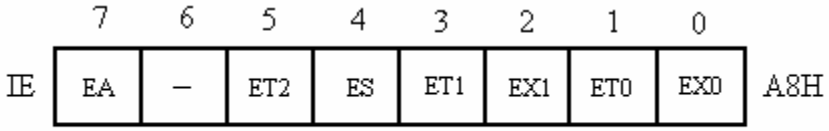
7. Interrupt

Interrupt = Programmunterbrechung
- 2 externe Interrupts
- 4 interne (drei Zeitgeber/Zähler 0 / 1 / 2 und eine serielle Schnittstelle / SPI)

Bei den 51'iger µC gibt es 2 Prioritätsebenen, die festlegen welcher Interrupt-Befehl zuerst abgearbeitet wird bzw. welcher Interrupt den anderen Interrupt unterbrechen darf. Bei dem T89C51AC2 gibt es vier Prioritätsebenen und noch mehr Interrupt-Quellen (mehr dazu im Kapitel 7.6). Bei Annahme eines Interrupts wird noch der laufende Befehl fertig abgearbeitet, die Adresse des folgenden Befehls (= Rückkehradresse) im Stack abgespeichert und der Interrupt Vektor wird in den Programmcounter (PC) geladen. Nach Abarbeiten des Interrupts soll das Programm an der Unterbrechungsstelle fortgesetzt werden. Daraus folgt das am Ende des Unterbrechungsprogramm ein ***„RETI“*** (Return from Interrupt) erfolgen muss.

Bei mehreren gleichzeitig anstehenden Interruptanforderungen mit gleicher Priorität wird folgende Abarbeitungsreihenfolge eingehalten:

1.) Externer Interrupt 0	Sprungadresse: 0003H
2.) Interrupt von Zeitgeber/Zähler 0	Sprungadresse: 000BH
3.) Externer Interrupt 1	Sprungadresse: 0013H
4.) Interrupt von Zeitgeber/Zähler 1	Sprungadresse: 001BH
5.) Interrupt der serielle Schnittstelle	Sprungadresse: 0023H
6.) Interrupt von Zeitgeber/Zähler 2	Sprungadresse: 002BH



DAS SFR IE

EA => Enable All control bit
Gemeinsame Steuerung der einzelnen Interrupt Quellen
„0“ = alle Interrupt sind gesperrt
„1“ = Interruptsperre aufgehoben
Das Bit ist per Software zu setzen und zu löschen.

ET2 => Enable Timer 2 interrupt control bit
Freigabe des Interrupts von Zeitgeber/Zähler 2 (*)

ES => Enable Serial port interrupt control bit
Freigabe des Interrupts der serielle Schnittstelle (*)

ET1 => Enable Timer 1 interrupt control bit
Freigabe des Interrupts von Zeitgeber/Zähler 1 (*)

EX1 => Enable eXternal interrupt 1 control bit.
Freigabe des externen Interrupts 1 (Bausteinanschluss INT 1) (*)

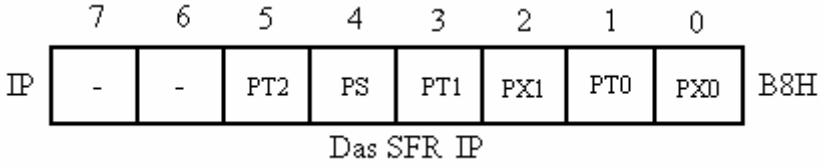
ET0 => Enable Timer 0 interrupt control bit
Freigabe des Interrupts von Zeitgeber/Zähler 0 (*)

EX0 => Enable eXternal interrupt 0 control bit.
Freigabe des externen Interrupts 0 (Bausteinanschluss INT 0) (*)

(*) => „0“ = Interrupt gesperrt ; „1“ = Interrupt freigegeben

- Die Bits sind per Software zu setzen und zu löschen.
- Nach dem Rücksetzen (RESET) enthält das SFR IE 0XX00000

PT2 => Priority controll bit Timer 2



Das SFR IP

Festlegung der Priorität des Interrupts von Zeitgeber/Zähler 1. (*)

PS => Priority controll bit Serial port
Festlegung der Priorität des Interrupts der seriellen Schnittstelle. (*)

PT1 => Priority controll bit Timer 1
Festlegung der Priorität des Interrupts von Zeitgeber/Zähler 1. (*)

PX1 => Priority controll bit eXternal interrupt 1
Festlegung der Priorität des externen Interrupt 1. (*)

PT0 => Priority controll bit Timer 0
Festlegung der Priorität des Interrupts von Zeitgeber/Zähler 0. (*)

PX0 => Priority controll bit eXternal interrupt 0
Festlegung der Priorität des externen Interrupt 0. (*)

(*) => „0“ = niedere Priorität (Low Level Priority) ; „1“ = hohe Priorität (High Level Priority)

- Die Bits sind per Software zu setzen und zu löschen.
- Nach dem Rücksetzen (RESET) enthält das SFR IP XXX00000

7.1 Interrupt-Vektoradressen

Adresse / H	Interrupt	Keil C51 Nr.
0003	Externer Interrupt 0	0
000B	Timer 0 Interrupt	1
0013	Externer Interrupt 1	2
001B	Timer 1 Interrupt	3
0023	Serieller Port Interrupt	4
002B	Timer 2 Interrupt	5

7.2 Interne Interrupt-Anforderung

- Durch Setzen von TF0, TF1, RI, TI oder SPIF per Software kann der jeweilige Interrupt simuliert werden. Sind die Interrupts gesperrt, können die Bits auch per Polling (= zyklisches Abfragen durch die Software) abgefragt werden, sie müssen dann allerdings auch per Software rückgesetzt werden.

7.3 Externe Interrupt-Anforderung

Die Eingänge der externen Interrupts sind Low-aktiv. Ferner können sie ***Pegel***- oder ***Flankengesteuert*** sein. Beim flankengesteuerten Interrupt wird, wenn nach dem 1. Befehlszyklus beim 2.Befehlszyklus ein Low Pegel am INT-Anschluss registriert wird, das

Interrupt anfordernde Bit IEx (x steht für 0 oder 1) gesetzt. Es wird nach der Annahme des Interrupt ***automatisch wieder gelöscht, wenn es flankengetriggert*** war.

Bei pegelgesteuertem INT-Anschluss bleibt das Bit IEx solange gesetzt wie am INT-Anschluss Masse anliegt und wird bei +Spannung am INT-Anschluss wieder gelöscht. Dies kann zu folgende Fehler führen:

1. Liegt das Signal sehr lange an, kann die Interruptroutine mehrmals durchlaufen werden.

2. Ist das Signal zu kurz, geht diese Interruptanforderung verloren, da sie nicht gespeichert wird.

Das SFR TCON

- TF1 bis TR0 werden zur Timereinstellung verwendet. (siehe Zeitgeber)
- IT0 => Interrupt 0 Type control bit
  Festlegung des Verhaltens bei einem Interrupt am INT 0 Eingang
  „0“ = Interrupt pegelgesteuert
  „1“ = Interrupt flankengesteuert
  Das Bit ist per Software zu setzen und zu löschen.
- IE0 => external Interrupt 0 Edge flag
  Speichert eine Interrupt-Anforderung am INTO-Eingang.
  Ein Low Pegel (bei IT0 = 0) bzw. eine negative Flanke (bei IT0 = 1) am INTO-Eingang setzt IE0. Das Bit kann auch per Software gesetzt werden, um z.B. einen Interrupt 0 zu testen. Bei flankengesteuertem Interrupt setzt die Annahme des Interrupt das Bit IE0 automatisch zurück.
- IT1 => Interrupt 1 Type control bit
  Festlegung des Verhaltens bei einem Interrupt am INT 1-Eingang entsprechend IT0, nur für Interrupt 1
- IE 1 => external Interrupt 1 Edge flag
  Speichert eine Interrupt-Anforderung am INT 1 -Eingang entsprechend IE0, nur mit IE 1 für Interrupt 1
- Nach dem Rücksetzen (RESET) enthält das SFR TCON 00000000
- Durch setzen von IE0 und IE1 lässt sich die Anforderung eines externen Interrupts simulieren.
- Vor jedem Aktivieren der externen Interrupts sollten die externen Interruptsflag (IE0 und IE1) gelöscht werden.

7.4 Ablauf einer Interrupt-Bearbeitung

- Zwischen Auftreten eines Interrupt-Signal und Beginn der Interrup-Service-Routine vergeht einige Zeit:
  - 1 Maschinenzyklus für das Speichern des Signals,
  - 1 Maschinenzyklus für das Pooling (abfragen) der Anforderung und
  - 2 Maschinenzyklus für den Aufruf (LCALL) der Service-Routine.

- Die Verzögerung zwischen Auftreten einer Interrupt-Anforderung und der Interrupt-Ausführung kann allerdings noch wesentlich länger dauern:

1. Wenn gerade ein Interrupt gleicher oder höherer Priorität abgearbeitet wird.

2. Wenn der Befehlszyklus, in dem das Polling durchgeführt wird, nicht der letzte Zyklus des auszuführenden Befehls ist.

3. Wenn gerade ein RETI-Befehl oder ein Zugriff auf eines der SFR IE oder IP durchgeführt wird.

Da die Special-Function-Register SFR (z.B. Akku, PSW, usw.) nicht in den Registerbänken enthalten sind, müssen diejenigen SFR gesondert gerettet werden, die von der Interrupt-Routine verändert werden. (PUSH- und POP-Befehle).

Programmbeispiel Assembler: Interrupt-Service Routine

```
;* Eine Interrupt-Service-Routine  (Interrupt-Unterprogramm)
;* muss sichern, was sie an Registern und SFR zerstört)

   Registerbank_x EQU  00001000B
Interrupt_x:        PUSH PSW          ; PSW im Stack sichern
                    PUSH ACC          ; Akku im Stack sichern
                    MOV PSW,#Registerbank_x
;* Registerbank_x = 00000000 fuer Registerbank 0
;*                  00001000 fuer Registerbank 1
;*                  00010000 fuer Registerbank 2
;*                  00011000 fuer Registerbank 3

;* Eigentliches Interrupt-Programm
                    POP ACC           ;alten Akku-Inhalt holen
                    POP PSW           ; PSW zurueckholen und gleichzeitig alte
                                      ; Registerbank wieder einstellen
                    RETI              ; RETurn und IIPx ruecksetzten
;* Anschliessend wird das unterbrochene Programm fortgesetzt
                    END
```

Programmbeispiel C:

siehe Bsp. 5.2 Programm Blinken

7.5 Interrupt – Abarbeitung bei C51

- func_name( ) interrupt x [using m]
  Festlegung der Funktion als ISR mit x :
  Interruptvektoradresse = 0003 + x · 8

- Das Interrupt – Attribut muss als Argument eine Integerkonstante erhalten. Ausdrücke mit Operatoren sind nicht zulässig.

Ein Interrupt bewirkt:

- speichert (sichert) auf dem Stack: ACC, B, DPH, DPL und PSW im Bedarfsfall

- Wird ohne Registerbank – Umschaltung gearbeitet (using m), so werden alle in der ISR – Funktion benötigten Arbeitsregister (R_n) auf dem Stack abgelegt.
- Vor Verlassen der ISR – Funktion werden alle zuvor abgelegten Registerinhalte wieder hergestellt

- Die Funktion wird mit dem RETI – Befehl beendet.

Regeln:

- keine Parameter ➔ globale Variablen

- kein Rückgabewert ➔ globale Variablen

- keine direkte Aufrufe von Interrupt – Funktionen

- Funktionen, die von einer Interrupt-Prozedur aufgerufen werden, müssen mit der gleichen Registerbank wie die Interrupt – Prozedur arbeiten. Der Compiler generiert fallweise absolute Registerzugriffe, für die die aktuelle Registerbank hergenommen wird, wenn nicht explizit eine using – Anweisung vorliegt. Ein Fehler entsteht, wenn das Unterprogramm eine andere Registerbank annimmt, als derzeit eingestellt ist. Der Anwender hat dafür Sorge zu tragen, dass die Registerbank den Erfordernissen entspricht, der Compiler kann dies nicht überprüfen.

7.6. Zusatz T89C51AC2

Der µC verfügt über zwei weitere Interruptquellen:

Interrupt	Vektoradresse	Priorität
PCA	33h	5
ADC	43h	9

Die Freigabe erfolgt über die SFR's IEN0 und IEN1. (siehe Datenblatt)

7	6	5	4	3	2	1	0
EA	EC	ET2	ES	ET1	EX1	ET0	EX0

Bit Number	Bit Mnemonic	Description
7	EA	**Enable All Interrupt bit** Clear to disable all interrupts. Set to enable all interrupts. If EA=1, each interrupt source is individually enabled or disabled by setting or clearing its interrupt enable bit.
6	EC	**PCA Interrupt Enable** Clear to disable the PCA interrupt. Set to enable the PCA interrupt.
5	ET2	**Timer 2 Overflow Interrupt Enable bit** Clear to disable Timer 2 overflow interrupt. Set to enable Timer 2 overflow interrupt.
4	ES	**Serial Port Enable bit** Clear to disable serial port interrupt. Set to enable serial port interrupt.
3	ET1	**Timer 1 Overflow Interrupt Enable bit** Clear to disable timer 1 overflow interrupt. Set to enable timer 1 overflow interrupt.
2	EX1	**External Interrupt 1 Enable bit** Clear to disable external interrupt 1. Set to enable external interrupt 1.
1	ET0	**Timer 0 Overflow Interrupt Enable bit** Clear to disable timer 0 overflow interrupt. Set to enable timer 0 overflow interrupt.
0	EX0	**External Interrupt 0 Enable bit** Clear to disable external interrupt 0. Set to enable external interrupt 0.

IEN0 Register; Adresse $A8_H$; Resetwert: 00_H

7	6	5	4	3	2	1	0
-	-	-	-	-	-	EADC	-

Bit Number	Bit Mnemonic	Description
7	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
6	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
5	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
4	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
3	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
2	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.
1	EADC	**ADC Interrupt Enable bit** Clear to disable the ADC interrupt. Set to enable the ADC interrupt.
0	-	**Reserved** The value read from this bit is indeterminate. Do not set this bit.

IEN1 Register; Adresse $E8_H$; Resetwert: 00_H

Es lassen sich 4 unterschiedliche Prioritätsebenen einstellen.
SFR: IPL0 IPL1 IPH0 IPH1

Die Kombination der beiden einzelnen Bits stellen die Prioritätsebene ein.

00 ➔ niedrigste Prioritätsebene
01 ➔ ....
10 ➔ ...
11 ➔ höchste Prioritätsebene