Was ist präemptives Scheduling und wie kann es realisiert werden?
The scheduling in which a running process can be interrupted if a high priority process enters the queue and is allocated to the CPU is called **preemptive scheduling**. In this case, the current process switches from the running queue to ready queue, and the high priority process utilizes the CPU cycle.

Es wird realisiert durch Interrupts die die aktuellen Prozessen unterbrechen und Coroutinen, die den Kontroll-Flüss wechseln

• Wie kann man eine quasiparallele Abarbeitung von Prozessen durch ein präemptives Scheduling erreichen?
Es kann aber zu einer "quasi-gleichzeitigen" Ausführung mehrerer Prozesse kommen, indem sich die betreffenden Prozesse in kleinen Zeiteinheiten auf der CPU abwechseln. Sind die "kleinen Zeiteinheiten" hierbei genügend klein gewählt, so erhält ein Anwender vor dem System den Eindruck, als ob alle Prozesse parallel ablaufen.

• Was wäre ein passendes Beispiel für einen nebenläufigen Zugriff auf gemeinsame Daten?
Wenn ein Interrupt ausgelöst wird ,der die Ready-Liste der Schedular manipulieren will. und der Scheduler auf der andern Seite hat ein Prozess am Laufen, der selber was an der Readyliste tut dann passieren komische Effekte

• Was ist ein Programmable Interval Timer?
Der ist ein Wecker, der nach ein bestimmten Zeit periodisch Interrupts auslöst.

• Wie funktioniert der Programmable Interval Timer (PIT)?
(sehe Vorlage)

• Wie kann man diesen konfigurieren (Beschreibt eine Konfigurationssequenz!)?

• Was ist ein Interrupt? https://www.electronicshub.org/types-of-interrupts-and-how-to-handle-interrupts/

Interrupt is a signal which has highest priority from hardware or software which processor should process its signal immediately.

• Wie kann die CPU Interrupts unterscheiden? interrupts have highest priority than other signals, there are many type of interrupts but basic type of interrupts are
Hardware Interrupts: If the signal for the processor is from external device or hardware is called hardware interrupts.
Software Interrupts: Software interrupt can also divided in to two types. They are

- **Normal Interrupts:** the interrupts which are caused by the software instructions are called software instructions.

- **Exception:** unplanned interrupts while executing a program is called Exception. For example: while executing a program if we got a value which should be divided by zero is called a exception.

Hardware:

A hardware interrupt request (IRQ) is an electronic signal issued by a hardware device which is external to the processor, to communicate that the device needs attention from the operating system (OS)[3] or, if there is no OS, from the "bare-metal" program running on the CPU.

Software

A software interrupt is requested by the processor itself upon executing particular instructions or when certain conditions are met. Every software interrupt signal is associated with a particular interrupt handler.

• Wie werden Interrupts behandelt?

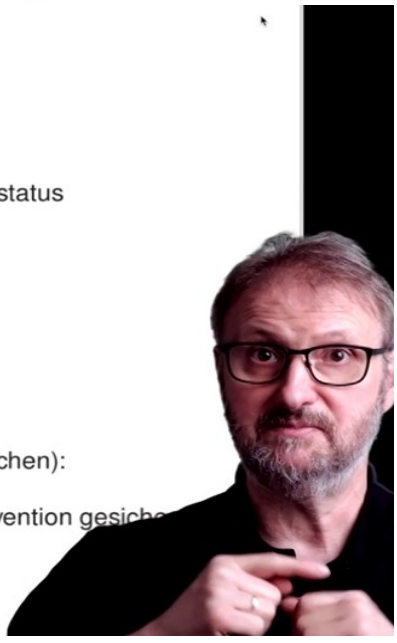
Die Behandlungsroutine

1. ... *quittiert* den Interrupt am Gerät bzw. am *Interrupt Controller*
   - ... um ggf. das Signal auszuschalten bzw. erneut zuzulassen
2. ... *überprüft* den Zustand des Gerätes
   - ... ggf. Daten vom Gerät abholen bzw. dem Gerät zuführen
3. ... *weckt* ggf. wartende Prozesse *auf*
4. ... und *kehrt zurück* zum unterbrochenen Programm

Unterbrechungsbehandlung

- die *Hardware* sichert i.A. nur den minimalen Prozessorstatus
  - wobei "minimal" ein dehnbarer Begriff ist
    - nichts bis einige hundert Bytes!
  - typischerweise (bei den gängigen Prozessoren):
    - *Statusregister*
    - *Programmadresse* (PC bzw. IP)
- die *Software* sichert, was sie selbst verändern wird
  - auch das ist sehr dehnbar
  - typischerweise (bei den gängigen Prozessoren/Sprachen):
    - *flüchtige Register* der CPU
    - nicht flüchtige Register werden nach Prozedurkonvention gesich

Rechner- und Betriebssysteme — Traps/Interrupts



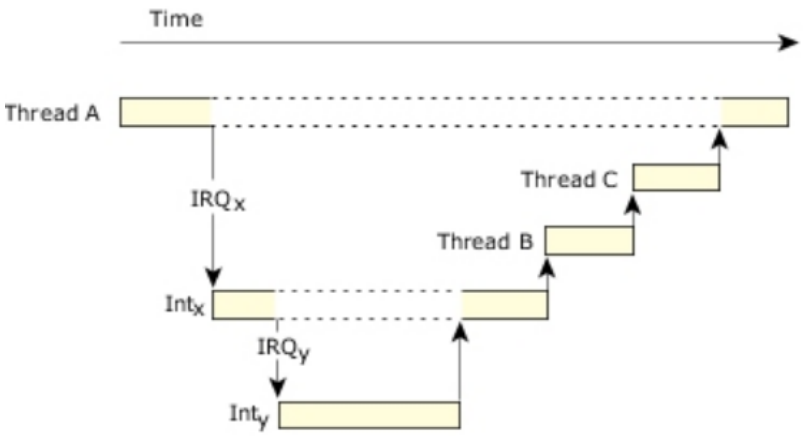
• Können neue Interrupts während einer Interruptsbehandlung auftreten? • Wenn ja, werden diese dann umgehend behandelt? • Wenn nein, werden die blockierten Interrupts durch das Gerät später nochmal ausgelöst?

Nein aber können verschachtelte (nested) interrupt sein. Es passieren immer bei Interrupts mit höhne Prioritäten und wird immer der Interrupt mit der niedrigsten Schicht in der Verschachtlung d.h Interrupts mit höchste Priorität wird erst behandelt dann der abgebrochene alter interrupt und so weite bis zu user mode (no interrupts any more)

Normally, an **interrupt service routine** proceeds until it is complete without being interrupted itself in most of the systems. However, If we have a larger system, where several devices may interrupt the microprocessor, a **priority** problem may arise.

If you set the **interrupt enable flag** within the current interrupt as well, then you can allow further interrupts that are **higher priority** than the one being executed. This "interrupt of an interrupt" is called a **nested interrupt**. It is handled by stopping execution of the original service routine and storing another sequence of registers on the **stack**. This is similar to **nested subroutines**. Because of the automatic decrementing of the stack pointer by each interrupt and subsequent incrementing by the RETURN instruction, the first interrupt service routine is resumed after the second interrupt is completed, and the interrupts are serviced in the proper order. Interrupts can be nested to any depth, limited only by the amount of **memory available for the stack**.

For example, In the following diagram, Thread A is running. Interrupt IRQx causes interrupt handler Intx to run, which is preempted by IRQy and its handler Inty. Inty returns an event causing Thread B to run; Intx returns an event causing Thread C to run.



Image

• Zwei Prozesse A und B sollen in einer Endlosschleife immer wieder ihren Namen ausgeben. Diese Ausgabe dauert 10 Takte. Wenn Prozess A eine Zeitscheibe von 10 Takten und Prozess B eine Zeitscheibe von 100 Takten zugewiesen wird, welche Ausgaben sind zu erwarten?

(Pending)

Durch Überlappungen wird Eine Meschung aus den beiden Namen am Ende jede 10 Takten sein. Da die beiden prozesse aud der Readyliste zugreifen und nicht synchronisiert arbeiten d.h. nicht erst läuft 10 Takten für A dann 10 für B sonder jetzt nimmen die beiden Prozesse an die 10 Takten teilen d.h. also Sei z.b. A hat der name a1a2...a10 und B hat b1...b10 dann das output sollte in der ersten 10 Takten eine Mischung aud a‘s und b‘s und danach der Name von b 9 mals.