Hochschule Bochum
Bochum University
of Applied Sciences

Grundlagen Rechnertechnik

Prof. Dr. Peter Gerwinski

20. November 2012

2.7 Struktur von Assembler-Programmen

Unbedingte Verzweigung
Beispiel: Endlosschleife von „Nothing“

```
        jmp 0
```

Bedingte Verzweigung
Beispiel: Kopierschleife von „Mice“

```
ptr     dat #0
start   mov #12, ptr
loop    mov @ptr, <dest
        djn loop, ptr
        spl @dest
        add #653, dest
        jmz start, ptr
dest    dat #833
        end start
```

2.7 Struktur von Assembler-Programmen

Selbstmodifizierender Code
Beispiel: Selbsterkennung von „Fini“

```
num     dat         #−2
start   mov  num, <pos
        jmp  start
pos     dat         #−3
        end  start
```

3 Architekturmerkmale von Prozessoren

3.1 Speicherarchitekturen

Bezeichnungen

- *Bit* = 0 oder 1 – kleinste Einheit an Information
- *Byte* = Zusammenfassung mehrerer *Bits*
  zu einer Binärzahl, die ein Zeichen (*Character*) darstellen kann,
  häufig 8 Bits (*Oktett*)
- *Speicherwort* = Zusammenfassung mehrerer Bits
  zu der kleinsten adressierbaren Einheit, häufig 1 Byte
- *RAM* = *Random Access Memory* = Hauptspeicher
- *ROM* = *Read Only Memory* = nur lesbarer Speicher

3 Architekturmerkmale von Prozessoren

3.1 Speicherarchitekturen

Verschiedene Arten von Speicher

- *Prozessor-Register*
  können direkt mit ALU verbunden werden,
  besonders schnell (Flipflops),
  überschaubare Anzahl von Registern
- *Hauptspeicher*
  kann direkt adressisert und mit Prozessor-Registern abgeglichen werden,
  heute i. d. R. dynamischer Speicher (Kondensatoren)
- *I/O-Ports*
  sind spezielle Speicheradressen, über die
  mit externen Geräten kommuniziert wird
- *Massenspeicher*
  liegt auf externem Gerät, wird über I/O-Ports angesprochen,
  Festplatte, Flash-Speicher, ...

3 Architekturmerkmale von Prozessoren

3.1 Speicherarchitekturen

- *Von-Neumann-Architektur*
  Es gibt nur 1 Hauptspeicher, in dem sich sowohl die Befehle
  als auch die Daten befinden.
  Vorteil: Flexibilität in der Speichernutzung
  Nachteil: Befehle können überschrieben werden.
  $\longrightarrow$ Abstürze und Malware möglich
- *Harvard-Architektur*
  Es gibt 2 Hauptspeicher. In einem befinden sich die Befehle,
  im anderen die Daten.
  Vorteil: Befehle können nicht überschrieben werden
  $\longrightarrow$ sicherer als Von-Neumann-Architektur
  Nachteile: Leitungen zum Speicher (Bus) müssen doppelt vorhanden sein,
  freier Befehlsspeicher kann nicht für Daten genutzt werden.
- Weitere Kombinationen
  Hauptspeicher und I/O-Ports gemeinsam oder getrennt,
  Hauptspeicher und Prozessorregister gemeinsam oder getrennt

3 Architekturmerkmale von Prozessoren

3.1 Speicherarchitekturen

Beispiele:

- Intel IA-32 (i386, Nachfolger und Kompatible):
  Von-Neumann-Architektur (plus Speicherschutzmechanismen),
  Prozessorregister und I/O-Ports vom Hauptspeicher getrennt
- Atmel AVR (z. B. ATmega):
  Harvard-Architektur (Befehlsspeicher als Flash-Speicher grundsätzlich auch schreibbar),
  Prozessorregister und I/O-Ports in gemeinsamem Adressbereich mit Hauptspeicher
- 6502 (heute: Renesas-Mikro-Controller):
  Von-Neumann-Architektur,
  I/O-Ports in gemeinsamem Adressbereich mit Hauptspeicher,
  Prozessorregister und Hauptspeicher getrennt

3 Architekturmerkmale von Prozessoren

3.2 Registerarchitekturen

- Mehrere Register, einzeln ansprechbar
- *Akkumulator*: Nur 1 Register kann rechnen.
- *Stack-Architektur*: Stapel, „umgekehrte Polnische Notation“

Je nachdem, auf welche dieser Arten die Register eines Prozessors organisiert sind, muß er auf völlig unterschiedliche Weise programmiert werden.

3 Architekturmerkmale von Prozessoren

3.2 Registerarchitekturen

Beispiele:

- Intel IA-32 (i386, Nachfolger und Kompatible):
  Mehrere Register, für verschiedene Zwecke spezialisiert (unübersichtlich),
  Fließkommaregister: Stack-Architektur
- Atmel AVR (z. B. ATmega):
  32 Register
- 6502 (heute: Renesas-Mikro-Controller):
  3 Register: A, X, Y. Nur A kann rechnen $\longrightarrow$ Akkumulator
- Java Virtual Machine (JVM):
  Stack-Architektur

3 Architekturmerkmale von Prozessoren

3.3 Befehlssätze

- *Complex Instruction Set Computer (CISC)*

  Umfangreiche Befehlssätze, mächtige Befehle
  $\longrightarrow$ komfortable manuelle Programmierung in Assembler
  $\longrightarrow$ längere Abarbeitungszeit der einzelnen Befehle

  Realisierung: „Prozessor im Prozessor“ – *Mikroprogramme*

  Beispiel: IA-32

- *Reduced Instruction Set Computer (RISC)*

  wenige, wenig mächtige Befehle
  $\longrightarrow$ Programmierung in Assembler für Menschen unkomfortabel
  $\longrightarrow$ schnelle Abarbeitung der Befehle

  Beispiel: Atmel AVR

- Weitere Befehlssatzarchitekturen: VLIW, EPIC (z. B. IA-64)
- Orthogonaler Befehlssatz

3 Architekturmerkmale von Prozessoren

3.3 Befehlssätze

Von den von C her bekannten Operationen stehen typischerweise nur die Zuweisungsoperationen zur Verfügung: =, +=, −=, ∗=, /=, |=, &=, ~=, <<=, >>=

Verschiedene Struktur der Befehle je nach Registerarchitektur:
Beispiel: `c = a + 2 * b`
Mehrere Register, einzeln ansprechbar
C:

```
R = b;
R *= 2;
R += a;
c = R;
```

Assember:

```
movl  (b), %eax
imull $2, %eax, %eax
addl  (a), %eax
movl  %eax, (c)
```