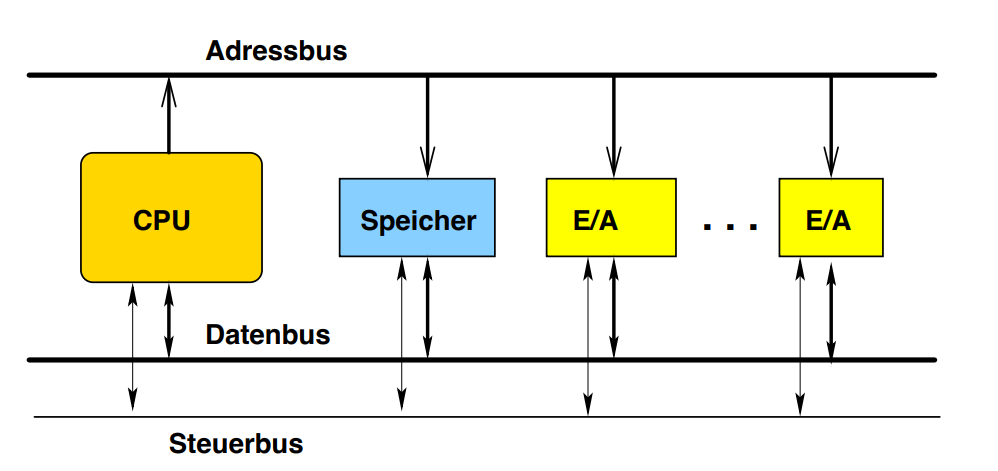
**Hardware Grundlagen**

Von-Neumann-Rechner

- Problemunabhängige Maschinenarchitektur  
- revolutionär: Programmgesteuerter Universalrechner   
(Daten und Programm gleichzeitig im Speicher)



Komponenten

Prozessor

- führt Befehle aus und verändert Daten

Speicher

- Feld von gleichartigen Speicherzellen, die Werte aufnehmen können

Ein-/Ausgabegeräte

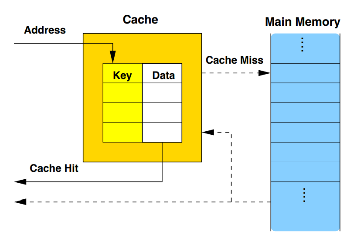
Speicher

Hauptspeicher (Main Memory)

ROM (Read Only Memory)  
 - Inhalte auch ohne Strom gespeichert  
 - Enthält z.B. BIOS

RAM (Random Access Memory)

- Speichern und Auslesen von Befehlen und Daten  
 - enthält pro Problem eine Befehlskette (Programm)

 Cache-Speicher

- schneller Assoziativspeicher  
 - Nach Anlegen eines Schlüsselwertes werden alle Cache-Lines gleichzeitig nach  
 dessen Auftreten durchsucht

Bestandteile des Prozessors

Rechenwerk (ALU) Erlaubt Datenmanipulation z.B. Rechenoperationen

Steuerwerk Koordiniert Ausführung der Instruktionen

Register schneller Mehrzweckspeicher  
 - Mehrzweckregister (Rechnen und Adressieren)  
 - Gleitpunktregister (Gleitkommarechnungen)  
 - Steuerregister (Steuer- und Zustandsinformationen)

Betriebszustände des Prozessors

Maschinenzustand AN/AUS/BOOTEND

Funktionszustand definiert durch gültigen Registersatz

Privilegierungszustand USER/SYSTEM (Schutzkonzept)

Befehlszyklus

- Sequenziell  
- Sprungbefehl (JUMP)  
- Prozedur-/Methodenaufruf  
- Unterbrechung (Interupt)

Unterbrechungskonzept

Ursachen

programmbezogen (synchron) Adressfehler, Division durch Null,   
 (Software-Interrupt, Trap) System-Call, …

systembezogen (asynchron) Disk-Interrupt, Timer-Interrupt, …  
(Hardware-Interrupt)

Wirkung

- unterbricht den sequenziellen Befehlszyklus

- Interrupt-Vektor enthält Einsprung Adressen der Behandlungsroutinen

- nach Abarbeiten einer asynchronen Unterbrechung bzw. von System-Calls wird das Programm normal fortgesetzt

Ablauf

Interrupt-Handler-Routine (liegt im BS-Kern)

Bei Annahme einer Unterbrechung:  
 - Umschalten in den SYSTEM-Modus  
 - sofort weitere Unterbrechungen sperren  
 - Maschinenzustand retten (Statusregister, allgemeine Register)

Beenden

durch speziellen Maschinenbefehl (Return from Interupt (RFI))

Register zurückladen  
 - Unterbrechungssperre aufheben  
 - Rücksprung ins Benutzerprogramm (Rückkehr in den USER-Modus)

nächsten Benutzerbefehl ausführen

Wie kommt man in den SYSTEM-Modus

Gibt keinen expliziten Maschinenbefehl   
 (Stattdessen: System-Calls (von Benutzerprogrammen aus aufrufbare Routinen im BS)

POSIX System-Calls zur Prozessverwaltung

pid = fork() erzeugt identischen Kindprozess  
 Rückgabewert pid - - 0

pid = waitpid (pid, &statloc, options) Warten bis Kindprozess fertig  
 ein Kind: pid = –1, statloc enthält exit-Status

s = execve (name, argv, environp) Ausführen eines Programms aus einer Datei

exit (status) Prozess beenden, exit-Status setzen

s = kill (pid, signal) Signal an einen Prozess senden

System-Calls des Dateisystems

- open, read, write, stat, lseek, close, chmod  
- mkdir, chdir, rmdir  
- link, unlink  
- mount, umount