AMIGA home schooling

Episode 1

by Gunnar von Boehn

Structure

Episode 1
Das Design des Amigas – **CHIPSET**

Episode 2
Werde ein 68K JEDI Master

Episode 3
Programmiere eine Demo

Das Design des Amiga

Physikalische CHIPS:

- **CPU** ➔ Führt alle Instruktionen aus
- **RAM** ➔ enthält alle Program/Instruktionen, alle Daten, alle Grafiken, alles Audio und den Bildschirm
- **AGNUS** ➔ DMA Kontroller holt die Daten aus dem RAM für die Chips und agiert als Torwächter für die CPU
- **DENISE** ➔ erzeugt das Video-out, enthält die Farbregister, und erzeugt die Sprites
- **PAULA** ➔ DJ; Audio Ausgabe

Das CHIPSET des AMIGA

Die physikalischen Chips : Agnus Denise und Paula enthalten die logischen Amiga Einheiten die wir programmieren und nutzen:

Die Planes, die Sprites, die Audiokanäle, die DMA-kanäle, den Blitter, den Copper

CUSTOM CHIPS

Die Logikeinheiten des Chipset

- 32/256/1024 Farbregister in DENISE
- 6/8/10 Planes in DENISE
- 8 Sprites in DENISE

- 4/8 Audiokanäle in PAULA

- DMA Logik für Planes und Sprites =AGNUS
- Blitter in AGNUS
- Copper in AGNUS

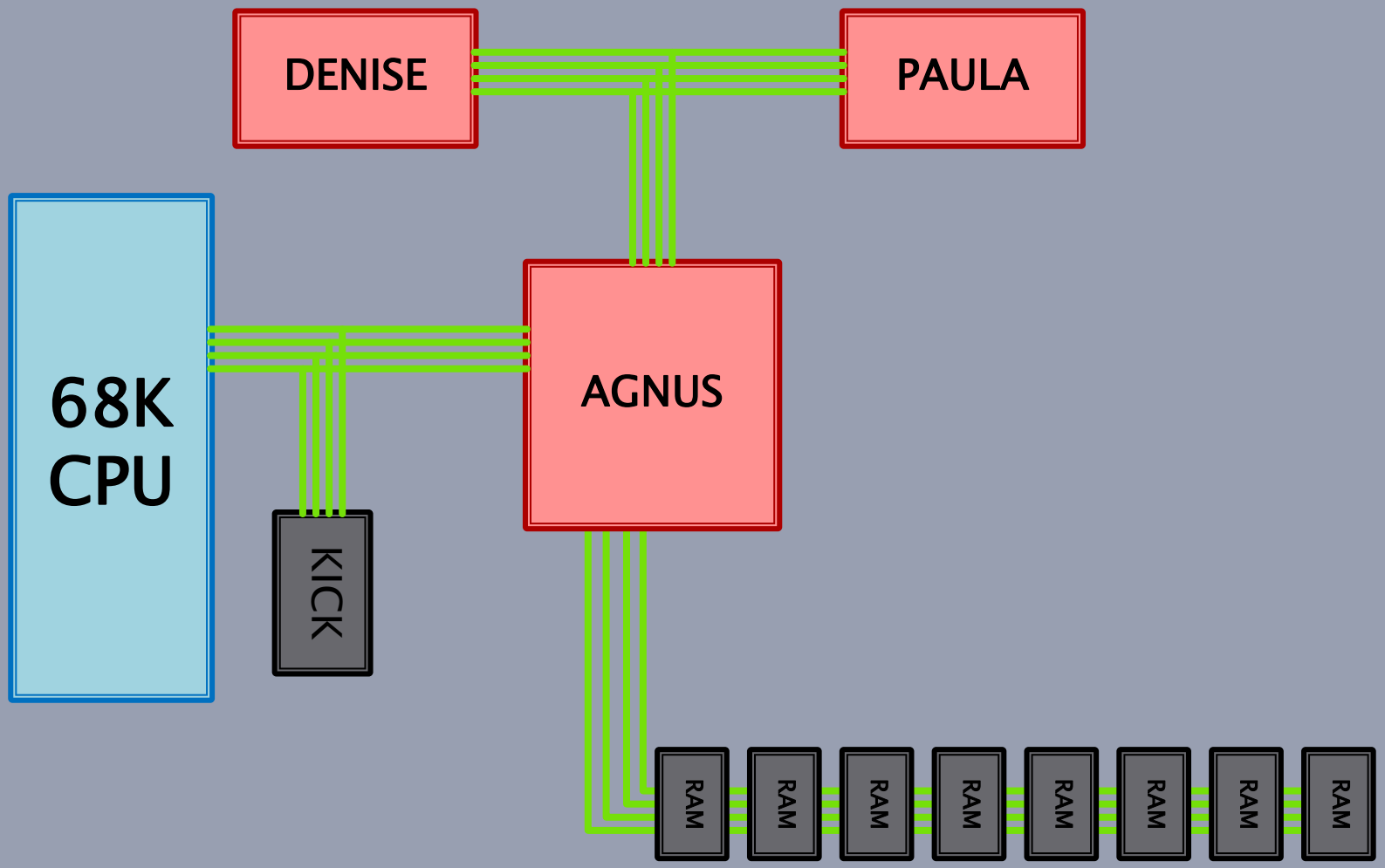
Logikeinheiten und Register

Alle Amigalogikeinheiten sind programmierbar über REGISTERS

➔Register sind keine Speicher welche Nummern speichern können.

➔Die Register des Amigachipset sind in der Memory Addressrange von
$DFF000 – $DFF200
Sie sind 16bit breit

AMIGA 500 Schema



AMIGA home schooling

Episode 2

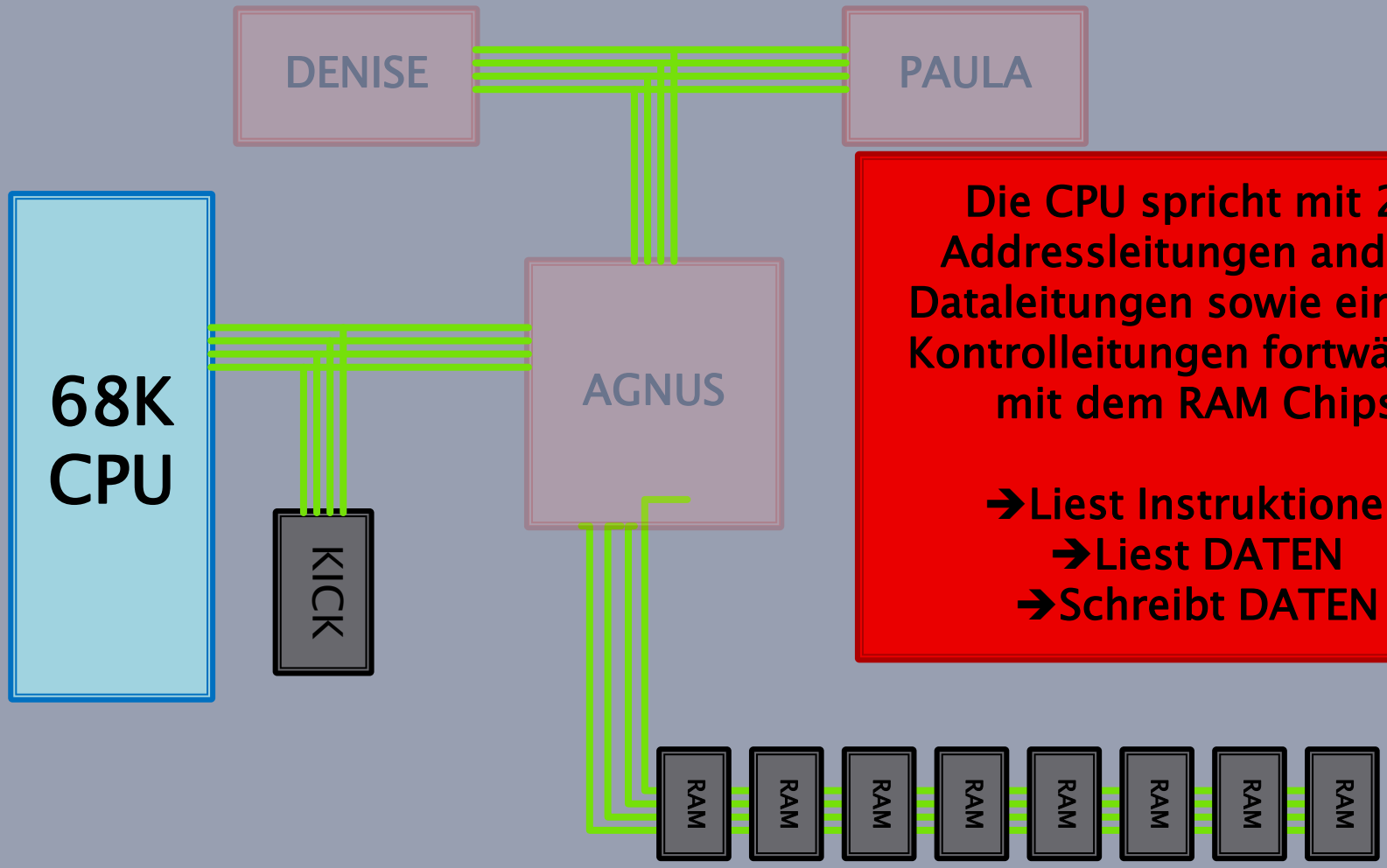
by Gunnar von Boehn

Episode 2: Become a 68k JEDI – Master

Wichtig zu wissen

- Die CPU spricht die ganze Zeit mit dem RAM
- 68K == 2 Operanten Maschiene
- REGISTER

CPU spricht zum RAM



Die CPU spricht mit 24 Addressleitungen and 16 Dataleitungen sowie einigen Kontrolleitungen fortwärend mit dem RAM Chips

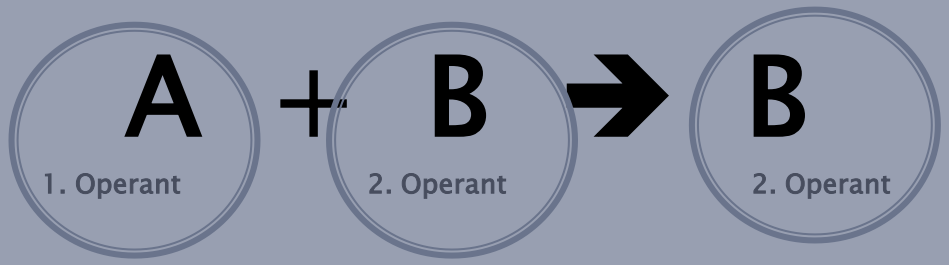
➔Liest Instruktionen
➔Liest DATEN
➔Schreibt DATEN

2 Operaten Maschine

Der 68k ist eine 2 Operanten Maschiene

Jeder Befehl kann bis zu 2 OPERANTEN benutzen

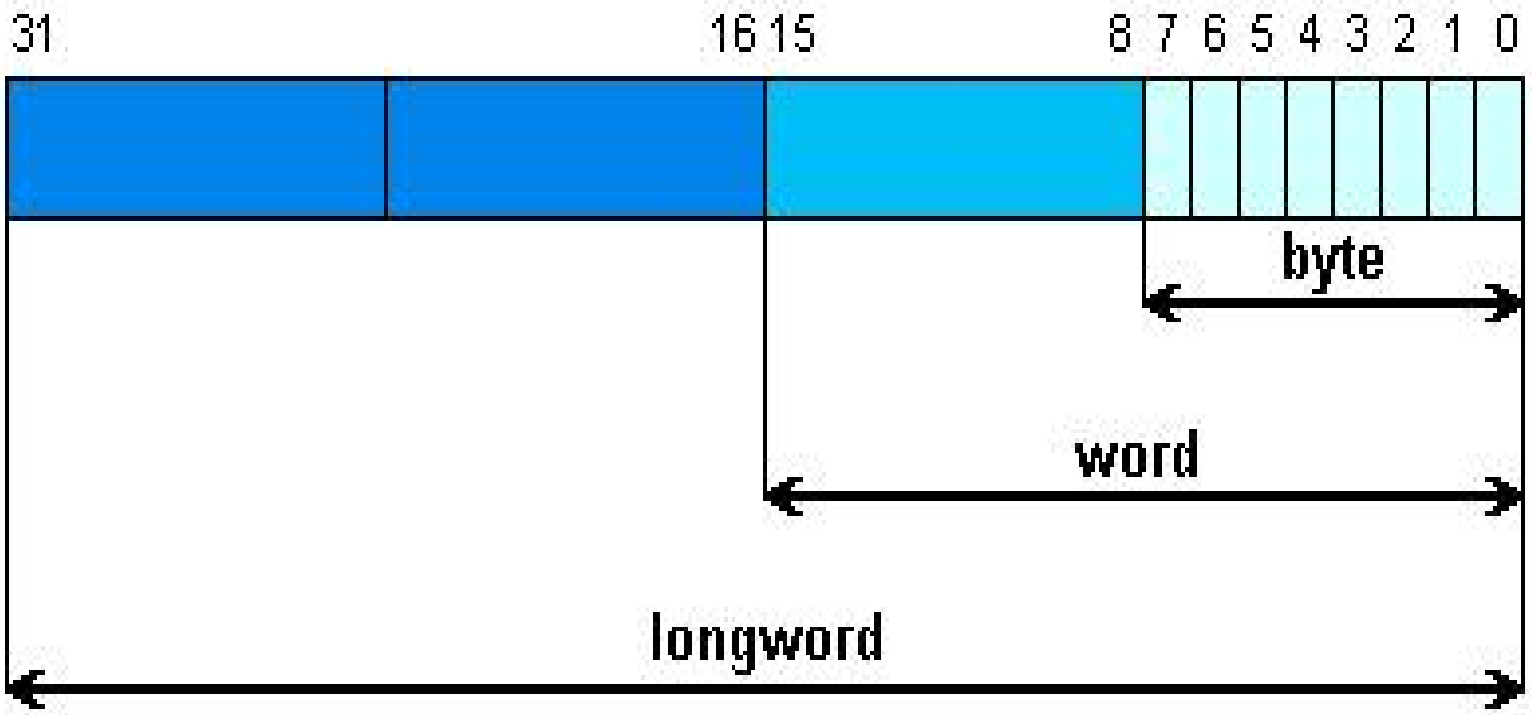
- ➔Source (Quelle)
- ➔Destination (Ziel) (kann auch Quelle sein)



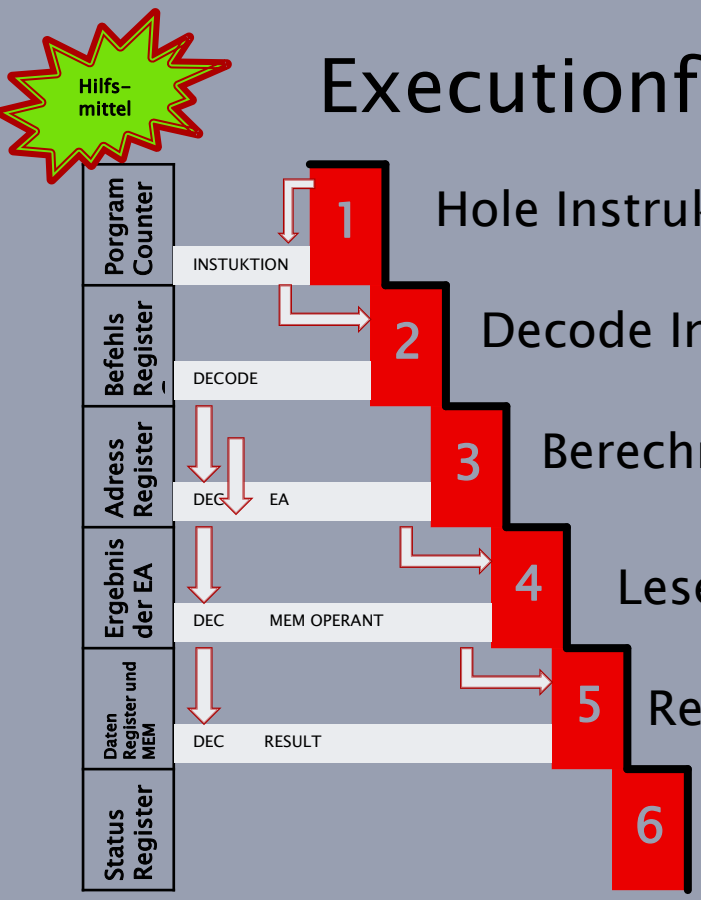
REGISTER

- REGISTER können Zahlen/Werte speichern
- PC = PROGRAM COUNTER
- ADDRESS REGISTER
  (A0, A1, A2, A3, A4, A5, A6, A7) (Apollo=16)
- DATEN REGISTER
  (D0, D1, D2, D3, D4, D5, D6, D7) (Apollo=32)
- SR = STATUS REGISTER

REGISTER



Die CPU PIPELINE



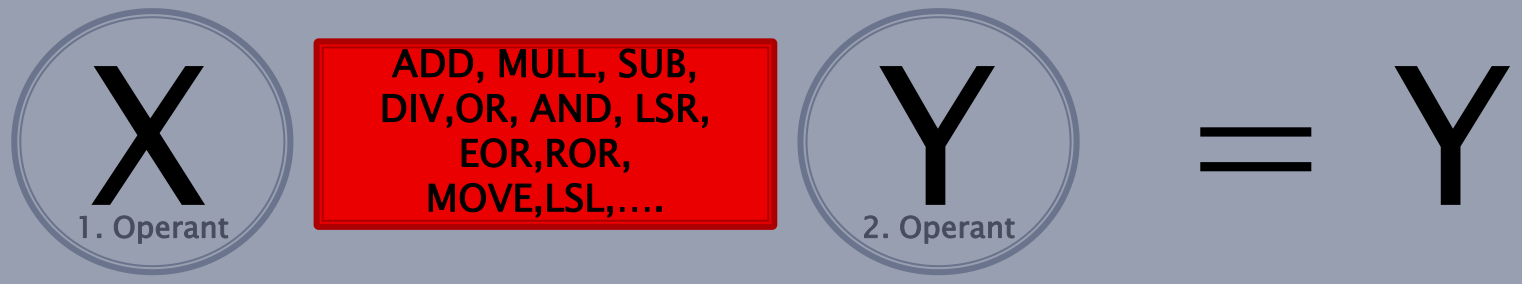
Executionflow der 68K Pipeline

1. Hole Instruktion ➔ FETCH UNIT <- RAM
2. Decode Instruktion ➔ DECODE UNIT
3. Berechne Memory Adresse ➔ EA
4. Lese DATA ➔ DATA CACHE <- RAM
5. Rechen die Mathe ➔ ALU
6. Schreibe Ergebnis zurück -> RAM

Was macht die ALU

ALU = Arithmetic Logic Unit

Die ALU ist die Arithmetic und Logic-Einheit, sie macht die eigentliche „Arbeit“ der Instruktionen



EA UNIT

- #100 Zahl (immediate)
- D0 Daten Register
- $1200 Inhalt einer Speicherstelle
- (A0) Memore-Pointer
- (A0)+ Mem -Pointer + increment
- -(A0) Mem -Pointer , decrement

100(A0) Mem -Pointer+Offset

10(A0,D0*4) Mem -Pointer + Index

Beispiele

- move.l #123,D0
- add.l #10,D0
- move.w #$000F,$DFF180
- move.w D0,$DFF180

Vergleich

- add.l #100,$8000

68000	68020/30	68040	68060	68080
32	16	3	3	1

Mini Programme

- Ein Regenbogen

```
        move.w #$FFFF,D0
LOOP
        move.w   D0,$DFF180
        dbra     D0,LOOP

        rts
```

Mini Programme

```
LOOP
        clr.w       $DFF180             ; Schwarz
        move.w #$0F00,$DFF180   ; ROT
        btst        #6,$BFE001          ; Mausknopf
        bne         LOOP

        rts
```

Wichtigsten Befehle

- move kopiere
- add
- sub
- mul
- div
- cmp vergleiche

Wichtigsten Befehle

- cmp
- bne/beq/bgt/blt …
- dbra
- bsr / rts
- lea

Siehe auch

- www.apollo-core.com CODING