

- Aus welchen Komponenten besteht ein Computer?
- Lassen sich die Komponenten sinnvoll gruppieren?

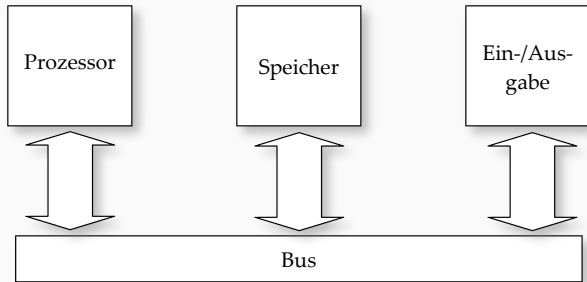
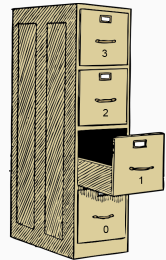


Abbildung 1: Schematische Darstellung der von-Neumann-Architektur

- Enthält Programmcode **und** Daten
- Besteht aus einzelnen Speicherzellen
 - Jede Speicherzelle hat eine Adresse, über die auf ihren Inhalt zugegriffen werden kann
 - Übliche Größe: 1 Byte pro Zelle



- Der Prozessor führt den Programmcode aus
 - Der Code liegt in **Maschinensprache** vor
- Dabei werden Daten aus dem Arbeitsspeicher geladen, verändert und wieder gespeichert

Maschinencode (1)

- Befehle, die vom Prozessor direkt ausgeführt werden können
 - Programmcode einer Hochsprache (z. B. C++ oder Java) muss erst von einem **Compiler** in Maschinencode übersetzt werden
- Für Menschen ohne Hilfsmittel vermutlich nicht verstehbar:

```
88 58 b7 53 99 21 03 3e e0 8f 14 48 0e 7e 49 c1 a8 05 4b 35 38 ca 81 b9 85 66 cd 84 1d 69 1b 77 1d ca 86 d4 ce 53 20 d7 de d6 80 9d 00 4e 47 5b 2c 09 81 ad b8 0c cb 25 31
e8 8d 3b 1a 6f af 67 e8 3d f0 f2 09 7e 85 43 c1 ca 77 2f 4f af a6 ff cc 99 7b be a3 bf a6 20 0e cf 3a 6e b5 95 d7 dd c9 ef 19 fd 55 fd 01 6c 05 ce 7b 62 c7 9a 89 f7 e7
03 e0 24 84 b9 00 1c 50 72 d9 af cf 84 1d 1b 72 7c c1 cf 99 7c 36 90 e7 5f fe db 19 7c bd 99 17 49 3c 0b fc aa 75 d7 01 3b 1f 33 f1 fe b7 01 bf 6a f1 35 9e 1f de 9e 0d 41
93 61 03 d8 18 04 4f 3f c6 8c fc 72 f5 77 59 bd 0c ab a8 53 5d 0c 3d c8 cc ff 33 31 79 72 d5 b8 b9 52 bd 6b bf 26 00 78 7f 82 cd 24 00 3d 04 a0 73 00 4c 75 01 82 00
f6 83 00 58 01 38 02 3b 0e 81 ff 04 1a 83 76 ed ef 54 04 c0 0a 00 43 01 b1 13 80 60 37 ab c1 cc 4a 70 e0 0a da dc f3 0b 2f 02 e8 dd ea 0f 20 80 9b 0c 02 18 70 b1 13 80 2a
fb 61 b8 66 09 0b 4f 61 6b 6e 87 85 3b a6 03 21 00 68 0a 0c 1e 42 00 19 14 18 22 50 64 60 9b a3 16 b4 6a d0 61 83 95 4b 60 3e a1 59 c3 f3 4a 68 0c 27 8c a5 aa df 11 21 67
e1 5d af 0e a0 11 70 66 15 36 49 40 9b 06 3c c1 dd c0 e9 d1 59 6e c8 c5 c8 2c 88 81 09 79 1b f4 06 42 cf 0c d3 e1 0b a8 92 a1 d9 f2 e3 ec f9 33 5b 78 dd bc 02 14 87 fd 5e
b2 f1 5e b2 d1 ec c2 3f 7c df b0 7f 2a 00 be 57 d3 10 4a e8 3d 55 67 f7 93 ec a3 b7 0f a4 24 57 09 3b c4 db 88 cf 99 b8 a3 f4 9d aa d1 73 3c 3f 65 bc 49 f8 b9 63 ba 6c da
31 49 3c 3f e0 9f ee 78 f3 02 9a 32 9f f2 df 3c 1f 79 fe f7 34 5f 78 7d eb b9 cc 09 4c e1 e4 a7 ec f7 02 3f 1b 7d 08 2e ec c4 fd f4 fc b4 20 28 0c c6 fe 03 f2 aa e5 01 c8
5c f9 fb b4 08 79 32 28 41 2a a1 00 36 01 fc b0 b9 d3 11 06 9f a6 1b 80 10 ff a3 3a 3d 07 60 13 c0 49 87 00 3a 94 a3 c8 01 90 00 08 7c 63 ab b6 20 24 72 08 80 57 82 9c 64
00 08 0c 1c 32 ca 2b b0 01 db dc ed 4d 00 dd 1f fb ad 84 bd bc c3 26 80 91 17 26 01 c0 fb 5f d5 36 a9 74 57 fa e0 76 39 38 0e ab 21 78 f5 b6 bc d0 d5 d0 72 71 02 2c ae db
f3 40 06 f9 34 90 80 31 6f 42 00 29 90 10 c6 73 c8 a1 08 57 d7 40 06 24 04 d7 a8 18 cc d7 76 b4 88 c2 7d c6 cc 63 c9 ab be 77 4a 02 70 00 dd 00 b5 11 d7 6c c6 d5 21 b8
7a 0c 57 9b c7 cb e3 f7 53 a4 c0 13 72 63 31 06 62 8f 7e a2 d3 5b cf 58 9d 25 19 98 05 9f 3a a9 87 2a 80 63 46 e0 ab cd 3c 04 bd 3f 00 73 91 87 32 0d 7c 17 fc 0c 5b 38
38 a4 66 f6 f7 a8 06 1d 53 a2 f3 b4 67 fa 4a 7e ed f9 4d 8f fe 34 80 6a 32 40 c6 a4 1d b3 f9 04 2d 3d fe ac 6f 09 7a 93 c8 a3 a4 a7 b1 ab 4f cf f3 9b 84 9f 07 fc 0c 5b 38
b9 7e 35 e0 e3 80 9f 5e 9f a0 4f f6 f5 24 fd 0c f8 b9 f0 83 0b 4d 58 fe 63 cf ff 53 01 b9 f0 fe f3 a4 74 8b 06 ff 6a 8c f5 ac c0 89 be 45 98 ea e3 ca ef 02 54 3f 32 30 72
cc d2 df 22 7c 96 24 80 8d 20 06 2a 80 dd c8 f2 37 aa 10 40 13 c0 31 0b fc 24 81 25 05 ab 94 d7 ff 60 7a 2a 22 02 44 00 32 04 30 04 60 2a 03 1c 08 e2 63 6d b0 00 98 8b 40
8c 75 7d e4 d7 32 39 e1 21 90 00 a6 3f cc 04 50 f0 56 09 6d 10 bc ff 32 c6 dd 01 2b b7 e6 04 ad da 96 13 bc 6a 3b 2e 4d 93 c1 b6 3c aa 02 cf 14 21 f8 92 02 06 cc 0b 8c b9
6a c1 a8 86 b5 3b 0a 73 9f 32 93 5b f8 be 47 a4 b5 db ca 66 d1 c3 5a 2b 8d f3 79 f7 be f9 c8 6a d3 9a ab ec cb f3 f5 f6 de 9a ab bd b2 da a0 c3 c5 1a aa af 1f 5e 98 ca c0
b0 82 f9 05 86 1a c0 db 6b 85 e1 99 cf a7 d7 57 23 bf ac 08 38 e6 76 03 d2 eb 5b a0 57 3c d6 89 3c 8c 91 d9 ea 6b 4e dd f1 97 ec 33 b2 5f f7 e8 33 7b ef f1 fc 73 54 dd
1f 23 b8 04 22 62 f3 71 48 d8 d1 d8 af 4f 00 e6 69 3d 78 7b c7 9b db 40 77 ab ff 7a 74 57 c7 f9 5a cf 35 a0 b7 3c bf 01 bf 92 fe 06 fc 3c a1 07 36 0a c0 a7 1d 3d 91 60 87
14 2d 95 c7 a3 f2 e4 ba 19 51 f2 58 60 bc 02 3f 6d f2 9f a0 e3 7f 78 7c c7 eb 37 60 bd ff f2 3d 07 a3 0a eb be a9 14 4a d1 3b 90 8f 2d 41 1c fd 5d b5 e7 80 4b 00 c7 11 0a
```

- In der Regel besteht der Befehlssatz eines Prozessors aus vergleichsweise elementaren Instruktionen
- Grobe Einteilung:
 1. Arithmetisch/logische Befehle
 2. Sprungbefehle
 3. Transferbefehle

- Ergänze die Lücken zu den drei Befehlskategorien auf dem Aufgabenblatt

- Gruppenarbeit: Bringt die Schritte der Befehlsverarbeitung in die richtige Reihenfolge

1. Befehl laden
 2. Befehl dekodieren
 3. Operanden laden
 4. Befehl ausführen
 5. Ergebnis speichern
 6. Befehlszähler erhöhen
- Der **Befehlszähler** (auch: **instruction pointer**) ist ein spezielles Register, das die Adresse des als nächstes auszuführenden Befehls enthält
 - Unter einem **Operanden** versteht man einen Wert, auf den ein Befehl angewendet wird

Schematischer Aufbau eines Prozessors

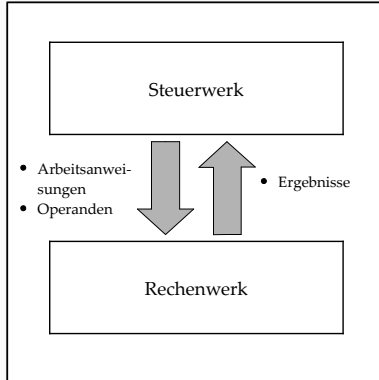


Abbildung 2: Schematischer Aufbau eines Prozessors

- Sogenannte **Register** dienen zur Speicherung von (Zwischen)ergebnissen und weiteren Daten im Prozessor

- Vervollständige Grafik und Text zum Aufbau eines Prozessors auf dem Aufgabenblatt

- Wir betrachten einen fiktiven Prozessor, bei dem jeder Maschinenbefehl aus 4 Bytes besteht:

Byte 0	Byte 1	Byte 2	Byte 3
Opcode	Adresse 1	Adresse 2	Adresse 3

Beispielhafte Befehlsabarbeitung (2)

- Der Opcode ist ein Wert, der angibt, welcher Befehl auszuführen ist.
Für unser Beispiel definieren wir folgende Opcodes:

Op- code	Bedeutung
<hr/>	
10	Addition: Addiere die Werte an Adresse 1 und Adresse 2, speichere das Ergebnis an Adresse 3
20	Multiplikation: Multipliziere die Werte an Adresse 1 und Adresse 2, speichere das Ergebnis an Adresse 3
30	Sprung: Fahre mit der Befehlsausführung an Adresse 1 fort
31	Bedingter Sprung: Fahre mit der Befehlsausführung an Adresse 1 fort, wenn die Werte an Adresse 2 und Adresse 3 gleich sind.

Beispielhafte Befehlsabarbeitung (3)

Adresse	Speicherinhalt	Adresse	Speicherinhalt
0	10	200	7
1	200	201	8
2	201	202	25
3	202	203	3
4	20	204	12
5	203	205	4
6	205	206	18
7	207	207	0
...

- Erster Befehl (ab Adresse 0)

Byte 0	Byte 1	Byte 2	Byte 3
10	200	201	202

- Opcode ist 10 \Rightarrow Addition
- Adresse 1 ist 200 \Rightarrow Erster Operand steht an Adresse 200 (Wert: 7)
- Adresse 2 ist 201 \Rightarrow Zweiter Operand steht an Adresse 201 (Wert: 8)
- Adresse 3 ist 202 \Rightarrow Speichere das Ergebnis $15 (= 7 + 8)$ an Adresse 202
- Erhöhe den Befehlszähler um 4 (da ein Befehl 4 Byte "breit" ist)

- Führe den nächsten Befehl des Programms auf der vorigen Folie aus
- Führe nach dem gleichen Schema das Programm auf dem Aufgabenblatt “Ein Programm in Maschinencode” aus

Modell vs. Realität (1)

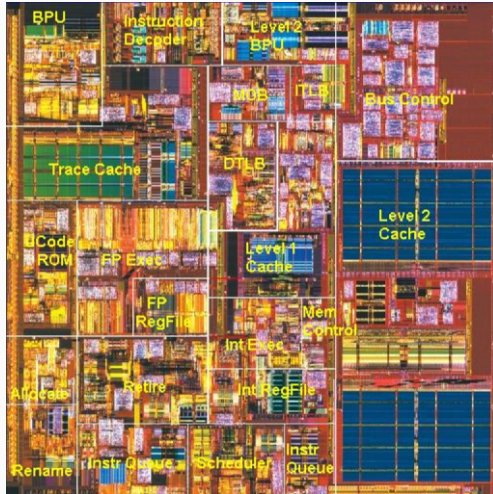


Abbildung 3: Pentium 4 Northwood (Quelle: <http://www.chip-architect.com>)

Modell vs. Realität (2)

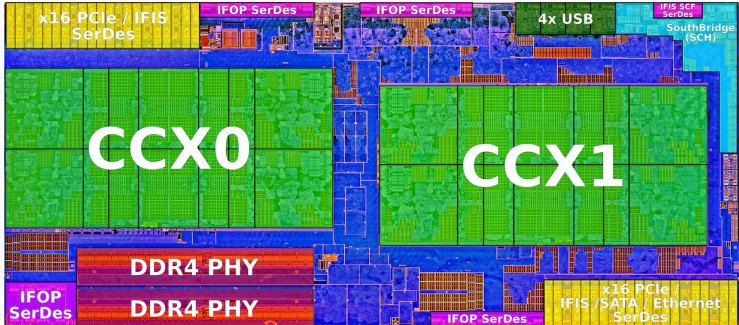


Abbildung 4: AMD Zen 8-Core (Quelle:
https://en.wikichip.org/wiki/File:amd_zen_octa-core_die_shot_%28annotated%29.png)

- Was macht das Rechenwerk, während das Steuerwerk einen Befehl lädt oder dekodiert?
- Was macht das Steuerwerk, während das Rechenwerk rechnet?

Pipelining

- Grundidee: Überlappung der Ausführungsphasen
 - Während der erste Befehl dekodiert wird, wird bereits der nächste geladen usw.

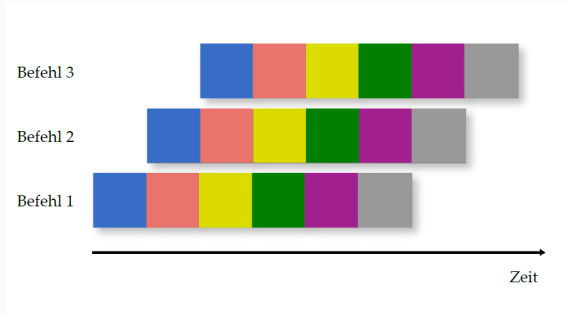


Abbildung 5: Pipelining