

Interrupts freigeben und sperren

Bitte nicht stören?

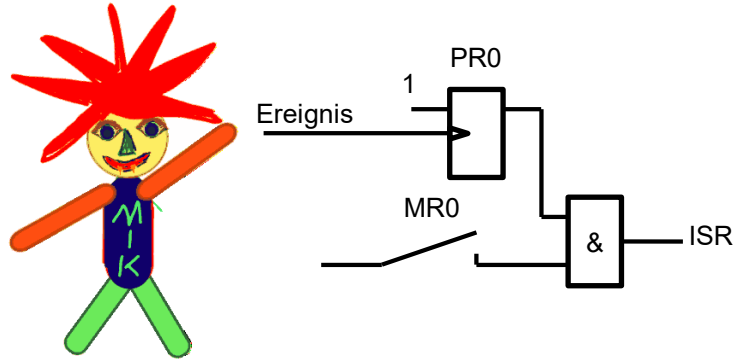


Ich bin Mik, Dein Mikrocontroller



Interrupts freigeben und sperren

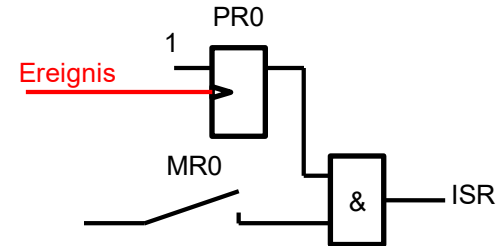
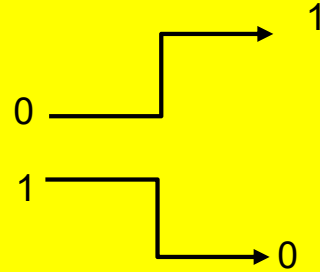
Ein Ereignis kann einen Interrupt auslösen.
`InterruptIn meinInterrupt(PA_0)`



Interrupts freigeben und sperren

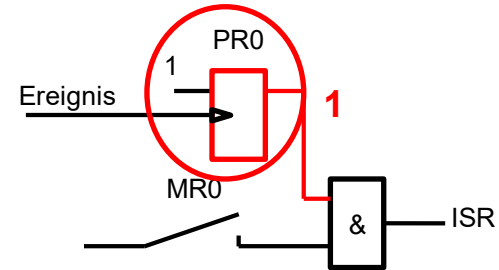
Ereignisse z. B. können sein:

- Steigende Flanke an PA_1
`meinInterrupt.rise(&ISR);`
- Fallende Flanke an PA_1
`meinInterrupt.fall(&ISR);`



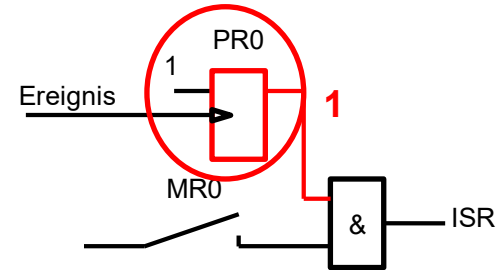
Interrupts freigeben und sperren

Das Ereignis bewirkt, dass ein Flipflop, das sogenannte Pendingbit PRx, hier PR0 gesetzt wird.



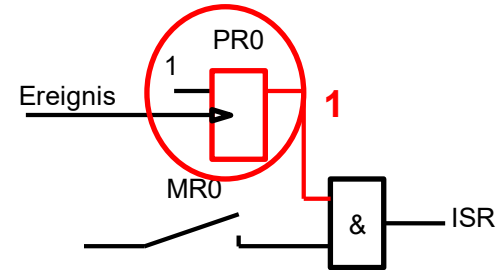
Interrupts freigeben und sperren

Damit die ISR ausgeführt wird sind weitere Voraussetzungen zu erfüllen:



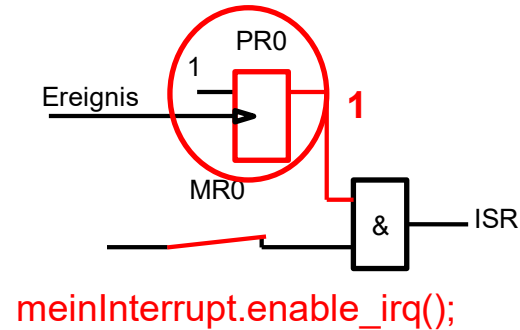
Interrupts freigeben und sperren

1. Ein Bit Namens MRx Maskregister x, hier MR0
muss gesetzt sein:
`meinInterrupt.enable_irq();`



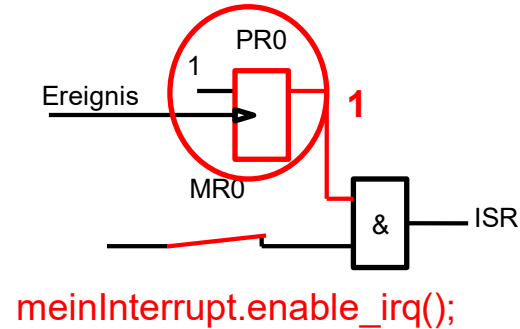
Interrupts freigeben und sperren

1. Ein Bit Namens MRx Maskregister x, hier MR0 muss gesetzt sein:
`meinInterrupt.enable_irq();`



Interrupts freigeben und sperren

Damit kann jeder Interrupt individuell:
Freigegeben werden: `meinInterrupt.enable_irq();`
Oder
Gesperrt werden: `meinInterrupt.disable_irq();`
Man nennt das: **Individuelle Freigabe**



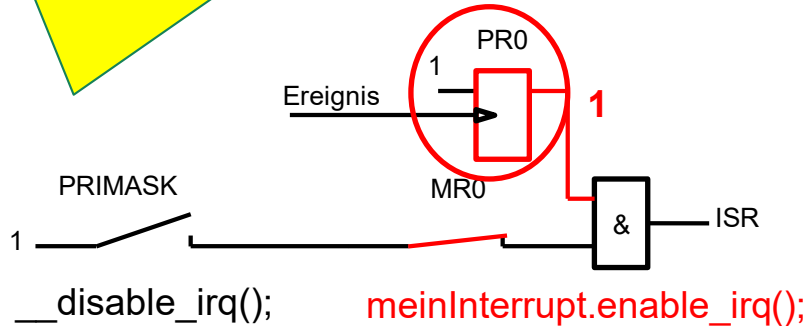
Interrupts freigeben und sperren

2. Globale Freigabe

Mit dem Bit PRIMASK (Priority Mask) können alle Interrupts

Gesperrt: `__disable_irq();`

Oder



Interrupts freigeben und sperren

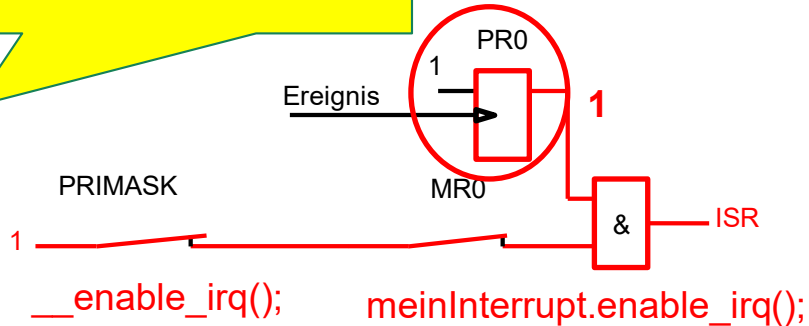
2. Globale Freigabe

Mit dem Bit PRIMASK (Priority Mask) können alle Interrupts

Gesperrt: `__disable_irq();`

Oder

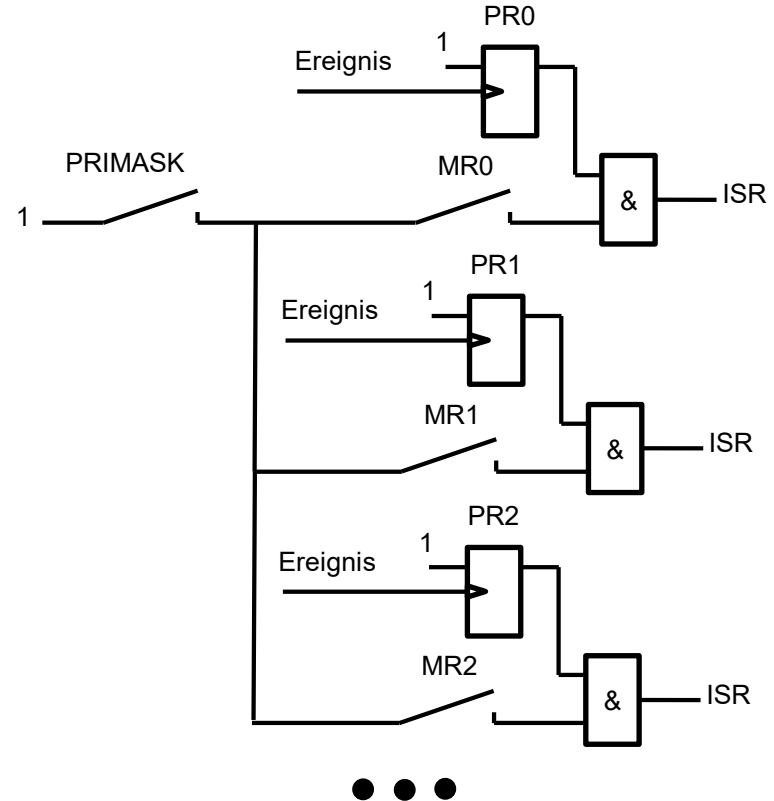
Freigegeben: `__enable_irq();`
werden



Interrupts freigeben und sperren



Mit der globalen Freigabe können alle Interrupts auf einmal gesperrt werden!



Interrupts freigeben und sperren



Standardmäßig sind bei MBED immer alle Freigaben erteilt. D.h. Wir müssen uns um die Freigaben meist nicht kümmern.

