

INTERRUPT

Reagieren auf unerwartete Ereignisse



INTERRUPT

Der μ Controller muss schnell und unmittelbar jederzeit bereit sein auf folgende Ereignisse zu reagieren:

- Not-Aus-Taster
- Lichtschranke
- Andere Signale mit höchster Priorität



INTERRUPT

Funktionsprinzip:



INTERRUPT

Funktionsprinzip:



INTERRUPT

Funktionsprinzip:



Hauptprogramm



INTERRUPT

Funktionsprinzip:

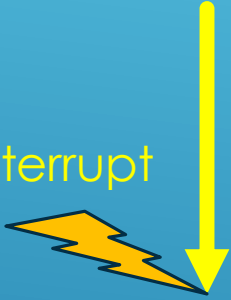


INTERRUPT

Funktionsprinzip:

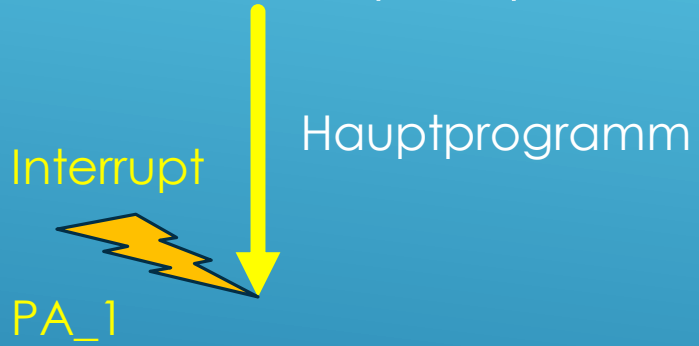
Hauptprogramm

Interrupt



INTERRUPT

Funktionsprinzip:



Beim Interrupt handelt es sich um ein Ereignis wie z.B. Ein Tastendruck auf PA_1, verursacht vielleicht durch eine Lichtschranke.

Ereignisse sind immer Veränderungen am Port, z.B. Der Port geht von 0 nach 1 oder von 1 nach 0. Bei Tastendruck geht PA_1 von 0 nach 1



INTERRUPT

Funktionsprinzip:

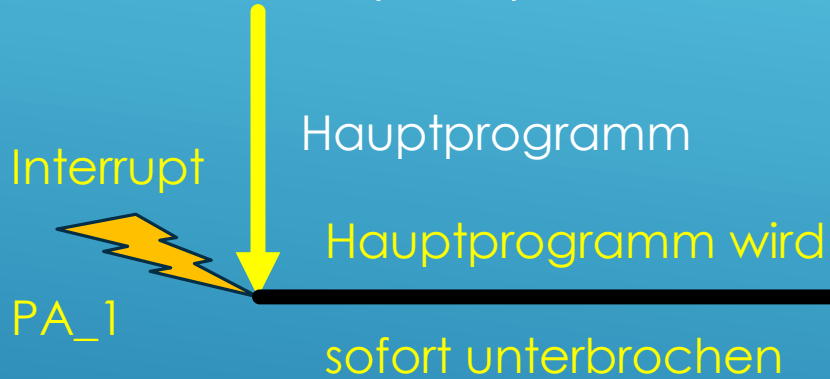


Beim Interrupt handelt es sich um ein Ereignis wie z.B. Ein Tastendruck auf PA_1, verursacht vielleicht durch eine Lichtschranke.



INTERRUPT

Funktionsprinzip:



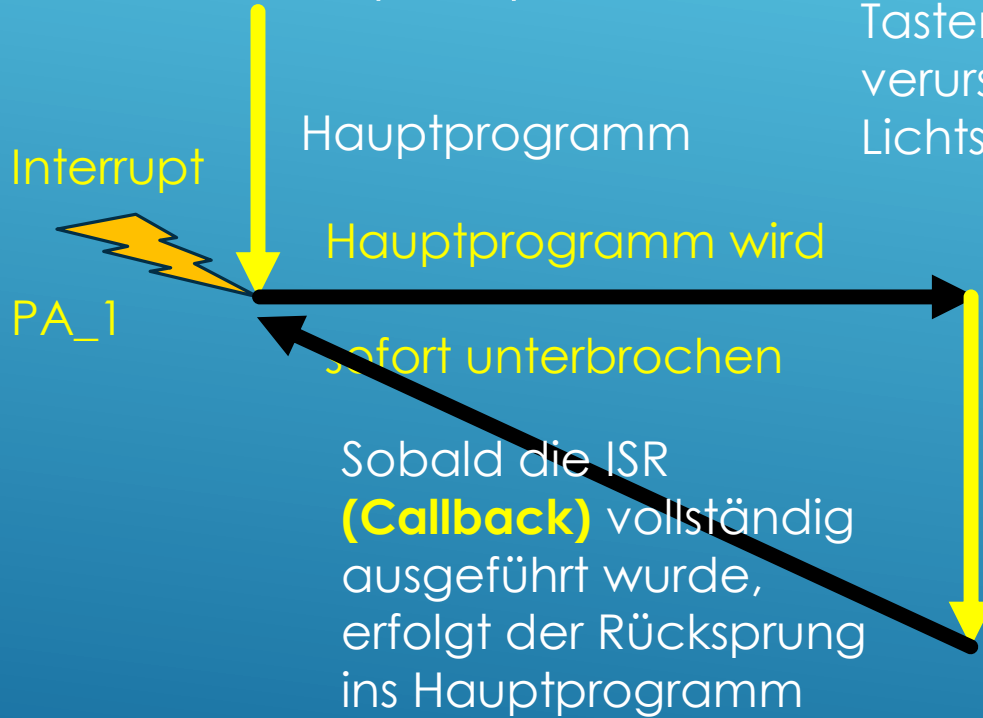
Beim Interrupt handelt es sich um ein Ereignis wie z.B. Ein Tastendruck auf PA_1, verursacht vielleicht durch eine Lichtschranke.

Ein spezielles Unterprogram, die sogenannte Interrupt Service Routine kurz ISR bzw. **Callback** wird automatisch gestartet und führt die programmierten Instruktionen aus.



INTERRUPT

Funktionsprinzip:



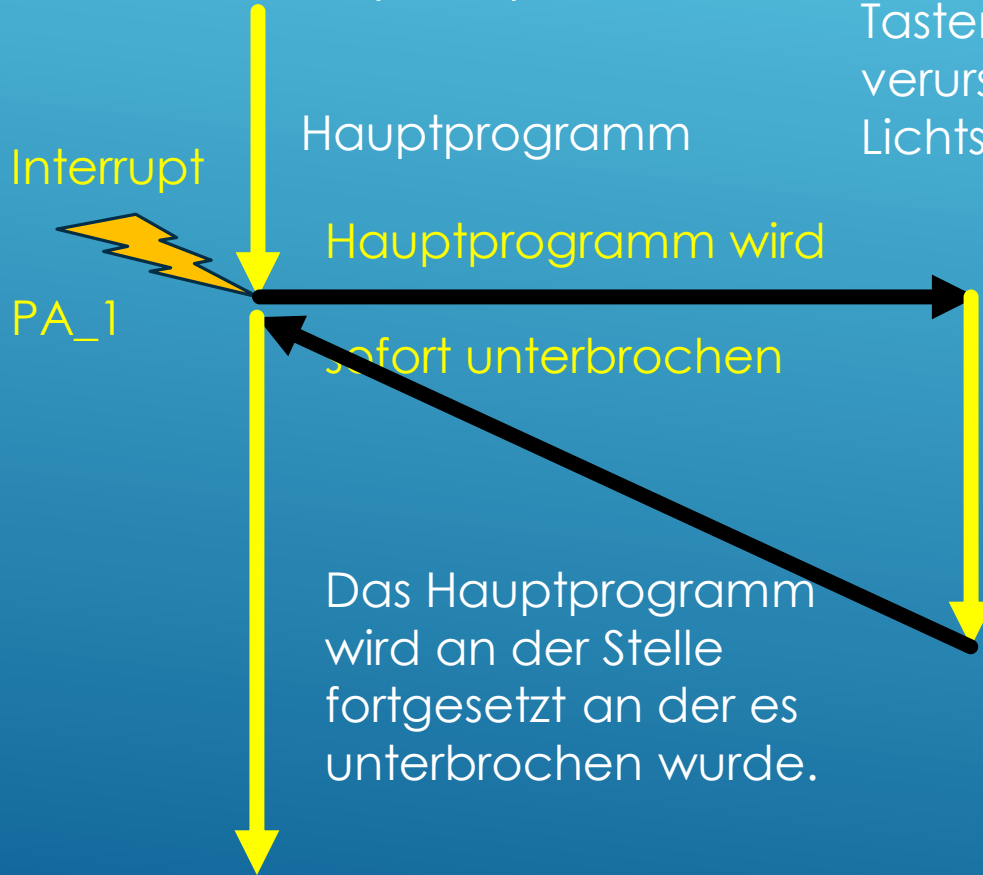
Beim Interrupt handelt es sich um ein Ereignis wie z.B. Ein Tastendruck auf PA_1, verursacht vielleicht durch eine Lichtschranke.

Ein spezielles Unterprogramm, die sogenannte Interrupt Service Routine kurz ISR ISR bzw. **Callback** wird automatisch gestartet und führt die programmierten Instruktionen aus.



INTERRUPT

Funktionsprinzip:



Beim Interrupt handelt es sich um ein Ereignis wie z.B. Ein Tastendruck auf PA_1, verursacht vielleicht durch eine Lichtschranke.

Ein spezielles Unterprogramm, die sogenannte Interrupt Service Routine kurz ISR ISR bzw. **Callback** wird automatisch gestartet und führt die programmierten Instruktionen aus.



INTERRUPT

Voraussetzungen:

1. Einen Port als InterruptIn deklarieren: `InterruptIn taste(PA_1);`
2. Bei unserem Mikrocontrollerboard wird noch ein PullDown benötigt:
`taste.mode(PullDown);` (Das hat was mit der Taste, nicht aber mit dem Interrupt zu tun)
3. Interrupt Service Routine zuweisen: `taste.rise(&Name_des_Unterprogramms);`
für steigende Flanke an PA_1 oder
`taste.fall(&Name_des_Unterprogramms);`
für fallende Flanke an PA_1 oder beides
4. Den Interrupt freigeben: `taste.enable_irq();` Macht den Interrupt scharf.
5. Die Interrupt Service Routine schreiben. Es handelt sich um ein ganz normales Unterprogramm.



INTERRUPT

Programmbeispiel:

```
DigitalOut led(PC_0);
```

```
InterruptIn taste1(PA_1);
```

PA_1 als Interruptquelle mit Namen taste festlegen

```
void taste1ISRrise(void)
```

```
{ led=1; }
```

```
void taste1ISRfall(void)
```

```
{ led=0; }
```

```
int main()
```

```
{
```

```
    taste1.mode(PullDown);
```

```
    taste1.rise(&taste1ISRrise);
```

```
    taste1.fall(&taste1ISRfall);
```

```
    taste1.enable_irq();
```

```
    while (true) {
```

```
    }
```

```
}
```



INTERRUPT

Programmbeispiel:

```
DigitalOut led(PC_0);
```

```
InterruptIn taste1(PA_1);
```

```
void taste1ISRrise(void)
```

```
{ led=1; }
```

```
void taste1ISRfall(void)
```

```
{ led=0; }
```

```
int main()
```

```
{
```

```
    taste1.mode(PullDown);
```

```
    taste1.rise(&taste1ISRrise);
```

```
    taste1.fall(&taste1ISRfall);
```

```
    taste1.enable_irq();
```

```
    while (true) {
```

```
    }
```

```
}
```

PA_1 als Interruptquelle mit Namen taste festlegen

Unterprogramm taste1ISRrise schaltet die led ein



INTERRUPT

Programmbeispiel:

```
DigitalOut led(PC_0);
```

```
InterruptIn taste1(PA_1);
```

```
void taste1ISRrise(void)
```

```
{ led=1; }
```

```
void taste1ISRfall(void)
```

```
{ led=0; }
```

```
int main()
```

```
{
```

```
    taste1.mode(PullDown);
```

```
    taste1.rise(&taste1ISRrise);
```

```
    taste1.fall(&taste1ISRfall);
```

```
    taste1.enable_irq();
```

```
    while (true) {
```

```
    }
```

```
}
```

PA_1 als Interruptquelle mit Namen taste festlegen

Unterprogramm taste1ISRrise schaltet die led ein

Unterprogramm taste1ISRfall schaltet die led aus



INTERRUPT

Programmbeispiel:

```
DigitalOut led(PC_0);
```

```
InterruptIn taste1(PA_1);
```

```
void taste1ISRrise(void)
```

```
{ led=1; }
```

```
void taste1ISRfall(void)
```

```
{ led=0; }
```

```
int main()
```

```
{
```

```
  taste1.mode(PullDown);
```

```
  taste1.rise(&taste1ISRrise);
```

```
  taste1.fall(&taste1ISRfall);
```

```
  taste1.enable_irq();
```

```
  while (true) {
```

```
  }
```

```
}
```

PA_1 als Interruptquelle mit Namen taste festlegen

Unterprogramm taste1ISRrise schaltet die led ein

Unterprogramm taste1ISRfall schaltet die led aus

Unsere Tasten benötigen PullDown



INTERRUPT

Programmbeispiel:

```
DigitalOut led(PC_0);
```

```
InterruptIn taste1(PA_1);
```

```
void taste1ISRrise(void)
```

```
{ led=1; }
```

```
void taste1ISRfall(void)
```

```
{ led=0; }
```

```
int main()
```

```
{
```

```
    taste1.mode(PullDown);
```

```
    taste1.rise(&taste1ISRrise);
```

```
    taste1.fall(&taste1ISRfall);
```

```
    taste1.enable_irq();
```

```
    while (true) {
```

```
    }
```

```
}
```

PA_1 als Interruptquelle mit Namen taste festlegen

Unterprogramm taste1ISRrise schaltet die led ein

Unterprogramm taste1ISRfall schaltet die led aus

Unsere Tasten benötigen PullDown

Bei steigender Flanke (0->1=Tastendruck) taste1ISRrise aufrufen



INTERRUPT

Programmbeispiel:

```
DigitalOut led(PC_0);
```

```
InterruptIn taste1(PA_1);
```

```
void taste1ISRrise(void)
```

```
{ led=1; }
```

```
void taste1ISRfall(void)
```

```
{ led=0; }
```

```
int main()
```

```
{
```

```
    taste1.mode(PullDown);
```

```
    taste1.rise(&taste1ISRrise);
```

```
    taste1.fall(&taste1ISRfall);
```

```
    taste1.enable_irq();
```

```
    while (true) {
```

```
    }
```

```
}
```

PA_1 als Interruptquelle mit Namen taste festlegen

Unterprogramm taste1ISRrise schaltet die led ein

Unterprogramm taste1ISRfall schaltet die led aus

Unsere Tasten benötigen PullDown

Bei steigender Flanke (0->1=Tastendruck) taste1ISRrise aufrufen

Bei fallender Flanke (1->0=losgelassen) taste1ISRfall aufrufen



INTERRUPT

Programmbeispiel:

```
DigitalOut led(PC_0);
```

```
InterruptIn taste1(PA_1);
```

```
void taste1ISRrise(void)
```

```
{ led=1; }
```

```
void taste1ISRfall(void)
```

```
{ led=0; }
```

```
int main()
```

```
{
```

```
    taste1.mode(PullDown);
```

```
    taste1.rise(&taste1ISRrise);
```

```
    taste1.fall(&taste1ISRfall);
```

```
    taste1.enable_irq();
```

```
    while (true) {
```

```
    }
```

```
}
```

PA_1 als Interruptquelle mit Namen taste festlegen

Unterprogramm taste1ISRrise schaltet die led ein

Unterprogramm taste1ISRfall schaltet die led aus

Unsere Tasten benötigen PullDown

Bei steigender Flanke (0->1=Tastendruck) taste1ISRrise aufrufen

Bei fallender Flanke (1->0=losgelassen) taste1ISRfall aufrufen

Interrupt freigeben.



INTERRUPT

Jetzt probieren wir das aus!!!

