**Schwerpunkte**

Von Neumann Architektur

Konzeptionelle und technische Merkmale

1. Sequentielle Verarbeitung vermeidet Inkonsistenzen

2. Anweisungen und Daten können im laufenden Betrieb verändert werden (im Gegensatz zu den früheren fest verdrahteten Computern)

3. Modell und Programmierung gut beherrschbar

4. Programmschleifen sind möglich

Bestandteile

- ALU = Arithmetic Logic Unit = Rechenwerk

- Control Unit = Steuerwerk

- Memory Unit = Hauptspeicher

- I/O-Unit = Eingabe-/Ausgabe-System

- Bussystem: Transport von binär kodierten Daten und Anweisungen

Modell: Prüfung auf Vollständigkeit. Wer hat das gemacht?

Wie heißen universelle Rechner, die alle mathematischen Aufgaben lösen können?

- Turing-Maschinen

Technik:

Cache (wozu braucht man es? Was ist es? Wie funktioniert es?)

- schneller Zwischenspeicher in Form von SRAM, entweder in der CPU integriert (L1-Cache) oder optional auf separatem Chip mit High Speed Alternative Bus (L2) oder separat auf Motherboard (L3)

Memory Wall

Unterschiedliche Leistung und Geschwindigkeit

- CPUs verdoppeln Geschwindigkeit alle 1,5 Jahre

- DRAM verdoppelt Schnelligkeit alle 10 Jahre

Was hat man gemacht, um den Von-Neumann-Flaschenhals zu beschleunigen?

- Von-Neumann-Flaschenhals: Das zentrale Bussystem muss Daten und Befehle nacheinander transportieren und wird bei leistungsfähigen Rechnern überfordert

Lösung:

- höhere Integration des Chipsatzes in VLSI-Chips (= very large scale) mit Millionen von Transistoren

- ab x386 Teilung des Bussystems in:

- schnelle North Bridge:

- für Grafik

- für RAM

- für PCI-e

- langsame South Bridge:

- für restliche Peripherie

- für Bios (ROM)

CPU-Architektur:

Risc & Cisc – was ist das und wo wurde/wird es verwendet?

CISC:

- konventionelle Prozessoren, nachträglich CISC genannt

- Complex Instruction Set Computing

- komplexer Befehlssatz mit ca. 300 Befehlen

- Befehle sind Microcode, die CPU aus einem ROM ausliest

- dieser ROM war früher separat auf dem Mainboard, später in der CPU embedded

- verwendet bei Intel x86 CPUs und AMD

- hält Anwendungs-Programme kleiner

- weniger Register, weniger RAM

- Interpretation beansprucht Taktzyklen

RISC:

- Reduced instruction set Computer

- ca. 100 Befehle

- 50 - 200 Register

- meisten Befehle auf Schaltungsebene und einen einzigen Taktzyklus schnell

- deutlich besser optimierbar für Hochsprachencompiler

- mehr Leistung, weniger Energieverbrauch

- in Smartphones und Tablets

Verschiedene Arten von Betriebssystemen - Klassifizierung

- Nach Computertyp

- Nach Betriebsart

- Nach Anzahl gleichzeitiger Aufträge

- Nach Anzahl gleichzeitiger Nutzer

Nach Einsatzort: Klein-/Server-BS

- in Großrechnern: z. B. z/OS, z/Linux

- in Servern: Windows Server, Linux

- in PCs: Windows 8, Windows 10, MacOS

- in Tablets und Smartphones: iOS, Andorid

- in Embedded Systems: Linux, QNX

- in IoT-Geräte/Sensoren: TinyOS, RIOT, Amazon FreeRTOS

Nach Betriebsart: (Echtzeit vs stapelverarbeitung – Unterschiede, welche Anforderungen) Interaktiv-, Dialog-

- Batch/Stapelverarbeitung: DOS, erste Windows Versionen

- Dialog: Ab Windows XP, Linux

- Echtzeit: VxWorks, Windows CE, QNX

Multitasking/singletasking

- Singletasking: DOS

- Multitasking: alle modernen OS

Multiuser/Singleuser

- Single User: DOS, 68K-MacOs, Android, iOS

- Multi User: Unix, Linux, Windows Server

Abstraktion von Betriebssystemen:

Systemaufrufe

- als Methode mit der eine Applikation eine Funktion vom BS aufruft  
( mit Beispiel/ welche Schnittstelle)

- Anwendungen greifen zum Speichern von Daten nicht direkt auf die Festplatte zu. Sie verwenden stattdessen Systemaufrufe, um deren Funktionen zum Speichern zu nutzen

- OS stellen Anwendungen ihre normierten Funktionen über Systemaufrufe (= System calls = Syscalls) zur Verfügung

- Systemaufrufe bilden oft die einzige Schnittstelle zwischen Anwendungen und dem OS

- Unix und Derivate führen eine Liste mit definierten Systemaufrufen, über die die Funktionen des Kernel erreichbar sind

Unterschiede in der Rechteebene – Kernelmodus/Usermodus

Kernelmodus:

- volle Nutzungsrechte für die Hardware

- Funktionen laufen im privilegierten Kernelmodus

- Alle Maschinenbefehle

- Abschaltung des Speicherschutzes

Usermodus:

- deutlich weniger Rechte, um Computer und Daten zu schützen

- nicht sicherheitskritische Maschinenbefehle

- nicht alle Speicherbereiche

Welche Aufgaben hat der Kernel?

- Steuert und verwaltet Hardware, laufende Prozesse, Ressourcen, Anwendungen und Daten

- Lädt Gerätetreiber für Netzwerk, Dateisystem, Grafik...

- Steuert Speicherzugriffe, alle Prozesse, I/O-Aufgaben und die Systemsicherheit

- Startet und interagiert mit der grafischen Benutzeroberfläche bzw. der Kommandozeile

- Startet und steuert Anwendungen

- Startet Systemdienste, die im Hintergrund laufen, z.B. eine Druckerwarteschlange oder Netzwerkdienste wie den TCP/IP-Stack oder DHCP und DNS

Welche Aufgaben hat das USer Interface?

- Stellt eine für Menschen mit ihren Sinnesorganen wahrnehmbare Schnittstelle zur Verfügung

- Entweder eine Kommandozeile zur Bedienung mittels Tastatur oder eine grafische Benutzeroberfläche

- Interpretiert die Aktionen des Benutzers und setzt sie in für die Maschine verständliche Anweisungen um

Multiprocessing & Multitasking

* Was ist ein Prozess? (per Definition, kennzeichnende Teile)

- Ein Prozess ist ein Programm in Ausführung, die laufende Instanz eines Programms.

- Komponenten: Eigener Adressraum, Process Identifier, Benutzerangaben, Kindprozesse und Threads, Weitere Angaben zur Ausführung des Prozesses z. B. Befehlszähler

- Logischer Aufbau: Code, Daten, Heap (frei zuweisbarer Speicher), Stack (LIFO-Speicher, zB für Unterprozesse)

* Was ist eine Prozesstabelle, was steht da drin

- Enthält Daten zu Prozessen, wird vom Scheduler benötigt, um den Prozessen CPU-Zeit zu gewährleisten

- da steht drin: Stackpointer, Prozesszustand, PID, Befehlszeiger, Register, Memory limits, Liste geöffneter Dateien, Verschiedene Buchführungs- und Statusdaten

* Was bewirkt ein Prozess, welche Bestandteile kann er haben

- Das Prozessmodell allgemein soll PC-Ressourcen abstrahieren bzw. Aktionen des Prozessors werden in zusammenhängende Aufgabenstränge abstrahieren und in pseudoparallelen Abläufen darstellen. All dies wird Programmen und dem Nutzer zur Nutzung bereitgestellt.

- Bestandteile: Eigener Adressraum, Process Identifier, Benutzerangaben, Kindprozesse und Threads, Weitere Angaben zur Ausführung des Prozesses z. B. Befehlszähler

- Logischer Aufbau: Code, Daten, Heap (frei zuweisbarer Speicher), Stack (LIFO-Speicher, zB für Unterprozesse)

* Was ist ein Thread, worin unterscheiden sich Prozess und Thread?

- Threads sind vergleichbar mit Prozessoren, jedoch Prozessen untergeordnet und als parallele Verarbeitungsstränge innerhalb eines Prozesses zu verstehen.

- Threads des selben Prozesses teilen sich Adressraum, globale Variablen, geöffnete Dateien. Im Gegensatz dazu können Prozesse nicht auf Adressräume etc. anderer Prozesse zugreifen

* Prozesszustände

- neu, bereit, laufend, blockiert, terminiert

Folie 8-18 circa

Race conditions:

- warum treten sie auf? 2 Bedingungen

- Nebenläufigkeit von Prozessen

- Gemeinsame Daten werden genutzt, z. B. globale Prozessvariablen

- was kann man tun, um die zu vermeiden

- Alle Interrupt ausschalten, TSL, TSL-Maschinenbefehl oder XCHG, Semaphoren

Gegenseitigen/gleichzeitigen Zugriff auf kritische Regionen vermeiden

- Dieses Konzept nennt man auch Mutual Exclusion

Wissen was eine kritische Region ist

- Eine kritische Region ist ein Zeitraum, in dem ein Prozess auf Variablen im geteilten Memory zugreift. In diesem Zeitraum darf kein anderer Prozess auf die selben Variablen zugreifen, weil es sonst zu Race Conditions kommen kann. Dies lässt sich durch eins der Mutual Exclusion Verfahren vermeiden.

Methoden kennen, mit denen vermieden werden kann, dass 2 Prozesse gleichzeitig in ihre kritische Region reingehen:

→ Sperrvariablen (Test, Set und Lock)

→ Semaphoren

Funktion, Vor- und Nachteile, Was sind die Unterschiede, warum ist das eine besser als das andere?

- Alle Interrupts ausschalten: Falls Prozess nicht beendet werden kann, gibt es keine Interrupts die zur Verfügung stehen, TSL: aktives Warten verschwendet CPU-Zeit, Interrupt kann dazwischen geraten, TSL-Maschinenbefehl: aktves Warten, Semaphoren: keins der beschriebenen Nachteile

Teststeuerung zur Vermeidung von Race conditions (Gemeint ist vermutlich Prozesssteuerung)

Bis Folie 51 circa

Deadlocks (nicht so im Detail)

* Was ist das?
* Foliensatz 2 ab Folie 60
* Bild auf Folie 62 – so funktioniert ein Deadlock, das macht ihn aus
* Beispiel auf Folie 61
* Was tun BS wie Windows oder Unix, um Deadlocks zu vermeiden und warum tun sie das so?

- Vogel-Strauß-Algorithmus (nichts tun): Deadlocks können entstehen müssen aber nicht, Verhindern oder Vermeiden von Deadlocks ist sehr teuer und aufwendig, es wird in Kauf genommen dass seltene Deadlocks durch Reboot kuriert werden

Scheduling:

Was gibt es für Scheduling Algorithmen (die tatsächlich angewandt werden: multilevel feedback sceduling, multilevel queue sceduling, not frequently used, aging, working set clock

- Batchsysteme: First Come First Serve, SJF, SJN, SRTN

- interaktive OS:

- Round Robin:

- gewährleistet ausgewogene Verteilung

- erzeugt viel Overhead, da keine Optimierung vorgenommen wird

- berücksichtigt nicht die unterschiedliche Wichtigkeit von Prozessen

- Priority-Scheduling:

- CPU-Zeit wird nach Priorität vergeben

- Verhungern von unwichtigen Prozessen wird verhindert durch Dynamic-Priorities oder durch Kombination mit Round Robin

- benutzt Multilevel-Queue-Scheduling: Queues werden nach Priorität gescheduled, Prozesse innerhalb des selben Queues werden per Round Robin gescheduled

- Multilevel-Feedback-Queue-Scheduling:

Prozesse, die schon viel CPU-Zeit bekommen haben werden in eine Queue der niedrigeren Priorität herabgestuft, umgekehrt in eine höhere

- Fair Share Scheduling:

- Scheduler vergibt CPU-Zeit an die Benutzer von Prozessen möglichst gleichmäßig (Linux: Completely Fair Share Scheduling, sehr komplex)

SSD

Was ist ein Block, was ist ein Sektor → Übliche Größen (512 Byte, 4K, im Dateisystem kann das dann auch größer werden)

Vergleich SSD und Festplatte:

* Wo liegt der Unterschied, was ist an einer Festplatte anders? Nicht alles ist an einer SSD besser

- HDD ist deutlich billiger und leichter erhältlich auf Markt, für viel mehr Storage, längere Lebenszeit

* Festplatte behält ihre Lebensdauer und kann beliebig oft wieder beschrieben werden – bei SSDs gibt es eine Lebenszeitgrenze
* Im Foliensatz 1
* Was war der Nachteil der SSDs?

- begrenzte Lebensdauer: SLC 100.000 Schreibvorgänge, MLC 1.000, (Oxidschicht um Floating-Gate übersteht nicht beliebig viele Schreibzyklen), außerdem deutlich teurer bei weniger Storage

* Und was tun die Hersteller, um den Nutzern klarzumachen, die Lebensdauer dieser Dinger ist begrenzt

- Angabe der Lebensdauer in TBW (Total Bytes Written) in Terabytes, außerdem. SSDs werden schon bei Produktion mit einem Überschuss an Speicherzellen versehen, Wear-Leveling bedeutet Controller verteilt Schreibvorgänge möglichst gleichmäßig auf alle Zellen

Interrupt:

* Was ist der Unterschied zwischen dem Interrupt-Handling in Einzelprozessormaschinen und Multiprozessormaschinen

- neu bei Multi: Interrupts müssen auf alle CPUs und zwischen den CPUs verteilt werden, das Interrupt-Handling kann an die Hardware delegiert werden, CPUs können sich gegenseitig aus dem Suspend erwecken, Maschinenbefehle zum Sperren und Erlauben wirken je Prozessor was bei der Synchro berücksichtigt werden muss

* Was kommt als Entscheidung hinzu

- Prozesse laufen jetzt nicht mehr sequenziell in Zeitscheiben, sondern tatsächlich in Zeitscheiben

* Was muss da gemacht werden

- entweder Spinlock und Sperrvariablen pro Prozessor oder gemeinsamer Bus muss jedes mal per TSL-Maschinenbefehl gesperrt werden, ansonsten Time Sharing / Space Sharing / Gang Sharing

* Welche Bausteine händeln das Interrupt-Thema in Multiprozessorsystemen

- spezielle APICs (Application Programmable Integrated Circuits) im Intel-Chipsatz und in jeder CPU kommunizieren miteinander

Multiprocessing

Was gehört bei Multiprozessorsystemen zusätzlich an Aufgaben für das BS dazu

* Welcher Prozessor übernimmt eine Aufgabe

- Symmetrisch: Alle gleich; asymmetrisch: Master teilt Slaves die Prozesse zu

* Wie wird der seperate Cache für jeden einzelnen Prozessor gemeinsam verwaltet (wie wird das gemacht? Protokoll)

- MESI-Protokoll, CPUs machen Bus-Snooping

Virtualisierung

* Virtualisierungmethoden
* Typ 1-, Typ2-Virtualisierung

- Typ-1: Direkt auf Hardware (Bare metal hypervisor):

- geringer Overhead (muss nicht ganze Hardware emulieren)

- im Produktiveinsatz

- Typ-2: Läuft auf OS

- keine Treiber nötig für Gastsystem, da Hardware emuliert wird

- mehr Overhead

- für Testumgebungen

* Vollvirtualisierung
* Paravirtualisierung

- OS des Gastsystems wird an das Laufen auf Hypervisor angepasst

- keine Hardwareaufrufe, sondern Gastsystem kommuniziert mit Hypervisor über spezielle APIs, bessere Leistung der VMs

* Vor- und Nachteile der Methoden

Eigentlich alle aktuell laufenden Serversysteme sind irgendwie virtualisiert

Grundbegriffe kennen:

Typ1-Virtualisierung – Hypervisor läuft direkt auf der Hardware

Typ2-Virtualisierung – Hypervisor läuft im BS

Beides zusammen = Vollvirtualisierung

* Das Gast-BS weiß nicht, dass es in einer virtuellen Umgebung läuft und arbeitet genauso, wie in einer physischen ↷ mehr Aufwand für den Hypervisor

Paravirtualisierung

* Das System arbeitet angepasst auf die Virtualisierung in einer speziellen Umgebung, Befehle können angepasst werden. Das BS weiß, dass es in einer virtuellen Umgebung arbeitet
* Was sind die Voraussetzungen dafür
* Was müssen Sie von dem BS kennen, das paravirtualisiert wird, damit Paravirtualisierung überhaupt möglich wird

- Quelltext des OS muss offen sein, damit Kernel an das Laufen auf einem Hypervisor angepasst werden kann

Praxisaufgabe: Windowsserver in einem Hyper-V installiert

* Was müssen sie machen, damit das BS überhaupt läuft?
* Den Hypervisor aktualisieren
* Hypervisor = Typ1-Vollvirtualisierung
* Wenn das BS installiert ist, Berechtigungen vergeben
* Was sind Freigabe-Berechtigungen
* Was sind NTFS-Berechtigungen
* Welche haben Wirkung in welchem Zugriff
* Wie können Sie arbeiten, um möglichst nur mit einer Berechtigungsebene arbeiten zu müssen?
* mit NTFS-Berechtigungen, damit sich das auf beide Zugriffsarten (remote und lokal) auswirkt