Script generated by TTT

Title: Eingebettete_Systeme (06.11.2012)

Date: Tue Nov 06 16:20:58 CET 2012

Duration: 59:51 min

Pages: 23

Teil I: Betriebssysteme und Systemsoftware

- Prof. J. Schlichter
  - Lehrstuhl für Angewandte Informatik / Kooperative Systeme, Fakultät für Informatik, TU München
  - Boltzmannstr. 3, 85748 Garching

Email: schlichter@in.tum.de

Tel.: 089-289 18654

URL: http://www11.informatik.tu-muenchen.de/

Übersicht

Einführung

Synchronisation und Verklemmungen

Prozess- und Prozessorverwaltung

Speicherverwaltung

Prozesskommunikation

Zusammenfassung

Fragestellungen

Dieser Abschnitt beschäftigt sich mit den Adressräumen für Programme und deren Abbildung auf den physischen Arbeitsspeicher einer Rechenanlage:

Programmadressraum vs. Maschinenadressraum.

Direkte Adressierung, Basisadressierung.

Virtualisierung des Speichers; virtuelle Adressierung, insbesondere Seitenadressierung (nur sehr kurz).

Einführung

Speicherabbildungen

Dieser Abschnitt behandelt einige Mechanismen zur Abbildung von Programmadressen auf Maschinenadressen des Arbeitsspeichers.

Direkte Adressierung

Basisadressierung

Seitenadressierung

Die unmittelbare Nutzung des physischen Adressraums (Arbeitsspeichers) bei der Anwendungsentwicklung ist nicht empfehlenswert. Probleme sind folgende:

- Kenntnisse über Struktur und Zusammensetzung des Arbeitsspeichers notwendig.
- Kapazitätsengpässe bei der Arbeitsspeichergröße.

⇒ deshalb Programmerstellung unabhängig von realer Speichergröße und -eigenschaften.

Adressräume

Organisation von Adressräumen

Fragmentierung

Forderungen an Adressraumrealisierung

Maschinenadressraum

Arbeitsspeicher = Folge von fortlaufend nummerierten Bytes (ab 0). ***Maschinenadresse*** = Nummer des Bytes.

Programmadressraum

Programmadressen = Adressen im Programm (z.B. für Variable).

Die Menge der zulässigen Programmadressen ist der ***Programmadressraum*** . Dieser ist prozessspezifisch, d.h. Programmadressen haben nur Programm-lokale Bedeutung z.B. als Sprungziele.

Speicherabbildung

Abbildung der Programmadressen auf Maschinenadressen (durch CPU).

direkte Adressierung,

Basisadressierung,

Seitenadressierung

Organisation von Adressräumen

Single-Threaded Adressraum

niedrige Adresse — hohe Adresse

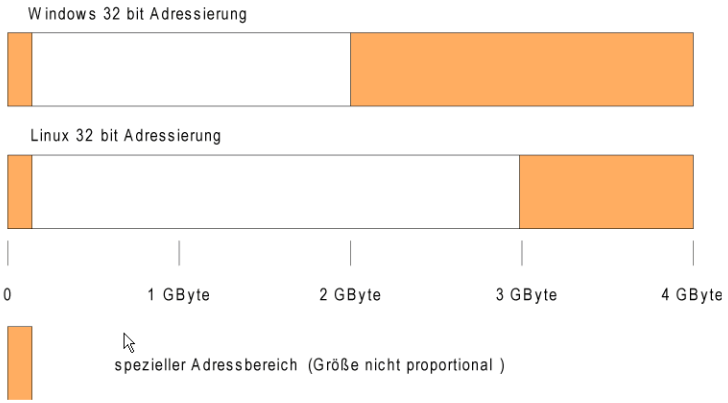
Multi-Threaded Adressraum

niedrige Adresse — hohe Adresse

[Beispiel - Adressräume]

32 Bit große Adressräume

Programmcode, statische Daten, Halde und Laufzeitkeller jeweils in einem für die Anwendung zugänglichen Bereich des Adressraums



Die unmittelbare Nutzung des physischen Adressraums (Arbeitsspeichers) bei der Anwendungsentwicklung ist nicht empfehlenswert. Probleme sind folgende:

- Kenntnisse über Struktur und Zusammensetzung des Arbeitsspeichers notwendig.
- Kapazitätsengpässe bei der Arbeitsspeichergröße.

⇒ deshalb Programmerstellung unabhängig von realer Speichergröße und -eigenschaften.

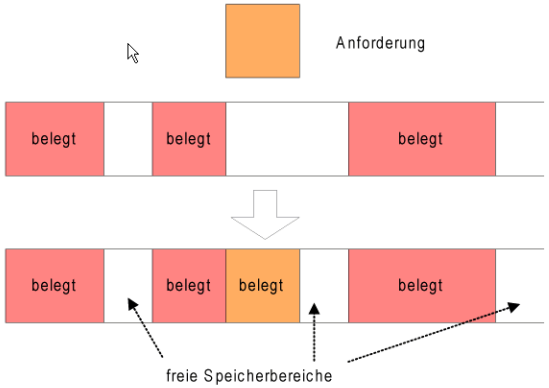
[Adressräume]

[Organisation von Adressräumen]

[Fragmentierung]

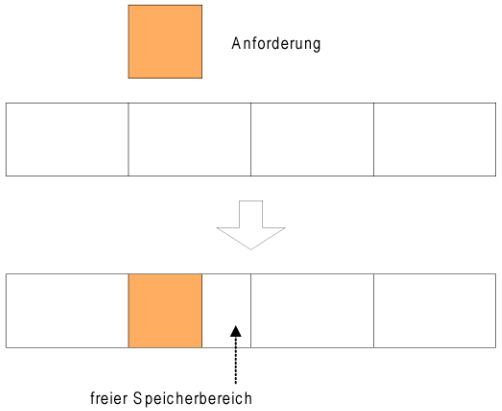
[Forderungen an Adressraumrealisierung]

Es wechseln sich benutzte und unbenutzte Speicherbereiche innerhalb des Adressraums ab. Speicheranforderungen werden jeweils genau erfüllt.



Interne Fragmentierung

Der Speicher ist in Bereiche fester Größe untergliedert und Speicheranforderungen werden nur in Vielfachen dieser festen Grundgröße befriedigt.



Aus der Sicht der Anwendungsprogrammierung können für einen Adressraum eine Reihe wichtiger Forderungen an dessen Realisierung gestellt werden.

- Homogene und zusammenhängende Adressbereiche.
- Größe des genutzten Adressraums unabhängig von der Kapazität des physischen Adressraums (Arbeitsspeichers).
- Erkennen fehlerhafter Zugriffe.
- Erkennen von Überschneidungen zwischen Halde und Keller sowie zwischen mehreren Laufzeitkellern.
- Schutz funktionstüchtiger Anwendungen gegenüber fehlerhaften Anwendungen.
- Kontrollierbares und kontrolliertes Aufteilen der Speicherressourcen auf alle Anwendungen.
- Speicherökonomie, minimale Fragmentierung.

Speicherverwaltung

Fragestellungen

Dieser Abschnitt beschäftigt sich mit den Adressräumen für Programme und deren Abbildung auf den physischen Arbeitsspeicher einer Rechenanlage:

- Programmadressraum vs. Maschinenadressraum.
- Direkte Adressierung, Basisadressierung.
- Virtualisierung des Speichers; virtuelle Adressierung, insbesondere Seitenadressierung (nur sehr kurz).

Einführung

Speicherabbildungen

Dieser Abschnitt behandelt einige Mechanismen zur Abbildung von Programmadressen auf Maschinenadressen des Arbeitsspeichers.

Direkte Adressierung

Basisadressierung

Seitenadressierung

Direkte Adressierung

Programmadresse = Maschinenadresse; Probleme

Verschiebbarkeit,

Programmgröße,

Speicherausnutzung.

Verschiebbarkeit

Programmgröße

Programme größer als Arbeitsspeicher; Programmierer zerlegt Programm in Segmente (Nachladen bei Bedarf). Man spricht von der sogenannten ***Overlay-Technik*** (veraltet).

Forderung

Die Programmgröße sollte unabhängig von der realen Arbeitsspeichergröße ist.

Speicherausnutzung

Programme bestehen aus Modulen, die nur zu bestimmten Zeiten verwendet werden.

Forderung

Arbeitsspeicher beinhaltet nur die momentan bzw. in naher Zukunft notwendigen Ausschnitte des Programms. Nutzen der ***Lokalitätseigenschaft*** von Programmen:

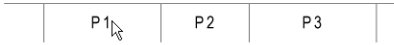
durch Datenstrukturen z.B. Arrays, oder

Programmstrukturen: Prozeduren, Schleifen.

In einem Mehrprozesssystem sind i.d.R. mehrere Programme im Arbeitsspeicher. Bei direkter Adressierung werden die Programme beim Laden fixiert und müssen dann bis zu ihrem Ende an derselben Stelle bleiben.

Problem

Externe Fragmentierung des Arbeitsspeichers. Sei der Arbeitsspeicher zunächst lückenlos mit Programmen P1, P2, P3 gefüllt.



Nach Beendigung des Programms P2 entsteht eine Lücke.

Forderung

In Mehrprogramme/Mehrprozesssystemen sollten daher Programme ***verschiebbar*** sein.

Programmadresse = Maschinenadresse; Probleme

Verschiebbarkeit,

Programmgröße,

Speicherausnutzung.

Verschiebbarkeit

Programmgröße

Programme größer als Arbeitsspeicher; Programmierer zerlegt Programm in Segmente (Nachladen bei Bedarf). Man spricht von der sogenannten ***Overlay-Technik*** (veraltet).

Forderung

Die Programmgröße sollte unabhängig von der realen Arbeitsspeichergröße ist.

Speicherausnutzung

Programme bestehen aus Modulen, die nur zu bestimmten Zeiten verwendet werden.

Forderung

Arbeitsspeicher beinhaltet nur die momentan bzw. in naher Zukunft notwendigen Ausschnitte des Programms. Nutzen der ***Lokalitätseigenschaft*** von Programmen:

durch Datenstrukturen z.B. Arrays, oder

Programmstrukturen: Prozeduren, Schleifen.

Generated by Targeteam

Basisadressierung

Die Basisadressierung hat eine einfache Abbildungsvorschrift:

Maschinenadresse = Basisadresse + Programmadresse

Die Basisadresse ist programmspezifisch.

Die Programmadressen aller Programme beginnen jeweils mit Null.

Der Arbeitsspeicher besteht aus zwei Klassen:

Belegtbereiche : Speicherbereiche sind Programmen zugeordnet. Verwaltung in Belegtliste.

Freibereiche : Speicherbereiche, die momentan keinem Programm zugeordnet sind, d.h. frei sind. Verwaltung Freibereichsliste.

Speicherverwaltungsstrategien

Aufgabe: Finden eines zusammenhängenden Arbeitsspeicherbereichs, der groß genug ist, um das Programm zu speichern.

first-fit

Durchsuche die Liste der Freibereiche vom Anfang an und nimm den ersten passenden Frei-Bereich: Spalte nicht benötigten Speicher ab und füge ihn als freien Bereich in die Freibereichsliste ein.

next-fit

Durchsuche die Liste der Freibereiche nach first-fit, jedoch beginne die Suche dort, wo man bei der letzten Suche aufgehört hat.

best-fit

Durchsuche die Liste der Freibereiche vom Anfang an und nimm den passenden Frei-Bereich, der die Speicheranforderungen des Programms am besten erfüllt: Spalte nicht benötigten Speicher ab und füge ihn als freien Bereich in die Freibereichsliste ein.

worst-fit

Durchsuche die Liste der Freibereiche vom Anfang an und nimm den Frei-Bereich, der die Speicheranforderungen des Programms am schlechtesten erfüllt: Spalte nicht benötigten Speicher ab und füge ihn als freien Bereich in die Freibereichsliste ein.

Buddy-Systeme

Speicheranforderungen werden in Größen von Zweierpotenzen vergeben, d.h. eine Anforderung wird auf die nächste Zweierpotenz aufgerundet

Anforderung 280 Bytes ⇒ Belegung von 512 Bytes = 2^9

am Anfang besteht der Arbeitsspeicher aus einem großen Stück.

Speicheranforderung von 2^k : ist Speicherbereich dieser Größe vorhanden, dann Unterteilung eines Speicherbereichs der Größe 2^{k+1} in 2 Speicherbereiche der Größe 2^k Bytes.

Freigabe eines Speicherbereichs von 2^k : falls auch entsprechendes Partnerstück frei ist, dann Verschmelzung der beiden Speicherbereiche zu einem Speicherstück der Größe 2^{k+1} .

Speicherverwaltung

Fragestellungen

Dieser Abschnitt beschäftigt sich mit den Adressräumen für Programme und deren Abbildung auf den physischen Arbeitsspeicher einer Rechenanlage:

Programmadressraum vs. Maschinenadressraum.

Direkte Adressierung, Basisadressierung.

Virtualisierung des Speichers; virtuelle Adressierung, insbesondere Seitenadressierung (nur sehr kurz).

Einführung

Speicherabbildungen

Dieser Abschnitt behandelt einige Mechanismen zur Abbildung von Programmadressen auf Maschinenadressen des Arbeitsspeichers.

Direkte Adressierung

Basisadressierung

Seitenadressierung

Die virtuelle Adressierung wurde Ende der 50er Jahre eingeführt. Ziel ist

Virtualisierung des Speichers,

Verstecken von realen Beschränkungen, wie Speichergröße,

Speicher als sehr großes Feld gleichartiger Speicherzellen zu betrachten.

Die Seitenadressierung ("paging") ist die Grundform der virtuellen Adressierung.

Ansatz

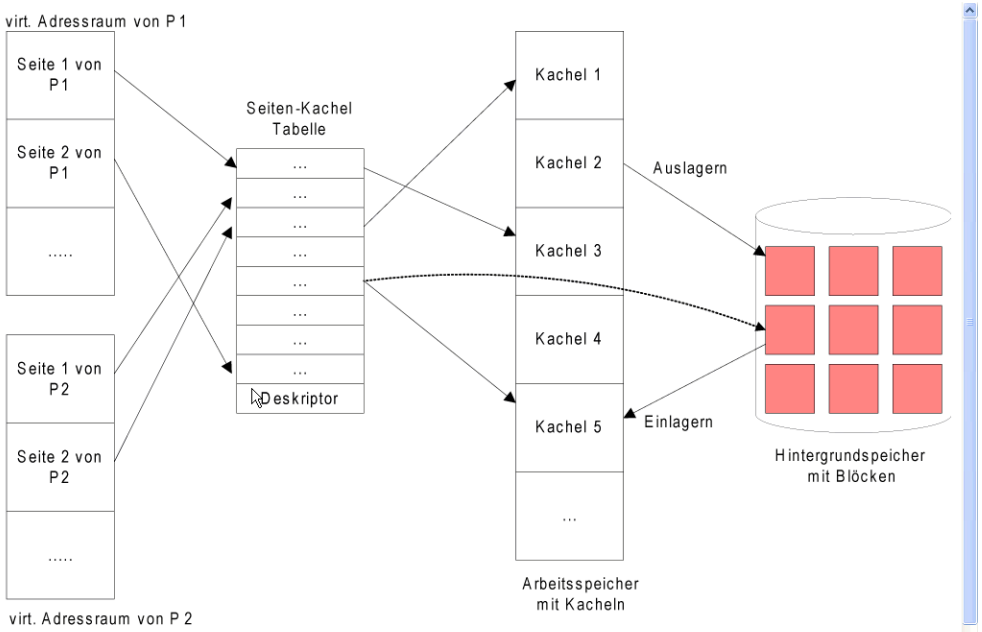
Der Programmadressraum, der sogenannte virtuelle Adressraum eines Prozesses wird in aufeinanderfolgende ***Seiten*** (engl. page) gleicher Größe unterteilt. Man spricht deshalb von virtuellen Adressen des Prozesses, anstatt von seinen Programmadressen.

Der Maschinenadressraum, also der physische Adressraum des Arbeitsspeichers, wird in ***Kacheln*** (engl. frame) unterteilt. Seiten und Kacheln sind i.d.R. gleich groß.

Eigenschaften der Seitenadressierung

virtueller Speicher - Arbeitsspeicher

Vorteile



Die Seiten eines Prozesses können im Arbeitsspeicher oder auf dem Hintergrundspeicher (Platte) gespeichert sein.

Die Kacheln nehmen die Seiten der Prozesse auf.

Zugriff auf virtuelle Adresse ⇒ zugehörige Seite muss einer Kachel zugeordnet sein.

Die Zuordnung, welche Seite in welcher Kachel gespeichert ist, und wo sich die Seite auf dem Hintergrundspeicher befindet, erfolgt mittels der ***Seiten-Kacheltabelle*** , die die ***Seitendeskriptoren*** enthält.

Seite nicht im Arbeitsspeicher ⇒ ***Seitenfehler*** ⇒ Einlagerung der Seite bei Bedarf ("Demand Paging").

Falls alle Kacheln belegt ⇒ Auslagern einer Seite gemäß einer ***Seitenersetzungsstrategie.***

Ziel dieser Strategien ist es, eine möglichst günstige Seite auszuwählen, und diese auf den Hintergrundspeicher auszulagern. Mögliche Strategien:

- ***FIFO*** (first-in first-out): Verdrängen der ältesten Seite, einfach zu implementieren.
- ***LRU*** (Least recently used): Verdrängen der am längsten nicht genutzten Seite; wird am häufigsten realisiert, wobei LRU approximiert wird, da eine exakte Realisierung zu aufwendig ist.

Bei der Seitenadressierung werden durch eine flexible Speicherabbildung alle Probleme der direkten Adressierung gelöst. D.h. die Programme können:

verschoben werden,

größer als der Arbeitsspeicher sein,

auch ausschnittsweise im Arbeitsspeicher sein.

Zusätzliche positive Eigenschaften

Es können ***gemeinsame Speicherbereiche*** zwischen Prozessen realisiert werden.

Es ist ein differenzierter Zugriffsschutz innerhalb eines Prozesses möglich.

Speicherverwaltung

Fragestellungen

Dieser Abschnitt beschäftigt sich mit den Adressräumen für Programme und deren Abbildung auf den physischen Arbeitsspeicher einer Rechenanlage:

Programmadressraum vs. Maschinenadressraum.

Direkte Adressierung, Basisadressierung.

Virtualisierung des Speichers; virtuelle Adressierung, insbesondere Seitenadressierung (nur sehr kurz).

Einführung

Speicherabbildungen

Dieser Abschnitt behandelt einige Mechanismen zur Abbildung von Programmadressen auf Maschinenadressen des Arbeitsspeichers.

Direkte Adressierung

Basisadressierung

Seitenadressierung