# MOUSE

## Mario’s Own Unique Singleboard Engine

## … ein Microcomputer Hardwareprojekt

## VCFB 2015

Mario Keller

- Warum baut man einen Computer?

MOUSE

- Warum baut man einen Computer?
- Warum hat das 30 Jahre gedauert?

MOUSE

- Warum baut man einen Computer?
- Warum hat das 30 Jahre gedauert?
- Was ist ein Computer für mich?

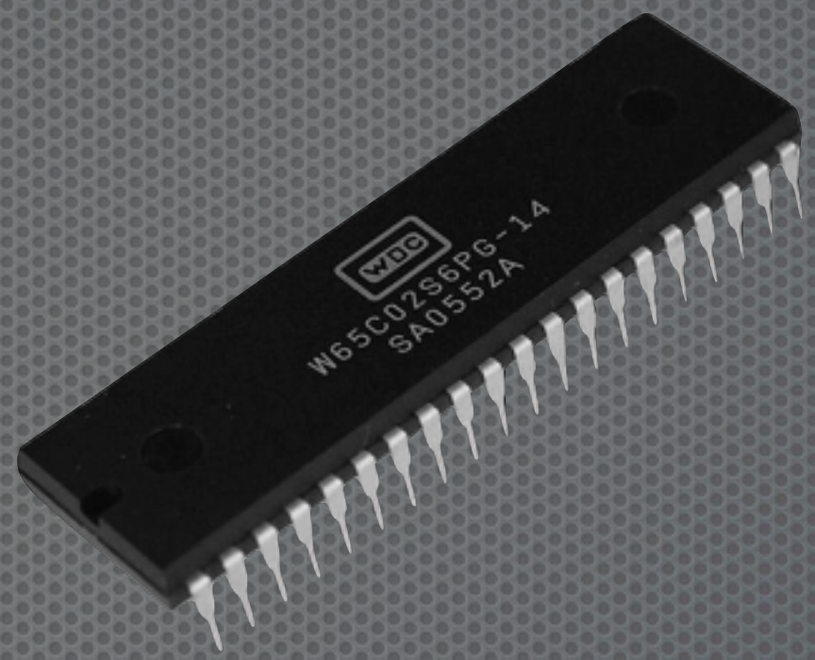
MOUSE

- Warum baut man einen Computer?
- Warum hat das 30 Jahre gedauert?
- Was ist ein Computer für mich?
- Wie fängt man das Ganze an?

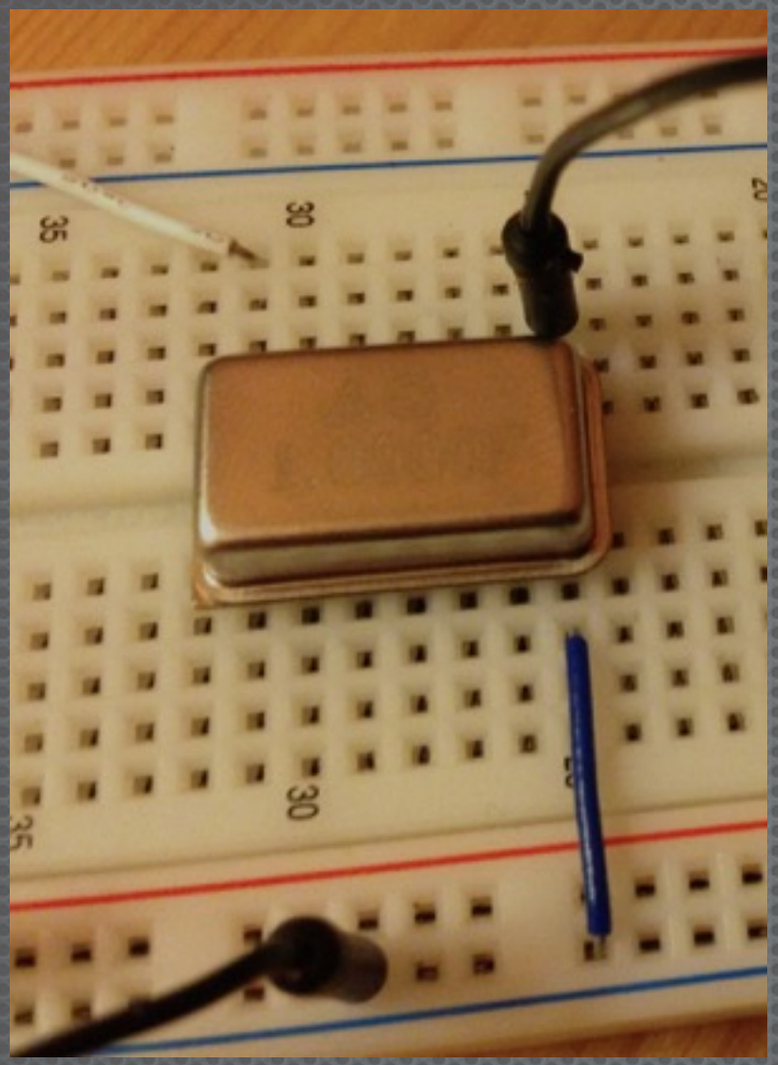
MOUSE

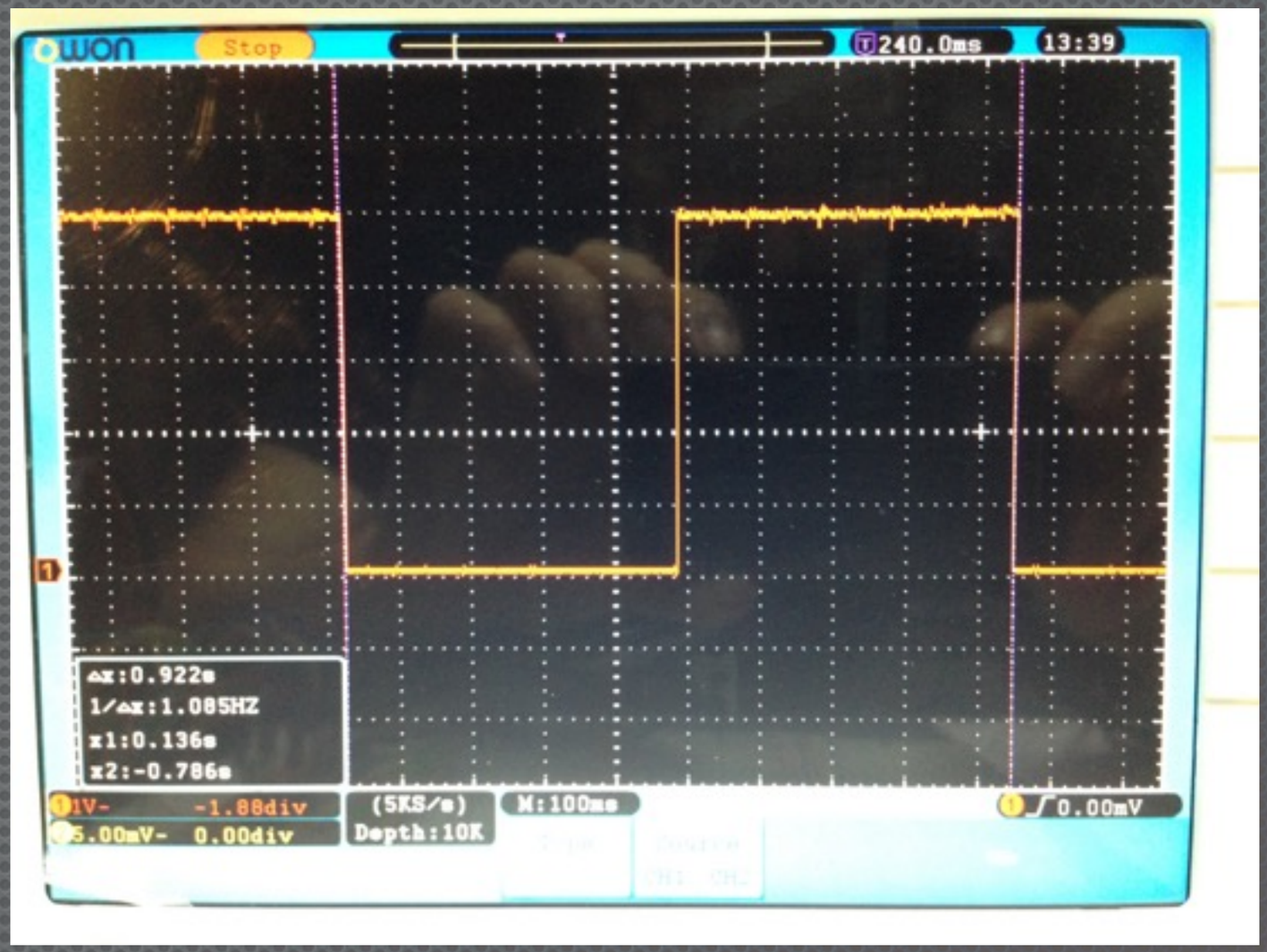
- Warum baut man einen Computer?
- Warum hat das 30 Jahre gedauert?
- Was ist ein Computer für mich?
- Wie fängt man das Ganze an?
- Hilfe im Netz: http://6502.org
  inkl. sehr gutem Forum

- CPU: 65C02
  - Vorkenntnisse
  - einfacher Takt
  - kein Bus-Multiplexing
  - benötigt nur 5V Spannung
  - wird immer noch produziert

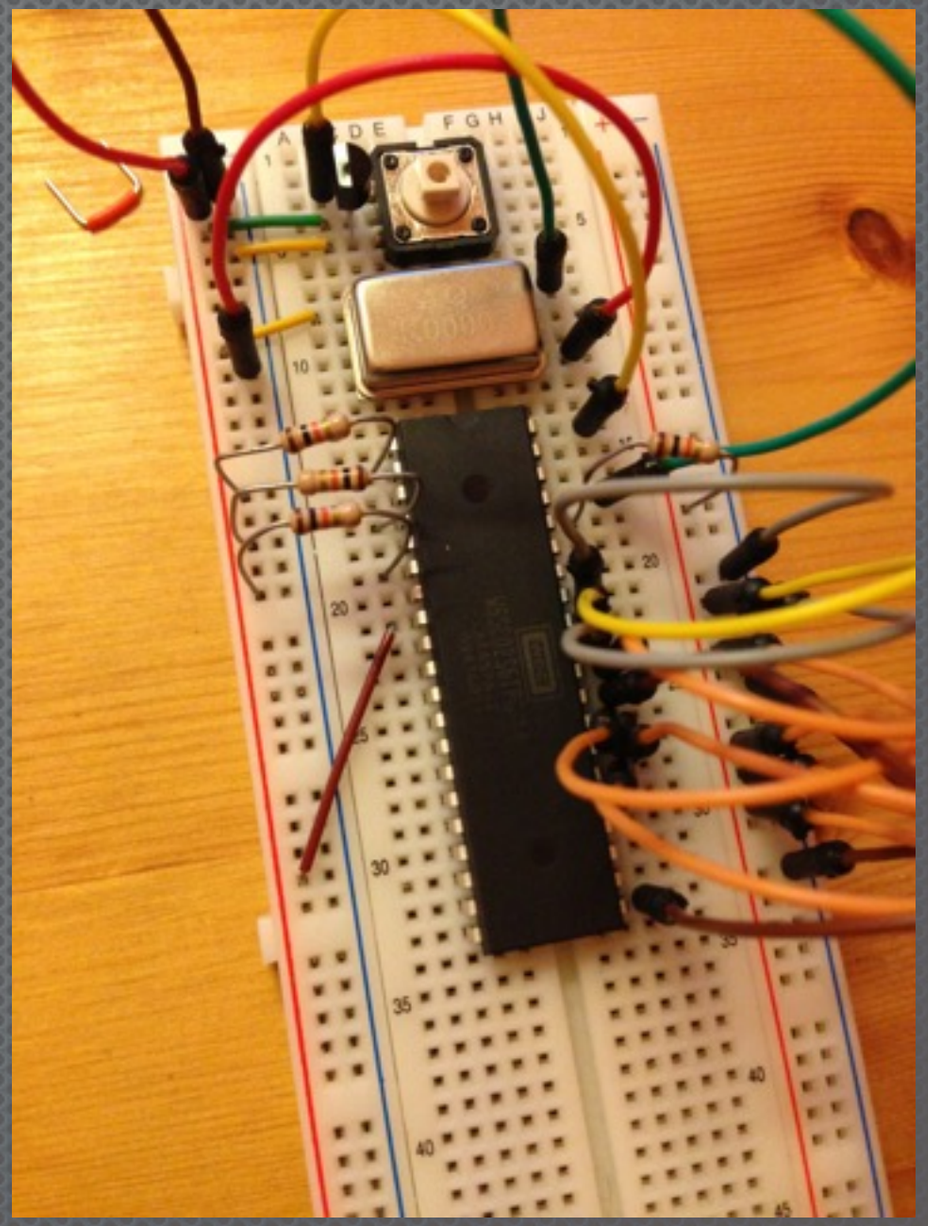


- **Schritt 1: Takt erzeugen**

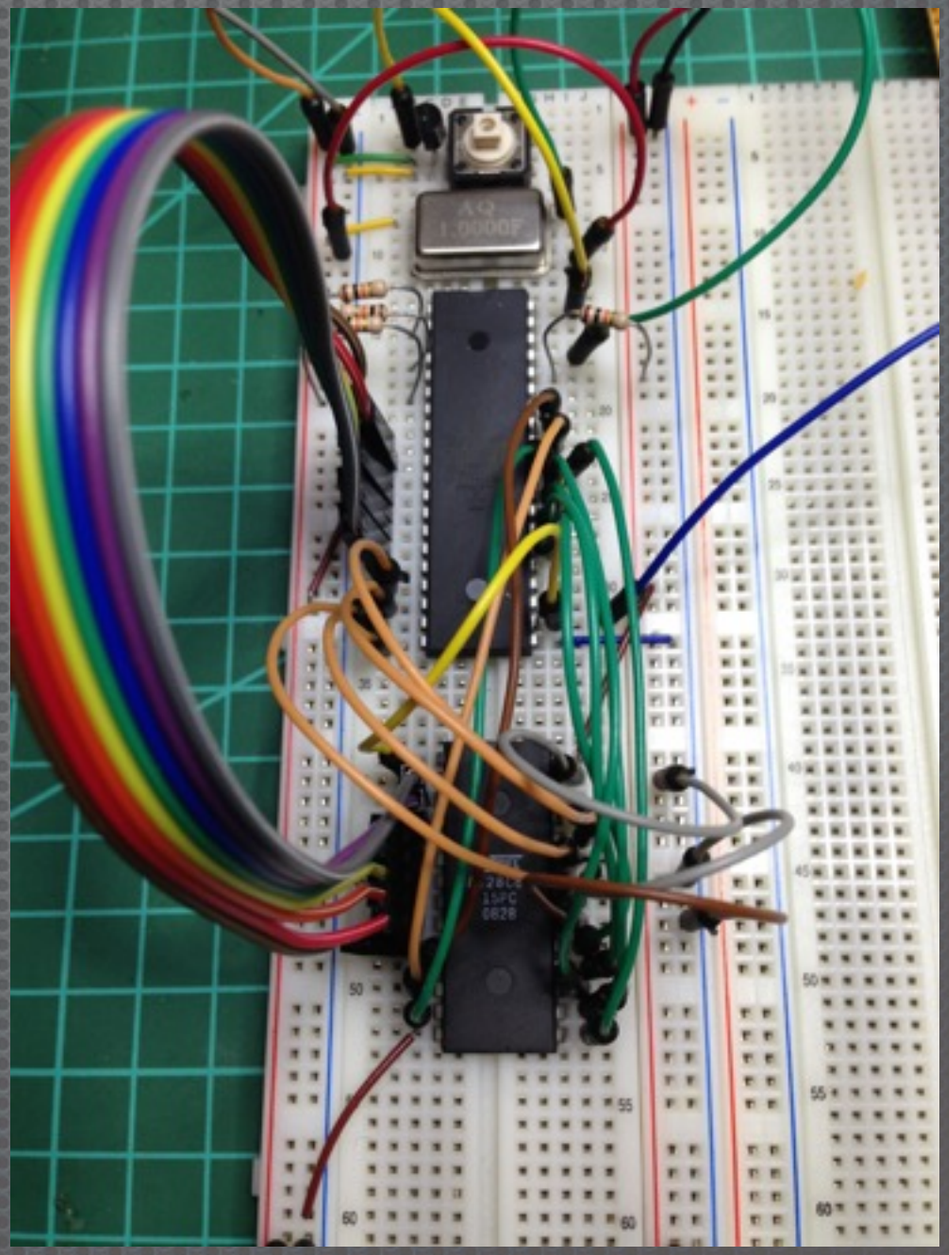




- Schritt 2: Free Run
  - NOP (No Operation)
  - Datenbus auf 0xEA
  - PullUp Widerstände
  - Reset-Logik



- Schritt 3: Speicher
  - EEPROM mit Daten
  - Adressbus verbunden
  - Datenbus verbunden
  - R/W verbunden
  - /CE + /OE auf Masse



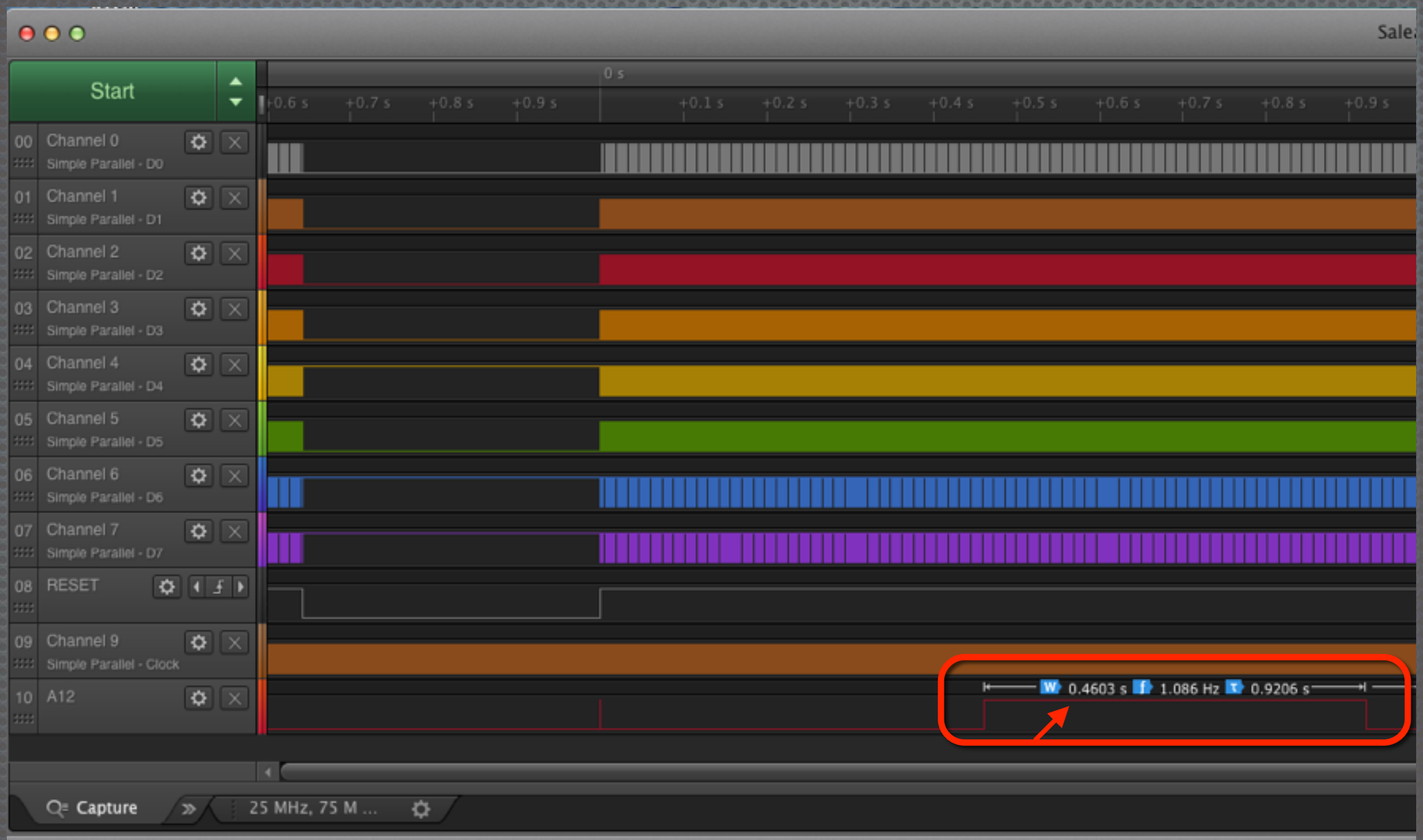
- # ROM only Testprogramm

```
.org $e000                ; start at $e000
.outfile "romtest.bin"    ; name the file

.scope
        LDX #$00          ; A2 - 2 cycles
outer1: LDY #$00          ; A0 - 2 cycles
inner1: NOP               ; EA - 2 cycles \
        INY               ; C8 - 2 cycles   7 cycles
        BNE inner1        ; D0 - 3 cycles /
        INX               ; E8 - 2 cycles
        BNE outer1        ; D0 - 3 cycles
        JMP $f000         ; 4C - 3 cycles
.scend

                          ; ( 256 * 7 cycles + 7 ) * 256 + 3 = 460547 cycles

.advance $f000            ; fill up with $00 to $f000

.scope
        LDX #$00          ; set X to 0
outer2: LDY #$00          ; set Y to 0
inner2: NOP               ; do nothing
        INY               ; increment Y
        BNE inner2        ; if Y != 0 branch to inner loop
        INX               ; if Y == 0 increment X
        BNE outer2        ; if X != 0 branch to outer loop
        JMP $e000         ; jump back to first loop
.scend

.advance $fffc            ; fill up to reset vector
.word    $e000            ; set reset to $e000
.word    $0000            ; fill last vector to $0000
~
~
```

```
        1111 1100 0000 0000
        5432 1098 7654 3210

$E000 - 1110 0000 0000 0000
$F000 - 1111 0000 0000 0000
```

```
0x0000  A2 00 A0 00 EA C8 D0 FC E8 D0 F7 4C 00 F0 00 00
0x0020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x0040  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x0060  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x0080  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00A0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x0100  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x0120  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x0140  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x0160  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x0180  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x01A0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x01C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x01E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

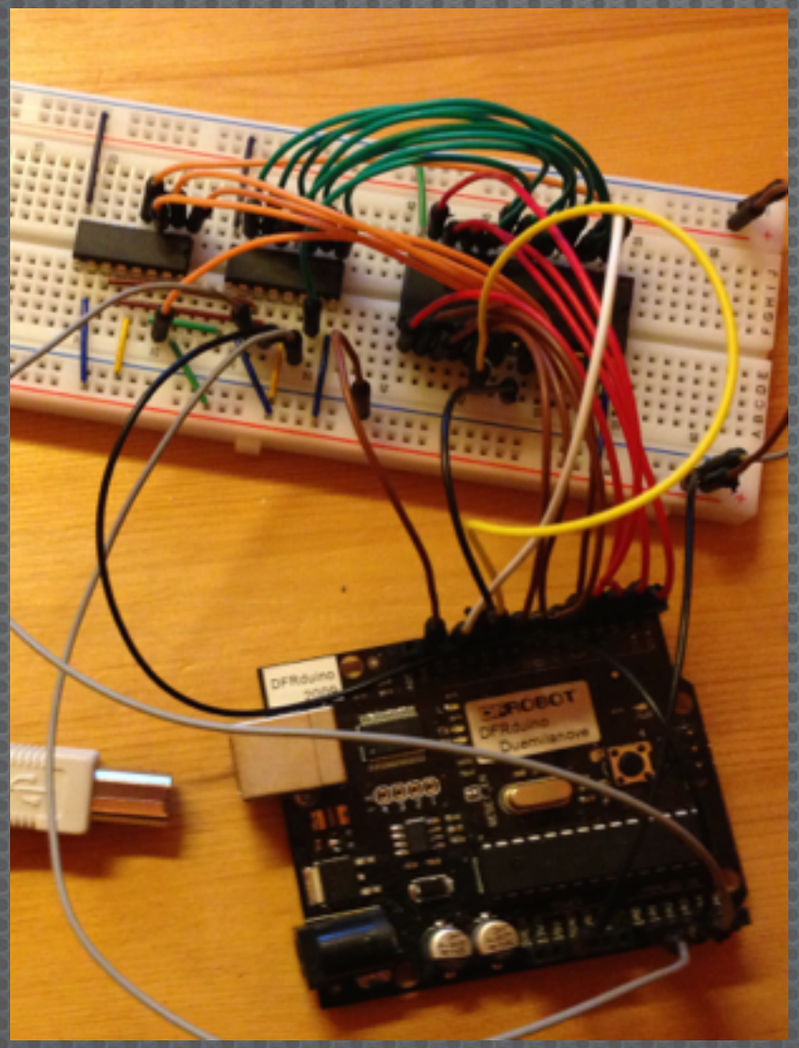
- **Ausgabe Logic-Analyser**



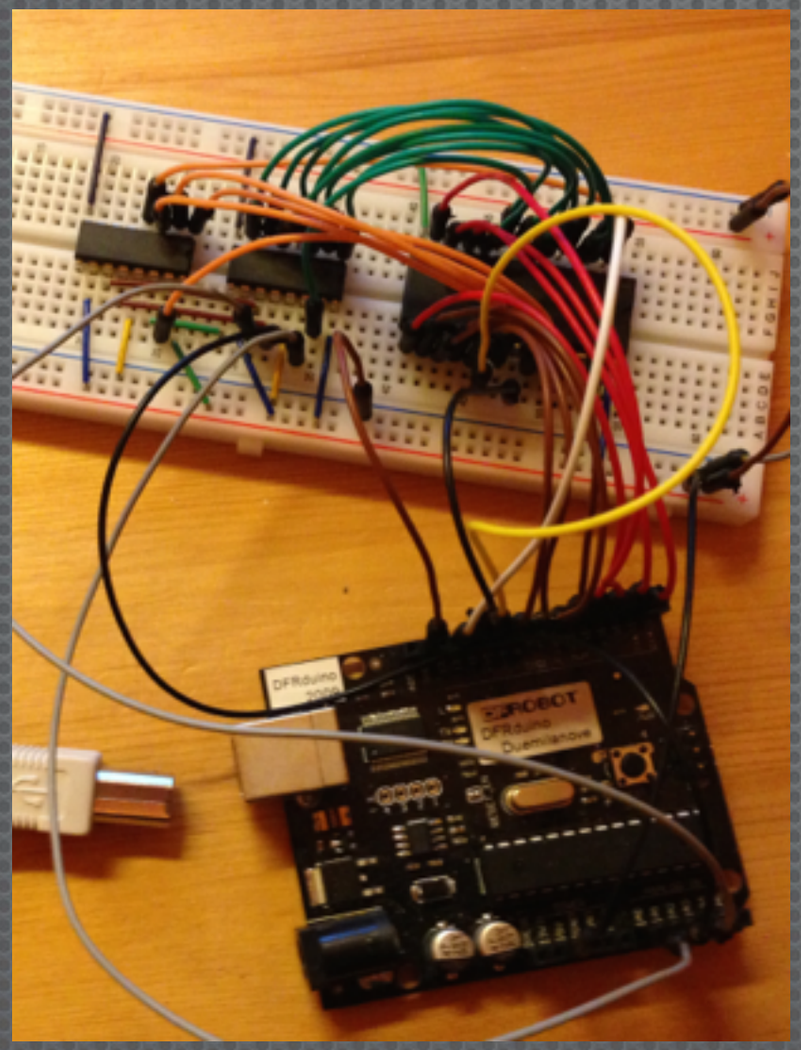
- Moment mal …

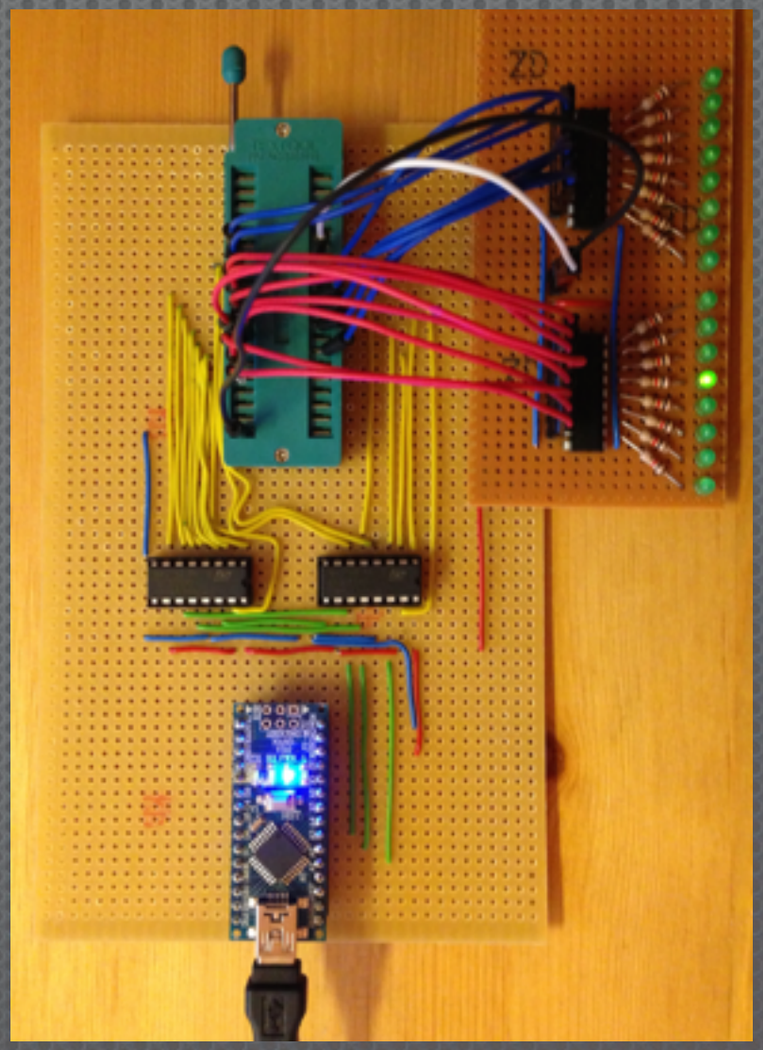
- Moment mal …
  Woher kommen eigentlich die Daten auf dem EEPROM?

- **Moment mal …**
  **Woher kommen eigentlich die Daten auf dem EEPROM?**

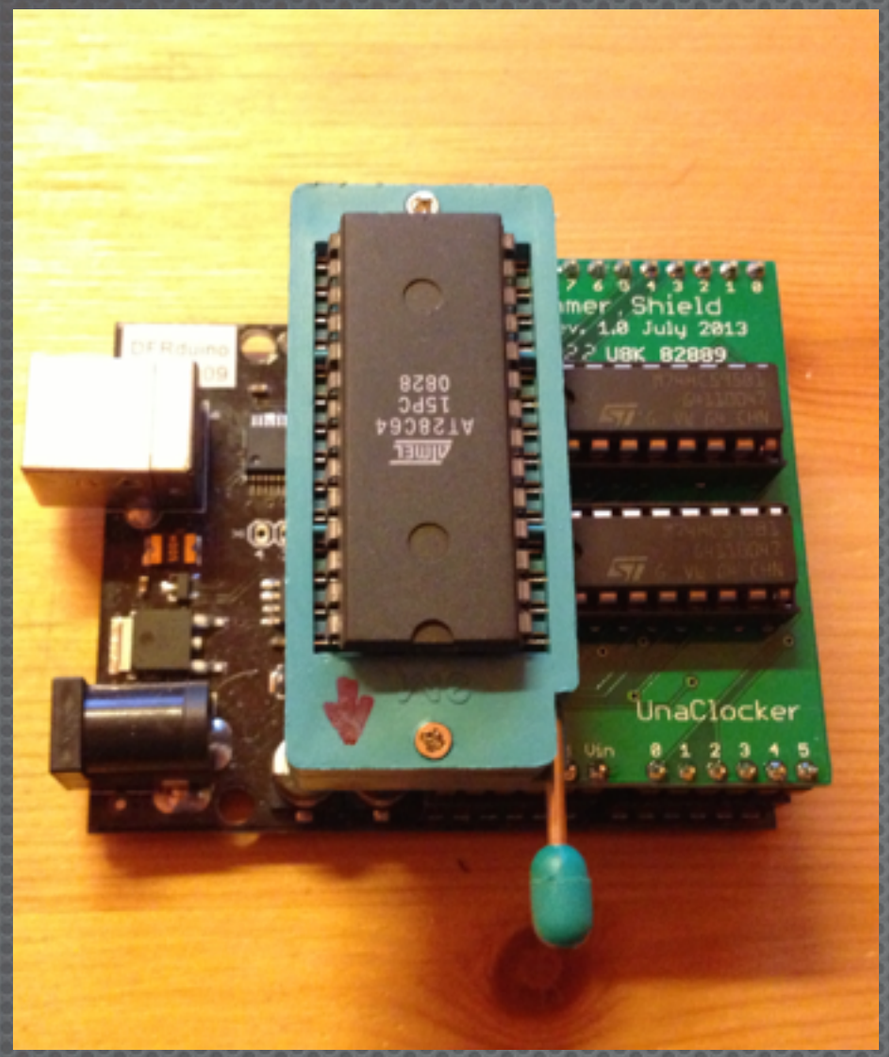


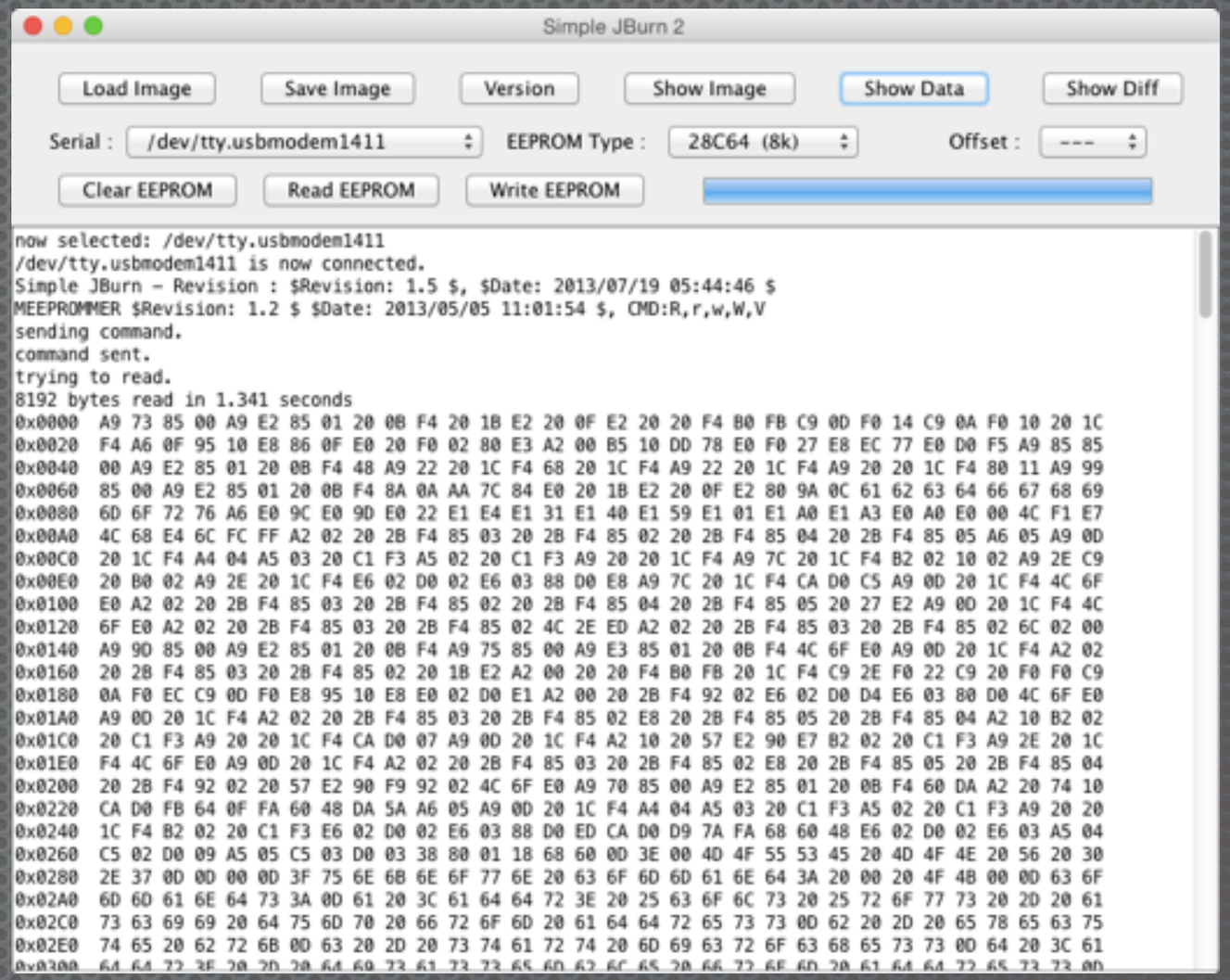
- **Moment mal …**
  **Woher kommen eigentlich die Daten auf dem EEPROM?**





- # Spin-Off: MEEPROMMER




https://github.com/mkeller0815/MEEPROMMER

- … und wo kommen jetzt die Daten her?

- … und wo kommen jetzt die Daten her?

- Ophis 65xx Cross Assembler
  - in Python geschrieben
  - kennt 6502, 65C02, und weitere
  - Macros, Label, lokale Label
  - gut dokumentiert

https://github.com/michaelcmartin/Ophis

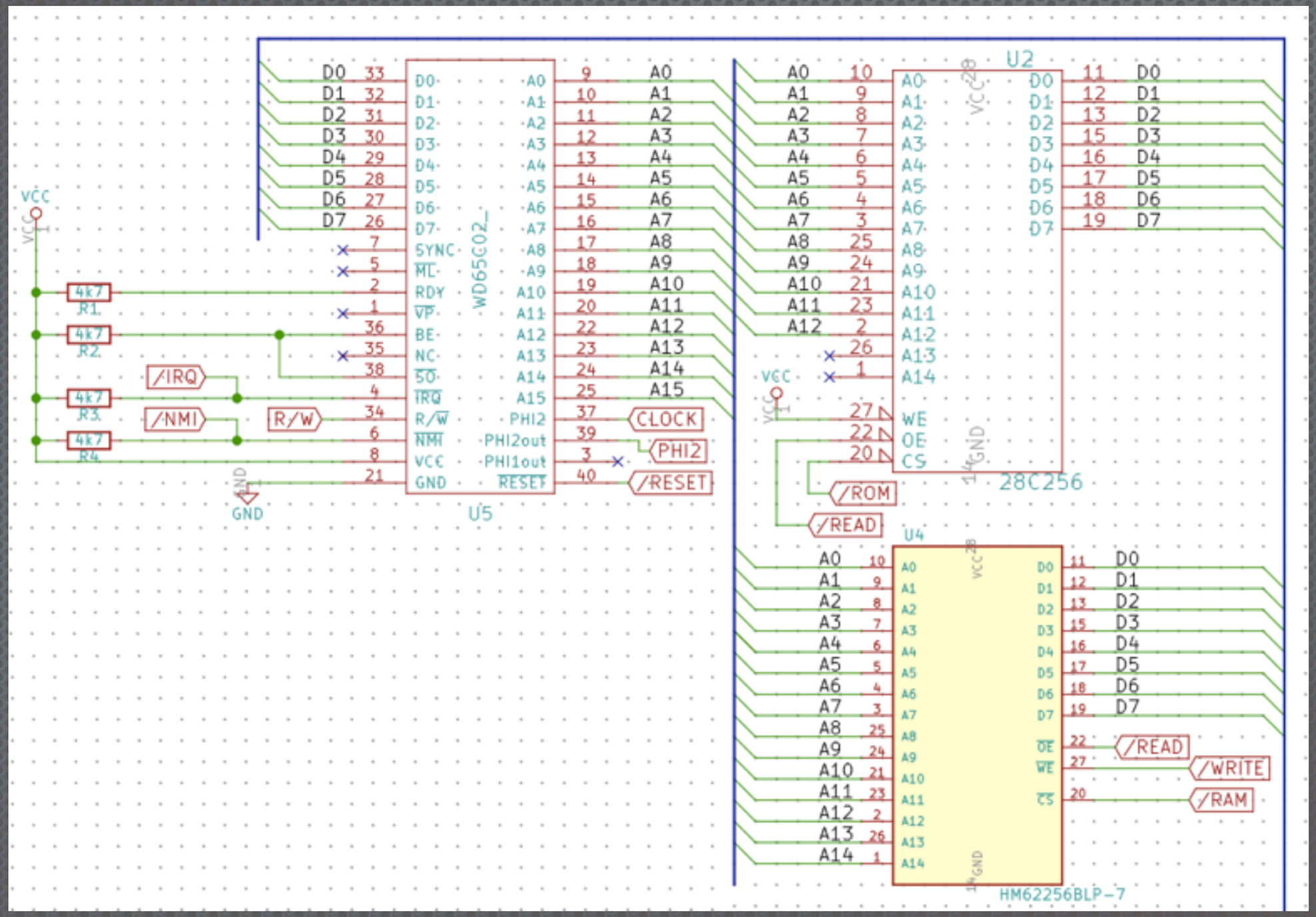
- … und woher weiß ich das es funktioniert?

- Py65
  - in Python geschrieben
  - Simuliert 6502, 65C02 und 65Org16
  - py65mon Monitor und Tracer
  - (fast) gut dokumentiert :-)

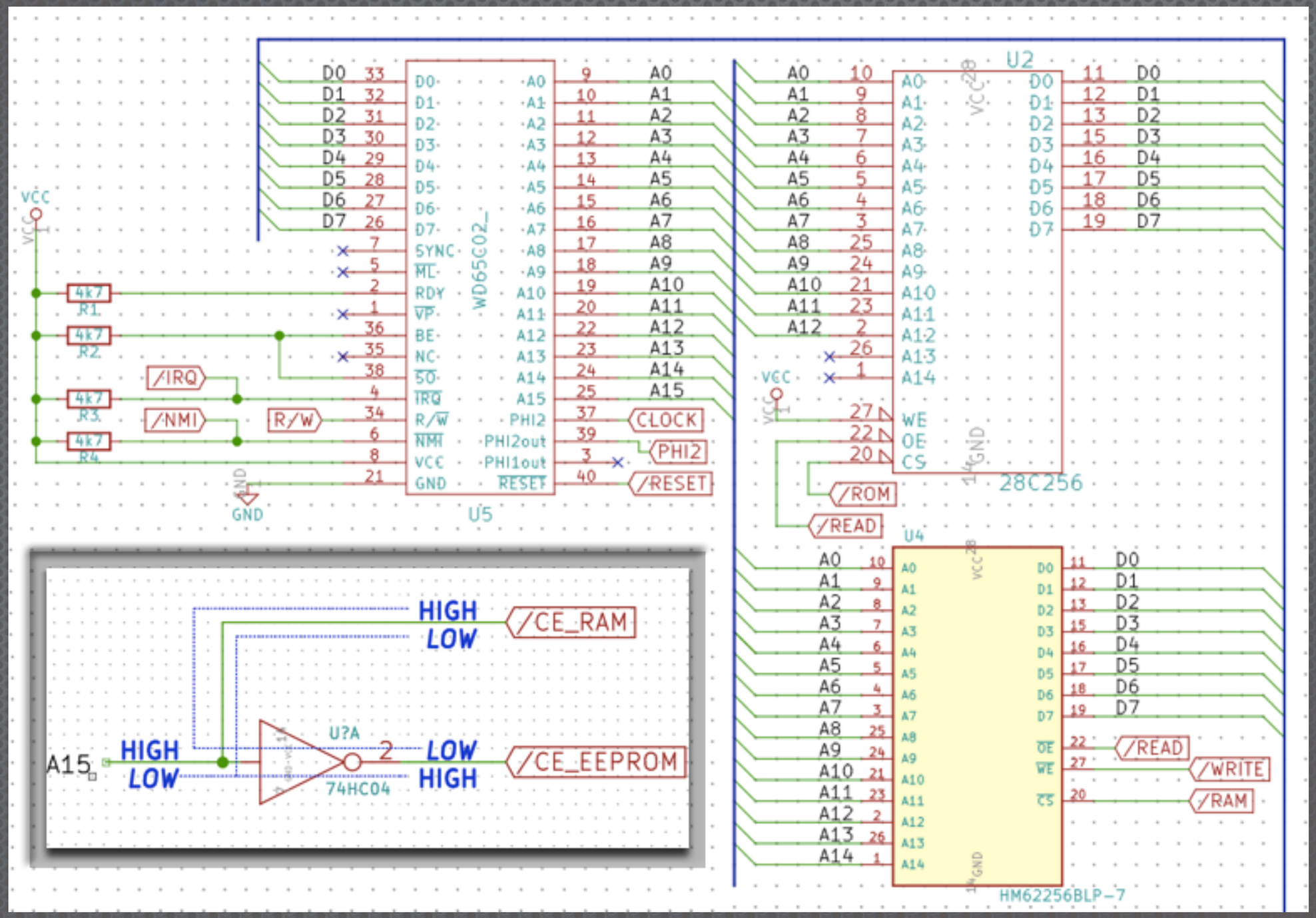
https://github.com/mnaberez/py65

- Mehr als ein Baustein an der CPU?

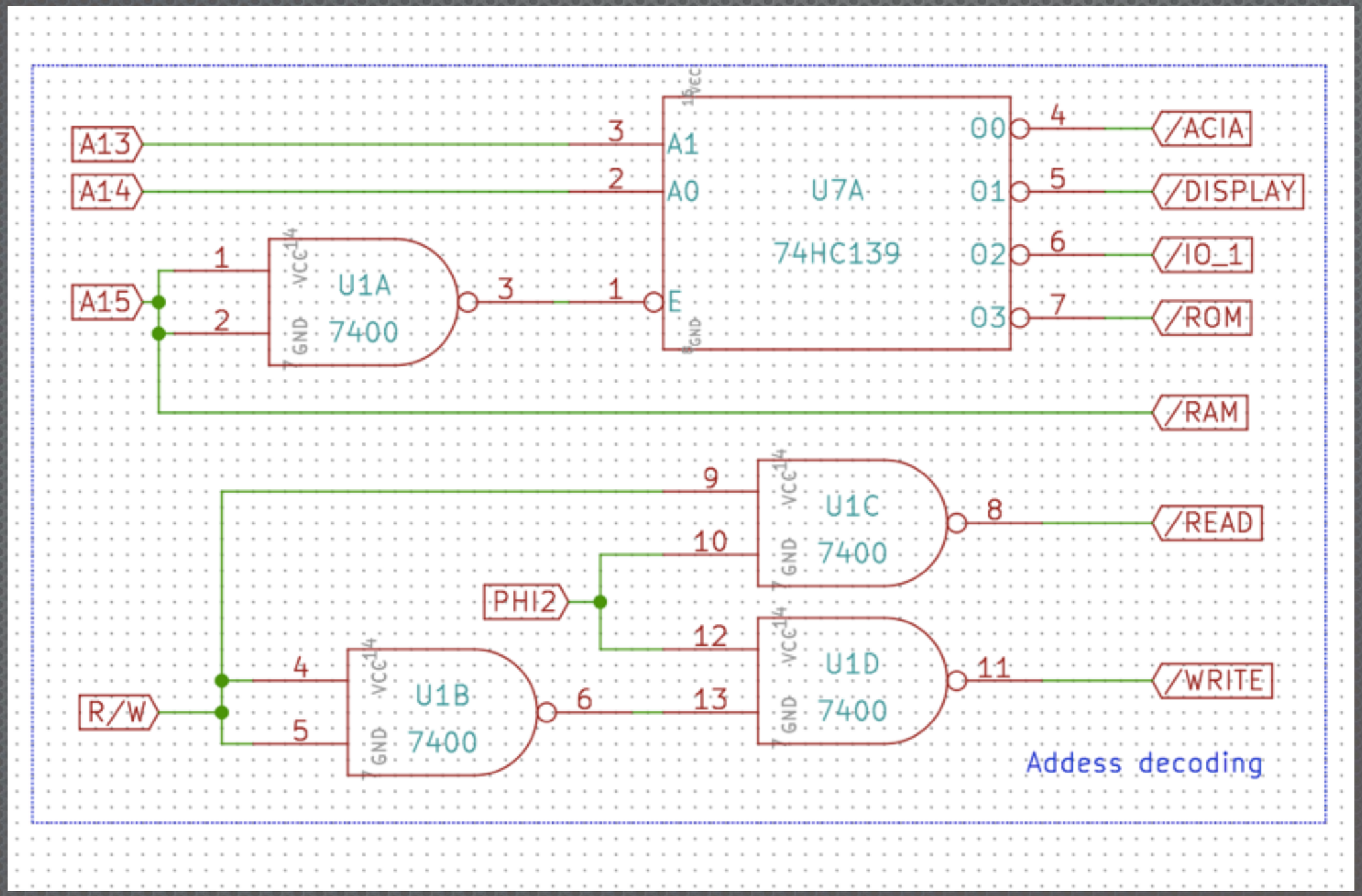
- **Mehr als ein Baustein an der CPU?**



- **Mehr als ein Baustein an der CPU?**

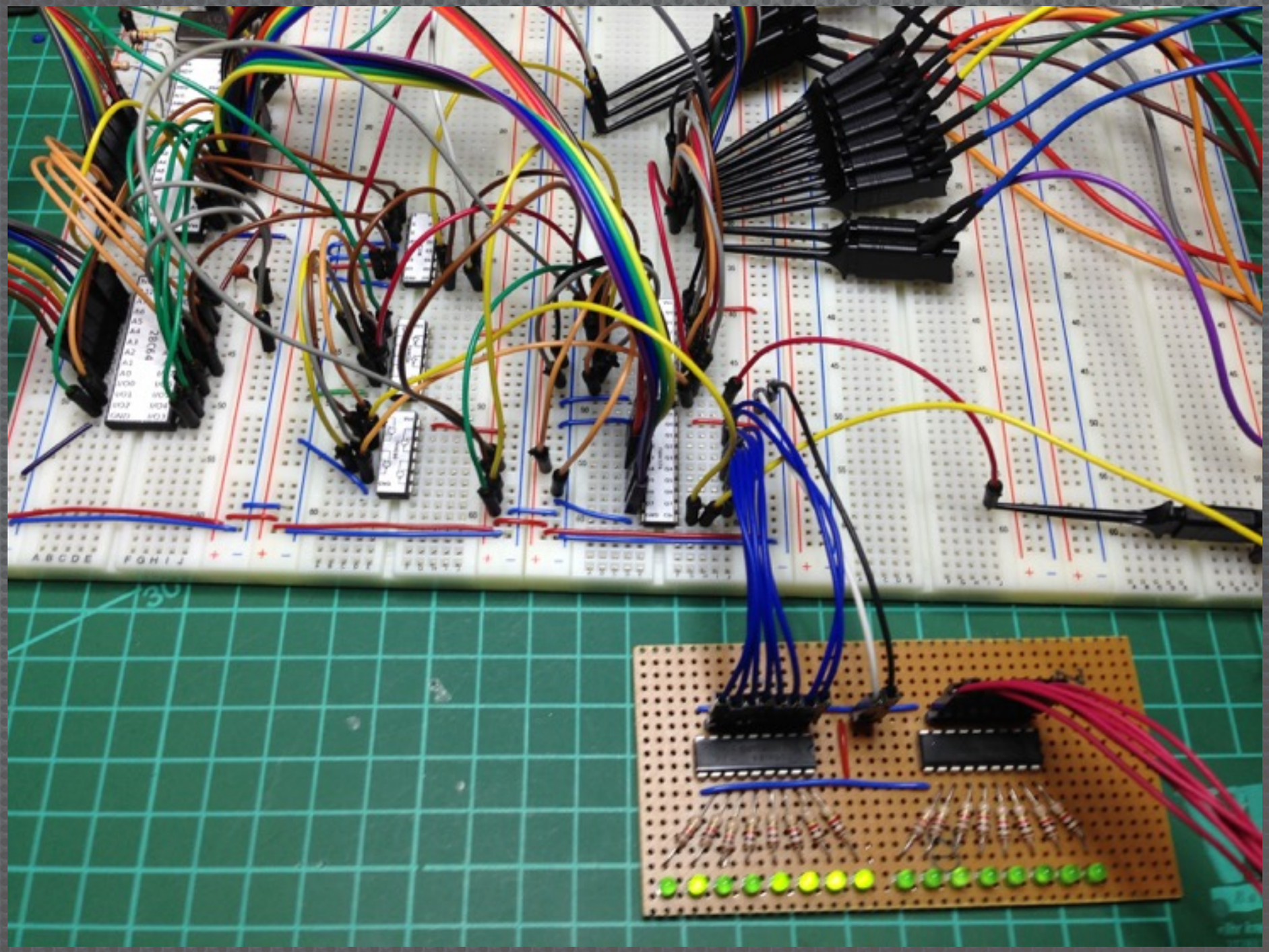


- # MOUSE Adressdekodierung



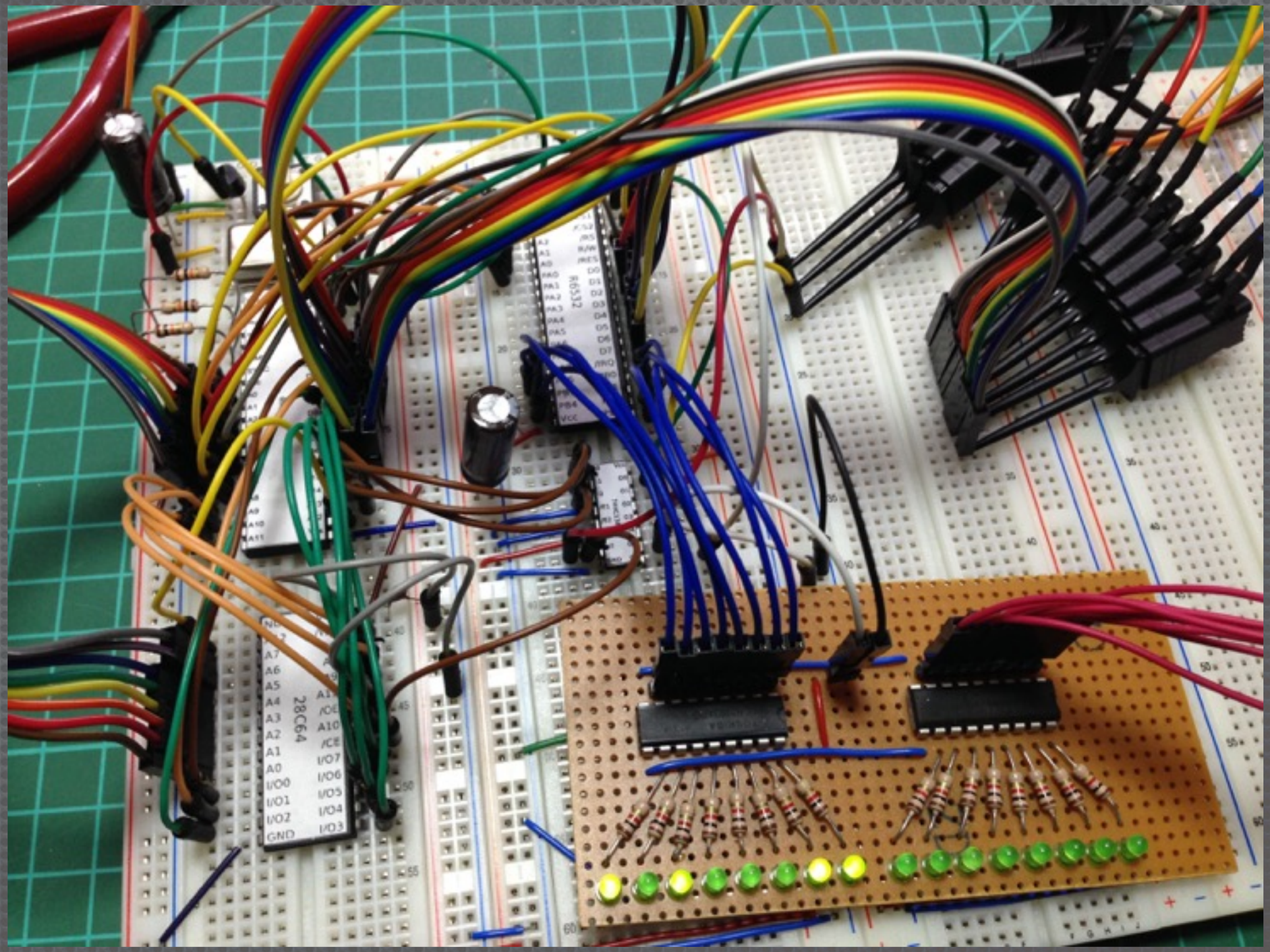
- Kommunikation:
  - 74HC574 (8bit Output Latch)
  - R6532 RIOT (Ram I/O Timer)
    - 128 Byte SRAM
    - 2x8bit I/O Ports
    - 8bit Timer mit Interrupt
  - M6850 ACIA (serielle Schnittstelle)

- 74HC574 Output

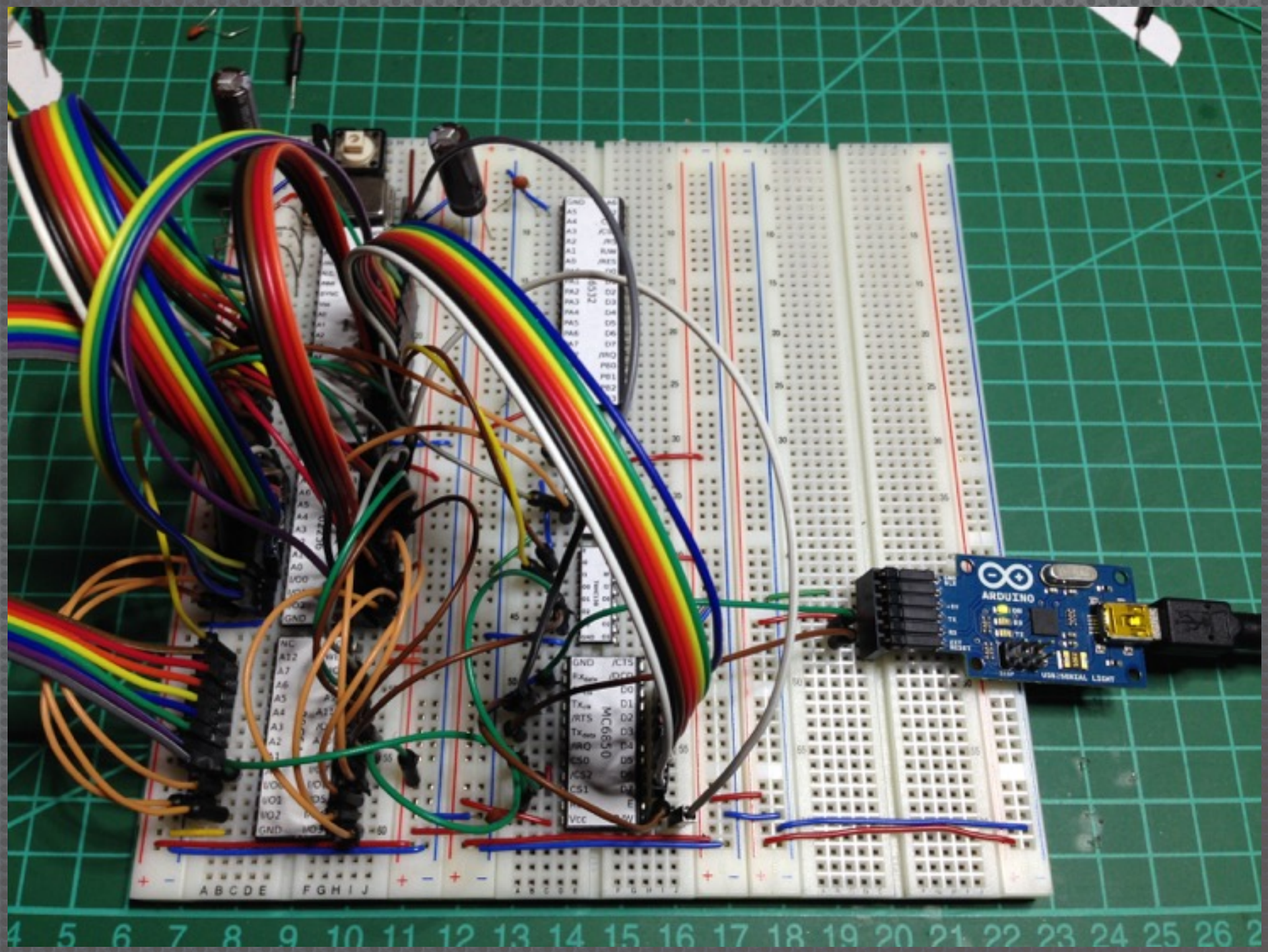


MOUSE

- R6532 RIOT

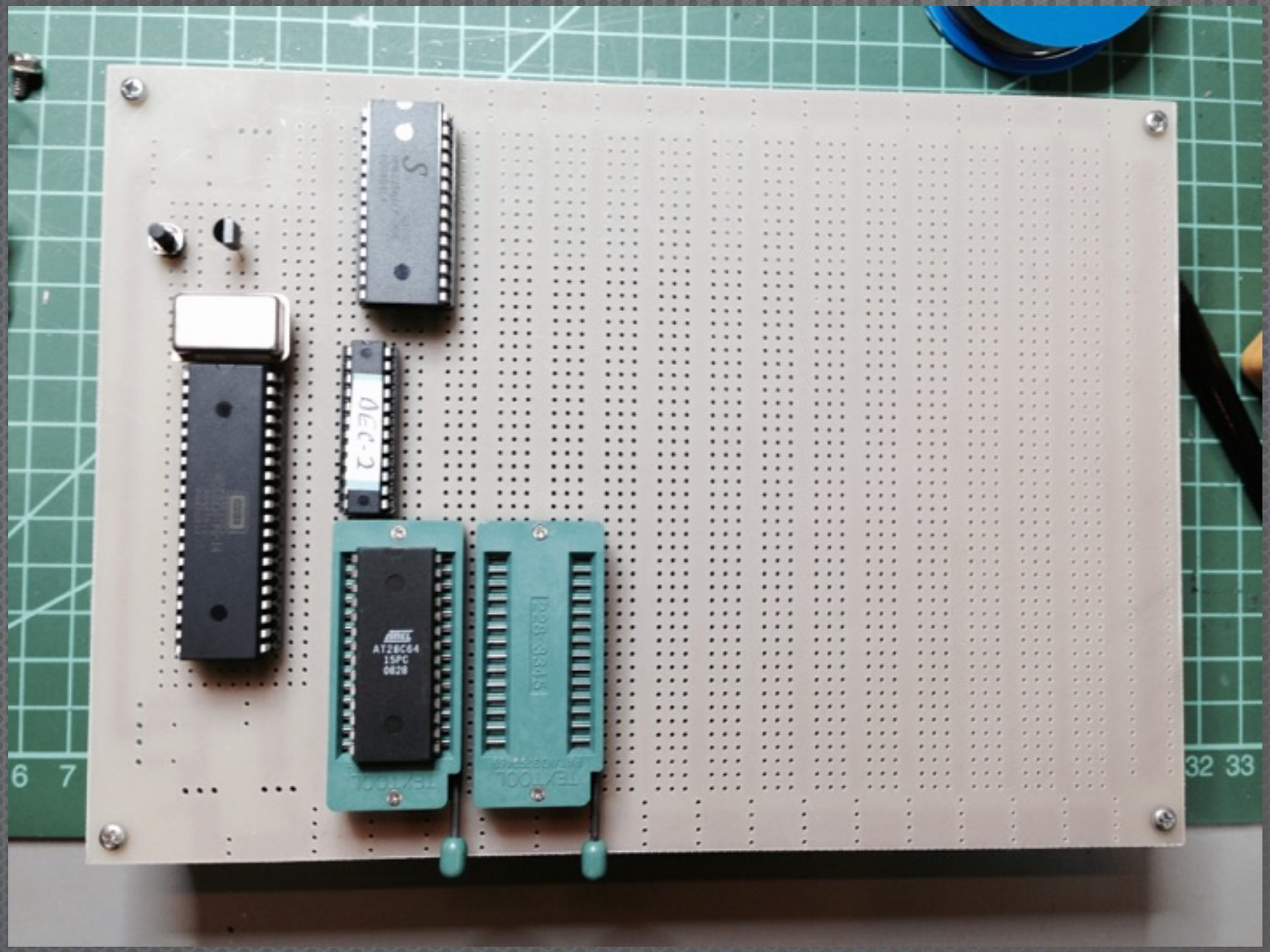


- M6850 ACIA + 32k SRAM



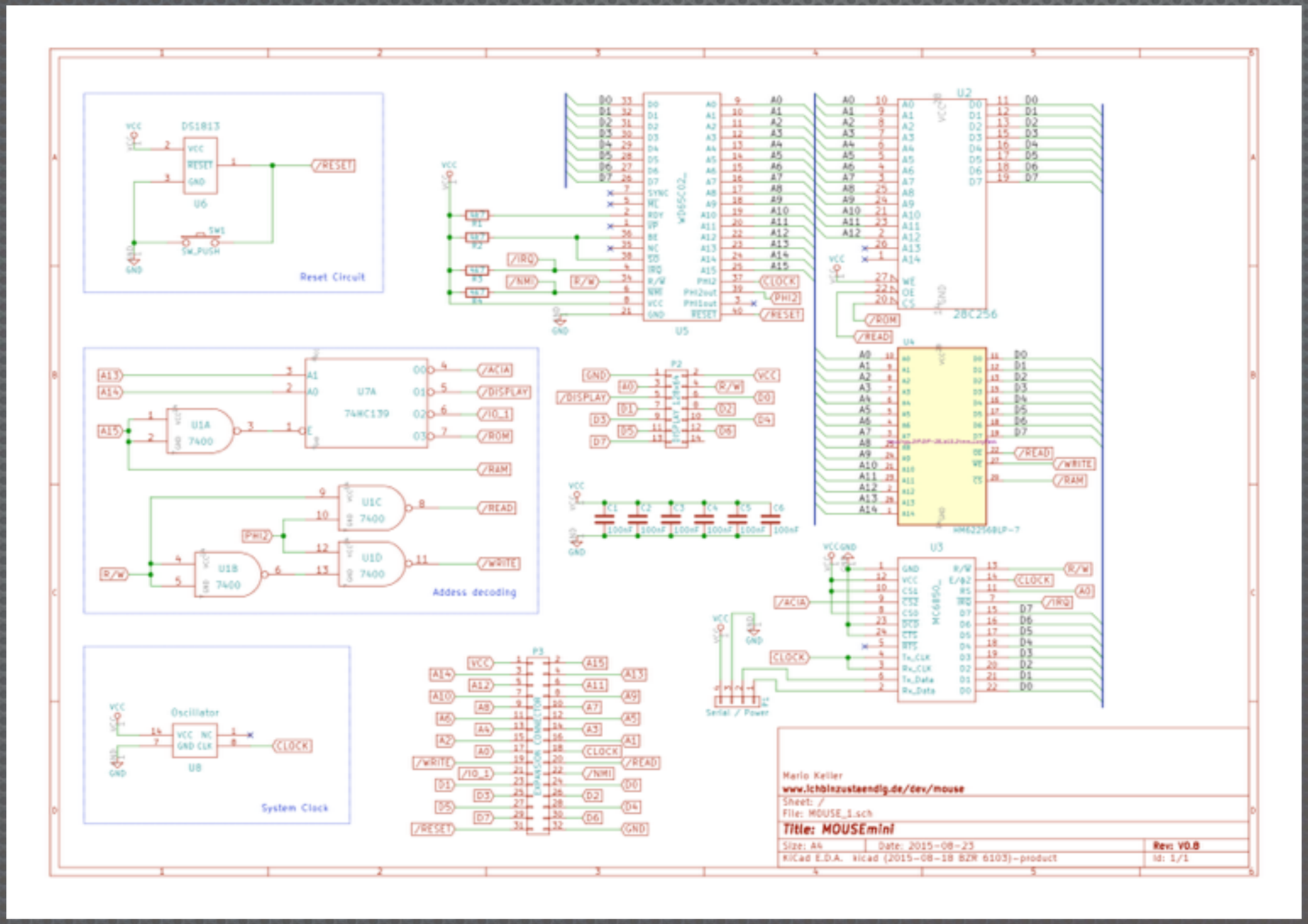
- Und nun in Hardware …

- Und nun in Hardware … FAIL :-(

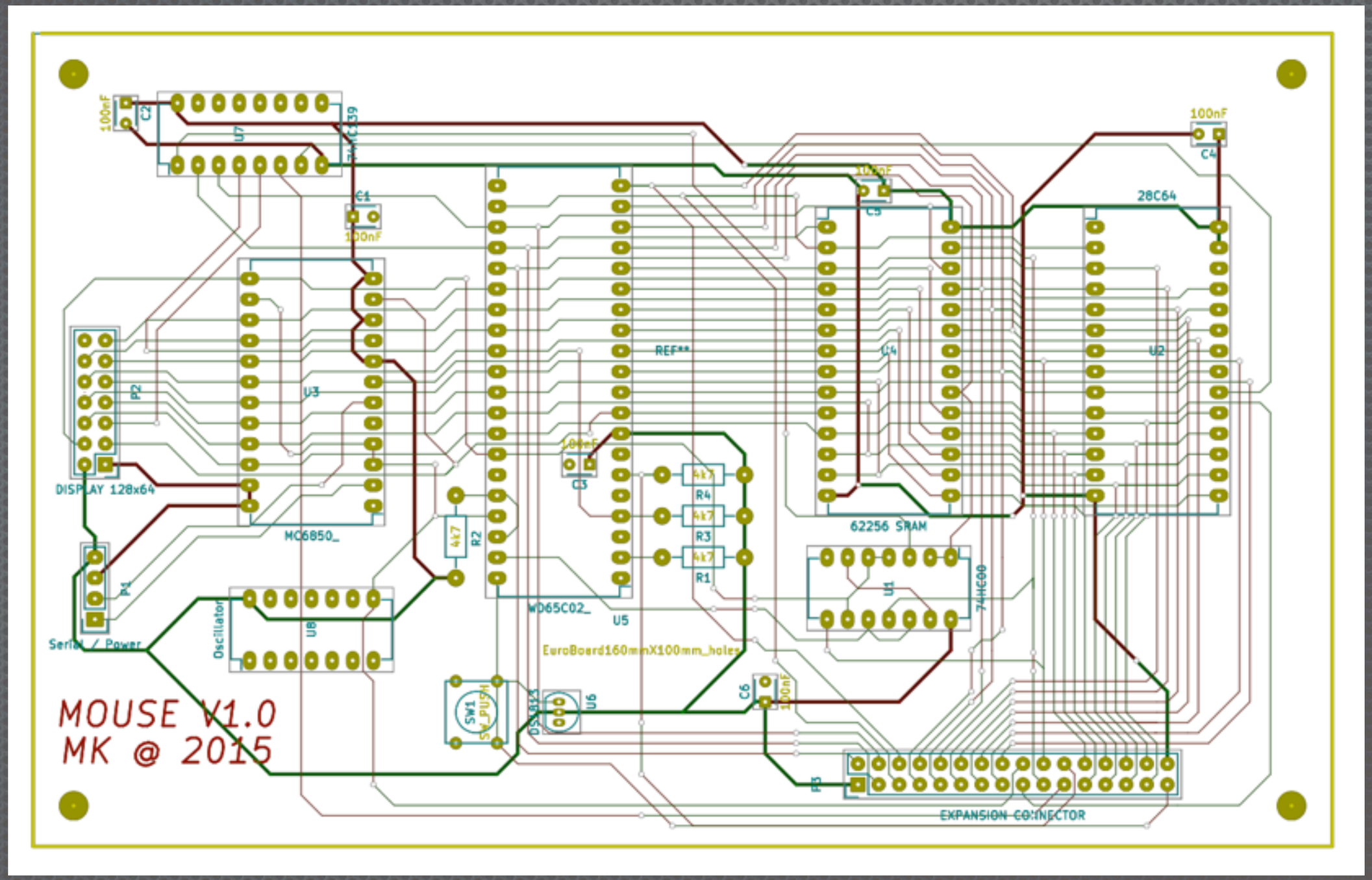


- Zweiter Versuch, diesmal richtig ...

- **Zweiter Versuch, diesmal richtig ...**

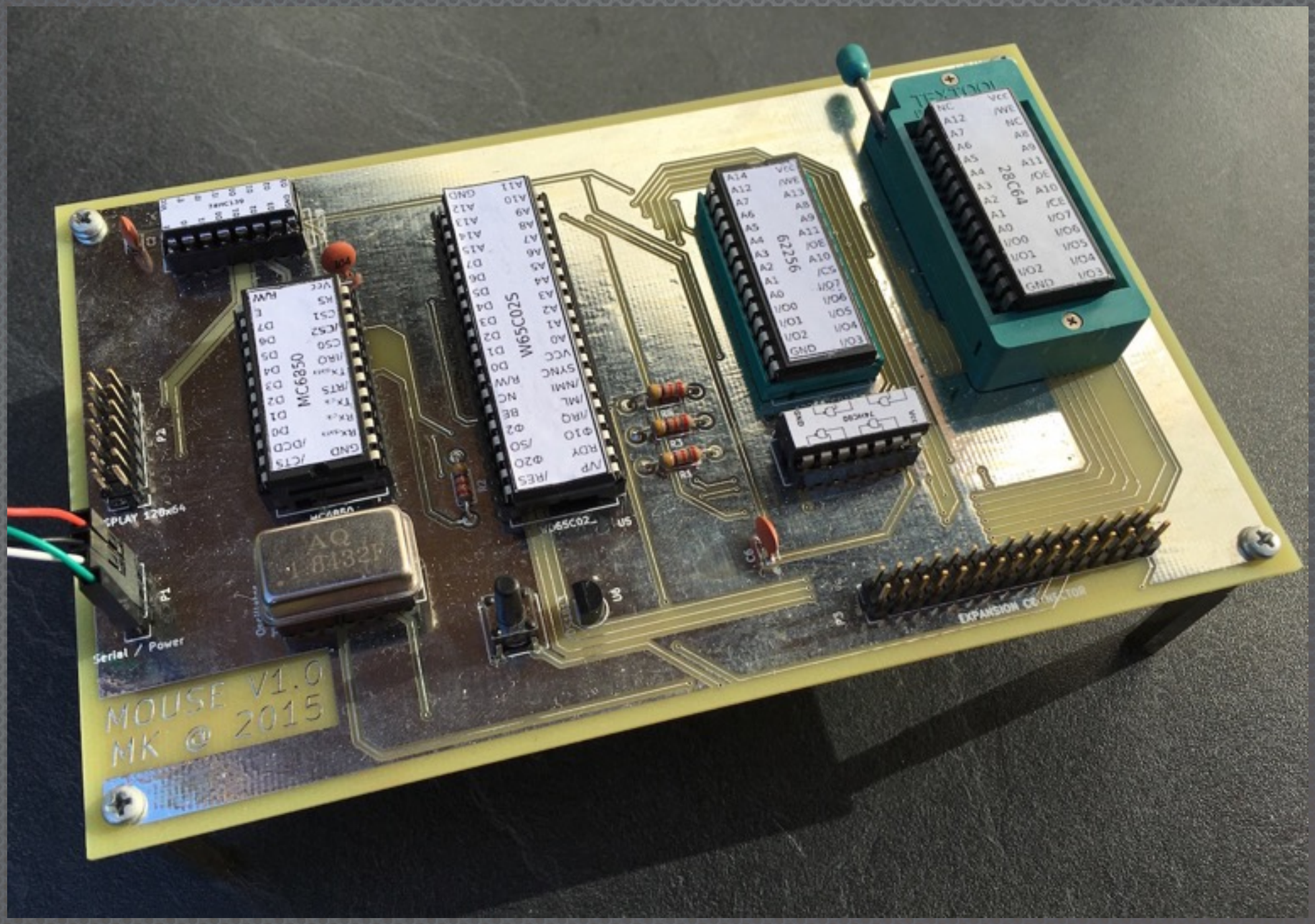


- **Zweiter Versuch, diesmal richtig … (fast)**



MOUSE

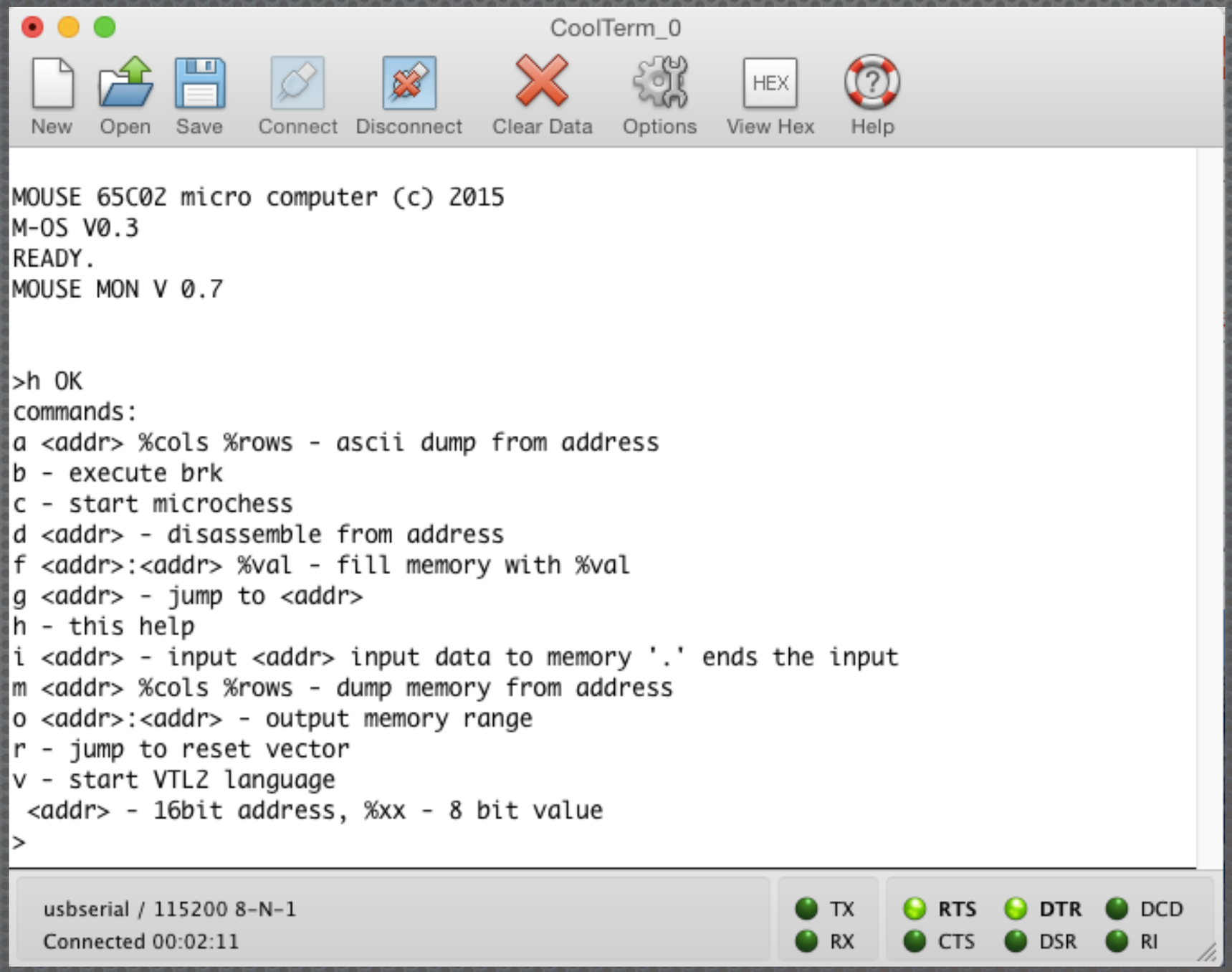
- **Nach einem Tag Fehlersuche:**



- und die Software … ?

- und die Software … ?
  - MIOS (Minimal Input/Output System)
    - Defines und Macros
    - ACIA Treiber
    - Kernel und Utilities
  - M-OS (MOUSE OS)
    - Main Programm
    - Monitor Programm
  - Programme
    - VTL2 (Programmiersprache)
    - Micro Chess
    - Disassembler

- **und die Software … ?**



MOUSE

Vielen Dank

für Eure Geduld :-)