Kap.2
Befehlsschnittstelle

Prozessoren,
externe Sicht

- **2.1** elementare Datentypen, Operationen
- **2.2** logische Speicherorganisation
- **2.3** Maschinenbefehlssatz
- **2.4** Klassifikation von Befehlssätzen
- **2.5** Unterbrechungen
- **2.6** Prozesse

- Speicher „spiegeln“ den Zustand eines Rechensystems, bzw. eines Programms

  „Konfiguration“

- Speicher „spiegeln“ den Zustand eines Rechensystems, bzw. eines Programms

  „Konfiguration“

- Wir unterscheiden Hauptspeicher und Registerspeicher.

Hauptspeicher

Hauptspeicher

- „großes“ adressierbares Array

Hauptspeicher

- „großes“ adressierbares Array
- i.d.R. Byte-Adressierung, d.h. jedes Byte besitzt eine eigene, eindeutige Adresse

Hauptspeicher

- „großes“ adressierbares Array
- i.d.R. Byte-Adressierung, d.h. jedes Byte besitzt eine eigene, eindeutige Adresse
- größere Datentypen (z.B. int) sind mehreren Bytes zugeordnet
  (little endian, big endian: siehe unten)

- Aus Peformance Gründen ist Schreiben einzelner Bytes/Bits oft gar nicht möglich

- Aus Peformance Gründen ist Schreiben einzelner Bytes/Bits oft gar nicht möglich

- Ausrichtung an Wortgrenzen:

  Alignment

Alignment

- zusätzliches Problem: ausgerichteter (aligned) und nicht ausgerichteter (misaligned) Zugriff auf Speicher:

Objektadresse	Ausgerichtete Byteabstände	Nicht ausgerichtete Byteabstände
Byte	0,1,2,3,4,5,6,7	ausgeschlossen
Halbwort	0, 2, 4, 6	1, 3, 5, 7
Wort	0, 4	1,2,3, 5,6,7
Doppelwort	0	1,2,3,4,5,6,7

- "misaligned" Zugriffe (die in der Regel erlaubt sind) sparen Speicherplatz aber kosten Zeit bzw. erfordern zusätzlichen Hardwareaufwand

- maximale Größe:
  m Bits für Adresse

- maximale Größe:
  m Bits für Adresse
  - 2^m adressierbare Speicherzellen

- maximale Größe:
  m Bits für Adresse
  - 2^m adressierbare Speicherzellen
  - 16-bit Adressen bei DEC PDP-11

- maximale Größe:
  m Bits für Adresse
  - 2^m adressierbare Speicherzellen

  - 16-bit Adressen bei DEC PDP-11
  - 24-bit Adressen bei IBM Großrechnern

- maximale Größe:
  m Bits für Adresse
  - 2^m adressierbare Speicherzellen

  - 16-bit Adressen bei DEC PDP-11
  - 24-bit Adressen bei IBM Großrechnern
  - 64-bit bei DEC Alpha, SuperSPARC

Wie sind die Bytes numeriert?

Wie sind die Bytes numeriert?

- little endian
  am wenigsten signifikanter Teil enthält die niedrigste Byte Adresse

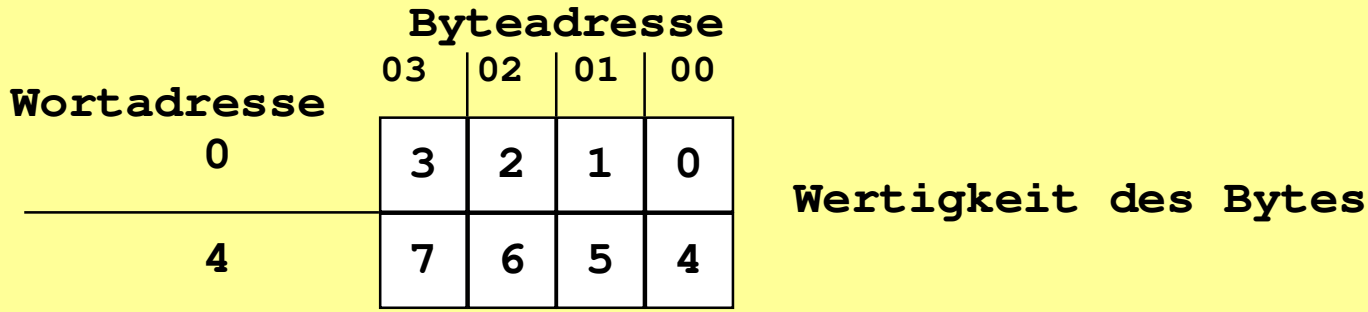
Wie sind die Bytes numeriert?

- little endian
  am wenigsten signifikanter Teil enthält die niedrigste Byte Adresse

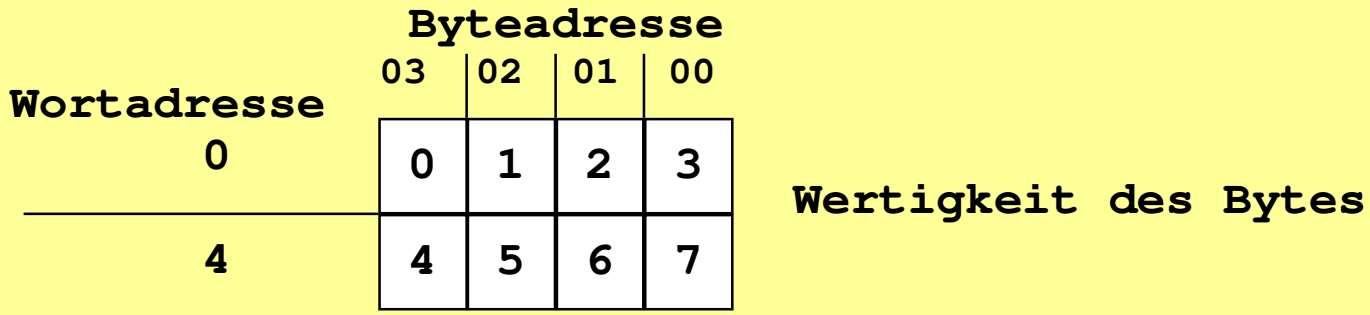
- big endian
  der signifikanteste Teil enthält die niedrigste Byte Adresse

Little Endian - Big Endian

- Little Endian: z.B. 80x86, DEC PDP11/VAX, DEC ALPHA



- Big Endian: z.B. MIPS, MC680x0, IBM 360/370



- von Rechner zu Rechner verschieden
- manche unterstützen beide Formate
- i.d.R. merkt der Benutzer nichts davon, es sei denn man möchte auf einzelne Bytes explizit zugreifen
- bei Strings sind big endians natürlicher, bei Zahlen little endians

- Soviel zur
  logischen (Haupt-)Speichersicht

  strukturelle Eigenschaften später

- jetzt:

Registerspeicher

Registerspeicher

- Viel kleiner als Hauptspeicher, aber auch viel schneller

Registerspeicher

- Viel kleiner als Hauptspeicher, aber auch viel schneller

- kann genutzt werden, um lokale Informationen in Registern zu halten und so die Ausführung von Programmen zu beschleunigen

Designparameter

Designparameter

- Größe
  je nach Architektur unterschiedlich große Registerspeicher

Designparameter

- Größe
  je nach Architektur unterschiedlich große Registerspeicher
  - kleine Registerbank: wenige Bit zur Adressierung, aber für schnelle Ausführung ist dann schnelle Speicherarchitektur notwendig

Designparameter

- Größe
  je nach Architektur unterschiedlich große Registerspeicher
  - kleine Registerbank: wenige Bit zur Adressierung, aber für schnelle Ausführung ist dann schnelle Speicherarchitektur notwendig
  - große Zahl: Befehlsgröße wächst, Ausführungszeit klein

- Funktionalität

- Funktionalität
  - homogen: alle Register haben gleiche Funktionalität
    - einfach zu nutzen, man muß eventuell angeben wie Inhalt zu interpretieren ist

- Funktionalität
  - homogen: alle Register haben gleiche Funktionalität
    - einfach zu nutzen, man muß eventuell angeben wie Inhalt zu interpretieren ist
  - inhomogen: „special purpose register“
    - erleichtert schnelle Ausführung von speziellen Operationen

Registerfunktionen

- Adressregister
- Datenregister
- Spezialregister

Spezialregister

Spezialregister

- **Akkumulator-Register** (ACCU)
  - speichert einen Operanden
  - unterstützt Ausführung von ‘Schieben’

Spezialregister

- **Akkumulator-Register** (ACCU)
  - speichert einen Operanden
  - unterstützt Ausführung von ‘Schieben’
- **Programmzähler** (Befehlszähler)
  - erzeugt durch Inkrementieren aufeinanderfolgende Programmadressen
  - bei Sprungbefehlen neuen Wert laden

Instruktionsregister

- nimmt den aus dem Speicher gelesenen Befehl auf

- **Instruktionsregister**
  - nimmt den aus dem Speicher gelesenen Befehl auf
- **Statusregister**
  - Informationen, die aus Operationen resultieren (Überlauf, Übertrag, Vorzeichen)

- **Stack-Register** (Stapel-Register)
  - aktueller Stand des Stackpointers
  - Verwaltung von Rücksprungadressen und lokalen Variablen bei Unterprogrammen

Beispiele

Beispiele

- **Motorola MC 68000 (1980--)**

Beispiele

- **Motorola MC 68000 (1980--)**
  - 16-bit Prozessor, interne Struktur 32 bit

Beispiele

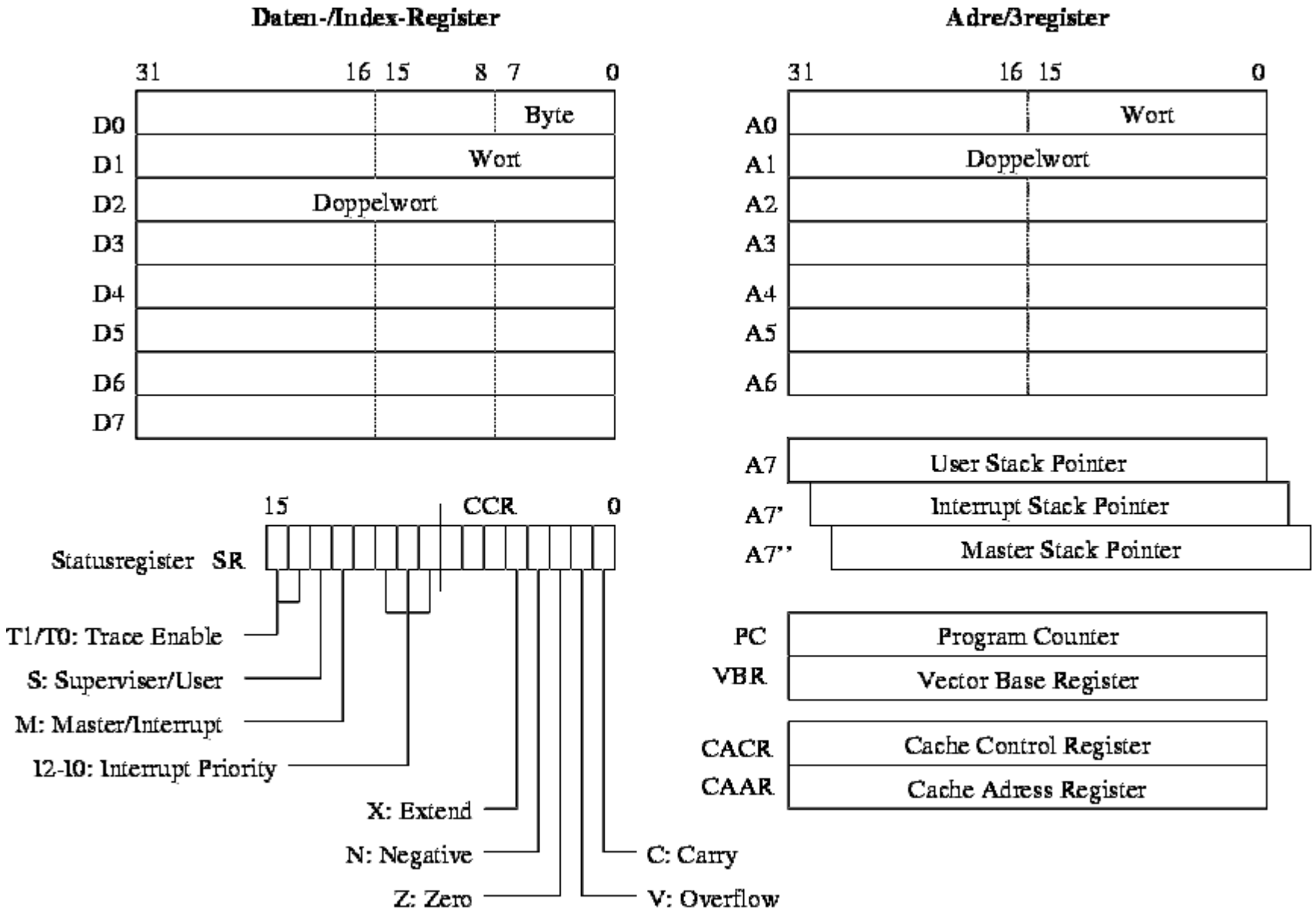
- **Motorola MC 68000 (1980--)**
  - 16-bit Prozessor, interne Struktur 32 bit
  - 16 weitgehend homogene 32-bit Register (8 Daten-, 8 Adress-)

Beispiele

- **Motorola MC 68000 (1980--)**
  - 16-bit Prozessor, interne Struktur 32 bit
  - 16 weitgehend homogene 32-bit Register (8 Daten-, 8 Adress-)
  - 5 Spezialregister

Beispiele

- **Motorola MC 68000 (1980--)**
  - 16-bit Prozessor, interne Struktur 32 bit
  - 16 weitgehend homogene 32-bit Register (8 Daten-, 8 Adress-)
  - 5 Spezialregister
  - optional 8 80-bit floating point Register


Daten-/Index-Register
31
16
15
8
7
0
D0
Byte
D1
Wort
D2
Doppelwort
D3
D4
D5
D6
D7
Adre/3register
A0
Wort
A1
Doppelwort
A2
A3
A4
A5
A6
A7
User Stack Pointer
A7'
Interrupt Stack Pointer
A7''
Master Stack Pointer
15
CCR
0
Statusregister SR
T1/T0: Trace Enable
S: Superviser/User
M: Master/Interrupt
I2-I0: Interrupt Priority
X: Extend
N: Negative
Z: Zero
C: Carry
V: Overflow
PC
Program Counter
VBR
Vector Base Register
CACR
Cache Control Register
CAAR
Cache Adress Register

MIPS I/II

MIPS I/II

- 32-bit RISC Prozessor

MIPS I/II

- 32-bit RISC Prozessor
- 32 weitgehend homogene 32-bit Register (bis auf Register mit Nr. 0 und 1)

MIPS I/II

- 32-bit RISC Prozessor
- 32 weitgehend homogene 32-bit Register (bis auf Register mit Nr. 0 und 1)
- 32 32-bit floating-point-Register (einzeln für single precision, paarweise für double precision)

- **Intel 80386/486 (1985-90)**

- **Intel 80386/486 (1985-90)**
  - 32-bit Prozessor

- **Intel 80386/486 (1985-90)**
  - 32-bit Prozessor
  - 16 Register, 10 32-bit, 6 16-bit
    jedes Register hat (mehr oder weniger) spezielle Aufgaben

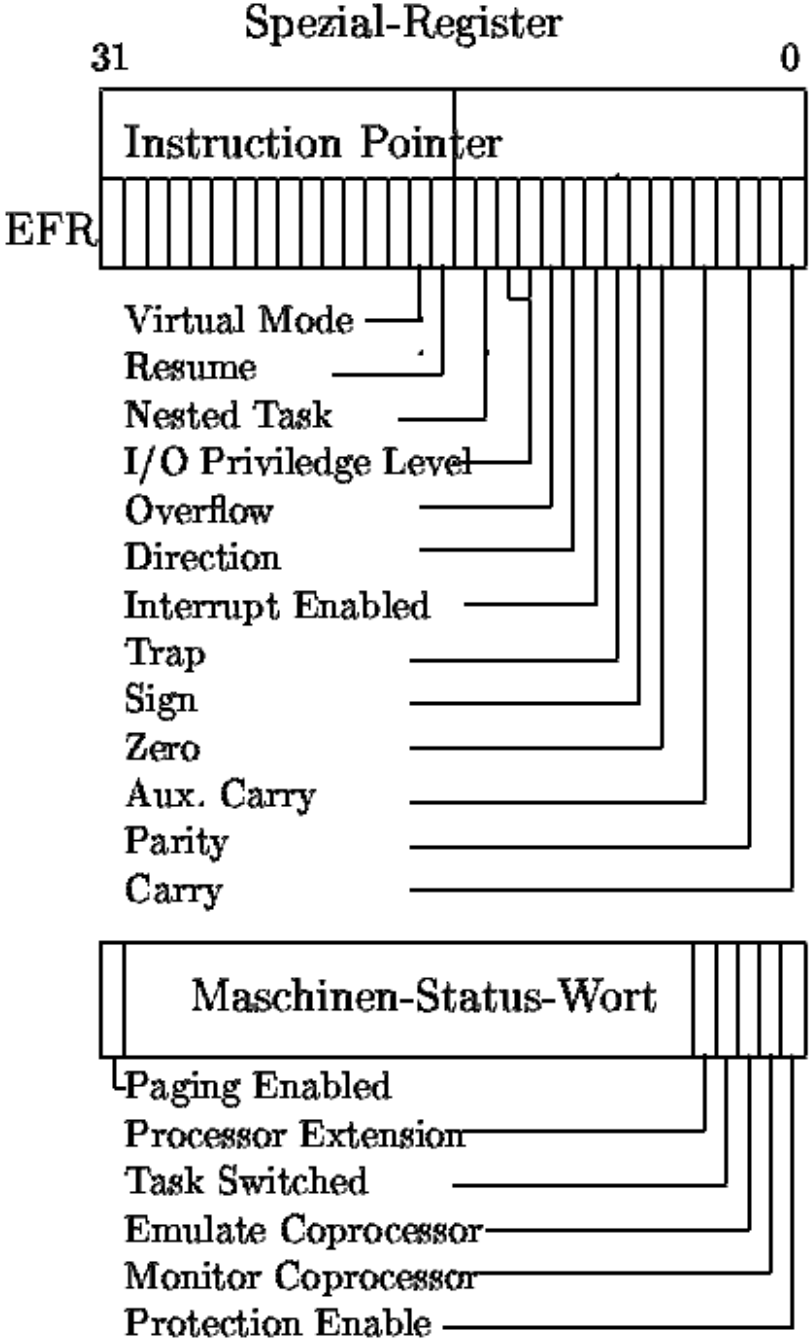
Intel 80386/486 (1985-90)

- 32-bit Prozessor
- 16 Register, 10 32-bit, 6 16-bit
  jedes Register hat (mehr oder weniger) spezielle Aufgaben
- 8 80-bit floating-point-Register

Allgemeine Register

	31	0
EAX	Arithm. Ergeb.	AX (AH | AL)
EDX	und Ein/Ausgabe	DX (DH | DL)
ECX	Zähl-Register	CX (CH | CL)
EBX	Basis-Register	BX (BH | BL)
EBP	Basis-Register	BP
ESI	Index-Register	SI
EDI	Index-Register	DI
ESP	Stack-Pointer	SP

Segmentregister

CS	Code-Segment
SS	Stack-Segment
DS	Daten-Segment
ES	Daten-Segment
FS	Daten-Segment
GS	Daten-Segment

Spezial-Register