type ST2
ON	OFF	ON	OFF	2764	2764
ON	OFF	ON	OFF	27128	27128
OFF	ON	OFF	ON	27256	27256
ON	OFF	OFF	ON	2764/128	27256
OFF	ON	ON	OFF	27256	2764/128

SW2: Kernal Select									
1	2	3	4	5	6	7	8	EPROM	Addr. Offset
ON	OFF	OFF	OFF	OFF	OFF	OFF	OFF	ST1	0x0000
X	ON	OFF	OFF	OFF	OFF	OFF	OFF	ST1	0x2000
X	X	ON	OFF	OFF	OFF	OFF	OFF	ST1	0x4000
X	X	X	ON	OFF	OFF	OFF	OFF	ST1	0x6000
X	X	X	X	ON	OFF	OFF	OFF	ST2	0x0000
X	X	X	X	X	ON	OFF	OFF	ST2	0x2000
X	X	X	X	X	X	ON	OFF	ST2	0x4000
X	X	X	X	X	X	X	ON	ST2	0x6000

X: do not care

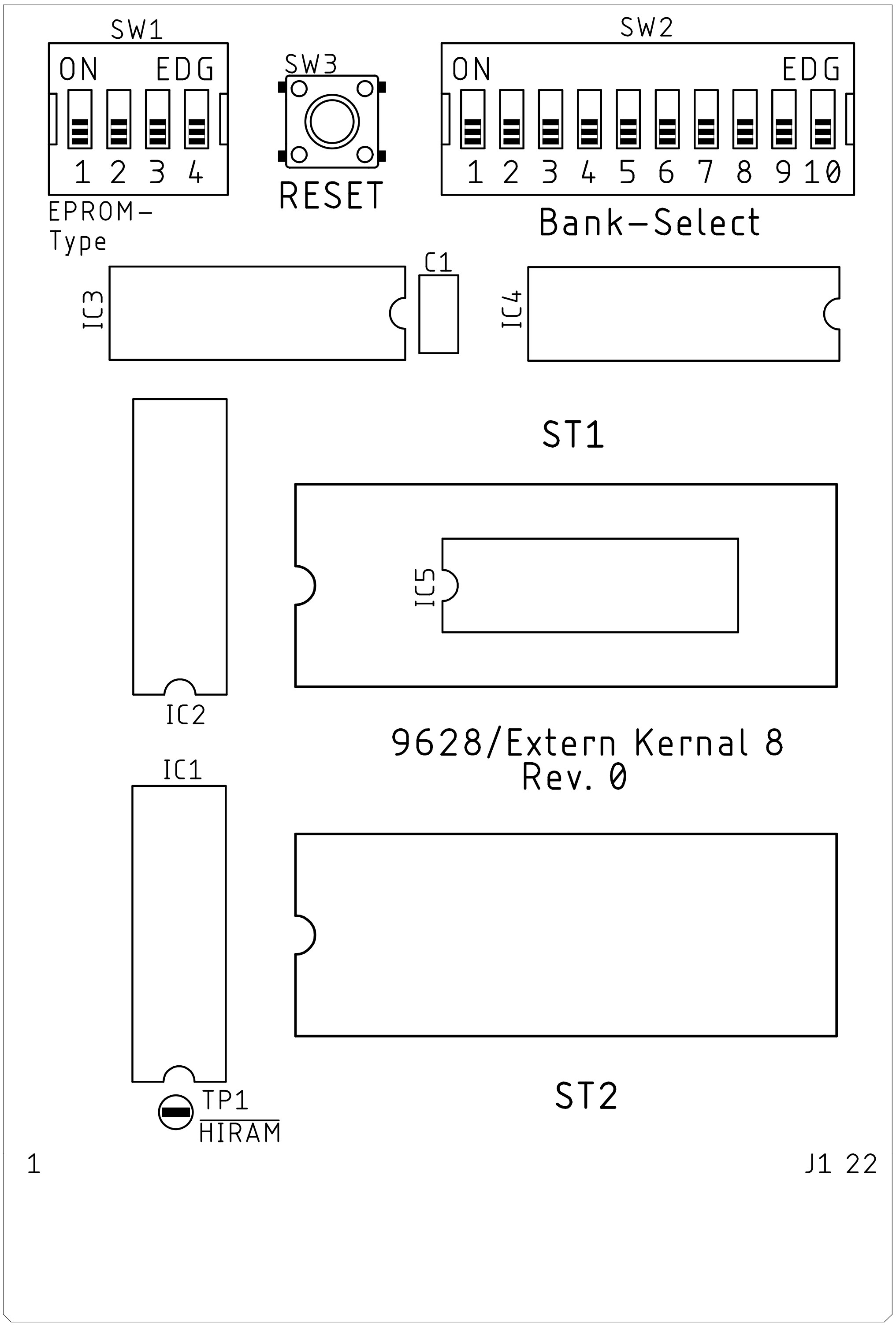
Mode Select		Position	
Switch	Function	OFF	ON
SW2-9	Game Stop	normal	stop
SW2-10	Cartridge On/Off	Off	On

DISCLAIMER

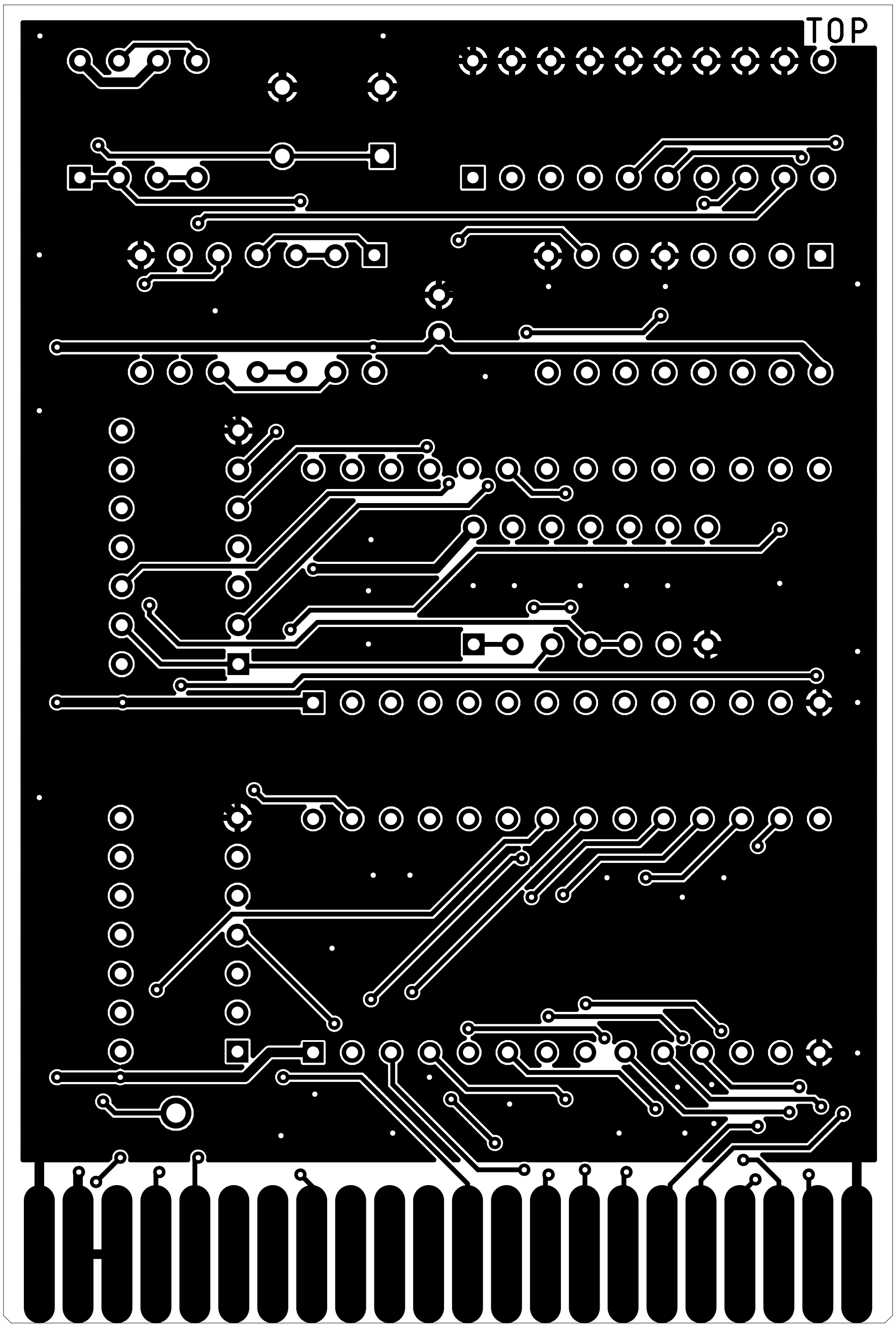
This schematic is the result of the reverse engineering of an original cartridge.
It serves documentation purposes only.
Use is on own risk.

Title:	REX Datentechnik 9628 Extern Kernal 8	Doc.-No.:	127-1-01-00
		Draft.:	Sven Petersen
Date:	21.05.2019 19:38	Rev.: 0	Page 1/1
File:	Extern_Kernal_REX9628		
			A3

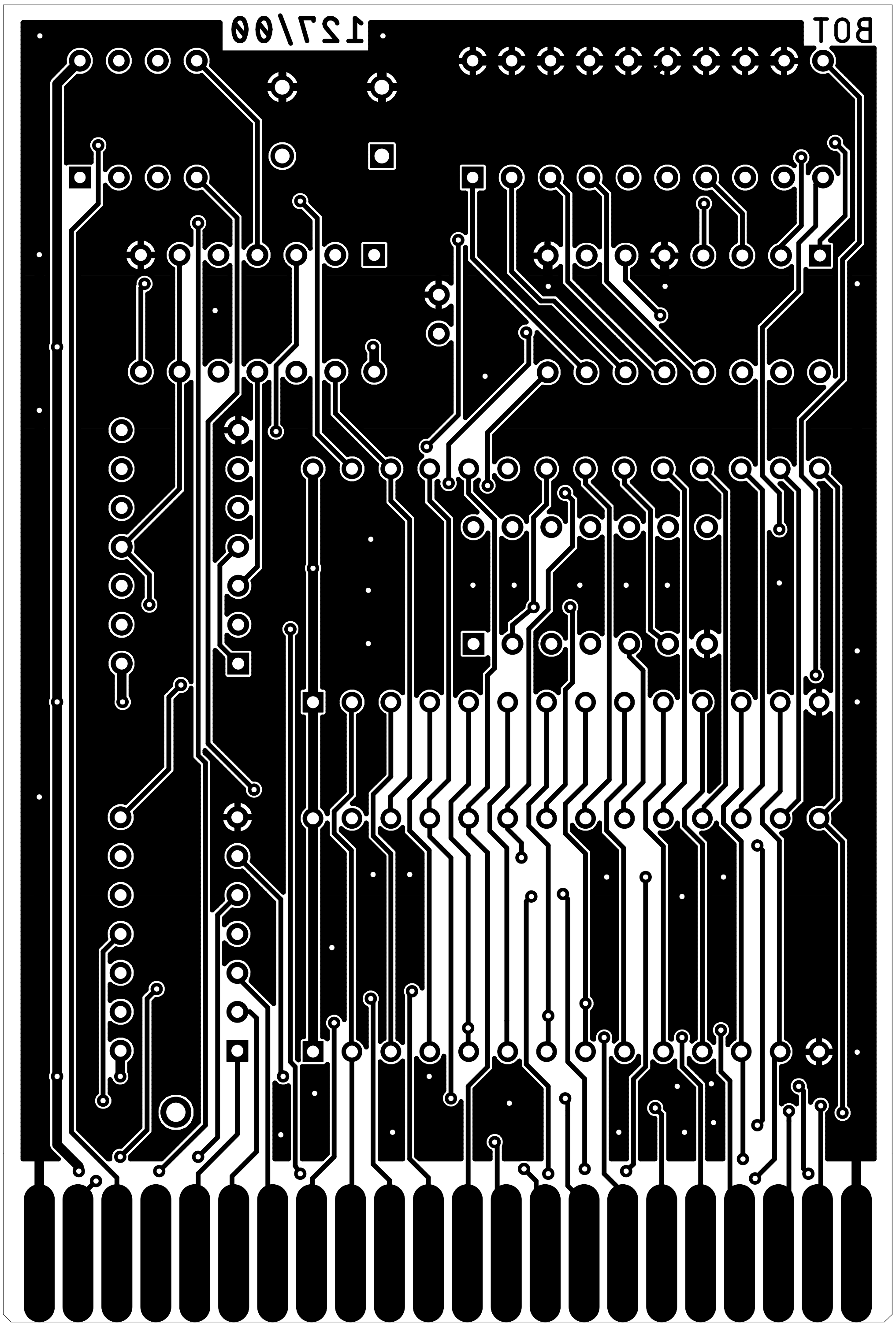
Sven Petersen 2019	Doc.-No.: 127-2-01-00	
	Cu: 35µm	Cu-Layers: 2
Extern_Kernal_REX9628		
21.05.2019 19:32		Rev.: 0
placement component side		



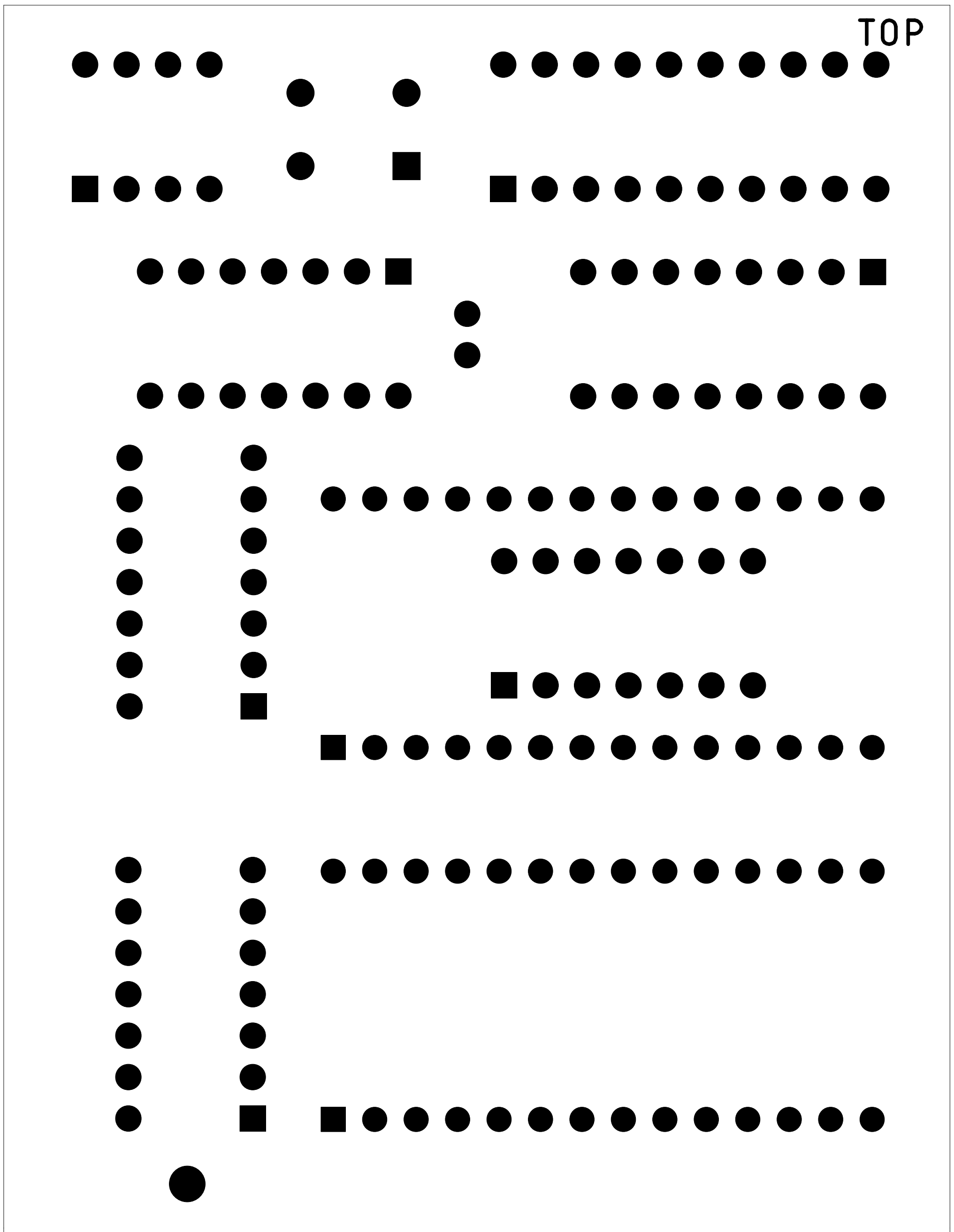
Sven Petersen 2019	Doc.–No.: 127–2–01–00	
	Cu: 35µm	Cu–Layers: 2
Extern_Kernal_REX9628		
21.05.2019 19:32		Rev.: 0
top	⬛	



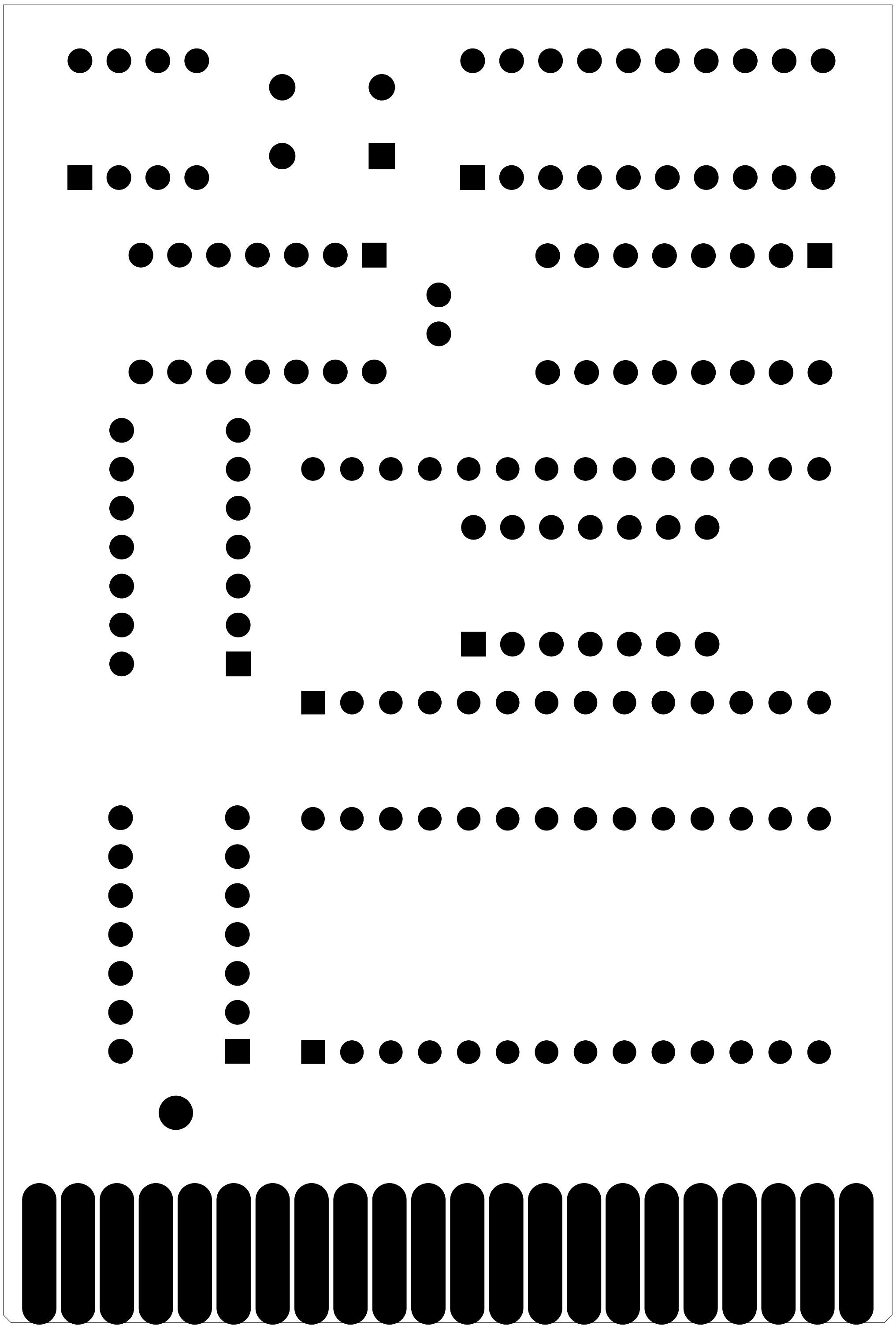
Sven Petersen 2019	Doc.-No.: 127-2-01-00	
	Cu: 35µm	Cu-Layers: 2
Extern_Kernal_REX9628		
21.05.2019 19:32		Rev.: 0
bottom		



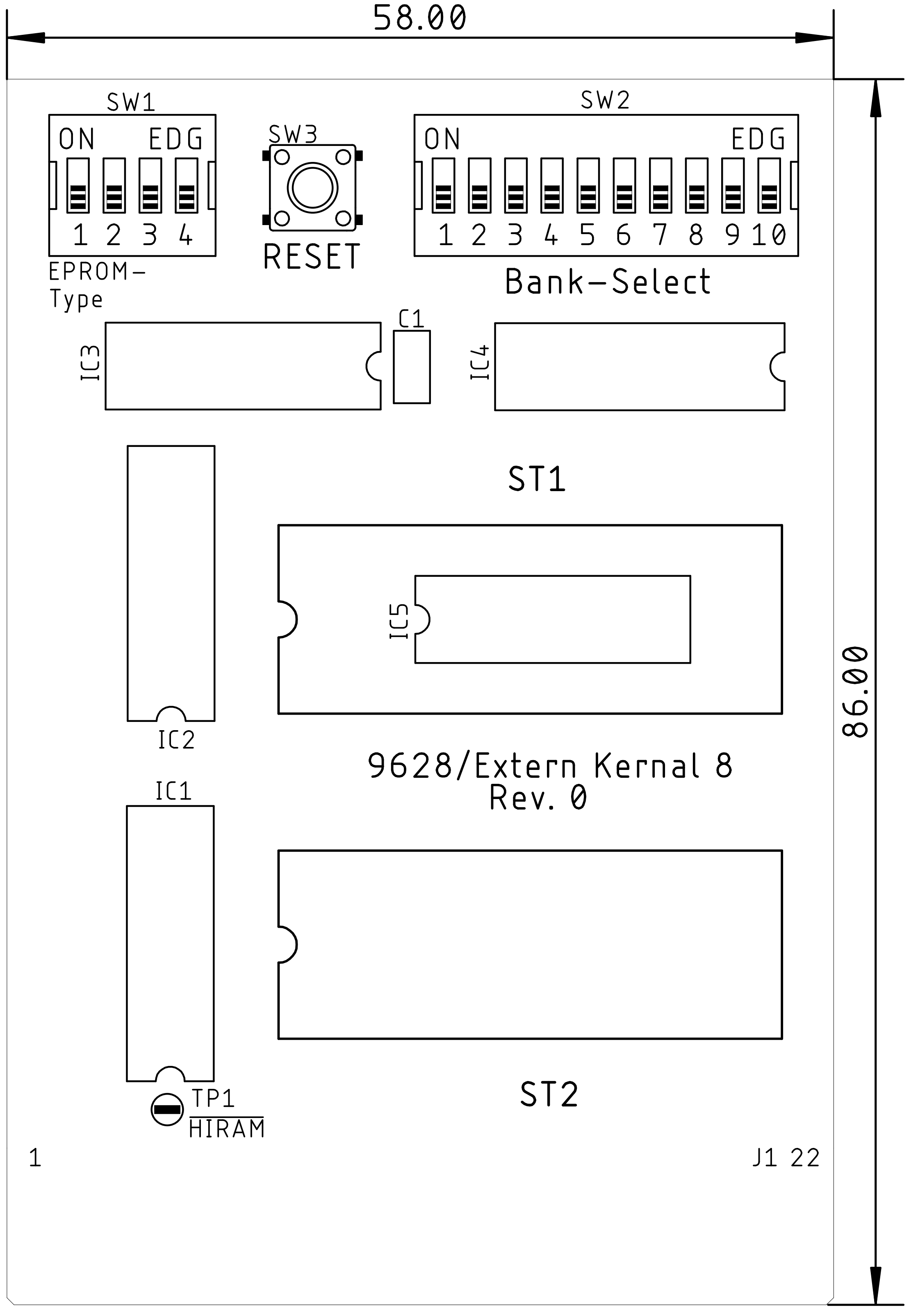
Sven Petersen 2019	Doc.-No.: 127-2-01-00	
	Cu: 35µm	Cu-Layers: 2
Extern_Kernal_REX9628		
21.05.2019 19:32		Rev.: 0
stopmask component side		



Sven Petersen 2019	Doc.-No.: 127-2-01-00	
	Cu: 35µm	Cu-Layers: 2
Extern_Kernal_REX9628		
21.05.2019 19:32		Rev.: 0
stopmask solder side		



Sven Petersen 2019	Doc.-No.: 127-2-01-00	
	Cu: 35µm	Cu-Layers: 2
Extern_Kernal_REX9628		
21.05.2019 19:32		Rev.: 0
placement component side measures		



Extern Kernal 8 – REX9628 Rev. 0

Functional Description

J1 is the edge connector for the C64 expansion port. ST1 and ST2 are the two EPROMs, that hold the kernals.

IC1 is responsible for the cartridge being able to run a kernal. $\overline{\text{GAME*}}$ is the output of the NAND gate IC1. It can switch on or off with SW2-10 to the configuration signal $\overline{\text{GAME}}$ of the expansion port. In case it is switched off, the cartridge is inactive. IC1 produces an extra chip select signal (taking a detour via $\overline{\text{GAME}}$ and $\overline{\text{ROMH}}$). The logic equation for the latter signal is:

```
!ROMH = (_HIRAM  & A15  & !A14  & A13  & !_AEC  & R__W  & !_EXROM  & !_GAME
        # A15  & A14  & A13  & !_AEC  & _EXROM  & !_GAME
        # _AEC  & _EXROM  & !_GAME  & VA13  & VA12 );
```

Equation 1

Obviously $\overline{\text{GAME}}$ is required for $\overline{\text{ROMH}}$ to get LOW (=active). A HIGH level on $\overline{\text{GAME}}$ (or an open SW2-10) will inhibit the EPROM access.

There is a difference between this and the internal kernel chip select: The internal kernel ROM can be switched off by software to get read access to the RAM in the same address space. This is done with the $\overline{\text{HIRAM}}$ signal. The latter signal is not contained in the 2nd line of equation 1, which is responsible for $\overline{\text{ROMH}}$ to be mapped to 0xE000 – 0xFFFF.

The logic equation for the internal chip select is:

```
!KERNAL = (_HIRAM  & A15  & A14  & A13  & !_AEC  & R__W  & _GAME
          # _HIRAM  & A15  & A14  & A13  & !_AEC  & R__W  & !_EXROM  & !_GAME );
```

Equation 2

A possible solution which fixes the problem is connecting the $\overline{\text{HIRAM}}$ signal (CPU 6510, pin 28) to the NAND-Gate via the solder pad "TP1". It is wired to an input pin of IC1.

The logic equation for the signal $\overline{\text{GAME*}}$ is:

```
!GAME* = ( PHI2 & A15 & A14 & A13 & R__W & BA & CS )
```

Equation 3

PHI is the system clock Φ_2, BA is the Bus Available signal, generated by the VIC-II chip.

SW2-9 can hold the $\overline{\text{IRQ}}$ signal low, when switched on. This was a game stop function is provided.

The first 8 switched of SW1 select the kernal. IC4 decodes these 8 switched into an octal number output on A0..A2. The bit A2 ($\overline{\text{E2*}}$)selects which EPROM is active. $\overline{\text{EA13*}}$ and $\overline{\text{EA14*}}$ will be converted into the address bits A13 and A14 of the EPROMs. This information is stored in a latch mechanism consisting of the bus driver IC2 and the Inverter IC3.

IC2 will let the three bits pass as long as the signal ROMH is low. IC5A inverts the bus chip-select $\overline{\text{ROMH}}$ for this purpose. Hence the information is passed through as long as $\overline{\text{ROMH}}$, which also serves as a chip select for the EPROMs is inactive (HIGH). The paired inverters are functioning as a storage element. As long as the bus drives are HIZ (ROMH = HIGH), it holds the logic level, that was asserted by the bus driver IC2 before.

The inverter IC5B generates the enable signal $\overline{\text{E1}}$ for EPROM ST1 from $\overline{\text{E2}}$ which is the chip enable signal of ST2.

The Address bit EA13 is connected to both EPROMs, while EA14 is switched separately by SW1 for each EPROM in order to configure the EPROM size/type for ST1 and ST2. In case the address bit is not required (for 27c128 and 27c64), it can be set to HIGH with SW1.

SW3 is a reset switch. When pressed, a system reset is performed.

The circuitry is reverse engineered from an original REX Datentechnik 9628 cartridge. It is provided "as is". It has some weak points: There are several open inputs, which act as HIGH in the LS logic family, but might be troublesome with other logic families. A good practice is, to connect them to a fix logic level. For IC1, that would be connecting pin 11 and pin12 and holding them high with a (10k) Pull-up resistor. SL1..SL8 on IC4 should also have a pull-up resistor (network).

The not used gates (IC2C, IC5D and IC5C) should also be connected to HIGH (+5V) on each input.

Further on, there is only one blocking capacitor for the whole cartridge. This might work, a better practice would be a blocking capacitor (100n) for each IC.

Extern Kenal 8 - REX 9628 Rev. 0
Bill of Material Rev. 0.0

Pos.	Qty	Value	Footprint	Ref.-No.	Comment
1	1	127-2-01-00	2 Layer	PCB Rev. 0	2 layer, Cu 35μ, HASL, 58.0mm x 86.0mm, 1.6mm FR4
2	1	100n	C-2,5	C1	ceramic capacitor, 2.5mm pitch
3	1	74LS30	DIL-14	IC1	LS required, NAND gate, 8 inputs, e.g. Reichelt LS 30
4	1	DIPSW-04V	DIPSW-04V	SW1	DIP-switch, 4 switches. E.g. Reichelt NT 04
5	1	DIPSW-10V	DIPSW-10V	SW2	DIP-switch, 10 switches. E.g. Reichet NT 10
6	1	JTP-1130	JTP-1130	SW3	6x6mm tact switch, e.g. Reichelt TASTER 9303
7	2	27C256	DIL28-6	ST1, ST2	27C128 and 27C64 are possible, too
8	2	28p. socket	DIL28-6	(ST1), (ST2)	precision round pin socket recommended, e.g. Reichelt GS 28P, modify for ST1, since IC5 is below ST1
9	1	74LS148	DIL-16	IC4	LS required, octal encoder, e.g. Reichelt LS 148
10	1	74LS00	DIL-14	IC5	LS required, quad nan gates, e.g. Reichelt LS 00
11	1	74LS04	DIL-14	IC3	LS required, hex inverters, e.g. Reichelt LS 04
12	1	74LS125	DIL-14	IC2	LS required, quad driver, e.g. Reichelt LS 125
13	1	16p. socket	DIL-16	(IC4)	optional, e.g. Reichelt GS 16P
14	3	14p. socket	DIL-14	(IC1), (IC2), (IC3)	optional, e.g. Reichelt GS 14P

Extern Kernal 8 – REX 9628 Rev. 0

Reverse Engineering

Introduction

An existing over 30 years old REX Datentechnik 9628 Extern Kernal 8 board (Date codes of the main parts: 1988) was reverse engineered to learn about its function and the way a kernal cartridge is different from a generic EPROM cartridge for the Commodore C64. The described reverse engineering method is mainly based on optical/photographic procedures. A decent knowledge of the GIMP, Photoshop or any other image processing software, that offers layers is required.

Reverse Engineering Method

Step 1: Test the Function

First it has to be proved, that the circuit is working. This is not always necessary, but quite helpful to know that all parts on the board are still working. You might need to measure some parts in order to identify them. Also, it is good to know a bit about the function and gain as much information as possible before the final schematics is drawn.

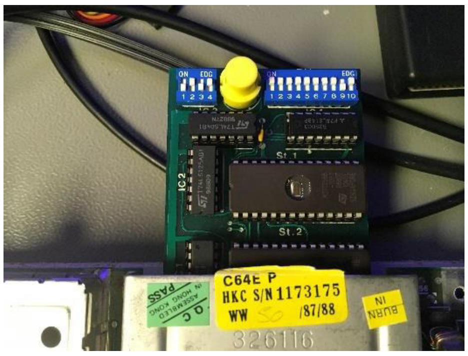


Figure 1: REX9628 in the expansion port of the C64

Step 2: Take a Good Picture

One or more high resolution pictures is required to document the position and reference number of parts on the PCB.

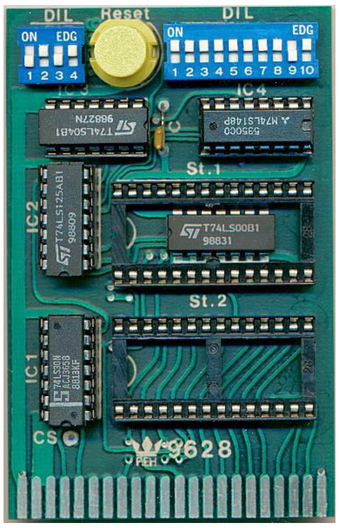


Figure 2: A scan of the assembled REX9628 board

A scanner can do a great job, here. In case the parts on the PCB are not too high, that scanned picture looks very decent and sharp enough. All pictures should be checked to be sharp and all parts are visible before the next step.

Step 3: Catalog the Parts and Unsolder

In this step, the parts have to be identified and their values have to be documented. In this case, it is fairly easy, because most parts already have reference numbers (like IC1, IC2 etc.) which are printed on the PCB and all values are printed on them clearly. The DIP-Switches are standard, the RESET switch, too. The capacitor value is most probably 100n and does not really matter.

In other designs you might have to define the reference numbers/designators and write them over or beside the part on that picture from Step 2, then document the values referencing to their designators. The GIMP is a software tool that helps with this job. Some parts have to be measured after unsoldering.

Identifying the part values or manufacturer part numbers is a critical step. In case a part cannot be identified or replaced with a suitable other part, it might make a reverse engineering impossible. The identification of connectors and small SMD parts can be very challenging. A catalog of SMD codes can be very helpful. Search for “SMD code book”.

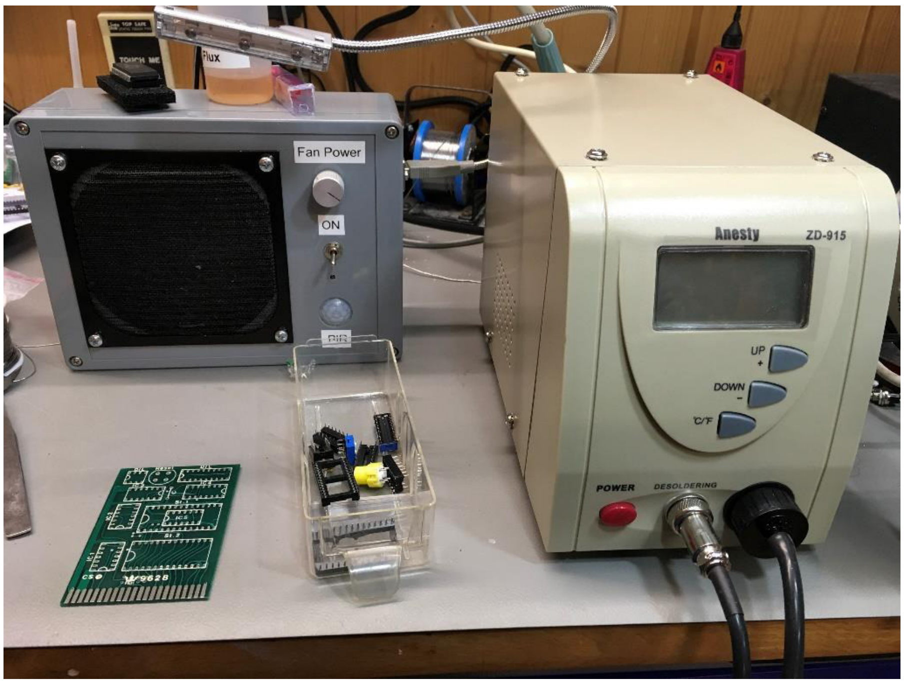


Figure 3: Unsoldered REX9628

The parts have to be kept well in a way, that they cannot be confused. Many little (anti-static) zip-lock bags and/or jars, paper labels and a pencil can be very helpful. The simplicity of the described project has allowed to drop the parts into a plastic box and keep the ICs on an anti-static foam mat.

Step 4: Scanning the PCB and Image Processing

This reverse engineering method is based on image processing. First, both sides of the PCB are scanned in a decent resolution (e.g. 300dpi). Later the scans are assembled with an image processing software like the GIMP. The goal is to resemble a PCB CAD view of the board with both layers in different colors viewed from above the component side.

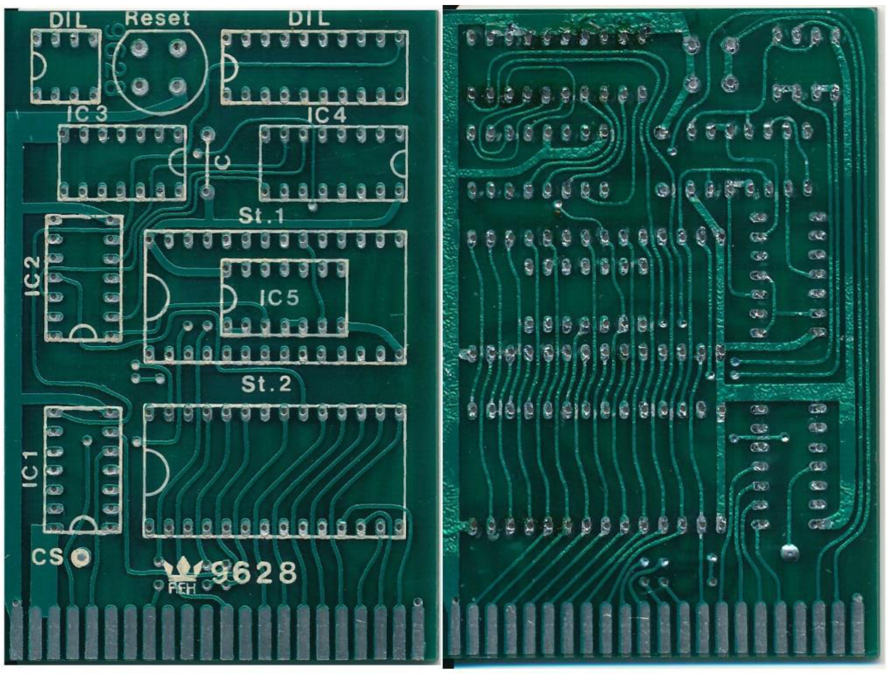


Figure 4: Both sides of the PCB

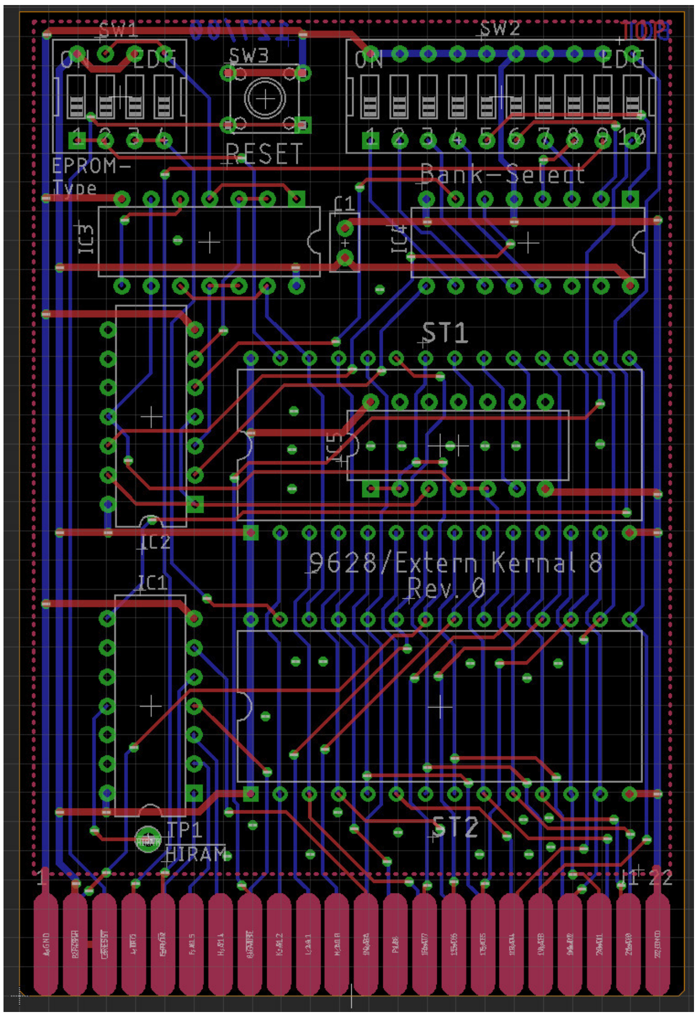


Figure 5: CAD view of (a different) PCB layout

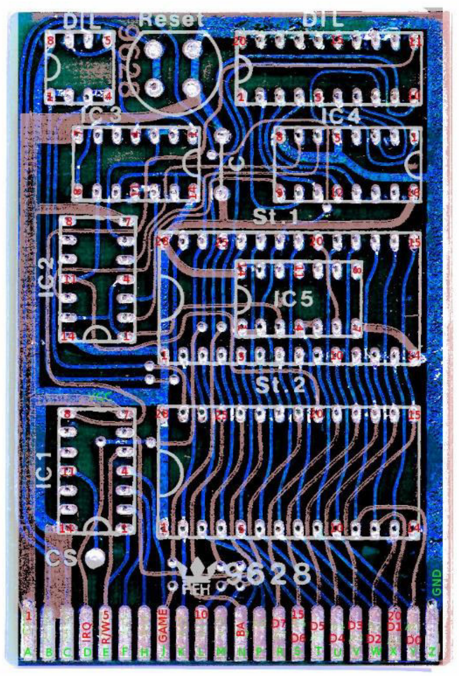


Figure 6: Photographic overlay of the copper layers (bottom layer is blue, top layer is red)

The process with the GIMP:

- Create a new image, a bit bigger in size as the scans of the PCB to allow alignment
- Set the background color of the image to black.
- Insert the scan of the solder side right above the background layer
- Mirror this scan
- Replace the color of the non-copper areas with “transparent”. This might not work perfectly. It might even be required to erase some still visible non-copper areas in case they are disturbing.
- Dye the remaining copper traces blue (or your preferred CAD tool color setting)
- Insert the scan of the component side above the bottom layer
- Replace the color of the non-copper areas with “transparent”
- Set the opacity of the top layer to about 75%, so the bottom layer becomes visible
- Align both layers
- Add a “silk screen” layer, if required. The REX9628 already had such a position print
- Add one or two layers with pin numbers to help counting pins later. The numbering of the edge connector on the solder side is a bit odd, so the numbering of those pads was required.
- Add as many signal names as known

You need to store the file for later use and also a print to follow the copper traces with a pencil.

Step 5: Datasheet and CAD Library Work, Placement

Getting the required data sheets and creating the part libraries in the CAD program has to be done before starting to draw the first schematic.

Since usually the parts that belong to the same functional group in the schematics are close together on the PCB, too, it is advised to place all components resembling their relative placements on the PCB. After the copper traces have been “transformed” into signals on the schematic, this schematic will usually be rearranged.

Step 6: Traces to Signals

Paper and a pen/pencil are the main tools of this step. The traces are followed with the pencil on the print obtained from Step 4. The pin numbers assigned before are now very helpful. The pencil marks the traces which are already followed. Concentrated work is required and multiple checking, since a mistake will cause a fail of the reverse engineering or at least lead to a time-consuming debugging. In doubt, the image processing software (GIMP) will provide a better view and finally, it is required to check connections with a beeper.

Making use of signal names/labels is more useful than drawing a signal cluster/ravel, that rather resembles the PCB view from before. The signals should be named reasonably, the naming might change later. This is the first step to gain understanding of the circuitry.

The result of this step is shown in Figure 7: First pass of the schematicsFigure 7.

Step 7: Trying to understand the circuitry and rearranging the schematic

The next step requires electronic expertise. The circuit has to be reviewed and understood. Any oddities have to be carefully investigated on the physical PCB.

In case of the REX9628, it was the $\overline{\text{IRQ}}$ being switched to ground with the DIP-switch that seemed to be strange. It turned out to be the "Game Stop" mechanism.

After the functions of the components and circuitry was understood (the 74LS148 required a datasheet with a logic table), the schematic was rearranged rather reflecting the function than the placement on the PCB. Some Signal names were improved. See Figure 8.

Finally, the gained knowledge was documented. The board layout (refer to Figure 5) was lastly created.

Required Tools

- Digital Camera
- Scanner
- Image Processing software like GIMP or Photoshop (elements)
- Soldering iron
- Solder sucker/De-soldering station (see Figure 3)
- E-CAD program (like Eagle, KiCAD etc.)
- Multimeter/LCR-Meter
- Printer
- Pencil
- Good concentration
- A sense for electronics and some experience

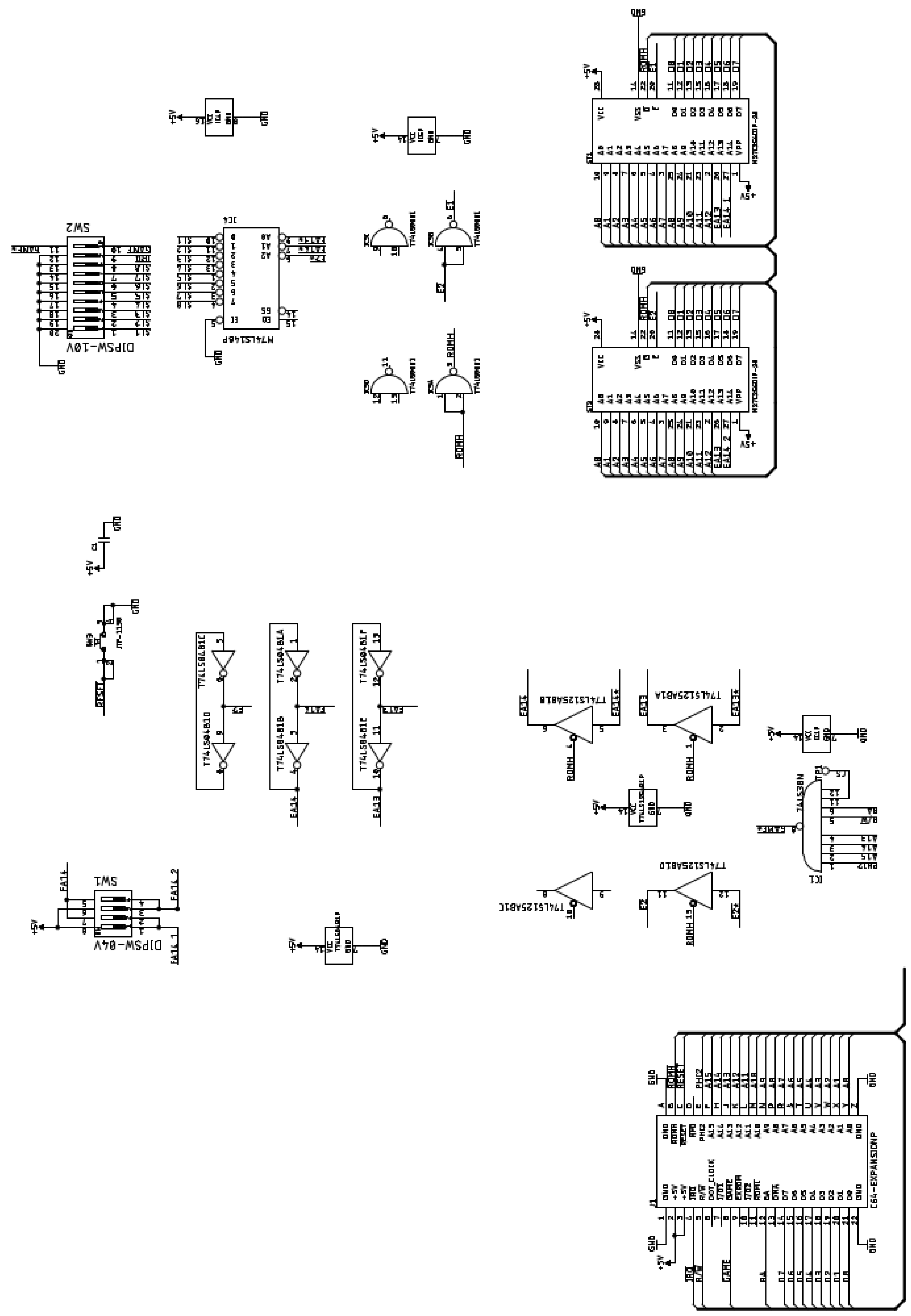
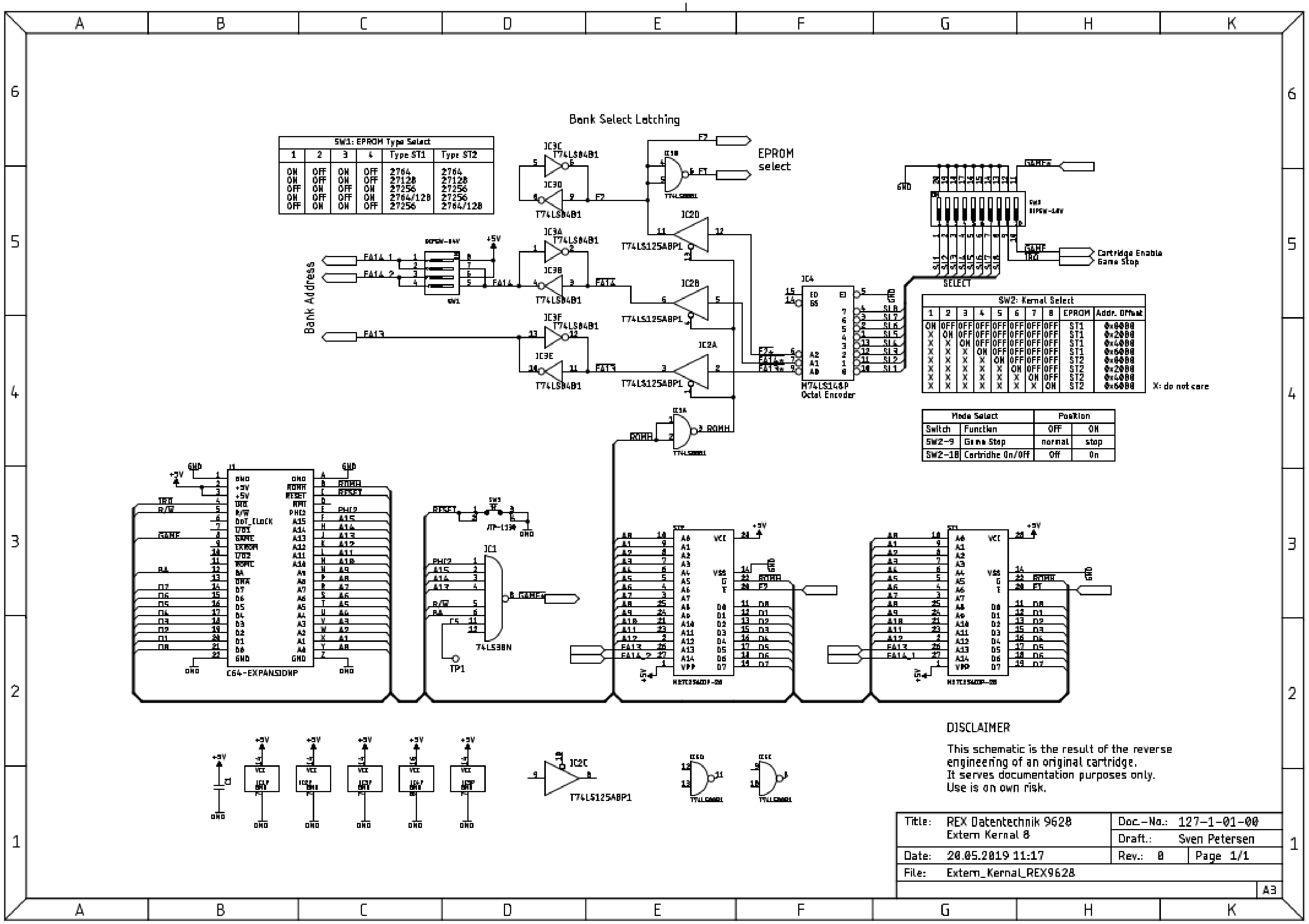


Figure 7: First pass of the schematics (Result of Step 6)



SW1: EPROM Type Select					
1	2	3	4	Type ST1	Type ST2
ON	OFF	ON	OFF	2764	2764
ON	OFF	ON	OFF	27128	27128
OFF	ON	OFF	ON	27256	27256
ON	OFF	OFF	ON	2764/128	27256
OFF	ON	ON	OFF	27256	2764/128

SW2: Kernal Select									
1	2	3	4	5	6	7	8	EPROM	Addr. Offset
ON	OFF	OFF	OFF	OFF	OFF	OFF	OFF	ST1	0x0000
X	ON	OFF	OFF	OFF	OFF	OFF	OFF	ST1	0x2000
X	X	ON	OFF	OFF	OFF	OFF	OFF	ST1	0x4000
X	X	X	ON	OFF	OFF	OFF	OFF	ST1	0x6000
X	X	X	X	ON	OFF	OFF	OFF	ST2	0x0000
X	X	X	X	X	ON	OFF	OFF	ST2	0x2000
X	X	X	X	X	X	ON	OFF	ST2	0x4000
X	X	X	X	X	X	X	ON	ST2	0x6000

X: do not care

Mode Select		Position	
Switch	Function	OFF	ON
SW2-9	Game Stop	normal	stop
SW2-10	Cartridhe On/Off	Off	On

DISCLAIMER

This schematic is the result of the reverse engineering of an original cartridge.
It serves documentation purposes only.
Use is on own risk.

Title:	REX Datentechnik 9628 Extern Kernal 8	Doc.-No.:	127-1-01-00
		Draft.:	Sven Petersen
Date:	20.05.2019 11:17	Rev.: 0	Page 1/1
File:	Extern_Kernal_REX9628		
			A3

Figure 8: Final schematic of the REX9628

Motivations for Reverse Engineering

The author is working as a hardware developer for almost 25 years, mostly in engineering services and has taken part in a number of reverse engineering projects. Those were done out of different motivations.

A copyright infringement must not be the intention behind this kind of work. In most cases, a new hardware development is easier to accomplish than copying a PCB and it is usually cheaper. In many cases, a reverse engineering is not possible or require methods that cannot be conducted by an average engineering service, such as grinding a multi-layer PCB to scan all copper layers or intrusive methods to regain the programming information of a code protected micro-controller. A scenario of "we cannot buy the processor anymore, but we want to keep the code" is usually a show-stopper, because an emulation of this processor (or other vital parts) is an expensive undertaking.

In most cases, customers depend on their development service, but did not request the full documentation and source files. Over the years, it gets more and more difficult to produce the board due to obsolescence of parts, the original developer is not available anymore and when it starts to be threatening for a company, the CEO starts looking for an alternative, having little information consisting mostly of oral traditions. Usually a PCB design is "a project" for the development service and "the project" for the customer. And those are very often not able to determine, if the "files" they have received (if the have received them) are most recent and complete. Usually, they are not.

Also, in aviation or railway applications, the vehicles have a much longer life cycle than the electronic components built into them and it is a challenging task to keep those vehicles running and to get hold of the obsolete parts. In depth information about the electronics are not available in some cases. The electronic modules usually have to undergo costly certification processed, so there is enough funds to pay for a reverse engineering to get something, that is not too different from the original to shorten the recertification. In several cases, it is intended to produce some service manual, which requires a schematics.

To automatize the reverse engineering, a group of companies gathered, discussing the possibility of a CT scan (computed tomography) of a PCB, automatic optical inspection and measurement with a flying probe tester. Finally creating schematics, PCB production data and documentation. As of the knowledge of the author, this project was not led to a success.

The component identification is a very difficult part of the reverse engineering process. Absolute maximum values can be assumed, but not measured. Tolerances cannot be measures as well. Some steps require more expertise than developing a circuit. And finally, a specification is required to really verify the result. A specification of a module/PCB in a complex system is a moving target throughout a development process. The initial specification defines requirements, the final module has properties and the corresponding modules might have to be adjusted to those in order to get a running system.