

# Informatik II

SS 2005

Kapitel 3: Rechnerarchitektur

Teil 2: von Neumann Architektur



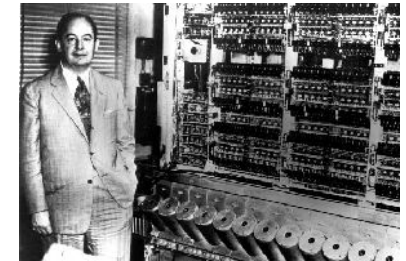
Dr. Michael Ebner  
Dipl.-Inf. René Soltwisch

Lehrstuhl für Telematik  
Institut für Informatik

## 3. Rechnerarchitektur

### Von Neumann Prinzipien (1946)

1. Rechner besteht aus vier Werken:
  - Haupt- bzw. Arbeitsspeicher für Programme und Daten
  - Steuerwerk
  - Rechenwerk (ALU)
  - Ein- / Ausgabewerk (I/O)
2. Programmsteuerung (universelle Hardware)
3. Gemeinsamer Speicher
4. Hauptspeicher besteht aus adressierbaren Zellen
5. Programm besteht aus einer Folge von Befehlen
6. Sprünge sind möglich (bedingte und unbedingte)
7. Speicherung erfolgt binär



## 3. Rechnerarchitektur

### Bestandteile eines von Neumann Rechners

- Haupt- bzw. Arbeitsspeicher für Programme und Daten
  - RAM, ROM
- Busse
  - Datenbus, Adressbus, Steuerbus
- Ein- / Ausgabewerk (I/O)
  - Keyboard, Maus, Scanner, .....
  - Drucker, Bildschirm, .....
  - Festplatten / Magnetbänder
- Steuerwerk (auch Leitwerk genannt)
- Rechenwerk
  - Register
  - Arithmetical Logical Unit (ALU)
- Steuerwerk + Rechenwerk = CPU

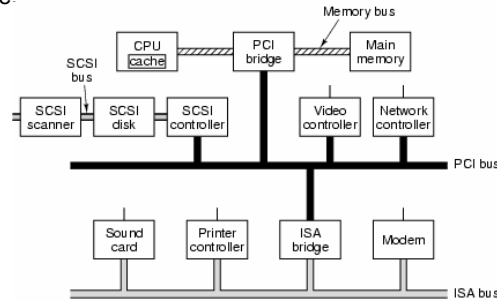
## 3. Rechnerarchitektur

### Speicher

- Speicherwerk (Hauptspeicher, Arbeitsspeicher).
- Persistente Speicher wie Festplatten sowie Caches und Register zählen wir logisch **nicht** zum Speicher.
- Der Arbeitsspeicher besitzt einen eingeschränkten Adressumfang
- Ergänzung durch Hintergrundspeicher (z.B. Plattenspeicher, Magnetbandspeicher, etc.), die persistent sind.
- Die **Zugriffsgeschwindigkeit** zum Arbeitsspeicher sollte der Arbeitsgeschwindigkeit der CPU angepasst sein.
- Je schneller der Speicher desto teurer → Speicherhierarchie
- Für die verschiedenen Einsatzbereiche der Speicher werden unterschiedliche Speicherarten verwendet, die sich unterscheiden hinsichtlich:
  - Speichermedium und physikalischem Arbeitsprinzip
  - Organisationsform
  - Zugriffsart
  - Leistungsparameter
  - Preis

Busse

- Ein gemeinsam genutztes Medium
- Die Anzahl gleichzeitig übertragbarer Bits heißt Busbreite.
- Busse können hierarchisch organisiert sein über Brücken
- CPU direkt am Front Side Bus
- North Bridge
  - AGP / PCI
- South Bridge
  - Seriell / Audio / USB / Firewire



Beispiel eines älteren PCs

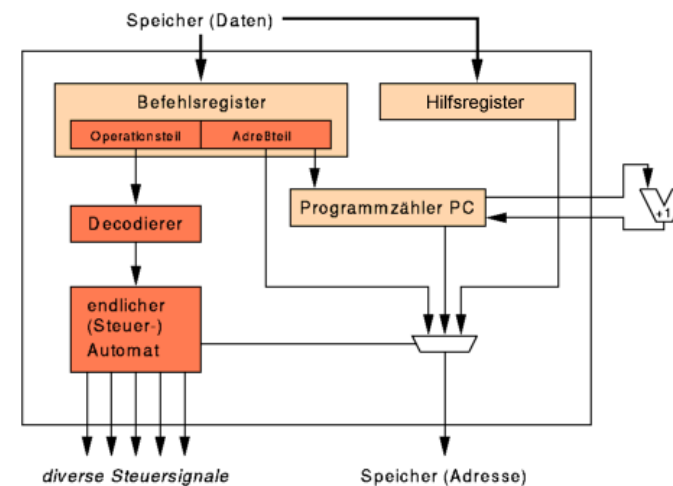
Beispiel Busse

Schnittstelle	Übertragungsgeschw.	Max. Kabellänge
USB 1.0/1.1	1,5 MByte/s	5 m
USB 2.0 Hi-Speed	60 MByte/s	5 m
Firewire	50 MByte/s	4,5 m
Serielle Schnittstelle	0,12 MByte/s	100 m
Parallele Schnittstelle	0,12 MByte/s	5 m
SCSI (SCSI 1)	5 MByte/s	6 m
Fast SCSI (SCSI 2)	10 MByte/s	3 m
Ultra SCSI	20 MByte/s	1,5 m
Ultra Wide SCSI	40 MByte/s	1,5 m
Ultra 2 Wide SCSI	80 MByte/s	12 m
Ultra 3 Wide SCSI	160 MByte/s	12 m

Ein- und Ausgabe (I/O für input / output)

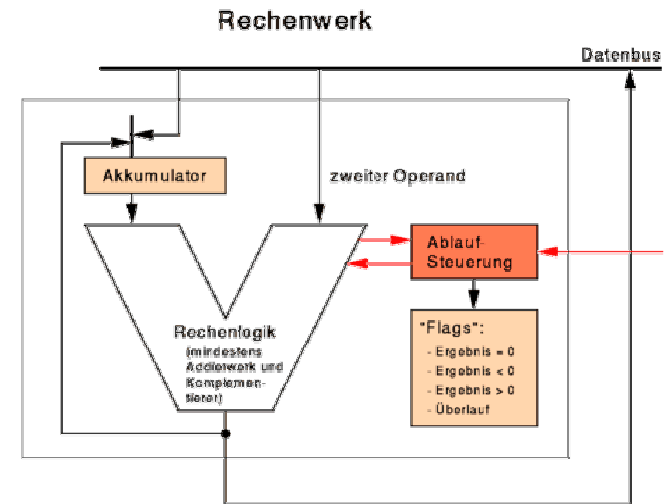
- In den Bereich I/O fallen alle Geräte, die Daten ausgeben oder Eingaben annehmen wie Monitore oder Soundkarten
- Auch die Festplatten sind nach von Neumann I/O Geräte
- DMA – Direct Memory Access
  - Bock Mode bei eigenem Cache
- I/O Geräte sind über den Bus mit Speicher und Prozessor verbunden

Steuerwerk



## Das Leitwerk (Steuerwerk, control unit)

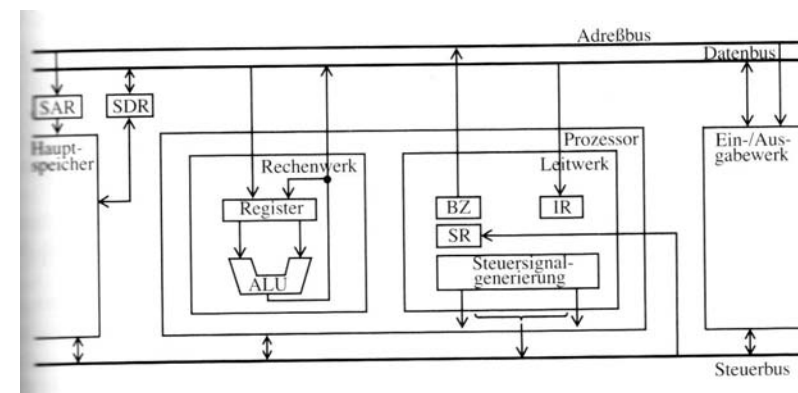
- Das Leitwerk steuert die Arbeitsweise des Rechenwerks durch schrittweise Interpretation der Maschinenbefehle
- Der Befehlszähler (PC) enthält die Adresse des nächsten auszuführenden Befehls; Das Leitwerk erhöht diesen Wert vor jeder Operation um 1. ( $PC := PC + 1$ )
- Das Befehlsregister (IR) enthält den aktuellen Befehl
- Das Statusregister (SR) nimmt Rückmeldungen aus dem System auf
- Das Leitwerk decodiert den Befehl
  - Der Operationsteil (auch Opcode genannt) bestimmt dabei welche Operationen ausgeführt werden sollen
  - Operanden werden durch Angabe von Registern oder Speicheradressen bestimmt
  - Direktoperanden können durch Konstanten angegeben werden
  - Decodierung i.d.R. durch Mikroprogramme
- Das Leitwerk erzeugt die nötigen Steuersignale für das Rechenwerk



## Das Rechenwerk


- Das Rechenwerk bildet zusammen mit dem Steuerwerk die CPU
- Es besteht aus einer (oder mehreren) ALU und Registern
- arithmetische Operationen (Addition, Subtraktion, ...)
- logische Operationen (UND, ODER, NICHT, ...)
- Verschiebe-Operationen
- u. U. Bitmanipulation
- Vergleichs- und Bit-Test-Operationen

## Rechneraufbau



Quelle: Rechenberg, Pomberger  
Informatik-Handbuch S. 301

## Der Befehlszyklus

- 
1. FETCH – Befehlsholphase
  2. DECODE – Dekodierungsphase
  3. FETCH OPERANDS – Operanden nachladen
  4. EXECUTE – Befehl ausführen
  5. UPDATE PC – PC auf den nächsten Befehl zeigen lassen

## Der Befehlszyklus

- Die Gemeinsame Arbeitsweise von Leitwerk und Rechenwerk wird durch den Maschinenbefehlszyklus beschrieben
- Der Befehlszyklus wird von der CPU ständig durchlaufen
- Die Befehle stehen im Speicher
- Das Leitwerk "weiß" jederzeit, welcher Befehl als nächster auszuführen ist
- Die Adresse (= Nummer der Speicherzelle) des nächsten auszuführenden Befehls steht in einem speziellen Register des Leitwerks, dem Befehlszähler (PC)
- Üblicherweise stehen aufeinander folgende Befehle in aufeinander folgenden Speicherzellen, der zuerst auszuführende Befehl hat die niedrigste Adresse
- Zu Beginn des Programms wird der PC mit dessen Startadresse geladen

## Ablauf des Befehlszyklus im Detail

- (1a) Befehlsholphase: Speicherzugriff auf die vom PC angezeigte Adresse.
- (1b) Der Befehl wird in das Befehlsregister des Leitwerks gebracht. Anschließend wird der PC erhöht, er zeigt dann auf den nächsten Befehl. Besteht ein Befehl aus mehreren Speicherworten, so setzt sich diese Phase auch aus mehreren Speicherzugriffen zusammen, bis der Befehl vollständig im IR steht.  
Das Befehlsregister (instruction register IR) ist untergliedert in Opcode-Register (OR, Befehlsregister) und Adress- Register (AR).
- (2) Der Befehl im OR wird decodiert (Befehlsdecoder) und der Ablaufsteuerung zugeführt. Diese kann als **Mikroprogramm** oder **hart verdrahtet** realisiert sein. Die Ablaufsteuerung erzeugt die für die Befehlsausführung nötigen Steuersignale.
- (3) Benötigt der Befehl Operanden, so wird deren Adresse aus dem Inhalt des AR ermittelt.
- (4a) Nun erfolgt der Speicherzugriff auf die so festgelegte Operandenadresse.
- (4b) Die Operanden werden in das vom Opcode spezifizierte Register des Rechenwerks oder in ausgewählte Speicherzellen gebracht.
- (5) Falls die Art des Opcodes weitere Teiloperationen erfordern, werden diese nun ausgeführt. Dabei kann auch der Inhalt des PCs verändert werden (Sprungbefehle, Prozeduraufrufe).

## Programm-Unterbrechungen (Interrupts) – I

- Der Programmablauf muss manchmal unterbrochen werden.
- Lesen von Daten, Terminal Eingaben usw.
- Die Unterbrechung erfolgt durch ein an die CPU gesandtes Signal, das interrupt request (IRQ) genannt wird.
- Die CPU veranlasst das gerade laufende Programm zu unterbrechen und eine Befehlsfolge auszuführen, die auf die Unterbrechung reagiert
- Nach dem Abarbeiten der interrupt service routine (ISR) fährt der Rechner mit der Ausführung des unterbrochenen Programms fort.

### Programm-Unterbrechungen (Interrupts) – II

- Als Reaktion auf eine Unterbrechungsanforderung geschehen zwei Dinge:
  - Retten des aktuellen Programmzustands: Der Maschinenstatus, d.h. der Inhalt der Register der CPU, muss festgehalten werden. Dies geschieht durch Wegspeichern der Registerinhalte des gerade unterbrochenen Programms (Zustandsvektor) auf den Stack.
  - Laden der Register mit dem Zustandsvektor der ISR (z.B. Startadresse der ISR): Der Zustandsvektor steht an einer festgelegten Adresse im Speicher (Interrupt-Vektor). Das Programm wird nun ab der Startadresse der ISR fortgesetzt.
- Die Unterbrechung darf nicht mitten in einer Befehlsausführung erfolgen (undefinierter Prozessorstatus!), sondern erst nachdem der gerade laufende Befehl vollständig abgearbeitet ist, also am Ende des Befehlszyklus.
- Nach Ende der ISR wird der ursprüngliche Zustandsvektor wieder vom Stack geholt und das Programm an der Unterbrechungsstelle fortgesetzt.
- Während des Rettens des Programmzustands und des anschließenden Ladens des Zustandsvektors der ISR darf keine neue Unterbrechungsanforderung auftreten (Zustandsinformation unvollständig!). Der Befehlssatz der Prozessoren enthält daher spezielle Befehle zum Sperren und Freigeben von Unterbrechungsanforderungen

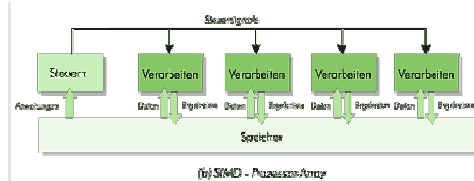
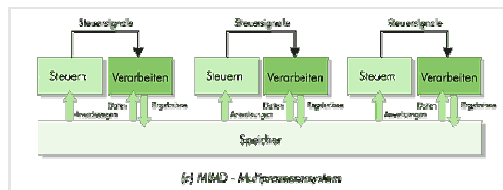
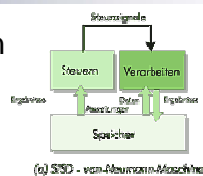
### RISC vs. CISC

- Die Intel x86 CPUs und Motorola 68000 Serie sind CISC Prozessoren – CISC (Complex Instruction Set Computer)
- Der RISC Prozessor ist einfacher aufgebaut (Reduced Instruction Set Computer)
  - weniger Befehle
  - gleiche Länge der Befehle
  - kürzere Ausführungszeit pro Befehl
  - pipelining
  - feste Verdrahtung
  - die arithmetischen Befehle werden nur auf Registern ausgeführt nicht auf den Speicher
  - Auf den Speicher kann nur über Lade- und Schreibbefehle zugegriffen werden
- Das Konzept des Pipelinings wurde verfolgt

### Nicht von Neumann Architekturen

- Klassifikation nach Flynn
  - Nach Daten und Instruktionen

SISD Single Instruction Single Data	MISD Multiple Instruction Single Data
SIMD Single Instruction Multiple Data	MIMD Multiple Instruction Multiple Data



### Was ist möglich?

- Der Earth Simulator:
  - 5,120 (640 8-way Knoten) 500 MHz NEC CPUs
  - 8 GFLOPS per CPU (41 TFLOPS total)
  - 2 GB (4 512 MB FPLRAM modules) per CPU (10 TB total)
  - shared memory in jedem Knoten
  - 640 x 640 crossbar switch zwischen den Knoten
  - 16 GB/s Bandbreite zwischen den Knoten
  - 20 kVA Stromverbrauch pro Knoten



Der Earth Simulator -- NEC SX Earth Simulation Center Yokohama – Japan

<http://www.es.jamstec.go.jp/>  
[www.top500.org](http://www.top500.org)

## Benchmarking

- Analyse der Geschwindigkeit von Rechnern (Performance)
- Was sagen die MIPS (million instructions per second [nicht unser MIPS Prozessor!!!]) und Flops (Floating point Operations) aus?
- So genannte Mixe aus verschiedenen Operationen, um die Leistung zu bewerten.
  - Der LINPACK-Benchmark ist wichtig für die Numerik
  - Es gibt viele weitere: SPAC, WHETSTONE, DHRystone, EDN, SSBA-Suite, ...usw.
- Die Thematik ist sehr komplex und würde alleine eine ganze Vorlesung füllen.

## Ausblick

