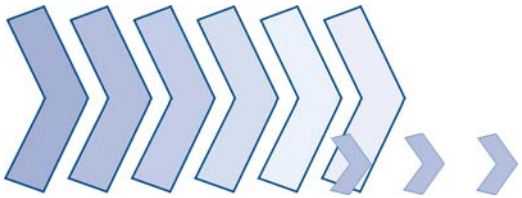
HUMBOLDT-UNIVERSITÄT ZU BERLIN



Technische Informatik 2
Befehlssatz: Eleganz vs. Effizienz

Prof. Dr. Miroslaw Malek

Sommersemester 2007

www.informatik.hu-berlin.de/rok/ca

Thema heute

- Einführung in Befehlssätze
- Grundbefehlssatz
- Befehlsverteilungen
- Entwurfsmethoden

Der Befehlssatz

"Man sollte erwarten, dass die menschliche Sprache direkt die Eigenschaften der menschlichen Intelligenz wiedergäbe, dass die Sprache ein direkter Spiegel der Seele sei, so wie es andere intelligente Systeme nicht könnten."

Noam Chomsky

Der Befehlssatz ist "Ein Spiegel der Computerseele"

Übersicht der Maschinen

Computer	Befehle	Op-code bits
CDC 3600	124	7
IBM 360/370	220-256	8
PDP-11	64	4-10
M6800	84	4-16
TMS990 (TI)	70	
Vax 11 / 750	304	variabel
Motorola 68000	über 130	4-16
Pentium II, III, IV	über 70	variabel
HP PA	170	variabel
UltraSPARC II	~70	variabel

Der minimale Befehlssatz

Van der Poel hat gezeigt, dass der kleinste Befehlssatz aus nur einem Befehl Bn besteht:

SUBTRAHIERE & SPEICHERE & VERZWEIGE, WENN NEGATIV

- Bn a,b,c ist definiert als:
  Subtrahiere den Operanden a vom Operanden b, speichere das Ergebnis nach b zurück und verzweige zur Adresse c, falls Ergebnis negativ.

Mehr Befehle - leichter zu programmieren -
aber komplizierte Kontrollstrukturen

Befehlssatz

↓

Einfachheit der Programmierung

↓

Komplexität der Schaltkreise

Befehlssätze (2)

Einfachheit des Programmierens
vs.
Prozessorkomplexität

Verbesserung der Kosten-Leistungsrate bei einem ”cleveren” Befehlssatz: Der Befehlssatz ist einer der wichtigsten Faktoren der Rechnerarchitektur:

- bestimmt die Verwaltung des Rechners
- bestimmt die Komplexität der Programme
- bestimmt die Komplexität von Schaltungen und Kosten

Befehlssätze (3)

Eleganz	vs.	Effizienz
Vollständigkeit		weniger Befehle = weniger Bits
symmetrisches MOV a, b und MOV b,a		Häufigkeit von Verwendung von Argumenten
Flexibilität (19 LOAD MOVE & 22 BRANCH Befehle auf HP-PA Prozessor)		Bandbreite der Operanden
Allgemeingültigkeit		Verhältnis der funktionalen & nichtfunktionalen Befehle
Mnemonische Signifikanz		sollte verschiedene Hoch-sprachen effizient unterstützen
Konsistenz der Adressierung		Raum und Zeit
Prozessor	**vs.**	**Programmieren**
Einfach		Komplex

Grundbefehlssatz

Typ	Operation	Beschreibung
Datenübertragung	Move (Transfer)	Transfer eines Wortes oder Blockes von der Quelle zum Ziel
	Store	Transfer eines Wortes vom Prozessor zum externen Speicher.
	Load (Fetch)	Transfer von Wort vom ext. Speicher zur CPU
	Exchange	Austausch der Inhalte von Quelle und Ziel.
	Clear (Reset)	Transfer eines Wortes bestehend aus Nullen zum Ziel.
	Set	Transfer eines Wortes bestehend aus Einsen zum Ziel.
	Push	Transfer eines Wortes von der Quelle zum obersten Kellerspeicherplatz (Top of Stack)
	Pop	Transfer eines Wortes vom obersten Kellerspeicherplatz (Top of Stack) zum Ziel.

Grundbefehlssatz (2)

Typ	Operation	Beschreibung
Datenmani-pulation	Shift	Links- (Rechts-) Schieben des Operanden
	Rotate	Links- (Rechts-) Schieben des Operanden auf einem geschlossenen Pfad.
	Convert (Edit)	Änderung des Datenformates, z.B.: Binär nach Dezimal.
Arithmetisch	Add	Berechnung der Summe von zwei Operanden.
	Subtract	Berechnung der Differenz von zwei Operanden.
	Multiply	Berechnung des Produktes von zwei Operanden.
	Divide	Berechnung des Quotienten von zwei Operanden
	Absolute	Ersetze den Operanden durch den Absolutwert
	Negate	Ändere das Vorzeichen des Operanden
	Increment	Addiere 1 zum Operanden
	Decrement	Subtrahiere 1 vom Operanden

Grundbefehlssatz (3)

Typ	Operation	Beschreibung
Logisch	And, Or, Not, Xor, Equivalence	Bitweises Ausführen der logischen Operation
Eingabe- und Ausgabe	Input (Read)	Transfer von Daten vom angegebenen Ein/Ausgabe Port oder Gerät zum Ziel; z.B.: Hauptspeicher oder Prozessorregister
	Output (Write)	Transfer von Daten zum angegebenen Quell-Ein/Ausgabe Port oder Gerät
	Start IO	Transfer von Befehlen zum Ein/Ausgabe Port, um die Ein/Ausgabe-Operation zu initiieren.
	Test IO	Transfer von Statusinformation vom Ein/Ausgabe-System zum angegebenen Ziel
	Halt IO	Transfer von Befehlen zum Ein/Ausgabe Port, um die Ein/Ausgabe-Operation zu beenden.

Grundbefehlssatz (4)

Typ	Operation	Beschreibung
Programmkontrolle	Jump (Branch)	Bedingungsloser Transfer, lade PC mit angegebener Adresse.
	Jump Conditional	Bedingungstest: In Abhängigkeit von der Bedingung lade entweder PC mit angegebener Adresse oder gehe zum nächsten Befehl
	Jump to Subroutine (Call)	Speichere jetzige Programmsteuerungsinfor-mation (PC, status register, usw.) an einen bekannten Ort z.B.: Top of the Stack, und springe zur angegebenen Adresse
	Return	Ersetze Inhalte des PC, Status Register, usw. mit der Information von dem bekannten Ort, z.B.: vom Top of the Stack
	Execute	Hole den Operanden von dem angegebenen Ort, und führe den Befehl aus; der PC wird nicht verändert.

Grundbefehlssatz (5)

Typ	Operation	Beschreibung
Programmkontrolle	Skip	Erhöhe PC, um den nächsten Befehl zu überspringen
	Skip Conditional	Teste angegebene Bedingung. In Abhängigkeit vom Ergebnis erhöhe PC oder tue nichts.
	Test	Teste angegebene Bedingung. Setze Flag(s) abhängig vom Ergebnis.
	Compare	Führe einen logischen oder arithmetischen Vergleich von zwei oder mehr Operanden durch. Setze Flaggen, die von dem Ergebnis abhängig sind.
	Set Control Variables	Große Klasse von Befehlen, welche die Kontrollen für Schutzzwecke einschalten: z.B.: Zeitgeberkontrolle, usw.(oft privilegierte Befehle).

Grundbefehlssatz (6)

Typ	Operation	Beschreibung
Programm-kontrolle	Halt	Anhalten der Programmausführung
	Wait (Hold)	Anhalten der Programmausführung. Testet kontinuierlich auf eine angegebene Bedingung. Falls die Bedingung erfüllt wird, dann wird die Abarbeitung wieder aufgenommen.
	No Operation	Es wird nichts getan. Das Programm läuft weiter.

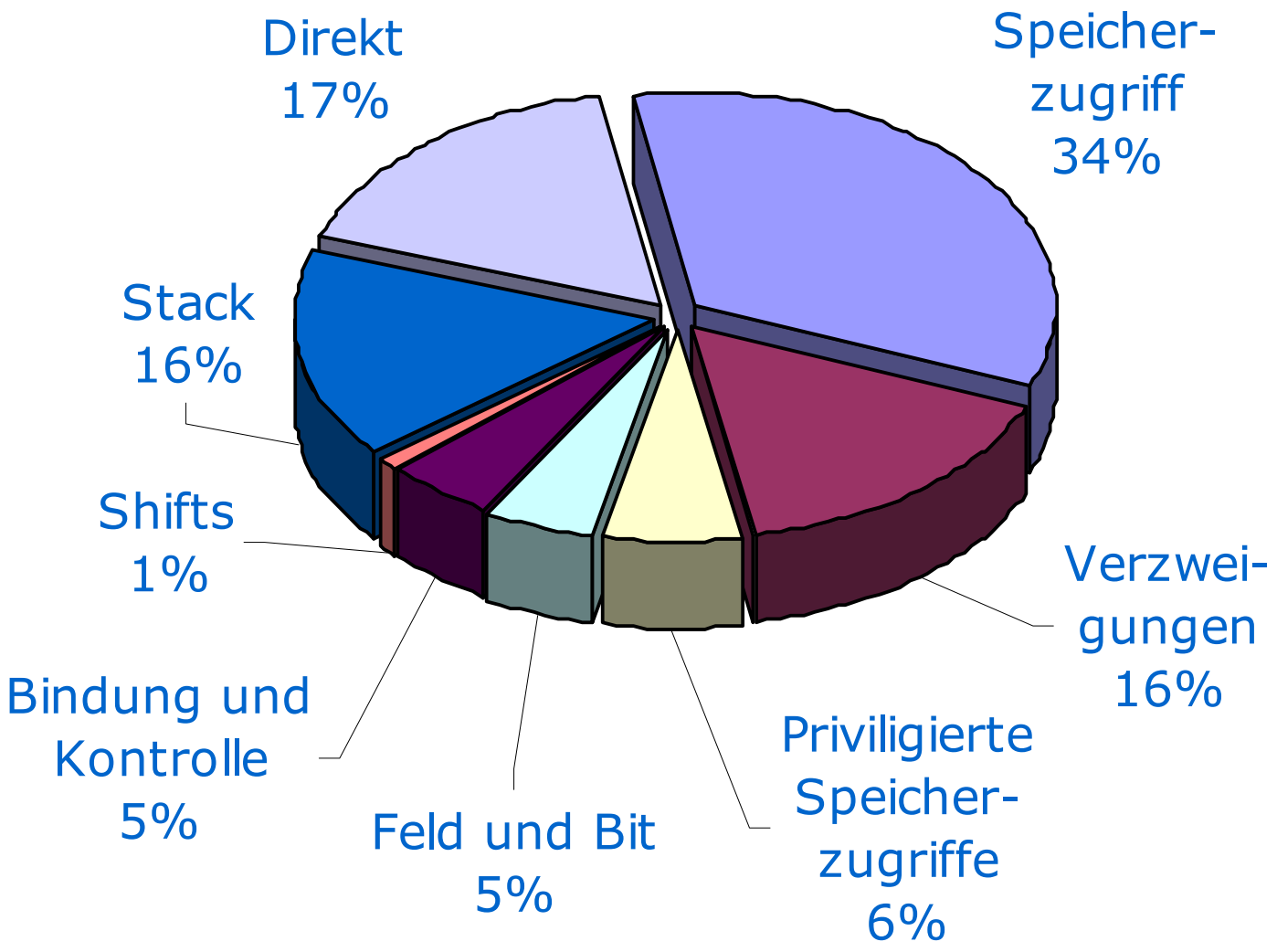
Beispiele eines erweiterten Befehlsatzes

Erweiterter Befehlssatz sinnvoll für spezialisierte Aufgaben wie z.B. 32 Bit Fließkomma-Arithmetik:

Befehlsumsetzung	Zyklenanzahl
Programmierung (normal)	250
Programmierung mit speziellen Befehlen zur Normalisierung und Differenzierung der Exponenten	75
Mikroprogrammiert	25
Festverdrahtet	2

Möglichkeiten: mehrstufige Unterprogrammbindung, automatische Indizierung, Operationen auf Vektoren und Felder & erweiterte Genauigkeiten.

Durchschnittlicher Instruktionsmix



Durchschnittliche Befehlshäufigkeit

LOAD	18%	Lädt ein Wort auf den Top of Stack
BCC	10%	Verzweigt je nach Status-Bedingung
STOR	7%	Speichert ein Wort auf den Top of Stack
LDXI	4%	Lädt immediate Wert in das Index-Register
DUP	3%	Dupliziert den Top of Stack
STAX	3%	Speichert Top of Stack in das Index-Register
BR	3%	Unbedingte Verzweigung
CMPI	3%	Vgl. immediate Wert mit dem Index-Register
LDX	3%	Lädt Index-Register vom Speicher
EXF	3%	Extrahiere Bit-Feld vom Top of Stack

Zusammenfassung des Gibson Befehlssatzes

Befehl	Wahrscheinlichkeit des Auftretens
Hauptspeichertransfer	0.31
Indizierung	0.18
Verzweigung	0.17
Fließkommaarithmetik	0.12
Festkommaarithmetik	0.07
Verschieben	0.04
Rest	0.11

Befehle mit variabler Länge nach Wahrscheinlichkeit

Befehl	Wahrscheinlichkeit des Auftretens p_i	Opcode
I1	0,5	1
I2	0,3	00
I3	0,1	011
I4	0,05	0100
I5	0,05	0101

Die durchschnittliche Anzahl der Opcode-Bits pro Befehl ist:

$$\sum_{i=1}^{5} p_i |c_i| = 1{,}8$$

Befehlssatzauswertung

- Für einen Satz von Benchmarks ist zu bewerten:
  - Programmlänge (# von Befehlen)
  - Ausführungszeit (Geschwindigkeit)
  - Kosten (Programmierungszeit , Speicherplatz)
- Beispiele von Benchmarks:
  - Arithmetik (z.B. Fließpunkt ADD, Matrix Operationen)
  - Zeichensuche
  - Sortieren
  - Bit Manipulation (z.B. bit test, set, reset)
  - Boolesche Matrix Transponierung
  - Code Konvertierung (z.B. ASCII nach Fließpunkt)
  - Speicher Manipulation (z.B. n Worte transferieren)
  - Specint95, Specfp95, Spec2000, Spec2006

Entwurfsmethoden

- Statistische Methoden
- Orientiert an höheren Programmiersprachen
- Methoden: Formale und Systematische
- Gruppierungsmethode
- Familien- oder aufwärtskompatibler Ansatz
- Zweckorientierte Methoden
- Bedingte Interpretationsmethode

Fragen?