Informatik Cheat Sheet

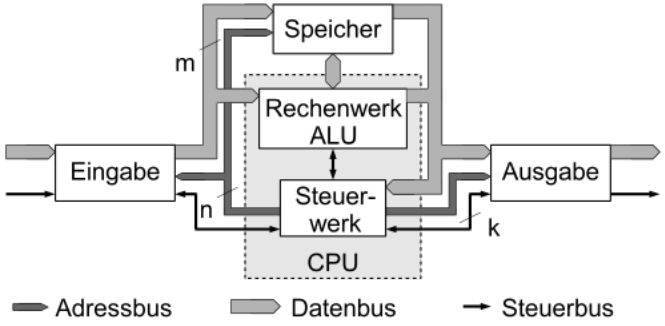
Rechnerarchitektur

Von-Neumann-Rechner

Kennzeichen:

- Der Rechner ist universell oder Turing mächtig
- Programm und Daten liegen im gleichen beschreibbaren Speicher auf den beliebig zugegriffen werden kann (RAM – Random Access Memory). Zwischen Programm und Programmdaten wird nicht unterschieden. Hier ist auch das grösste Risiko. Programm oder Daten können beabsichtigt oder unbeabsichtigt beschädigt werden
- Programm und Daten sind binär codiert
- Steuerung erfolgt über Befehle in einem Programm
- Ohne korrektes Programm ist der Rechner nutzlos
- Programm = Sequenz von Anweisungen/Entscheidungen
- Der Befehlszähler (spez. Speicher) kennt Adresse des nächsten Befehls
- Bedingte Befehle erlauben Entscheidungen über den Fortgang des Programms

Komponenten



Steuerwerk (Leitwerk/*Control Unit*) enthält Befehlsregister, Befehlsdecoder und den Befehlszähler:

- Interpretiert Programmcode
- Erstellt Verbindungen zwischen Speicher und Rechenwerk
- Steuert die Reihenfolge der Programmbefehle

Rechenwerk (Arithmetisch-Logische-Einheit, *Arithmetic Logic Unit (ALU)*) stellt logische und mathematische Operationen (NICHT, ODER, ..., Addition, Multiplikation, ...) zur Verfügung

Steuer- und Rechenwerk zusammen bilden heutzutage zusammen (mit weiteren Komponenten) die *Central Processing Unit (CPU)* (aka. Hauptprozessor)

Speicher(werk) (*Memory*) Speichert alle Daten (alles im selben Speicher). Der Speicher ist in fortlaufend durchnummerierte gleich grosse Zellen unterteilt. Die Nummer entspricht der Adresse der Speicherzelle

Eingabewerk Steuert die Eingabe aller Daten des Rechners

Ausgabewerk Steuert die Ausgabe aller Daten des Rechners

Ein- und Ausgabewerk zusammen sind letztendlich dedizierte Speicher. Sie sind heutzutage in der *Input/Output-Unit (I/O-Unit)* zusammengefasst

Die einzelnen Werke sind über Bus-Systeme (Daten-, Adress- und Steuerbus) miteinander verknüpft.

Register

Register (Teil des Steuer- und Rechenwerks) sind sehr schnelle, sehr kleine Speicher für die Zwischenspeicherung von Daten. Steuer- und Rechenwerk arbeiten i. d. R. mit Werten dieser Register

- Befehlsregister (im Steuerwerk)
- Befehlszähler (im Steuerwerk)
- Zustandsregister (im Steuerwerk)
- Interrupt-Register (im Steuerwerk)
- Akkumulator (im Rechenwerk)
- Statusregister (im Rechenwerk)
- Hilfs- und Arbeitsregister (*General Purpuse Register*, Steuer- und Rechenwerk)

Die Grösse der Register (i. d. R. $n \cdot 8$) charakterisiert einen Rechner:

- Akkumulator/Arbeitsregister definiert Grösse der in einem Zyklus bearbeitbaren Zahlen (Bei 32 Bit $\rightarrow 2^{32}$ mögliche Zahlen)
- Befehlszähler definiert die Grösse des direkt ansprechbaren Speichers
- Befehlsregister definiert die Anzahl möglicher Befehle

Für Register gilt: je grösser, desto teuerer. Üblich sind 8, 16, 32 und 64 Bit Register
Ein Register besteht aus n Flip-Flops, die einzeln angesprochen werden können

Programmablauf

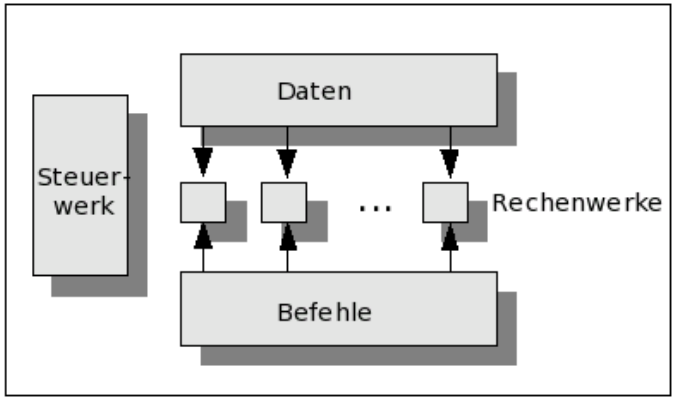
1. Aktuelles Befehlswort aus der Speicherzelle, auf die der Befehlszähler zeigt lesen und and das Steuerwerk übertragen
2. Befehlswort dekodieren und entsprechende Signale auf die Steuerleitungen schalten
3. Operanden aus dem Speicherwerk lesen und in die Register des Rechenwerks übertragen
4. Operation ausführen und Ergebnis in ein Register oder den Speicher schreiben
5. Befehlszähler um eins erhöhen (oder aufgrund eines Sprungbefehls auf einen anderen Wert setzen)

Von-Neumann-Flaschenhals

Die Trennung von Recheneinheit und Speicher führt dazu, dasss Daten sehr häufig übertragen werden müssen. Somit wird der Zuständige Datenbus zum Flaschenhals
(Von-Neumann-Flaschenhals). Auf dem Datenbus werden sowohl Programmcode als auch Daten übertragen und das ganze geschieht sequentiell.
Lösungsansätze:

- Schneller Zwischenspeicher zwischen CPU und Speicher (Cache)
- Getrennte Busse und Zwischenspeicher für Programmcode und Daten
- Vorhersagen von bedingten Programmverzweigungen (*branch prediction*)
- Parallele Strukturen

Harvard-Architektur



- Trennt Programm- und Datenspeicher (inkl. Zwischenspeicher)
- Nutzt getrennte Datenbusse
- kann mehrere Rechenwerke parallel nutzen

Linksverschiebung	Rechtsverschiebung
Programm und Daten werden gleichzeitig geladen	Nichtdeterministisches Verhalten aufgrund unterschiedlicher Laufzeiten
Programmcode kann nicht überschrieben werden	Nicht genutzer Speicherplatz der einen Art kann nicht für die andere genutzt werden
Befehlswortbreite und Datenwortbreite können unterschiedlich sein	

Super-Harvard-Architektur

Auf gemeinsamen Speicher wird über verschiedene Datenbusse parallel zugegriffen

Prozessor

Eigenschaften:

Befehlssatz Die Befehle, die hardwareseitig implementiert sind. Je grösser deren Anzahl, desto kürzere und damit schnellere Programme sind möglich. Ein Compiler macht aus einem Programmbefehl n Prozessorbefehle (Befehlsarchitektur)

Taktzyklus beschreibt die Dauer eines Zykluses (normalerweise in Hz also Taktzyklen pro Sekunde angegeben)

Taktzyklen pro Befehl (CPI) (*clock cyles per instruction*) gibt an, wieviele Taktzyklen ein Befehl im Durchschnitt benötigt (Einzelne Befehle gehen schneller 1 Taktzyklus, andere langsamer $>$ 1 Taktzyklus)

Befehl laden

Befehlszähler addressiert Speicherzelle, der Inhalt wird ins Befehlsregister übertragen (für arithmetisch-logische Operationen). Dazu sind Schaltnetze und Schaltwerke erforderlich.

Schaltnetz besteht ausschliesslich aus logischen Bauelementen (AND, ADD, ...)

- besteht aus 2^n Dateneingängen, n Steuereingängen und einem Ausgang. Mit Hilfe der Steuereingänge wird bestimmt, welches der n-Eingangssignale durchgeleitet wird
- Kombinationen von Schaltnetzen sind wieder ein Schaltnetz

Schaltwerk speichert intern Daten

- mindestens ein Dateneingang, ein Datenausgang und ein Takteingang
- Takteingang bestimmt, wann ein Speicherelement überschrieben wird (gelesen werden kann es immer)
- Unterscheidung zwischen pegel- und flankengesteuerten Schaltwerken (heute: Flanken), d. h. die Spannung muss einen bestimmten Wert übersteigen, damit Schreibvorgänge stattfinden

Allgemein: Schaltwerk $\rightarrow$ Schaltnetz $\rightarrow$ Schaltwerk Bei flankengesteuerten Schaltwerken können die Daten wieder ins selbe Schaltwerk geschrieben werden

Operanden laden, speichern

Mit Hilfe von Schaltnetzen und -werken werden Daten aufgrund der von der ALU berechneten absolute Speicheradresse manipuliert

Ablaufzeit eines Programms P

$$t_P = (\text{Anzahl Befehle}) \cdot \text{CPI} \cdot \text{Zeit pro Zyklus}$$

Die minimale Zykluszeit hängt davon ab, wie lange der Strom entlang des längsten möglichen Datenpfades für einen Befehl braucht. Optimierung muss sich immer nach diesem Befehl richten, auch wenn der Befehl nur selten gebraucht wird. Alle anderen Befehle müssen immer auf den langsamsten aller möglichen Befehle warten.

Load/Store unter Einbezug des Speichers ist wohl immer am langsamsten.

Befehle können gruppiert werden, so dass alle Befehle einer Gruppe etwa gleich lange Pfade haben. Langsamere Befehle beanspruchen dann einfach n Zyklen (so kommt dann der CPI-Wert zustande). Allerdings wird dann die Steuerung und das Design deutlich komplexer.

Pipelining

Mehrere Befehle werden gleichzeitig (überlappend ausgeführt)

Revision: 1.0, Datum: 29. September 2013