

Steckschwein Emulator && Unit-Tests

Wie wir ohne Hardware entwickeln.

Warum?

- Michael Steil (pagetable.com) – VCFB 2019
“Wie könnt Ihr Software für das Steckschwein entwickeln ohne Emulator?”
- Debugging/Tracing
- IDE-Integration
- Produktivität

Was braucht man?

- CPU-Emulator für 65(c)02
- Grafik (V9958)
- Memory-Logik aus dem CPLD
- I/O (VIA, SD-Card, Seriell, Joysticks)
- Sound (YM3812/OPL2)
- RTC (DC1306)

65(c)02 CPU-Emulator

- diverse Implementierungen verfügbar
- **fake6502** (<http://rubbermallet.org/fake6502.c>)
 - „zyklengenau“ - clockticks api
 - callback api (hook)

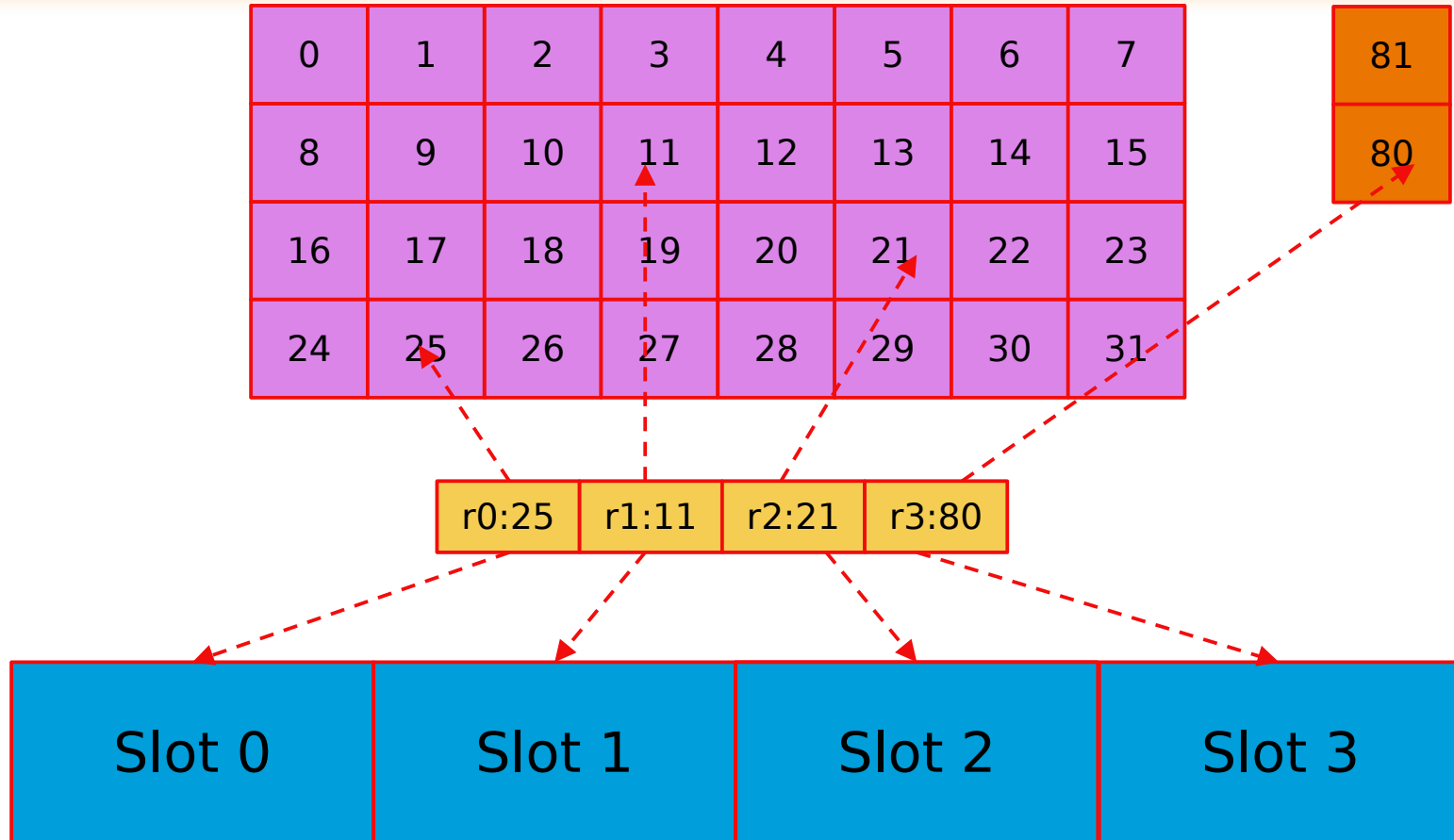
Grafik

- „Steil-Vorlage“ aus X-16 Emulator (Michael Steil)
- SDL2 (simple direct media layer)
 - einfache API / Programmier-Modell - Framebuffer
 - schnelle Umsetzung

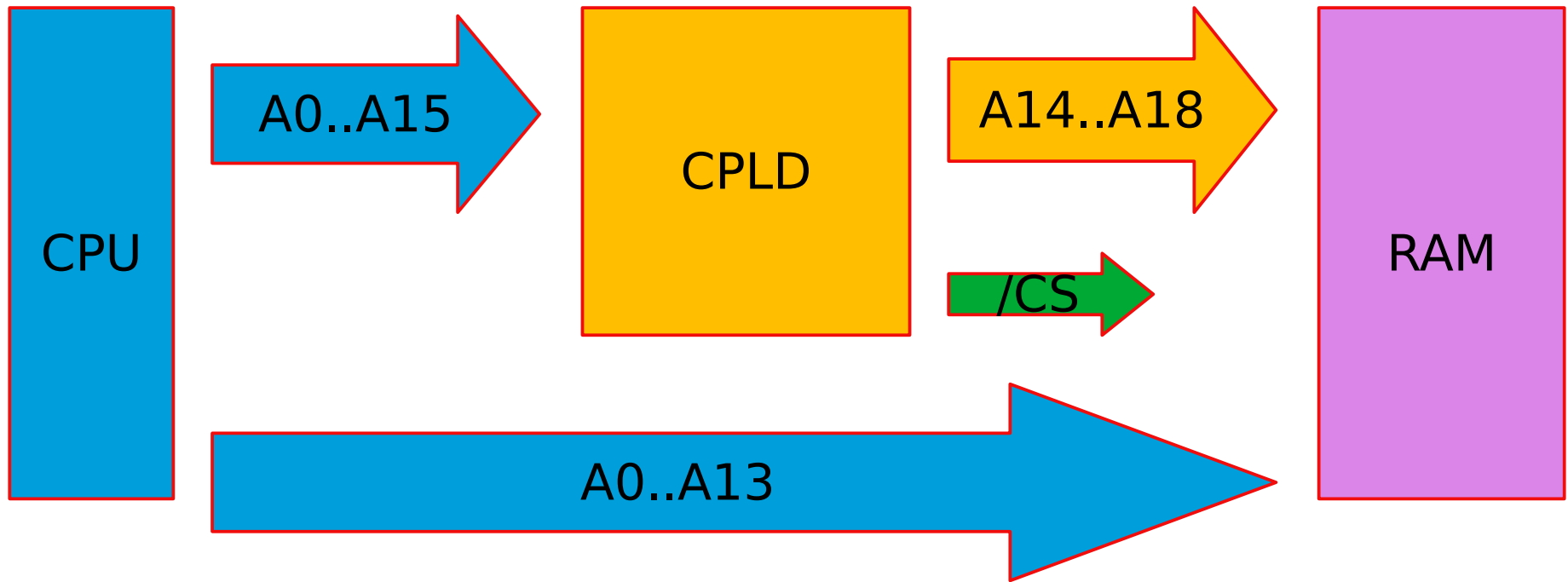
Grafik - V9938/V9958

- Emulation komplex und aufwändig
 - tiefgreifendes Verständnis der Logik vom V9958 nötig
 - Command-Engine – Lines, Blitt-Commands
 - Diverse Text-/ Grafik-Modes zu implementieren
- BlueMSX (<http://www.bluemsx.com/>)
 - Emulator für diverse MSX1,2,2+ -Plattformen
 - SDL1 :/
 - not maintained anymore :/

Memory-Logik aus dem CPLD



Memory-Logik aus dem CPLD



Memory-Logik aus dem CPLD

```
sig_cs_rom <= INT_banktable(conv_integer(CPU_a(15 downto 14)))(5) AND NOT io_select;  
sig_cs_ram <= not(INT_banktable(conv_integer(CPU_a(15 downto 14)))(5)) AND NOT io_select;
```

```
void memory_init() {  
    ram = malloc(RAM_SIZE);  
    rom = malloc(ROM_SIZE);  
}
```

```
uint8_t *get_address(uint16_t address, bool debugOn){  
  
    uint8_t *p;  
    uint32_t mem_size;  
  
    uint8_t reg = (address >> BANK_SIZE) & sizeof(ctrl_port)-1;  
    if((ctrl_port[reg] & 0x80) == 0){ // RAM/ROM?  
        p = ram;  
        mem_size = RAM_SIZE;  
    }else{  
        p = rom;  
        mem_size = ROM_SIZE;  
    }  
  
    uint32_t extaddr = ((ctrl_port[reg] & ((mem_size >> BANK_SIZE)-1)) << BANK_SIZE) | (address & ((1<<BANK_SIZE)-1));  
}
```

I/O – VIA, SD-Card, UART, Joysticks

- VIA
 - rudimentäre Implementierung
- SD-Card
 - init, block read/write
- UART (serial i/o)
 - aus QEMU – Projekt
 - „serial loopback“ (vgl. unix socat)
- Joysticks
 - SDL2 – Joystick API

Sound

- Sound (YM3812/OPL2)
 - MAME (multi-purpose emulation framework)

RTC

- RTC (DS1306)
 - Implementierung nutzt Systemzeit - „geht immer richtig“ ;)
 - nvram - wird in „home“ dir des Nutzers gesichert

DEMO

Ausblick

- IDE-Integration – VSCode
 - Debugger Extension (Debug Adapter)
 - alchemy65 (<https://github.com/AlchemicRaker/alchemy65>)
- Steckschwein-Debugging
 - „im 6502 Code im Steckschwein-Emulator debuggen“
 - Video-Speicher
 - RAM, ROM, Banks etc.

ASM & Unit-Tests

- Warum?
 - steckOS ist „komplex“ geworden
 - erfordert Modularisierung von Code
 - Produktivität
 - Test Driven Development
 - Regression Testing
 - Code Quality

ASM & Unit-Tests

- Voraussetzungen, Was brauchen wir?
 - Entkopplung des Codes von konkreter Hardware
 - Test-API
 - Mocks

ASM & Unit-Tests

- assertA <expect> - vgl. Akku mit <expect>
- assertX <expect> - vgl. X-Reg mit <expect>
- assertY <expect> - vgl. Y-Reg mit <expect>
- assert8, assert16, assert32 <expect> <address> - vgl. Bytes an Adresse mit <expect> (big endian)
- assertCarry <0|1> - vgl. Carry mit <expect>
- assertZero <0|1> - vgl. Zero mit <expect>
- assertString <expect> <address> - ...
- assertOut <expect> - vgl. expect mit char output
- fail <message> - fail mit Meldung

```
>
; hexout a binary number
;
.export hexout
.import char_out

hexout:
    pha
    phx

    tax
    lsr
    lsr
    lsr
    lsr
    jsr hexdigit
    txa
    jsr hexdigit
    plx
    pla
    rts

hexdigit:
    and #$0f           ;mask lsd for hex print
    ora #'0'           ;add "0"
    cmp #'9'+1         ;is it a decimal digit?
    bcc _out           ;yes! output it
    adc #6             ;add offset for letter a-f

_out:
    jmp char_out
```

```
.include "asmunit.inc" ; unit test api

.import hexout ; uut

.code

    lda #$7e
    jsr hexout

    assertOut "7E" ; assert output
    assertA $7e ; assert A is not destroyed

    lda #$e7
    jsr hexout

    assertOut "E7" ; assert output
    assertA $e7 ; assert A is not destroyed

    lda #$9f
    jsr hexout

    assertOut "9F"
    assertA $9f

    brk

.segment "ASMUNIT"
```

ASM & Unit-Tests

- Was noch? - Nichtfunktionale Tests
 - assertCycles <threshold> - vgl. verbrauchte Zyklen mit <threshold>
 - ... ein paar Macros für die Tests und Mocks...
 - cmp16, cmp32 – vgl. 16/32 Bit-Wert an Adresse
 - set16, set32 – setze 16/32 Bit Wert an Adresse

ASM & Unit-Tests

The screenshot displays the GitHub Actions interface for the repository 'Steckschwein / code'. The 'Actions' tab is active, showing a list of workflow runs for the workflow 'Build and Test SteckOS and Tools'. The interface includes a sidebar with navigation options like 'Code', 'Issues', 'Pull requests', 'Discussions', 'Actions', 'Projects', 'Wiki', 'Security', 'Insights', and 'Settings'. The main content area shows 'All workflows' with a search filter and a list of 52 workflow runs. Each run entry includes a status icon (green checkmark), a workflow name, a description, a link to the workflow file, the event that triggered the run, the status, the time taken, and the actor.

Actions New workflow

All workflows

Build and Test SteckOS and Tools

Management

- Caches
- Runners Beta

All workflows Filter workflow runs

Showing runs from all workflows

52 workflow runs

Event	Status	Branch	Actor
✓ Feature/ssw2.0 Build and Test SteckOS and Tools #53: Pull request #54 synchronize by mlauke	15 hours ago 54s	feature/ssw2.0	...
✓ +update Build and Test SteckOS and Tools #52: Commit 6ff418f pushed by mlauke	15 hours ago 30s	feature/ssw2.0	...
✓ Feature/ssw2.0 Build and Test SteckOS and Tools #51: Pull request #54 synchronize by mlauke	last week 32s	feature/ssw2.0	...
✓ +fix vdp char_out Build and Test SteckOS and Tools #50: Commit e699a96 pushed by mlauke	last week 36s	feature/ssw2.0	...
✓ Feature/ssw2.0 Build and Test SteckOS and Tools #49: Pull request #54 synchronize by mlauke	last week 28s	feature/ssw2.0	...
✓ +prepare rom write Build and Test SteckOS and Tools #48: Commit b6e9bc3 pushed by mlauke	last week 38s	feature/ssw2.0	...
✓ Feature/ssw2.0 Build and Test SteckOS and Tools #47: Pull request #54 synchronize by mlauke	last week 31s	feature/ssw2.0	...
✓ +clean, clean vram Build and Test SteckOS and Tools #46: Commit 5fa1a7b pushed by mlauke	last week 26s	feature/ssw2.0	...

Q&A