Wie erfolgt die Interrupt-Behandlung bisher? Wenn ein Interrupt eintritt, ab wann können neue Interrupts auftreten?

Bisher existiert eine Handle-Methode für jeden möglichen Interrupt. Wird der Interrupt ausgelöst, so werden alle anderen Interrupts gesperrt, die Handle-Methode ausgeführt und erst nach kompletter Ausführung werden alle Interrupts wieder reaktiviert.

Wozu dienen die Interrupt-Sperren (IntLock) und wann ist ihr Einsatz bisher notwendig?

Interrupt-Sperren dienen dazu, kritische Abschnitte zu schützen. Ihr Einsatz ist nötig, wenn eine Unterbrechung an einer bestimmten Stelle zur Folge hätte, dass das Ergebnis beeinträchtigt werden kann.

Interrupt locks are used to protect critical sections. Their use is necessary if an interruption at a certain point would have the effect of affecting the result.

Wozu dienen bei der Interrupt-Behandlung Prolog und Epilog?

Bei Prolog- und Epilog-Aufteilung wird die vorige Handle-Methode in einen Teil aufgeteilt, der zuerst ausgeführt wird (Prolog), und einen, der danach ausgeführt wird (Epilog).

Worin unterscheiden sie sich?

Im Prolog befinden sich Anweisungen, deren Ausführung essentiell ist und für die eine kurzzeitige Interrupt-Sperre notwendig ist. Für die Anweisungen im Epilog, die auch später ausgeführt werden können, gilt diese Sperre nicht mehr.

Welchen Vorteil bringt die Aufteilung?

Durch die Aufteilung werden nicht mehr während der gesamten Behandlung eines Interrupts alle anderen Interrupts gesperrt, sondern nur noch so lange, wie es unbedingt nötig ist.

Was ist ein Monitor?

Ein Monitor ist ein Konzept zur Synchronisation für eine Menge von Funktionen.

Wozu wird er verwendet? Welches Konzept ersetzt er?

Der Monitor wird verwendet, um für die gesamte Menge von Funktionen einen wechselseitigen Ausschluss zu garantieren. Er ersetzt somit das Konzept von manuellem Sperren und Synchronisieren, da über seine gröbere Granularität Einzelbetrachtungen überflüssig werden.

The monitor is used to guarantee mutual exclusion for the entire set of functions. It thus replaces the concept of manual locking and synchronization, since its coarser granularity makes individual observations superfluous.

Was ist ein Semaphor?

Ein Semaphor ist ein abstrakter Datentyp zur Steuerung von Prozessen. Er spiegelt ein Signal aus der Bahntechnik wider und es gibt ihn als zählenden oder binären Semaphor (letztere werden mit 1 initialisiert). Semaphoren werden durch wait() / V() / up() und signal() / P() / down() gesteuert.

Erreicht der Wert des Semaphors 0, so wird der aufrufende Prozess gesperrt und es entsteht eine Warteschlange (Queue). Wird der Wert wieder größer als 0, so werden die Prozesse wieder entsperrt und ausgeführt.

Was ist private Vererbung?

Erbt eine Klasse privat von der Elternklasse (K1), so kann die erbende Klasse (K2) alle Methoden und Attribute wie üblich nutzen (sofern sie für K2 sichtbar sind).

In K2 werden nun diese geerbten Methoden und Attribute als privat behandelt. Das bedeutet: Erbt nun erneut eine Klasse (K3) von K2, so kann es die Methoden und Attribute von K1 nicht mehr verwenden.

(Ähnlich verhält es sich z. B. unter Java mit „protected“.)