Informationstechnik - Unterichtsmitschrift

Timm Albers

2014

Betriebssysteme

Aufgaben

Bootvorgang

Der Bootvorgang beschreibt den Vorgang, der im Normalfall nach dem Einschalten eines Computers abläuft. Während dieses Vorgangs werden alle Hardwarekomponenten überprüft und das Betriebssystem geladen.

BIOS, BIOS-Einstellungen, BIOS-Fehlermeldungen

Das BIOS (= Basic Input Output System) führt während des Bootvorganges grundlegende Systemüberprüfungen (Hardware) durch und lädt das Betriebssystem.Die BIOS-Einstellungen ermöglichen das Verhalten des BIOS zu konfigurieren. Außerdem lassen sich grundlegende Einstellungen des Computers, wie etwa das Verhalten bei Überhitzungen des Prozessors oder die Systemuhr, konfigurieren.Sollten beim der grundlegenden Überprüfung des Systems oder dem Laden des Betriebssystems Fehler auftreten, so gibt das BIOS sogenannte BIOS-Fehlermeldungen aus.

POST

Der POST (= power-on-self-test) ist ein Programm, welches die grundlegenden Hardwarekomponenten eines Rechners prüft und bei nicht funktionsfähiger Hardware eine entsprechende Fehlermeldung ausgibt. Der POST wird vom BIOS zur Ausführung gebracht und ist das erste Programm, dass nach dem Einschalten eines Computers ausgeführt wird. Die Implementierung des POST hängt vom BIOS ab.Die meisten POSTs überprüfen zunächst die CPU und alle CPU nahen Bausteine, danach in gegebener Reihenfolge den CMOS-RAM, den

Cache-Speicher, die ersten 64 Kilobyte des Arbeitsspeichers und den Grafikspeicher, sowie die Hard- und Software zur Grafikausgabe.Sollten während des Überprüfens der Hardware Fehler auftreten, gibt der POST eine Fehlermeldung aus. Diese kann sich beispielsweise als Signalton äußern, wenn die Grafik noch nicht überprüft wurde oder nicht funktioniert.

CMOS-Setup

- Verliert der CMOS-Speicher seine Informationen gibt der Rechner eine Fehlermeldung * => Rechner bootet dann mit Standardeinstellungen

Aufgaben

Festplattenformatierung

- Die Festplatte besteht aus einem Stapel von Aluminiumscheiben (meistens), Glas oder Metalllegierung (5.25“; 3.5“; 1.8“ (Notebook)).
- Auf jeder Scheibe ist ein dünnes (ca. 1 Mikrometer) magnetisierbare Metalloxidschicht angebracht.
  - Abstand Lesekopf-Scheibe ca. 3nm. Der Abstand wird durch ein Luftpolster (aufgrund der Rotation) konstant gehalten.
- 800 GByte Speicherkapazität reichen für * 800 Stunden Aufzeichnung DVB-Ausstrahlung (640*480 Bildauflösung)* 80 Stunden Aufzeichnung HDTV-Ausstrahlung (1920*1080 Bildauflösung)
- **Formatierung von Festplatten**
  - Low-Level-Formatierung
    - Aufteilung in Spuren und Sektoren
    - Wird vom Hersteller vorgenommen
    - Nach Auslieferung im Normalfall nicht notwendig
    - Sektor:
      - Präambel: Beginnt mit einem bestimmten Bitmuster, damit die Hardware den Beginn eines Sektors erkennen kann. Sie enthält außerdem die Zylinder- und Sekorennummern.
      - Daten: Die Größe des Datenteils wird vom Formatierungsprogramm festgelegt (meistens 512 Byte große Sektoren (inkl. Präambel und ECC))
      - ECC (Fehlerkorrekturfeld): Enthält redundante Informationen, die zur Wiederherstellung der Daten nach Lesefehlern genutzt werden können. (ECC ca. 16 Byte)
    - Alle Festplatten haben Reserve-Sektoren, um Sektoren mit Herstellungsfehlern zu ersetzen.

- **Zylinderversatz**: Sektor 0 von Spur 1 ist gegenüber Sektor 0 von Spur 2 versetzt usw.
  - Bsp.: Anfrage benötigt 18 Sektoren, angefangen bei Sektor 0 der innersten Spur (vgl. Abb. S. 19 a). Das Lesen der ersten 16 Sektoren benötigt eine Plattenumdrehung. Für den 17. Sektor muss der Kopf neu positioniert werden. Bis der Kopf sich zur nächsten Spur bewegt hat, ist der 17. Sektor (Sektor 0 der nächsten Spur) bereits am Lesekopf vorbeirotiert.
    - => Es ist also eine volle Umdrehung für diesen Sektor nötig.
    - => Lösung: Versatz der Sektoren
  - Die Größe des Zylinderversatzes hängt von der Plattengeometrie (Größe) ab.
    - Bsp.: Festplatte mit 10000 Umdrehungen pro Minute benötigt 6 ms für eine Rotation, bei 300 Sektoren pro Spur taucht alle 20 Microsekunden ein neuer Sektor unter dem Kopf auf. Beträgt die Positionierungszeit zur nächsten Spur 800 Microsekunden, dann laufen in dieser Zeit 40 Sektoren am Kopf (Lesekopf) vorbei. Der Zylinderversatz sollte also 40 Sektoren betragen (statt drei wie in Abb. 5.26).

Low-Level-Formatierung (Spuren, Sektoren, Zylinder, Cluster) Die Low-Level-Formatierung unterteilt die Festplatte physikalisch in Spuren (aufeinanderliegende nennt man Zylinder) und Sektoren.Siehe Abb. 1 (A = Spur, B = Sektor, C = Block, D = Cluster)#### Partitionierung Unter einer Partitionierung versteht man den Vorgang des Anlegens von Partitionen auf einer#### Logische Formatierung #### Formatierung einer Festplatte AB 1

Aufgabe 1 Die Größe der Festplatte wird aufgrund der Low-Level-Formatierung reduziert. Hierbei ist die Größe der Präambeln, die Lücke zwischen den Sektoren, der Fehlerkorrekturcode und die Anzahl der Reservesektoren entscheidend.##### Aufgabe 2 Die Angabe der Festplattenkapazität ist deshalb kritisch zu betrachten, da einige Hersteller die Größe der Festplatte in unformatiertem Zustand angeben.##### Aufgabe 3##### Aufgabe 4 Um kontinuierliches Lesen ist meistens nur dann möglich, wenn der Festplattencontroller einen großen Buffer hat, der mehrere Sektoren zwischenspeichern kann. Sonst kann es häufig dazu kommen, dass nach dem Lesen eines Sektors fast eine komplette Umdrehung gewartet werden muss. Um dieses Problem zu umgehen und den Buffer des Controllers nicht zu groß anlegen zu müssen, werden die Sektoren verschachtelt. Das bedeutet, dass bei der Nummerierung der Sektoren jeweils einer (einfache Verschachtelung) oder zwei (doppelte Verschachtelung) ausgelassen werden.

TEILWEISE UNVOLLSTÄNDIG

- ... verwaltet.* die logischen Laufwerke werden hier Volums genannt, wobei verschiedene Volume-Typen möglich sind (einfaches Volume, übergreifendes, Stripeset)* weiterer Unterschied: Die Informationen über logische Laufwerke (eingerichtetete Volumes) werden nicht in der Partitionstabelle (die es hier auch gibt, damit andere BS die Platte nicht als unpartitioniert einstufen) abgelegt, sondern in einer Art Datenbank am Ende der Festplatte.

Software

Systemsoftware | Anwendungssoftware — | —Betriebssysteme | TextverarbeitungDiagnose- und Service-Tools | TabellenkalkulationNetzwerk- und kommunikationsprogramme | Branchenspezifische Software | „Freizeitsoftware“ Multimedia ### Systemsoftware #### von-Neumann-Rechner * Ausführbare Programme und zu bearbeitende Daten werden im gleichen Arbeitsspeicher abgelegt. (Unterscheidung durch Reihenfolge)* **Sequenzielles** Arbeiten (Befehl für Befehl hole, interpretieren, ausführen, Ergebnis speichern)* Befehle, die im Programm nacheinander folgen, werden ihrer Reihenfolge entsprechend im Speicher abgelegt. * Das Abarbeiten des nächsten Befehls wird durch die Erhöhung des Befehlszählers initiiert.* Durch Sprungbefehle kann von der Bearbeitung der Befehle in ihrer gespeicherten Reihenfolge abgewichen werden.* **Anwendung**: klassische Mikrocomputersysteme, wie auch der PC sind nach dem von-Neumann-Prinzip aufgebaut. * Es sind mindestens folgende Befehlstypen vorhanden: * arithmetische Befehle (Addition, Multiplikation) * Logische Befehle (UND, ODER, NICHT, ...) * Transportbefehle (MOVE, ...) * Ein-, Ausgabebefehle * Sprungbefehl (GOTO) * Verzweigungen (IF ... THEN ... ELSE ...) #### Harvard-Architektur * Instruktionen und Daten sind in getrennten Speichern untergebracht.* Prozessor benutzt getrennte Busse für Instruktions- und Datenzugriffe (effektiveres Arbeiten, da schnelleres paralleles Arbeiten möglich ist) d.h. die nächste Instruktion kann geholt werden, während noch Daten in den Speicher geschrieben werden.* Nachteil: Hoher Aufwand für die Realisierung dieser Architektur* Anwendung (heute): in speziellen Grafik-Chips. ### Aufgaben des BS (OS)

| **Anwendungen** | | ——————— | | semantische Lücke | | **Betriebssystem** | | **Hardware** | Das Betriebssystem verkleinert die semantische Lücke und erweitert damit **für den Anwender** des Rechensystems die Fähigkeiten der Hardware.#### Beispiel Benutzer soll Dateien im Hauptspeicher abspeichern können, ohne das Wissen darüber zu haben, ob: * genügend freier Speicherplatz vorhanden ist und* unter welcher physikalischen Speicherplatzadresse die Datei abgespeichert wird.=> Das Betriebssystem stellt dazu eine Maschine (virtuelle Maschine) zur Verfügung. **Virtuell:** der Benutzer kann nicht trennen, welche Aufgaben von der Hardware und welche vom Betriebssystem übernommen werden. #### Grundlegende Aufgaben des BS * Hochfahren (Booten) des

Rechnersystems * u.a. Laden des BS in den Arbeitsspeicher* Anpassung und Steuerung der verwendeten Hardware und Peripheriegeräte* Erkennung und Abfrage von Fehlersitationen* Verwalten des Arbeitsspeichers und des Dateisystems (Filesystem)* Bereitstellen von Dienst- und Anwendungsprogrammen zur Systempflege* Überwachung und Koordination der ströungsfreien Ausfürung von Anwendungsprogrammen* Vernetzung mit anderen Systemen#### TEILWEISE UNVOLLSTÄNDIG #### Anzahl der verwalteten Prozessoren * Zur Verarbeitung der Daten stehen mehrere Prozessoren (Universalprozessoren) zur Verfügung. * Es gibt mindestens einen Hauptprozessor und * weitere##### Unterscheidungsmerkmale* **Ein-Prozessor-BS** * Die meisten Rechner, die nach der von-Neumann-Architektur aufgebaut sind, verfügen über einen Universalprozssor.* **Mehrprozssor-BS** * Es ist mehr als ein Prozssor vorhanden * **Vorgehensweise:** Jedem Prozessor wird durch das BS eine eigene Aufgabe (Prozess) zugeteilt. * Es können nur so viele Aufgaben bearbeiten werden, wie Prozessoren vorhanden sind. * **Problem:** Koordinierung erforderlich, wenn die Anzahl verfügbarer Prozssoren und die der Aufgaben nicht gleich ist. Jede Aufgabe kann prinzipiell jedem Prozssor zugeordnet werden. Gibt es mehr Aufgaben als Prozessoren, so bearbeitet ein Prozssor mehrere Aufgaben „quasi-parallel“ (zeitlich verschachtelt). Sind mehr Prozessoren als Aufgaben vorhanden, so können mehrere Prozessoren die gleiche Aufgabe bearbeiten, wenn die Aufgabe in parallel bearbeitbare Teilaufgaben geteilt werden kann. #### Multithreading (bei Intel: Hyperthreading) Hierbei wird die Steuerlogik vervielfacht, d.h. der Prozessor kann zwischen zwei verschiedenen Threads hin- und herschalten. **Beispiel:** Muss z.B. ein Prozess ein Datenwort aus dem Speicher lesen, was mehrere Taktzyklen dauert, kann eine Multithreading-CPU während der Wartezeit einfach zu einem anderen Thread umschalten. Es bietet keine echte Parallelität und es läuft nur ein Prozess, **aber** die Zeit des Umschaltens zwischen den Threads ist erheblich reduziert. Aus Sicht des BS erscheint jeder Thread wie eine seperate CPU. => BS muss diese Technik unterstützen, da sonst eine ineffiziente Auswahl bzw. Zuteilung droht. #### Allgemein Alle Betriebssystemarten haben die gleichen typischen Systemaufgaben: * Prozessverwaltung* Dateiverwaltung* Speicherverwaltung* I/O-Geräteverwaltung### Scheduler / Scheduling #### Allgemein * Scheduling ist Teilaufgabe des BS* Bezeichnet die Verwaltung von mehreren Prozessen, die auf einem Rechner bearbeitet werden.* Strategie des Vorgehens wird Scheduling-Strategie (scheduling algorithm) genannt.* **Prozess-Scheduling:** Gibt es mehr Bedarf an Betriebsmitteln als vorhanden sind, so muss der Zugriff koordiniert werden.Dabei ist der Scheduler und seine Strategie beim Einordnen der Prozesse in Warteschlangen von besonderer Bedeutung. * **Bsp:** Einprozessorsystem, bei dem unabhängige Prozesse sequentiell ablaufen. Unterscheidung in zwei Arten von Scheduling-Aufgaben: 1. Planen der Jobausführung (Langzeit-Scheduling)Aufgaben dabei sind: * Es kommen nur so viele Nutzer mit ihren Jobs neu ins System, wie Nutzer sich abmelden. * Zu viele Nutzer: Zugang sperren bis Systemlast „erträglich“ (Datei-Server) * Ausführen von nicht-interaktiv ablaufenden Jobs zu bestimmten Zeiten (nachts) 2. Ak . . . * Effizienz / Auslastung der CPU * Höhe der Auslastung (40% - 90%) * Durchsatz * Zahl der Jobs pro Zeiteinheit ist ein Maß für die Systemauslastung

* Sollte maximal sein * Faire Behandlung * Gezielte Verteilung der Prozessorzeit, d.h. jeder Benutzer / Prozess sollte im Mittel den gleichen CPI-Zeitanteil erhalten. * Ausführungszeit * Zeit von Jobbeginn bis Jobende enthält alle Zeiten in Warteschalfen, der Ausführung und der Ein- und Ausgabe. Sollte minimal sein. * Scheduler kann von Ausfürhungszeit nur Wartezeit in Bereitliste beeinflussen. => Wartezeit minimieren. * Antwortzeit * Interaktive Prozesse sollten schnell reagieren, d.h. die Reaktion des Rechners auf eine Eingabe des Benutzers sollte minimal sein. 3. Einleitende Betrachtungen zur Scheduling-Strategie * Scheduling-Strategie auf einfachen Pcs mit „einfachen“ Anwendungen, d.h. Anwendungen, die nur wenig CPI-Zeit beanspruchen eher unwichtig, da es hier „fast“ nicht von Bedeutung ist welcher Prozess zuerst bearbeitet wird. * Dagegen benötigt z.B. das Rendern eines Videos (Farben optimieren in einzelnen Bildern eines Videos) extrem starke Rechenleistung (400.000 Einzelbilder eines einstündigen Videos). => Scheduling-Strategie wichtiger * Bei vernetzten Servern wird Scheduling-Strategie erheblich wichtiger (z.B. Entscheidung ob Rechenleistung für tägliche Statistiken oder Benutzeranfrage zur Verfügung gestellt wird) * Scheduler muss sich um * Auswahl des richtigen Prozesses und um die effiziente bzw. optimale Ausnutzung der CPI kümmern. * Eine sorgfälltige Auswahl ist erforderlich, da ein Prozesswechel aufwändig ist. Folgende Schritte müssen dazu erfolgen: * Wechsel vom Benutzermodus zum Kernmodus * Zustand des laufenden Prozesses muss gesichert werden, d.h. Speichern der Register in Prozesstabelle damit sie später wieder geladen werden können. * Speicherzuordnungstabelle speichern, d.h. es wird gespeichert unter welcher Adresse die Informationen zum jeweiligen Prozess zu finden sind. * Auswahl des nächsten Prozesses durch Scheduling-Algorithmus. * MMU (Memory Management Unit) muss mit der Speicherzuordnungstabelle des neuen Prozesses geladen werden. * Neuer Prozess muss gestartet werden * i.d.R. ist gesamter Speicher-Cache ungültig und muss neu geladen werden. * => diese Schritte bedeuten einen erheblichen Rechenaufwand. * ... * der **Informatiker** ist die **CPU** * **Zutaten** sind Eingabedaten * der **Prozess** ist die Aktivität, die daraus besteht, dass der Informatiker (CPU) das Rezept (Algorithmus, Programm) liest, die Zutaten (Daten) herbeiholt und den Kuchen backt. * Folgende Situation tritt ein: * Sohn kommt schreiend (von Biene gestochen) in die Küche. * Informatiker merkt sich, an welcher Stelle des Rezeptes er sich befindet (Zustand des aktuellen Rezeptes wird gespciehrt), holt das Erste-Hilfe-Buch und liest (Alogorithmus). * Informatiker (CPU) wechselt vom Prozess „Backen“ zum Prozess „medizinische Hilfe“ (höhere Priorität!) * Jedem Prozess liegt (hier) ein anderes Programm zugrunde. * Bienenstich behandelt => Informatiker (CPU) kehrt zum Prozess „Backen“ zurück. #### Prozesszustände1. Prozess blockert, weil er z.B. auf eine Eingabe wartet.2. Scheduler wählt einen anderen Prozess aus, weil dieser z.B. lange genug gelaufen ist und nun ein anderer Prozess (A) Rechenzeit erhalten soll.3. Scheduler wählt diesen Prozess (A) aus.4. Z.B. Eingabe vorhanden; Wechsel von Zustand „blockiert“ zum Zustand „rechenbereit“.**Rechnend:** Befehle werden von CPU ausgeführt.**Rechenbereit:** kurzzeitig gestoppt (der Prozess)**Blockiert:** Prozess nicht lauffähig, da bestimmte Ereignisse (Eingaben usw.) fehlen. #### C* Das Scheduling soll erfolgen, wenn: * Sich zwei

Prozesse im rechenbereiten Zustand befinden => Entscheidung erforderlich, welcher als nächster bearbeitet werden soll. * Ein Prozess beendet wurde. => Auswahl eines neuen Prozesses aus Menge der rechenbereiten Prozesse. * Ein laufender rechnender Prozess auf ein Eingabe / Ausgabe warten muss und daher ein neuer ausgewählt werden muss oder wegen eins Semaphors blockert wird.* **Bsp:** Prozess (A) wartet darauf, dass (B) kritische Region (verändert (z.B. eine Datenstruktur, die auch Prozess (A) benutzt) verlässt. Dann wird es durch Einteilung von (B) als nächsten Prozess ermöglicht, dass Prozess (A) bald weiter arbeiten kann. * Ein E-/A-Interupt auftritt. Scheduler muss entscheiden ob auf E/A wartenden Prozess ein rechenbereiter Prozess oder ein dritter Prozess, der evtl. auf E/A gewartet hat und blockert war, bearbeitet werden soll. * Allg: Das Scheduling bzw. die Entscheidungen können zu jedem k-ten Timer-Interrupt erfolgen.

(d) Scheduling-Strategien

Scheduling-Strategien (z.B. "Firstcome-First-served-Scheduling") lassen sich bezüglich ihres Umgangs mit Timer-Interrupts in **zwei Kategorien** einteilen. In

- **nicht unterbrechende (non preemptive)** d.h. der Scheduler lässt einen Prozess so lange laufen, bis er blockiert oder bis er freiwillig die CPU freigibt. **Innherhalb** eines Taktzyklus findet keine Entscheidung statt.

  Nachdem ein Timer-Interrupt abgearbeitet wurde, wird der Prozess wieder aufgenommen, der vor dem Interrupt lief. Es sei denn ein Prozess mit höherer Priorität hat auf solch eine Unterbrechung gewartet.

- **unterbrechende (preemtive)**. Dabei lässt die Scheduling-Strategie einen Prozess eine vorher festgelegte Zeit laufen, unterbricht diesen und wählt einen anderen rechenbereiten Prozess aus. Zum Schluss des Zeitintervals ist ein Timer-Interrupt erforderlich, um dem Scheduler die Kontrolle über die CPU zurückzugeben.

Diese Kategorien von Scheduling-Strategien kommen in den folgenden verschiedenen Systemen / Umgebungen zum Einsatz.

- **Stapelverarbeitungssystem**
  - First-Come-First-Served-Scheduling (non preemptiv)
  - Shortest-Job-First-Scheduling (non preemptiv)
  - Shortest-Remaining-Time-Next-Scheduling (preemptiv)
- **interaktives System**
  - Robin-Round-Scheduling (preemptiv)

  - Prioritäts-Scheduling (preemptiv)

- **Echtzeitsysteme**
  - Echtzeit-Scheduling: Die Echtzeitalgorithmen können statisch oder dynamisch sein. Bei statischen Algorithmen wird jedem Prozess eine feste Priorität zugeteilt. Das führt zum *unterbrechbaren Prioritäts-Scheduling*. Bei dynamische Algorithmen gibt es fast keine fest vorgegebenen Prioritäten. Hierbei meldet sich ein Prozess mit seiner Deadline an. Der Scheduler sortiert nach Deadlines. Ist die Deadline kürzer aös die des gerade laufenden / aktuellen Prozesses, wird der laufende Prozess unterbrochen (preemptiv).

(e) Ziele von Scheduling-Strategien in veschiedenen Systemen

- **Stapelverarbeitungssysteme**
  - **Durchsatz:** Maximieren der Jobs pro Stunde
  - **Durchlaufzeit:** Maximieren der Zeit vom Start bis zur Beendigung
  - **CPU-Ausnutzung:** CPU ist immer ausgelastet

Aufgabe

Fasse die wesentlichen Merkmale der folgenden Scheduling-Strategien stichwortartig (schriftlich!!!) zusammen.

1. **First-Come-First-Served-Scheduling** (FCFS) / **First-In-First-Out-Scheduling** (FIFO)

   non preemptiv

   Alle Prozesse werden in der Reihenfolge ihres Eingangs abgearbeitet.

   - Gute CPU-Auslastung
   - Schlechte Ressourcen-Auslastung
   - Nicht für Mehrbenutzersysteme geeignet

2. **Shortest-Job-First** (SJF) / **Shortest-Job-Next** (SJN) / **Shortest-Processing-Time** (SPT)

   non preemptiv

   Lässt sich immer dann einsetzen, wenn die für einen Prozess benötigte Rechenzeit sich gut aus Erfahrungswerten vorhersagen lässt.

   - Große Prozesse bekommen unter Umständen niemals die CPU zugeteilt

- Nicht für Mehrbenutzersysteme geeignet

2.1 **Shortest-Remaining-Time-Next** / **Shortest-Remaining-Time** (SRT) / **Shortest-Remaining-Processing-Time** (SRPT)

preemptiv

Erweitert Shortest-Job-First um die Fähigkeit einen Prozesswechsel durchzuführen, wenn ein Prozess eingeht, der eine kürzere Ausführungszeit aufweißt, als der aktuell laufende.

3. **Round-Robin** / **Zeitscheibenverfahren**

   preemptiv

   Die CPU wird einem Prozess für eine bestimmte, kurze Zeitspanne zugeteilt. Nach Ablauf dieser Zeitspanne wird der Prozess wieder hinten eingereiht.

   Weighted Round-Robin (WRR)

   preemptiv

   Die Zeitspannen können unterschiedlich lang sein.

4. **Prioriäts-Scheduling**

   Jedem Prozess wird eine Priorität zugeordnet, die dann die Reihenfolge der Abarbeitung festlegt.

   4.1 **Rate Monotonic Scheduling** (RMS)

   4.2 **Deadline Monotonic Scheduling** (DMS)

   4.3 **Multilevel Feedback Queue Scheduling** / **Shortest Elapsed Time** (SET)

Quellen

- Prozess-Scheduler (Wikipedia; Abrufdatum: 25.04.2014)

Übungen

Aufgabe 1

1. Welche typischen Systemaufgaben haben alle Betriebssysteme?

   - Prozessverwaltung
   - Dateiverwaltung
   - Speicherverwaltung
   - Geräteverwaltung (IO)

2. Erkläre zwei Betriebsarten eines Rechnersystems.

- Singleuser ein Benutzer zur Zeit
- Multiuser mehrere Benutzer zur selben Zeit
- Batchbetrieb Stapelverarbeitung
- Dialogbetrieb Dialog zu Usersystem (über Bildschirm, Tastatur, Maus usw.)
- Netzwerkbetrieb Einbindung in ein Netzwerk
- Echtzeit Verarbeitungszeit im Vordergrund

3. Was wird mit dem Begriff Multitasking (in Bezug auf Rechnersysteme) beschrieben? Gibt es unterschiedliche Vorgehensweisen? Erkläre.

    - mehrere Prozesse quase parallel
    - kooperativ (Programme teilen selbst ein) / präemptiv (BS behält Kontrolle)

Aufgabe 2

1. Was wird unter Scheduling verstanden und welche Aufgabe hat dieser allgmein?

    - zeitliche Begrenzung von Prozessen. Es wird festgelegt wann wie viel Prozessorzeit bereitgestellt wird.
    - Scheduler wird benötigt um Ordnung in die Abarbeitung von Prozessen / verschiedenen Anwendungen usw. zu bringen, damit der Prozessor effektiver ausgenutzt werden kann und Prozesse schneller ablaufen können.

2. Erkläre zum Scheduling das non-preemptive Verfahren.

    - Prozesse laufen so lange, bis sie von sich aus den Aktivzustand verlassen, d.h. sie werden nicht vorzeitig abgebrochen.

Aufgabe 3

Rechner können entsprechend ihrer Architektur (Von-Neumann, Harvard) eingeteilt werden. Erläutere die wesentlichen Unterschiede.

Bei der Harvard-Architektur

- CPU besitzt getrennte Busse für Instruktionen und Datenzugriffe, d.h. dass die nächste Instruktion geholt (gelesen) werden kann während noch Daten in den Speicher geschrieben werden.
- getrennte Speicher Instruktionen und Daten

Nachteil: hoher Aufwand für die Realisierung der Architektur

Aufgabe 4

...

Boot-Vorgang

- BIOS wird beim Einschalten gestartet. BIOS überprüft Größe des RAM und ob Tastatur und andere Geräte installiert sind. ISA-, PCI-Bussyteme werden nach angeschlossenen Geräten durchsucht. Geräte werden in Liste eingetragen und falls sie noch unbekannt sind sind, werden sie konfiguriert (E-A-Adressen, Interruptnummern)
- BIOS bestimmt Gerät (CD, Festplatte) von dem das BS geladen werden soll. Dabei wird ein Liste im CMOS-Speicher durchprobiert.
- erster Sektor des Boot-Gerätes wird in Arbeitsspeicher geladen und ausgeführt. Dieses kleine Programm liest die Partitionstabelle und ermittelt die aktive Partition. Von dieser Partition wird der Bootloader gelesen. Dieser wiederrum liest das BS ein.
- BS fragt BIOS nach Konfigurationseinstellungen => Gerätetreiber werden gesucht.
- Hintergrundprozesse einrichten, Login-Programm und grafische Oberfläche starten
- BS hat / übernimmt Kontrolle

Interrupt

- ohne Interrupt würde der Prozessor in einer Scheilfe arbeiten, d.h. er würde alle angeschlossenen Komponenten (Tastatur, Monitor, Schnittstellen usw.) regelmäßig abfragen bzw. ansteuern. **=> zeitaufwendig, da er auch abfragen muss, wenn keine Aktion gefordert ist.**
- Mit Interrupt-Verarbeitung wird der Prozessor in seiner momentanen Arbeit unterbrochen. Bei Darstellung auf dem Monitor könnte z.B. eine Tastatureingabe eingelesen werden.

Beispiel

Betätigung einer Taste löst einen Interrupt aus

- Prozessor liest Tastencode ein und führt entsprechende Aktion aus.
- nach erledigter Aktion wird das Programm an der Stelle fortgesetzt wo es unterbrochen wurde.

- Dazu werden logischer Zustand und die Inhalte der Register automatisch gespeichert, d.h. sie werden auf den Stapel-Speicher (Stack) gelegt und später wieder ausgelesen.

2014-06-19

Unterscheidung in Soft- und Hardware-Interrupt

- Den Software-Interrupts sind festgelgte Funktionen zugeordnet (z.B. BIOS)
- Den Hardware-Interrupts sind Systemkomponenten (Hardware-Komponenten) zugeordnet.

Software-Interrupt

Hardware-Interrupt Interrupts werden von externen Baugruppen (Erweiterungskarten, Tastatur etc) über **spezielle Leitungen**, den Interrupt-Leitungen, eingegliedert.

Im PC steht nur eine begrenzte Anzahl von Interrupt-Leitungen zur Verfügung. Die Verwaltung erfolgt über den Interrupt-Controller (im Chipsatz integriert).

Allgemein IRQ mit niedrigerer IRQ-Nr. haben die höchste Priorität.

Erfolgt ein Interrupt mit höherer Priorität während eines Interrupts mit niedrigerer Priorität, so wird das laufende Interrupt unterbrochen und zunächst der höherwertige bedient.

Interrupt-Controller

- koordiniert alle eintreffenden Unterbrechungsanforderungen, speichert sie im Anforderungsregister und leitet sie entsprechend ihrere Priorität zusammen mit der programmierbaren Unterbrechnungsnummern an die CPU.
- **Unterbrechungsanforderung**
  - Steuerbaustein im Interrupt-Controller sendet Unterbrechungsanforderung (INT) an CPU.
  - CPU quittiert mit INTA (acknowledge) und gibt gleichzeitig Datenbus frei.
  - INterrupt-Controller übernimmt **intern** Anforderungen mit höchster Priorität (höchstpriore Anforderung) vom Anforderungsregister in das Bedienungsregister und sendet **Kennzeichnungsnummer** dieser Anforderung (über Datenbus) zur CPU.

- NMI: Non Maskable Interrupt
  - Dieser Interrupt kann nicht maskiert, d.h. durch Programmierung unterdrückt werden.
  - **Beispiel:** nicht korrekte Parität im RAM führt zu einem NMI
    - => PC kommt zum Stillstand; Absturz
  - Meldung: Versorgungsspannung bricht zusammen
    - => z.B. Aktion: Spannung über Kondensatoren so lange halten bis Rettungsaktion / Sicherung abgeschlossen ist.

Übungen

Aufgabe 1

a) Wo ist der DMA-Controller angesiedelt? Erkläre. Manchmal direkt im Plattencontroller aber meistens existiert ein eintiger DMA-Stauerbaustein (z.B. auf der Hauptplatine)

b) Welche Register enthält der DMA-Controller und was wird in diesen gespeichert bzw. von deisen gesteuert? Erkläre.

- Speicheradressregister
- Bytezählregister
- ein / mehrere Kontollregister
  - gewünschter Ein- / Ausgabeport
  - Richtung der Übertragung
    - Lesen / schreiben
  - Übertragungseinheit
    - Byte / Wort pro Zyklus

$$\frac{Anzahl\ Daten}{Zyklus}$$

c) Stelle den Ablauf des Lesens von einer Platte ohne DMA und mit DMA stichwortartig dar. Zur Darstellung kannst du Bezug nehmen auf die Abb. 5.4 auf S. 1 des Arbeitsblattes vom 12.06.2014.

Lesen ohne DMA

- Plattencontroller liest Block byteweise bis gesamter Block im Controllerpuffer liegt

- Checksum berechnen
- Controller sendet Interrupt
- Wenn OS Rechenzeit: In Schleife Daten zeichenweise auslesen und in RAM laden

Lesen mit DMA

- CPU programmiert DMA-Controller
- Kommando an Plattencontroller: Daten in internen Speicher laden
- Checksum testen und DMA-Übertragung beginnen
- => Lesebefehl
- Speicheradresse auf Adressbus
- Schreiben in RAM
- DMA-Controller erhöht Speicheradresse und veringert Anzahl der zu übertragenden Zeichen
- Schleife so lange, wie zu übertragende Zeichen > 0: lesen
- DMA erzeugt Interrupt (Daten im Speicher) und signalisiert CPU, dass die Übertragung fertig ist

d) Es kann sein, dass mehrere Geräte eine Übertragung von Daten wünschen. Mit Hilfe welcher Verfahren entscheidet der DMA-Controller darüber, welches Gerät als nächstes bedient wird? Nenne zwei.

- Round-Robin
- Priorität

Aufgabe 2

a) Was wird mit den Begriffen Cycle Stealing und Burst-Modus bezeichnet? Erkläre.

Cycle Stealing

- Erwartet DMA-Controller eine Übertragung
  - => DMA bekommt die Daten
- falls der Prozesoor gleichzeitig den Bus verwenden will, muss er warten
- Gerätetreiber stiehlt einen Buszyklus von CPU
  - => CPU kurzzeitig blockiert

Burst Stealing

- Gerät wird von DMA beauftragt den Bus zu belegen, Übertragung durchzuführen und wieder freizugeben

b) Werden die Daten immer von Gerät zu Gerät übertragen oder immer über den DMA-Controller? Erkläre welche Möglichkeiten es gibt.

z.B. Fly-by-Modus DMA-Controller sagt Gerätecontroller Daten direkt in RAM zu schreiben

Alternative

- Geräte-Controller Daten => DMA-C.
- DMA-Controller startet zweiten Buszyklus oder Übertragungszyklus => Ziel (dauert länger, aber flexibel)

c) Nenne zwei Vorteile die aus der Verwendung eines internen DMA-Puffers entstehen.

- Vor Übertragung: Fehlerprüfung (durch Checksum)
- DMA-Übertragung nicht zeitkritisch
  - => Wenn Bus belegt, müsste Controller warten, Daten zwischenspeichern und verwalten (hohe Auslastung)

d) Ist die Verwendung / der Einsatz eines DMA-Controllers uneingeschränkt von Vorteil? Begründe deine Aussage. Nein, weil es zu viel Zeit benötigt und es mit der CPU schneller und sparsamer ist.

Aufgabe 3

a) Nenne vier Eigenschaften, die ein präzises Interrupt besitzt. (Ergänzung)

- Befehlszähler wird gesichert (an einer **bekannten** Stelle)
- Befehlszähler zeigt auf aktuellen Befehl, alle vorherigen Befehle sind vollständig abgearbeitet
- **kein** Befehl nach dem Aktuellen wurde schon ausgeführt
  - Auswirkungen bereits gestarteter Befehle sind wieder rückgängig gemacht worden.
- Ausführungszustand des aktuellen Befehls ist bekannt

b) Worin unterscheiden sich präzise von unpräzisen Interrupts im Wesentlichen? Erkläre stichwortartig.

- **präziser Interrupt**, wenn Kriterien aus **a)** erfüllt werden
- im Gegensatz zum präzisen Interrupt befinden sich beim unpräzisen Interrupt die Befehle in der Nähe des aktuellen Befehls in Unterschiedlichen Ausführungsstadien (vgl. Abb. 5, 6 b / Blatt 4)

c) Ein Gerät hat seine Aufgabe erledigt und sendet einen Interrupt. Stelle den weiteren Ablauf stichwortartig dar. (Interrupt-Controller, CPU, Prioritäten, Interruptvektor usw.)

- Gerät sendet Interrupt an Interruptcontroller (IC)
- IC entscheidet was zu tun ist (nach Priorität des Interrupts)
- IC legt Nummer zu Identifizierung des Gerätes, welches den Interrupt ausgelöst hat, auf Adressleitung
- CPU unterbricht die Arbeit und trägt die Nummer auf der Adressleitung als Index in Tabelle ein (**Interruptvektor**). Aus dieser Tabelle wird der **neue** Befehlszähler geholt. Dieser Befehlszähler zeigt auf die *interrupt service routine* (ISR)
- ISR signalisiert dem IC, dass die Bearbeitung des Interrupts begonnen hat => IC kann nächsten Interrupt bearbeiten.