

Interrupt-Controller

Interrupt-Controller: Interrupt-Signale von Komponenten des Rechners **entgegen** zu nehmen, und die CPU über das Vorliegen von einem (oder mehreren) Interrupts zu **informieren**.

- Die CPU ist dann für die **Abarbeitung des Interrupts** zuständig.
- Dies geschieht, indem eine sogenannte **Interruptbehandlungsroutine** aufgerufen wird.

Interruptbehandlungsroutine (ISR): eine Reihe von Anweisungen, die einem **bestimmten Interrupt zugeordnet** ist und deren Anweisungen auf einer CPU ausgeführt werden können.

Keine negative Beeinträchtigung

- Interrupt unterbrochene Prozess muss später **ohne negative Beeinträchtigung** weiter ausgeführt werden können
- Das vom Prozess erarbeitete Ergebnis darf sich **nicht unterscheiden**

Einen Interrupt nennt man eine **präzise Unterbrechung**, falls alle der folgenden Bedingungen erfüllt sind:

1. Der Programmzähler des unterbrochenen Prozesses wird an einer bekannten Stelle gesichert.
2. **Alle** Befehle des unterbrochenen Prozesses, die **vor** dem Befehl ausgeführt werden müssen, auf den der Programmzähler zeigt, sind vollständig abgearbeitet.
3. **Kein** Befehl des unterbrochenen Prozesses, der **nach** dem Befehl ausgeführt werden muss, auf den der Programmzähler zeigt, ist bereits abgearbeitet.
4. Der Ausführungszustand des Befehls des unterbrochenen Prozesses, auf den der Programmzähler zeigt, ist bekannt.

Einen Interrupt nennt man eine **unpräzise Unterbrechung**, falls mindestens eine der für einen präzisen Interrupt genannten Bedingungen nicht erfüllt ist.

Gründe für eine Interrupt-Auslösung:

- Auslösung aufgrund einer Speicherschutzverletzung.
- Auslösung durch einen Hardware-Taktgeber zur quasi-gleichzeitigen Ausführung mehrerer Prozesse.

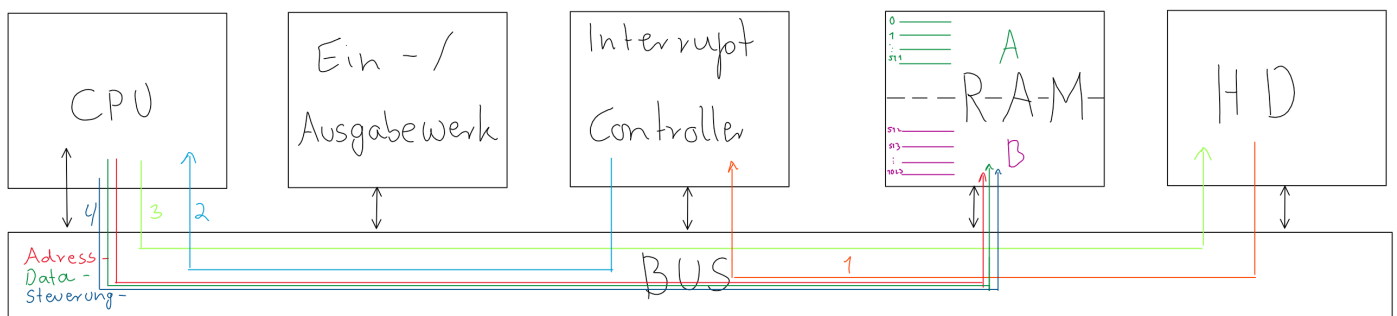
- Auslösung durch ein E/A-Gerät während der Kommunikation zwischen CPU und E/A-Gerät

Speicherschutzverletzung

→ Stellt das **Steuerwerk** während der *Abarbeitung eines Prozesses* fest, dass der *aktive Prozess* auf einen Speicherbereich im *Hauptspeicher* zugreifen möchte, der ihm **selbst nicht zugeordnet** ist, so handelt es sich um eine **Speicherschutzverletzung**, die **durch einen Interrupt angezeigt** wird.

→ **Reaktion:** Speicherschutzverletzung verursachende Prozess wird ohne zu speichern beendet

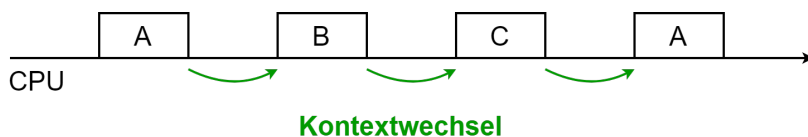
Interruptablauf - Beispiel: Eine Datei soll in den RAM-Speicher innerhalb eines Prozesses abgespeichert werden.



1. CPU schickt Anforderung eines Datenworts an die Festplatte und führt danach einen Kontextwechsel aus
2. HD löst Interrupt aus. Interrupt wird an Interrupt-Controller gesendet und in Queue gespeichert.
3. Der Interrupt-Controller meldet den Interrupt dem Steuerwerk.
4. Nach letzten Execute-Phase des aktuellen Prozesses werden Registerinhalte gesichert. Die Interruptbehandlungsroutine wird ausgeführt. Datenwort wird mit Speicherzelle und Befehl (wirte) wird an Hauptspeicher geschickt (Adressbus, Databus, Steuerbus)

Kontextwechsel

- Festplatte braucht mehr Zeit als die CPU
- CPU macht beim Warten einen Kontextwechsel -> ein anderer Prozess wird abgearbeitet
- Während Kontextwechsels werden Registerinhalte des vorherigen Prozesses gesichert und des neuen Prozesses geladen.



Kontextwechsel versus Interrupt



Der Unterschied zwischen Kontextwechsel und Interrupt, ist die Dauer der Unterbrechung

Kontextwechsel: sehr aufwendig, da viele Informationen auf unterschiedliche Register abgespeichert werden

Interrupt: arbeitet mit dem ISR und somit handelt es sich nur um ein Programm, dass für sich selbst sichert

Gründe für eine Interrupt-Auslösung

- Auslösung aufgrund einer Speicherschutzverletzung
- Auslösung durch einen Hardware-Taktgeber zur quasi-gleichzeitigen Ausführung mehrerer Prozesse.
- Auslösung durch ein E/A-Gerät während der Kommunikation zwischen CPU und E/A-Gerät.

Quasi-gleichzeitige Ausführung mehrerer Prozesse

→ indem sich die betreffenden Prozesse in **kleinen Zeiteinheiten auf der CPU abwechseln**

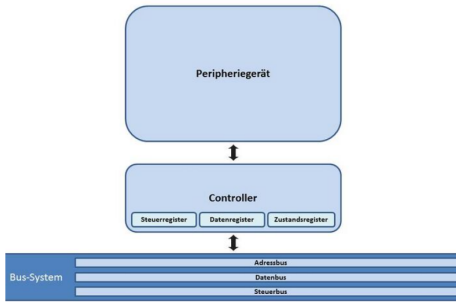
WER oder **WAS** bestimmt, wann es Zeit ist, einen Prozess zu unterbrechen und die CPU einem anderen Prozess zuzusprechen?

WER oder **WAS** bestimmt, welcher Prozess als nächstes die CPU bekommt?

→ **Wir brauchen ein Betriebssystem!**

Hardware-Taktgeber löst in sehr kleinen zeitlichen Abständen einen Interrupt auslöst

→ Wird entschieden, dass es Zeit für einen **Prozesswechsel** auf CPU ist, so kann ein *anderer Teil des Betriebssystems aufgerufen werden, der den nächsten Prozess auswählt und ihm die CPU übergibt.*



Kommunikation mit E/A-Geräte (Festplatte/Monitor/Maus/Tastatur/...)

→ CPU kommuniziert nicht direkt mit diesen Komponenten, sondern mit einem Controller, speziell für die Komponente zuständig

- **Steuerregister**
- **Datenregister:** Datenwort hinterlegen
- **Zustandsregister**

Datentransfer und Interrupts

- Die CPU sendet die Adresse des gewünschten **Datenwortes** an den **Festplatten**-Controller.
- Der **Festplatten**-Controller besorgt das Datenwort von der **Festplatte** und stellt es in seinem Datenregister zur Verfügung.
- Der **Festplatten**-Controller sendet einen Interrupt zum **Interrupt**-Controller.
- Der **Interrupt**-Controller nimmt den **Interrupt** entgegen und verwaltet ihn.
- Der **Interrupt**-Controller informiert die CPU über den Interrupt des **Festplatten**-Controllers.
- Die CPU startet die zugehörige **Interruptbehandlungsroutine**.
- Die **Behandlungsroutine** kopiert das **Datenwort** in ein Register auf der CPU.
- Die **Behandlungsroutine** sendet das Datenwort von der CPU zum Hauptspeicher.
- Der Hauptspeicher legt das **Datenwort** in der gewünschten Speicherzelle ab.
- Die **Interruptbehandlungsroutine** informiert den **Interrupt**-Controller darüber, dass der **Interrupt** fertig behandelt ist.

Revision #2

Created 1 October 2022 16:00:10 by Merith Holtmann

Updated 2 October 2022 10:34:03 by Merith Holtmann