Systemprogrammierung

Zwischenbilanz

Wolfgang Schröder-Preikschat

Lehrstuhl Informatik 4

18. Juli 2013

Gliederung

Lernziele und Lehrinhalte

Grundlagen von Betriebssystemen

Vorgänge innerhalb von Rechensystemen **ganzheitlich** verstehen

- **Zusammenspiel** { Hardware ↔ Software ⋮ Anwendung ↔ Betriebssystem ⋮ Treiber ↔ Gerät } **begreifen**

Grundzüge imperativer Systemprogrammierung (in C)

- im Kleinen praktizieren ⇝ Dienstprogramme
- im Großen erfahren ⇝ Betriebssysteme

Einführung in die Programmiersprache

Schlüsselwörter

```
auto     break    case     char     const    continue  default  do
double   else     enum     extern   float    for       goto     if
int      long     register return   short    signed    sizeof   static
struct   switch   typedef  union    unsigned void      volatile while
```

Operatoren, Selektoren, Klammerungen und andere „Satzzeichen“

```
!   "   %   &   '   (   )   *   +   ,   -   .   /
:   ;   <   =   >   ?   [   ]   ^   {   }   ~
```

Frage: Was macht dieses C-Programm?

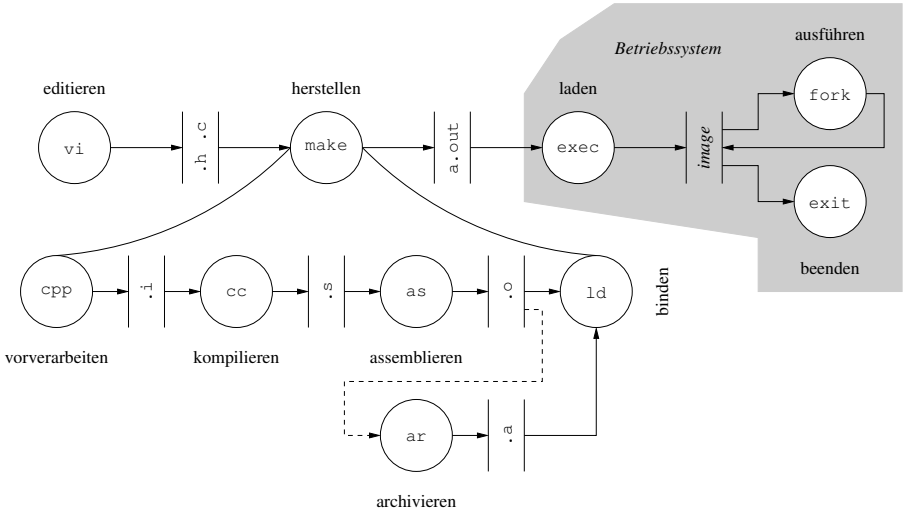
```
#include <stdio.h>

main(){char q=42,n=10,*s="main(){char q=42,n=10,*s=%c%s%c;printf(s,q,s,q,n);}%c";printf(s,q,s,q,n);}
```

Antwort: Es reproduziert sich selbst! (http://www.zyvex.com/nanotech/selfRep.html)

```
wosch@fangorn 41$ gcc magic.c
wosch@fangorn 42$ ./a.out
main(){char q=42,n=10,*s=*main(){char q=42,n=10,*s=%c%s%c;printf(s,q,s,q,n);}%c*;printf(s,q,s,q,n);}
wosch@fangorn 43$
```

Vom Quellprogramm zum Prozess. . .



Motivation

Rückgrat eines jeden Rechensystems

Betriebssysteme sind unerlässliches Handwerkszeug der Informatik

nicht alle müssen ein solches Handwerkszeug bauen/pflegen können

alle müssen jedoch mit dem Begriff/Produkt umgehen können

Betriebssysteme zu verstehen hilft, Phänomene zu begreifen

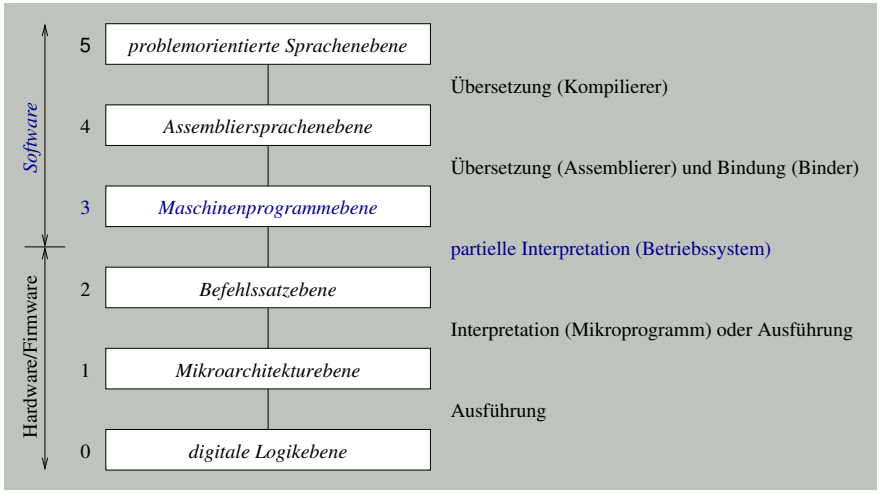
- unterschiedliches Systemverhalten erklären zu können
- Eigenschaften und Fehler auseinanderhalten zu können

Betriebssysteme immer im **Anwendungskontext** beurteilen:

- kein einzelnes System ist für alle möglichen Zwecke optimal geeignet

Strukturierte Organisation von Rechensystemen

Betriebssystem: **abstrakter Prozessor**/**virtuelle Maschine** für Programme der Ebene 3



Unterbrechungen und Ausnahmesituationen

Teilinterpretation

Programmunterbrechungen zeigen Ausnahmebedingungen an und bedeuten die **partielle Interpretation** von Maschinenprogrammen:

Trap synchron, vorhersagbar, reproduzierbar

Interrupt asynchron, unvorhersagbar, nicht reproduzierbar

- macht determinierte Programme nicht-deterministisch
- **Nebenläufigkeit**, **kritischer Abschnitt**

Ausnahmebehandlung bringt Kontextwechsel mit sich, die **abrupte Zustandswechsel** das ausführenden Prozessors bewirken:

- vom unterbrochenen Programm zum behandelnden Programm ↓BS
- vom behandelnden Programm zum unterbrochenen Programm BS↑

Hardware und Software sind (funktional) äquivalent: Emulation

- die Nachahmung der Eigenschaften von Hardware durch Software

Adressraum

Ausführungs- und Schutzdomäne von Programmen

physikalischer Adressraum (Hardware) . . . . . Ebene 2
- ist durch die jeweils gegebene Hardwarekonfiguration definiert
- nicht jede Adresse ist gültig, zur Programmspeicherung verwendbar

logischer Adressraum (Kompilierer, Binder, Betriebssystem) . . . Ebene 5/4/3
- abstrahiert von Aufbau/Struktur des Haupt- bzw. Arbeitsspeichers
- alle Adressen sind gültig und zur Programmspeicherung verwendbar

virtueller Adressraum (Betriebssystem) . . . . . Ebene 3
- auf Vorder- und Hintergrundspeicher abgebildeter log. Adressraum
- erlaubt die Ausführung unvollständig im RAM liegender Programme

Speicher

Zusammenspiel aneinander angepasster Funktionen zu gegenseitigem Nutzen

Laufzeitsystem (bzw. Bibliotheksebene) verwaltet den lokal vorrätigen Speicher eines logischen/virtuellen Adressraums
- Speicherblöcke können von sehr feinkörniger Struktur/Größe sein
  - einzelne Bytes bzw. Verbundobjekte
- Verfahrensweisen orientieren sich (mehr) an Programmiersprachen

Betriebssystem verwaltet den global vorrätigen Speicher (d.h. den bestückten RAM-Bereich) des physikalischen Adressraums
- Speicherblöcke sind üblicherweise von grobkörniger Struktur/Größe
  - z.B. eine Vielfaches von Seiten
- Verfahrensweisen fokussieren auf Benutzer- bzw. Systemkriterien

Datei

Abstraktion von Informationen (über-) tragenden Betriebsmitteln

Aufbewahrungsmittel für zu speichernde Informationen
- kurz-, mittel-, langfristige Speicherung
- bleibende Speicherung (persistente Daten)

Kommunikationsmittel für kooperierende Prozesse
- gemeinsamer (externer) Speicher
- Weiterleitung von Informationen

Abstraktionsmittel für den Betriebsmittelzugang
- Hardware: CPU, RAM, Peripherie, . . .
- Software: Adressräume, Prozesse, . . .

Abbildung symbolische Adresse $\mapsto$ numerische Adresse:
- einen Dateinamen auf auf eine Dateikopfnummer abbilden
- ein Dateiverzeichnis (auch) als Umsetzungstabelle verstehen

Namensraum

Namen Kontexte zuordnen

Namensräumen eine Struktur aufprägen und dadurch einem Namen in „benutzerfreundlicher Weise“ eine eindeutige Bedeutung geben können:

flache Struktur eines einzigen Kontextes

hierarchische Struktur mehrerer Kontexte (d.h. flacher Strukturen)
- **hierarchischer Namensraum**
  - Pfadnamen zur Navigation im Namensraum
  - spezielle Kontexte (UNIX-artiger Systeme)
  - Bindung und Auflösung von Namen
- **Hierarchie von Namensräumen**
  - Montieren von Dateisystemen

Dateisysteme und Namensräume sind (logisch) verschiedene Dinge:
- das eine organisiert den Hintergrundspeicher (zur Dateiablage)
- das andere dient der Identifikation von Objekten (nicht nur Dateien)

Prozess

Abstraktes Gebilde vs. Identität einer Programmausführung

Gewichtsklasse eine Frage der Isolation von Adressräumen

Federgewicht keine Isolation
- der „reine“ Kontrollfluss: Faden

Leichtgewicht vertikale Isolation
- vom Betriebssystemadressraum

Schwergewicht horizontale Isolation
- von allg. Programmadressräumen

Einplanung Reihenfolgen festlegen, Aufträge sortieren
- Ablaufplan zur Betriebsmittelzuteilung erstellen
- Ablaufzustände von Prozessen fortschreiben

- charakteristische Eigenschaften der Einplanung/Einlastung von UNIX

Koordinationsmittel

Sequentialisierung nicht-sequentieller Programme

Semaphor abstrakter Datentyp zur Signalisierung von Ereignissen
- unteilbare Operationen auf eine Koordinationsvariable
  - **P** und **V** manipulieren eine nicht-negative ganze Zahl
- zur blockierenden Synchronisation gleichzeitiger Prozesse

Botschaft Synchronisation kombiniert mit Datentransfer
- Primitiven (Semantiken) zum Botschaftenaustausch
  - {*no-wait*, *synchronization*, *remote-invocation*} *send*
- Rollenspiele bei der Interprozesskommunikation
  - gleich- vs. ungleichberechtigte Kommunikation
- Kommunikationsendpunktadressen und Verbindungen

Betriebsmittel vs. synchrone/asynchrone bzw. blockierende IPC:

konsumierbares Betriebsmittel Nachricht (bzw. Botschaft)

wiederverwendbares Betriebsmittel Nachrichtenpuffer

Stapelbetrieb

Stapelsysteme

abgesetzter Betrieb Satellitenrechner, Hauptrechner
- Entlastung durch Spezialrechner

überlappte Ein-/Ausgabe DMA, *Interrupts*
- nebenläufige Programmausführung

überlappte Auftragsverarbeitung Einplanung, Vorgriff
- Verarbeitungsstrom von Aufträgen

abgesetzte Ein-/Ausgabe *Spooling*
- Entkopplung durch Pufferbereiche

Mehrprogrammbetrieb *Multiprogramming*
- Multiplexen der CPU

- programmiertes dynamisches Laden von Überlagerungen (*Overlays*)

Mehrzugangsbetrieb

Interaktive Systeme

Dialogbetrieb Dialogstationen
- mehrere Benutzer gleichzeitig bedienen können

Hintergrundbetrieb Mischbetrieb
- Programme im Vordergrund starten

Teilnehmerbetrieb Zeitscheibe, *Timesharing*
- eigene Dialogprozesse absetzen können

Teilhaberbetrieb residente Dialogprozesse
- sich gemeinsame Dialogprozesse teilen können

Multiprozessorbetrieb Parallelrechner, SMP
- Parallelverarbeitung von Programmen

- Umlagerung (*Swapping*) kompletter Programme, virtueller Speicher

Echtzeitbetrieb

Zeitabhängige Systeme

- die im Rechensystem verwendete Zeitskala muss mit der durch die Umgebung vorgegebenen identisch sein
- **Zeit ist keine intrinsische Eigenschaft des Rechensystems**

weich auch „schwach“ . . . . . *soft*
- Terminverletzung ist tolerierbar

fest auch „stark“ . . . . . *firm*
- Terminverletzung ist tolerierbar, führt zum Arbeitsabbruch

hart auch „strikt“ . . . . . *hard*
- Terminverletzung ist keinesfalls tolerierbar, Ausnahmefall

- querschneidender Belang der gesamten Systemsoftware + Anwendung

Gliederung

Vertiefung

Ausgewählte Kapitel der Systemprogrammierung

Prozessverwaltung	• Einplanung (klassisch, Fallstudien) • Koroutinen, Programmfäden, Einlastung
Koordination	• ein-/mehrseitig, blockierend/nicht-blockierend • Verklemmungen (Gegenmaßnahmen, Auflösung)
Speicherverwaltung	• Adressräume, MMU (Pentium) • Disziplinen, virtueller Speicher, Arbeitsmenge
Dateiverwaltung	• Organisation des Hintergrundspeichers • Datenverfügbarkeit (RAID)

Und falls wir dann noch Zeit haben. . .

Sicherheit	• Zugriffsmatrix, Befähigung, Zugriffskontrollliste • Bell/LaPadula
Architektur	• Monolith, geschichtetes System, Minimalkerne • Selbst-/Paravirtualisierung