**Interruptquellen und Interruptfreigabe**

Durch Interrupts wird der normale Programmablauf unterbrochen. Dies bietet die Möglichkeit, auf bestimmte Ereignisse schnell und sicher zu reagieren (Bsp. Not-AUS). Abfallende Flanken oder Low-Signale können externe Interruptereignisse **(INT0/, INT1/)** auslösen, überlaufende Timer lösen Timer-Interrupts **(TF0, TF1, TF2)** aus, über die serielle Schnittstelle gesendete **(TI)** oder empfangene **(RI)** Zeichen können Interrupts auslösen.

Die Interrupt-Anfoderungs-Bits (s.u.) werden immer gesetzt, der entsprechende Interrupt wird jedoch nur dann bearbeitet, wenn die Interrupts freigegeben sind und die globale Interrupt-Freigabe **EA** = 1 ist. Zur Bearbeitung der Interrupts springt der Controller zu festgelegten Einsprungadressen im Programmspeicher. Dort wird kurzer Code direkt ausgeführt, oder bei umfangreicheren Aufgaben, zu einer Interrupt-Service-Routine (ISR) gesprungen.

Nach erfolgter Bearbeitung eines Interrupt setzt der Controller das Programm wieder an der Stelle des Auftretens der Unterbrechung fort. Dazu wird die Rücksprungadresse im Stapelspeicher (Stack) des Controllers gespeichert. Der Befehl **RETI** beendet die Interruptverarbeitung und schreibt die Rücksprungadresse vom Stapelspeicher (Stack) in den Programmzähler (PC).

## Interruptfreigabe

Register **IEN** (0A8h) bitadressierbar (Reset: 0x0000000b)

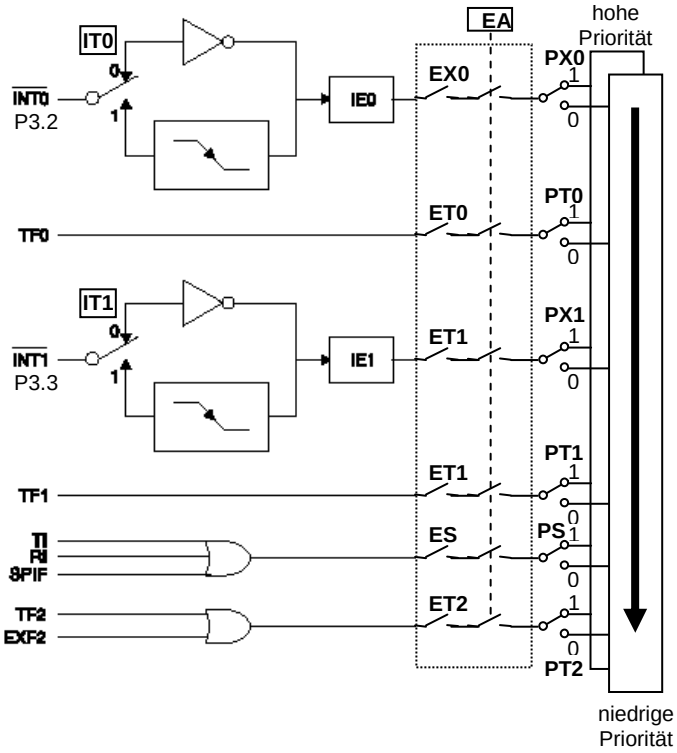
| **EA** | **-** | **ET2** | **ES** | **ET1** | **EX1** | **ET0** | **EX0** |
|---|---|---|---|---|---|---|---|
| AFh | AEh | ADh | ACh | ABh | AAh | A9h | A8h |
| | | | | | | | extInt0 |
| | | | | | | Timer 0 | |
| | | | | | externer Interrupt 1 (extInt1) | | |
| | | | | Timer 1 | | | |
| | | | Serielle Schnittstelle Interrupt | | | | |
| | | Timer 2 **(0: Interrupt gesperrt; 1: freigegeben)** | | | | | |
| | reserviert | | | | | | |
| Globale Interruptfreigabe: EA = 1 → Interrupts können individuell freigegeben werden! | | | | | | | |

## Interruptpriorität und Prioritätsebenen

Register **IP** (0B8h) bitadressierbar (Reset: 0x0000000b)

| **-** | **-** | **PT2** | **PS** | **PT1** | **PX1** | **PT0** | **PX0** |
|---|---|---|---|---|---|---|---|
| BFh | BEh | BDh | BCh | BBh | BAh | B9h | B8h |
| | | | | | | | extInt0 |
| | | | | | | Timer 0 | |
| | | | | | extInt1 | | |
| | | | | Timer 1 | | | |
| | | | seriell | | | | |
| | | Timer 2 **(0: niedrige Priorität; 1: hohe Priorität)** | | | | | |
| | reserviert | | | | | | |
| reserviert | | | | | | | |

**Interruptquellen und Steuerung (vereinfacht)**




## Einsprungadressen

| **Interrupt** | **Programmspeicher Einsprungadresse** | **Interrupt-Anforderungs-Bit** |
|---|---|---|
| Externer Interrupt 0 | **0003h** | IE0 |
| Timer0 - Überlauf | 000Bh | TF0 |
| Externer Interrupt 1 | **0013h** | IE1 |
| Timer1 – Überlauf | 001Bh | TF1 |
| Serieller Port | 0023h | TI oder RI |
| Timer2 – Überlauf oder externer Interrupt 2 | 002Bh | TF2 oder EXF2 |

Es sei hier angemerkt, dass der AT89c5131-Controller über weitere Interruptquellen verfügt und auch 2 zusätzliche Prioritäsebenen besitzt.

**Programmierung der externe Interrupts beim AT89C51..**

Mit den Portpins P3.2 und P3.3, an die auch jeweils ein Taster angeschlossen ist, können externe Interrupts ausgelöst werden.
Die Interrupts werden nur ausgelöst, wenn sie durch Setzen der Bits EX0 bzw. EX1 freigegeben sind. Wähle unbedingt die Flankensteuerung, da sonst solange Interrupts ausgelöst werden, wie an den Pins Low-Signal anliegt.

| Interrupt | | Einsprung Adresse | Port-Bit | Freigabe mit | Flanken-gesteuert | auslösende Flanke | gesetzes Bit bei Interrupt-Anforderung |
|---|---|---|---|---|---|---|---|
| ____ INT0 | externer Interrupt0 | 0003 | P3.2 | setb EX0 (IEN0) | setb IT0 (TCON) | abfallend | IE0 (TCON) |
| ____ INT1 | externer Interrupt1 | 0013 | P3.3 | setb EX1 (IEN0) | setb IT1 (TCON) | abfallend | IE1 (TCON) |

**Beispiel**

```
;*****************************************************************************************
;* Übung 8: Interruptprogrammierung
;* Beschreibung: Interruptgesteuertes Zählen der (Prell-)impulse eines Tasters
;*           Taster an Port 3.2, Nullsetzen mit Taster P3.3, 8 LEDs an Port2
;*****************************************************************************************
                include AT89c5131.inc           ;Registeradressen einbinden

                taster  bit     p3.2            ;Taster, dessen (Prell-) Impulse gezählt werden
                cReset  bit     p3.3            ;Taster zum Rücksetzen der gezählten Impulse (CounterReset)
                LEDs    equ     p2              ;Anzahl der gezählten Impulse dual anzeigen

;******** Programm ************************************************************************
                code at 0                       ;Anfangsadresse Programm
                ljmp    init                    ;Interrupt -Einsprung-Adresse überspringen

;******** Interrupt-Service-Routine ***************************************************
                org     0003h                   ;Interrupt-Einsprungadresse
extInt0:        inc     LEDs                    ;Port um 1 erhöhen, da Flanke aufgetreten
                reti                            ;Rückkehr aus dem Interrupt

;********* Initialisierung ********************************************************************
init:           mov     LEDs,#0                 ;Port mit LEDs = 0 setzen
                setb    IT0                     ;ext. Interrupt0 bei abfallender Flanke auslösen
                setb    EX0                     ;ext. Interrupt0 freigeben
                setb    EA                      ;globale Interruptfreigabe

;********** Hauptprogramm, das von den Interrupts unterbrochen wird ***************************
haupt:          jnb     cReset,haupt            ;Warten auf Interrupt oder Zähler Rücksetzbefehl
                mov     LEDs,#0                 ;Port mit LEDs = 0 setzen
                sjmp    haupt
```

**Aufgaben:**

- *Tausche die Funktionen der Taster aus (p3.3-Impulse zählen, p3.2-Nullsetzen)*
  *Achtung: die Interrupt-Einsprungadresse ändert sich!*

- *Zählen und Zähler-Rücksetzen soll interruptgesteuert funktionieren.*

- *Teste die Interrupt-Prioritäten, in dem der eine Taster immer bei Low-Signal Interrupts auslöst und der „zweite Taster versucht“, den Interrupt zu unterbrechen.*