5.3 Spezial-Funktions-Register

Die Spezial-Funktions-Register befinden sich im Adressbereich 80h bis FFh. Sie sind über direkte Adressierung anzusprechen. Sie lassen sich mit ihrer absoluten Hexadresse benennen oder, bei einem geeigneten Übersetzerprogramm, auch mit ihrer symbolischen Adresse. Einige SFR sind nur als komplettes Byte anzusprechen, andere sind bitadressierbar. Bei geeignetem Übersetzerprogramm sind auch diese einzelnen Bits mit ihrem symbolischen Namen zu benennen.

Beispiele:

Setzen des Timer-Modus-Registers auf den Wert 0Fh (auf die Funktion der Register soll hier nicht eingegangen werden):

Symbolische Adresse des SFR: TMOD
Absolute Adresse des SFR: 89h
nicht bitadressierbar

```
MOV TMOD,#0F
```
oder
```
MOV 89,#0F
```

Freigabe des Timer 0 durch Setzen des Timer Run Flags TR0 auf 1:

Symbolische Adresse des SFR: TCON
Absolute Adresse des SFR: 88h
bitadressierbar
Symbolische Adresse des Bit: TR0
Absolute Adresse des Bit: 8Ch

```
SETB TR0
```
oder
```
SETB 8C
```

SETB TCON.04h
SETB 88h.04h

Die Spezial-Funktions-Register enthalten alle Register, die der Controller für seine Arbeit, für die Ein- und Ausgabe und für interne Einstellungen benötigt.

Auf die Bedeutung der Register oder einzelner Bits wird eingegangen, wenn sie für die Realisierung bestimmter Funktionen benötigt werden.

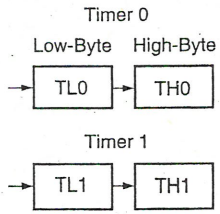
Liste der SFR-Register

Block	Symbol	Funktion	Adresse	Bitadr.
CPU	ACC	Akkumulator	E0H	•
	B	B-Register	F0H	•
	DPL	Data Pointer, Low-Byte	82H	
	DPH	Data Pointer, High-Byte	83H	
	PSW	Programmstatuswort	D0H	•
	SP	Stack Pointer	81H	
Ports	P0	Port 0	80H	•
	P1	Port 1	90H	•
	P2	Port 2	A0H	•
	P3	Port 3	B0H	•
	P4	Port 4	E8H	•
	P5	Port 5	F8H	•
	P6	Port 6 (nur Eingang; im 80515 nicht vorhanden)	DBH	
Serielle Schnittstelle	ADCON (80(C)515)	A/D-Wandler-Steuerregister	D8H	•
	PCON	Stromaufnahme-Steuerregister	87H	
	SCON	Steuerregister der ser. Schnittstelle	98H	•
	SBUF	Buffer der seriellen Schnittstelle	99H	
Timer 0/1	TCON	Steuerregister für Timer 0/1	88H	•
	TMOD	Modusauswahl für Timer 0/1	89H	
	TL0	Timer-0-Register, Low-Byte	8AH	
	TH0	Timer-0-Register, High-Byte	8CH	
	TL1	Timer-1-Register, Low-Byte	8BH	
	TH1	Timer-1-Register, High-Byte	8DH	
Watchdog	IEN0	Interrupt-Freigaberegister 0	A8H	•
	IEN1	Interrupt-Freigaberegister 1	B8H	•
	IP0	Interrupt-Priorität-Steuerung 0	A9H	
	IP1	Interrupt-Priorität-Steuerung 1	B9H	
Interrupt-System	IEN0	Interrupt-Freigaberegister 0	A8H	•
	IEN1	Interrupt-Freigaberegister 1	B8H	•
	IP0	Interrupt-Priorität-Steuerung 0	A9H	
	IP1	Interrupt-Priorität-Steuerung 1	B9H	
	IRCON	Interrupt-Request-Register	C0H	•
	TCON	Steuerregister für Timer 0/1	88H	•
	T2CON	Steuerregister für Timer 2	C8H	•
A/D-Wandler	ADCON (80(C)515)	A/D-Wandler-Steuerregister	D8H	•
	DAPR (80(C)515)	Steuerregister für programmierbare Referenzspannungen	DAH	
	ADDAT (80(C)515)	A/D-Wandler-Ergebnisregister	D9H	
Timer-2-Block	CCEN	Compare/Capture-Freigaberegister	C1H	
	CCL1	Compare/Capture-Register 1, Low-Byte	C2H	
	CCH1	Compare/Capture-Register 1, High-Byte	C3H	
	CCL2	Compare/Capture-Register 2, Low-Byte	C4H	
	CCH2	Compare/Capture-Register 2, High-Byte	C5H	
	CCL3	Compare/Capture-Register 3, Low-Byte	C6H	
	CCH3	Compare/Capture-Register 3, High-Byte	C7H	
	CRCL	Compare/Rel./Capture-Register, Low-Byte	CAH	
	CRCH	Compare/Rel./Capture-Register, High-Byte	CBH	
	TL2	Timer-2-Register, Low-Byte	CCH	
	TH2	Timer-2-Register, High-Byte	CDH	
	T2CON	Timer-2-Steuerregister	C8H	•
Modi für reduzierte Stromaufnahme	PCON	Stromaufnahme-Steuerregister	87H	

16 Die Zähler/Zeitgeber Timer 0 und 1

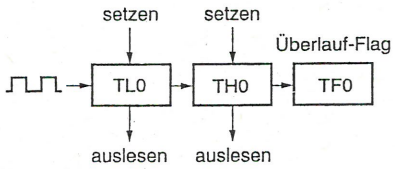
Timer 0 und Timer 1 bestehen aus jeweils zwei 8-Bit-Vorwärtszählern. Die Zähler lassen sich, je nach gewähltem Modus, als 8-Bit oder als 16-Bit-Zähler verwenden. Die 16-Bit-Zähler ergeben sich aus der Reihenschaltung zweier 8-Bit-Zähler.

Die 8-Bit-Zähler-Register sind die Spezial-Funktions-Register TL0 und TH0 für Timer 0 und TL1 und TH1 für Timer 1.



Die Zähler-Register lassen sich auf Anfangswerte setzen. Ab diesen Werten zählen sie nach dem Starten des Zählers aufwärts. Läuft der Zähler über, wird das Überlauf-Flag gesetzt. Nach einem Überlauf lässt sich der Zähler wieder auf einen Anfangswert setzen. Geschieht dies nicht, beginnt er bei dem Wert null.

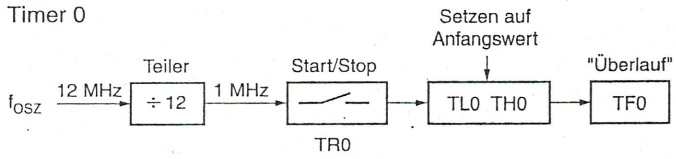
Timer 0 als 16-Bit-Zähler:



Das Überlauf-Flag kann einen Interrupt auslösen oder vom Programm abgefragt werden. Bei einer Abfrage per Programm ist es auch durch das Programm wieder zu löschen.

Der jeweilige Zählerstand lässt sich auch aus den Registern auslesen.

16.1 Einsatz der Timer als Zeitgeber



Bei einer Verwendung der Timer als Zeitgeber werden sie vom Systemtakt hochgezählt. Die Periodendauer des Systemtaktes wird über den Zähler addiert. Je nach Anfangswert des Zählers ergibt sich bis zum Überlauf eine bestimmte Zeitverzögerung. Diese Zeitverzögerung ist nicht abhängig vom laufenden Programm. Der Timer läuft, getaktet vom Systemtakt, als eigenständige Hardware-Baugruppe. Die Zeitverzögerung hängt also nur von der Quarzfrequenz ab.

Läuft der Zähler über, setzt er sein Überlauf-Flag TF0 oder TF1 auf 1. Damit lässt sich ein Interrupt auslösen, mit dem das laufende Programm unterbrochen und ein Unterprogramm als Interrupt-Service-Routine gestartet werden kann.

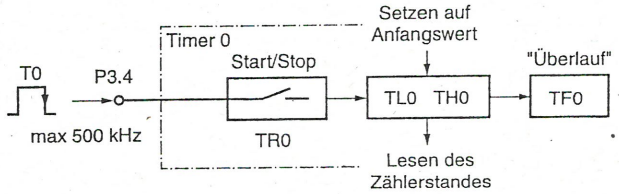
Wird nicht im Interruptbetrieb gearbeitet, wird das Überlauf-Flag bei jedem Programmzyklus abgefragt. Ist es gesetzt, wird darauf reagiert und das Flag zurückgesetzt.

Die Überlauf-Flags sind Bits in bestimmten Spezial-Funktions-Registern. Sie sind bitadressierbar.

Einstellen der Zeit und starten: z. B. Timer 0

1. Taktsignal stoppen:
   ```
   CLR TR0
   ```
2. Anfangswerte in beide Timerbytes laden:
   ```
   MOV TL0,#ANFL
   MOV TH0,#ANFH
   ```
3. Überlauf-Flag rücksetzen:
   ```
   CLR TF0
   ```
4. Taktsignal starten:
   ```
   SETB TR0
   ```

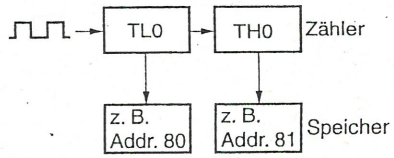
16.2 Einsatz der Timer als Ereigniszähler



Werden die Timer als Ereigniszähler eingesetzt, erhalten sie ihre Zählimpulse von einem externen Geber. Es werden die negativen Flanken der Eingangsimpulse gezählt. Dabei arbeiten die Zähler unabhängig vom laufenden Programm.

Lesen des Zählerstandes

Der 16-Bit-Zähler wird aus zwei Registern gebildet, die nacheinander auszulesen sind. Das ist kein Problem, wenn der Zähler während des Auslesens über TR0 angehalten werden kann.



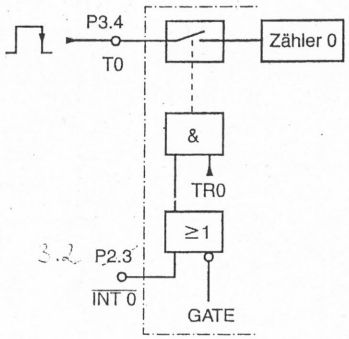
Will man bei laufendem Zähler den Zählerstand auslesen, ist zu berücksichtigen, dass während des Auslesens Zählimpulse eintreffen können, die den Zählerstand verändern. Es ist daher zu kontrollieren, ob während des Auslesens ein Übertrag in das höherwertige Byte aufgetreten ist. Ist das der Fall, ist das Lesen zu wiederholen.

Programm zum Auslesen des Zählerstandes des Timers 0:

```
LES_Z16:
    MOV 81,TH0; H-Byte nach 81
    MOV 80,TL0; L-Byte nach 80
    MOV A,TH0 ;TH_NEU = TH_ALT?
    CJNE A,81,LES_Z16 ;Nein:
    RET    ;Ja:
```

Freigabe des Zählers

Am Beispiel des Timers 0:
Der Zähler lässt sich über die Bits TR0 und GATE in Spezial-Funktions-Registern oder über ein externes Signal an Portpin P3.2 freigeben.



Interne Freigabe: `GATE = 0; TR0 = 1`
Externe Freigabe über P3.2 = 1: `GATE = 1; TR0 = 1`

16.3 Einstellen der Timer-Funktion

Die Timer-Funktion wird mithilfe des Timer-Modus-Registers TMOD eingestellt.

Timer-Modus-Register TMOD:

TMOD nicht bitadressierbar — Resetwert 00h

GATE	C/$\overline{T}$	M1	M0	GATE	C/$\overline{T}$	M1	M0
7							0

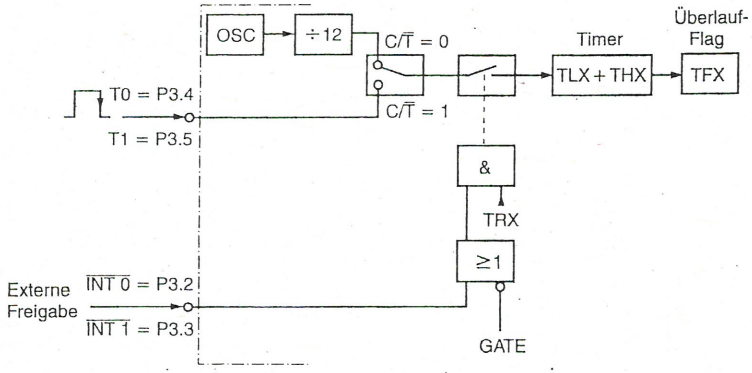
Timer 1 | Timer 0

Bedeutung der einzelnen Bits des Timer-Modus-Registers:

Bit		Funktion
GATE		Umschaltung interne/externe Freigabe:
0		Interne Freigabe über TR0 bzw. TR1
1		Externe Freigabe über P3.2 bzw. P3.3 plus interne Freigabe über TR0 bzw. TR1.
C/$\overline{T}$		Umschaltung Zeitgeber oder Zähler:
0		Funktion als Zeitgeber
1		Funktion als Zähler
M1	M0	Wahl des Arbeitsmodus: (Es werden nur die zwei gebräuchlichen Arbeitsmodi beschrieben)
0	1	Modus 1: TL0 und TH0 bzw. TL1 und TH1 bilden einen 16-Bit-Zähler
1	0	Modus 2: TL0 bzw. TL1 bilden einen 8-Bit Auto-Reload-Zähler. Bei Überlauf wird der im High-Byte TH0 bzw. TH1 stehende Wert in das Low-Byte kopiert. Das High-Byte bleibt unverändert.

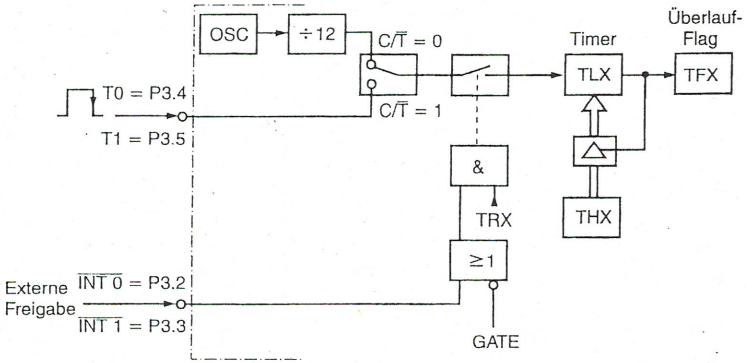
Blockschaltbild der Timer 0 und 1 in Modus 1

16-Bit-Timer/Counter:



Blockschaltbild der Timer 0 und 1 in Modus 2

8-Bit-Timer/Counter mit Auto-Reload:



16.4 Steuern der Timer

Die Timer werden über die Bits TF und TR im Timer-Control-Register TCON gesteuert.

Timer-Control-Register TCON: Adresse 88H

TCON bitadressierbar — Resetwert 00h

TF1	TR1	TF0	TR0				
7							0

Bedeutung der einzelnen Bits des Timer-Control-Registers:

Bit	Funktion
TF0 TF1	Timer-Überlauf-Flag: Das Flag wird beim Überlauf des Timers gesetzt. Beim Einsprung in die zugehörige Interrupt-Adresse wird es automatisch rückgesetzt. Bei Abfrage im Programmzyklus muss es durch das Programm gelöscht werden.
TR0 TR1 1 0	Timer-Run-Flag = Timer Freigabe: Das Flag muss durch das Programm gesetzt oder gelöscht werden. Timer startet Timer stopt

Beispiel 16.1

Initialisieren Sie den Timer 1 als Zeitgeber. Es soll eine Zeit für ein Blinklicht erzeugt werden. Timer 0 ist nicht zu beeinflussen.

Timer-Modus einstellen:

TMOD nicht bitadressierbar

GATE	C/T	M1	M0				
0	0	0	1	x	x	x	x

Assemblerprogramm:

```
INIT_T1:  ANL TMOD,#0Fh; 16-Bit-
          ORL TMOD,#10h  Zähler
          RET
```

Der Timer soll vom laufenden Programm auf einen Zeitwert gesetzt und auf Überlauf abgefragt werden. Nach jedem Zeitabschnitt wird das Blinklicht an P5.0 umgeschaltet.

Assemblerprogramm:

```
BLINK:
    JNB TF1,BL1;Timer Überlauf?
    CLR TR1; Ja: Timer stoppen
    MOV TL1,#ZL1; Timer setzen
    MOV TH1,#ZH1
    CLR TF1
      ; Überlauf rücksetzen
    CPL P5.0
      ; Blinklicht umschalten
    SETB TR1; Timer starten
BL1: RET
```

18.1 Interrupt-Quellen und Anforderungs-Flags

Externe Interrupt-Quellen

Interrupt 0:

Interrupt 1:

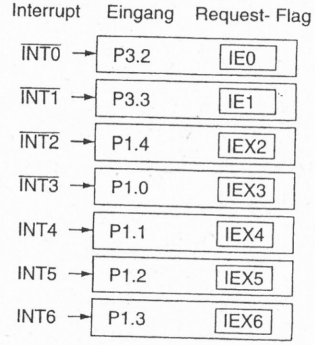
Interrupt 2:

Interrupt 3:

Interrupt 4:

Interrupt 5:

Interrupt 6:



Interne Interrupt-Quellen

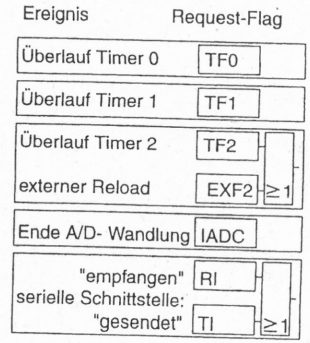
Überlauf-Interrupt Timer 0:

Überlauf-Interrupt Timer 1:

Überlauf-Interrupt Timer 2 oder externer Reload:

Ende-Interrupt A/D-Wandler:

Empfang oder Senden eines Zeichens der seriellen Schnittstelle:



Die Interrupt-Quellen setzen die angegebenen Request-Flags. Diese Flags sind Bits in Spezial-Funktions-Registern.

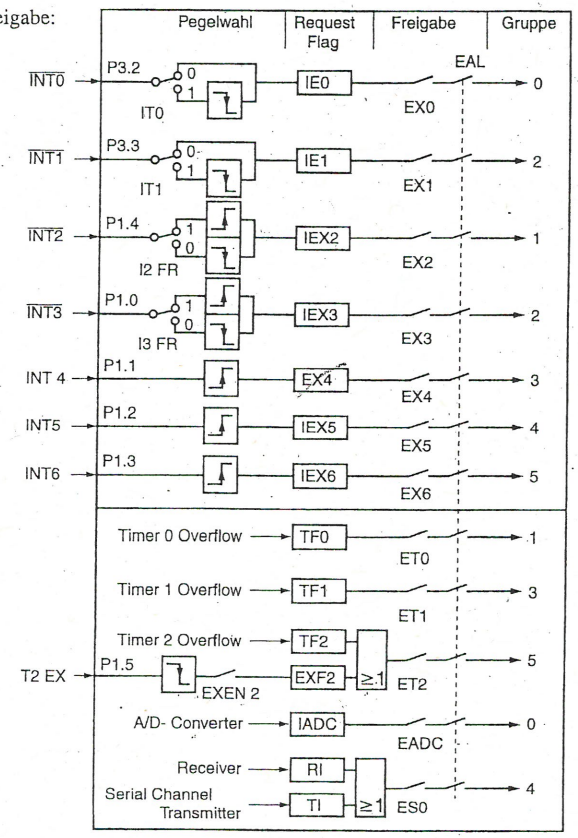
18.2 Pegelwahl und Interrupt-Freigabe

Bei einigen externen Interrupts kann gewählt werden, ob die Signalflanke oder der Signalpegel einen Interrupt auslösen soll. Diese Wahl erfolgt wieder über Bits in Spezial-Funktions-Registern.

Der Interrupt-Controller fragt zyklisch ab, welche Request-Flags gesetzt sind. Dabei werden jedoch nur die freigegebenen Flags berücksichtigt. Jedes Flag hat eine eigene Freigabe (Enable). Außerdem gibt es eine generelle Freigabe für alle Flags. Die Freigabe erfolgt ebenfalls durch Setzen bestimmter Bits in Spezial-Funktions-Registern.

Der Interrupt wird angenommen, wenn das Flag freigegeben und es von der Priorität her an der Reihe ist. Ehe jedoch der Interrupt-Controller den internen LCALL-Befehl für den Sprung in die Interrupt-Service-Routine erzeugt, muss der gerade laufende Befehl beendet werden.

Pegelwahl und Interrupt-Freigabe:



IEX4

Spezial-Funktions-Register für Pegelwahl, Request-Flag und Freigabe

TCON

TCON bitadressierbar Resetwert 00h

TF1		TF0		IE1	IT1	IE0	IT0
7							0

Bedeutung der einzelnen Bits:

Bit	Funktion
TF1	Interrupt-Request-Flag des Überlaufes von Timer 1. Wird beim Einsprung in die Interrupt-Routine automatisch gelöscht.
TF0	Interrupt-Request-Flag des Überlaufes von Timer 0. Wird beim Einsprung in die Interrupt-Routine automatisch gelöscht.
IE1	Interrupt-Request-Flag des externen Interrupts 1 an P3.3. IT1 = 1: IE1 wird gesetzt bei einer fallenden Flanke an P3.3. IE1 wird bei Einsprung in die Interrupt-Routine automatisch gelöscht. IT1 = 0: IE1 bleibt gesetzt, so lange L-Pegel anliegt. Die Interrupt-Routine muss veranlassen, dass der L-Pegel am Eingang P3.3 verschwindet.
IT1 0 1	Selektionsbit für Interrupt 1 (Interrupt 1 Type Select Bit). Interrupt wird durch L-Pegel ausgelöst. Interrupt wird durch eine fallende Flanke ausgelöst.
IE0	Interrupt-Request-Flag des externen Interrupts 0 an P3.2. IT1 = 1: IE0 wird gesetzt bei einer fallenden Flanke an P3.2. IE0 wird bei Einsprung in die Interrupt-Routine automatisch gelöscht. IT1 = 0: IE0 bleibt gesetzt, so lange L-Pegel anliegt. Die Interrupt-Routine muss veranlassen, dass der L-Pegel am Eingang P3.2 verschwindet.
IT0 0 1	Selektionsbit für Interrupt 0. Interrupt wird durch L-Pegel ausgelöst. Interrupt wird durch fallende Flanke ausgelöst.

T2CON

T2 CON bitadressierbar Resetwert 00h

	I3FR	I2FR					
7							0

Bedeutung der einzelnen Bits:

Bit	Funktion
I3FR	Auswahlbit zur Einstellung der aktiven Flanke des externen Interrupts 3 an P1.0 (Interrupt 3 Falling/Rising Edge).
0	Die fallende Flanke ist aktiv.
1	Die steigende Flanke ist aktiv.
I2FR	Auswahlbit zur Einstellung der aktiven Flanke des externen Interrupts 2 an P1.4.
0	Die fallende Flanke ist aktiv.
1	Die steigende Flanke ist aktiv.

IRCON

IRCON bitadressierbar Resetwert 00h

EXF2	TF2	IEX6	IEX5	IEX4	IEX3	IEX2	IADC
7							0

Bedeutung der einzelnen Bits des Interrupt-Request-Registers (Interrupt Request Control):

Bit	Funktion
EXF2	Interrupt Request Flag des externen Timer-2-Nachlademodus. Es wird bei einer fallenden Flanke an P1.5 gesetzt, wenn der externe Reload-Modus für den Timer 2 gewählt wurde. Ist der Nachlade-Interrupt des Timers freigegeben (EXEN2 = 1), wird in die Interrupt-Routine gesprungen. EXF2 muss per Software gelöscht werden.
TF2	Interrupt Request Flag des Überlaufes von Timer 2. Bei einem Überlauf wird es automatisch gesetzt. Es muss per Software rückgesetzt werden.
IEX6	Interrupt Request Flag des externen Interrupts 6. Es wird gesetzt, wenn eine steigende Flanke an P1.3 auftritt. Es wird beim Einsprung in die Interrupt-Routine automatisch gelöscht. Das Flag lässt sich auch per Software löschen.
IEX5	Interrupt Request Flag des externen Interrupts 5. Es wird gesetzt, wenn eine steigende Flanke an P1.2 auftritt. Es wird beim Einsprung in die Interrupt-Routine automatisch gelöscht. Das Flag lässt sich auch per Software löschen.
IEX4	Interrupt Request Flag des externen Interrupts 4. Es wird gesetzt, wenn eine steigende Flanke an P1.1 auftritt. Es wird beim Einsprung in die Interrupt-Routine automatisch gelöscht. Das Flag lässt sich auch per Software löschen.
IEX3	Interrupt Request Flag des externen Interrupts 3. Es wird gesetzt bei einer steigenden/fallenden Flanke an P1.0. Es wird beim Einsprung in die Interrupt-Ruotine automatisch gelöscht. Das Flag lässt sich auch per Software löschen.
IEX2	Interrupt Request Flag des externen Interrupts 2. Es wird gesetzt bei einer steigenden/fallenden Flanke an P1.4. Es wird beim Einsprung in die Interrupt-Routine automatisch gelöscht. Das Flag lässt sich auch per Software löschen.
IADC	Interrupt Request Flag des A/D-Wandlers. Es wird am Ende einer A/D-Wandlung automatisch gesetzt. Es muss per Software glöscht werden.

IEN0

IEN0 bitadressierbar Resetwert 00h

EAL		ET2	ES0	ET1	EX1	ET0	EX0
7							0

Bedeutung der einzelnen Bits des Interrupt-Freigaberegisters 0 (Interrupt Enable):
Generell gilt: Ein 0-Signal sperrt den Interrupt, ein 1-Signal gibt ihn frei.

Bit	Funktion
EAL 0 1	Generelles Freigabebit für alle Interrupts Es wird kein Interrupt bedient. Es gilt das individuelle Freigabebit des jeweiligen Interrupts.
ET2	Freigabebit des Timer-2-Interrupts
ES0	Freigabebit des Interrupts der seriellen Schnittstelle
ET1	Freigabebit des Timer-1-Interrupts
EX1	Freigabebit des externen Interrupts 1
ET0	Freigabebit des Timer-0-Interrupts
EX0	Freigabebit des externen Interrupts 0

IEN1

IEN1 bitadressierbar Resetwert 00h

EXEN2		EX6	EX5	EX4	EX3	EX2	EADC
7							0

Bedeutung der einzelnen Bits des Interrupt-Freigaberegisters 1:
Generell gilt: Ein 0-Signal sperrt den Interrupt, ein 1-Signal gibt ihn frei.

Bit	Funktion
EXEN2	Freigabebit des extern gesteuerten Nachlade-Interrupts des Timers 2
EX6	Freigabebit des externen Interrupts 6
EX5	Freigabebit des externen Interrupts 5
EX4	Freigabebit des externen Interrupts 4
EX3	Freigabebit des externen Interrupts 3
EX2	Freigabebit des externen Interrupts 2
EADC	Freigabe des A/D-Wandler-Interrupts

18.3 Interrupt-Prioritäten

Den Interrupts lassen sich vier Prioritätsstufen zuordnen. Dabei werden jeweils zwei Interrupts zu einer Gruppe zusammengefasst. Somit ergeben sich sechs Gruppen. Bei gleicher Priorität entscheidet die vorgegebene Reihenfolge der Abfrage, welcher Interrupt zuerst angenommen wird.

Die Prioritätsstufe null bis drei wird der Gruppe über je zwei Bits in den Interrupt-Prioritäten-Registern IP0 und IP1 zugeordnet.

IP0, IP1

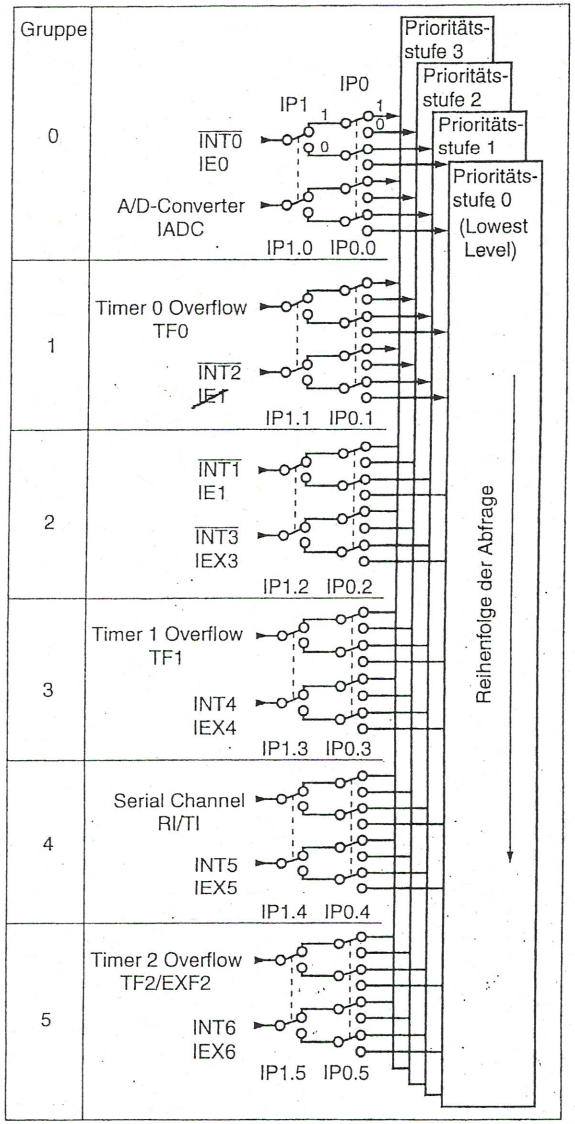
		IP0.5	IP0.4	IP0.3	IP0.2	IP0.1	IP0.0
IP0							
IP1		IP1.5	IP1.4	IP1.3	IP1.2	IP1.1	IP1.0
7							0

IP0 IP1 nicht bitadressierbar Resetwert 0 für alle IP

Bedeutung der einzelnen Bits der Interrupt-Prioritäten-Register:

Bit		Funktion	
IP1.X	IP0.X		
0	0	Prioritätsstufe	0
0	1		1
1	0		2
1	1		3

Prioritäten und Reihenfolge der Interrupt-Abfrage:



IEX2

18.4 Interrupt-Vektoren

Wird ein Interrupt-Request-Flag angenommen, erzeugt die Interrupt-Steuerung intern einen LCALL-Befehl mit einer Sprungadresse. Dabei ist jedem Flag eine bestimmte Sprungadresse fest zugeordnet. Diese festen Adressen nennt man Interrupt-Vektoren.

Interrupt-Vektoren:

Interrupt-Quelle	Request-Flag	Vektoradresse
Externer Interrupt 0	IE0	0003h
Timer-0-Interrupt	TF0	000Bh
Externer Interrupt 1	IE1	0013h
Timer-1-Interrupt	TF1	001Bh
Interrupt ser. Schnittst.	RI/TI	0023h
Timer-2-Interrupt	TF2/EXF2	002Bh
A/D-Wandler-Interrupt	IADC	0043h
Externer Interrupt 2	IEX2	004Bh
Externer Interrupt 3	IEX3	0053h
Externer Interrupt 4	IEX4	005Bh
Externer Interrupt 5	IEX5	0063h
Externer Interrupt 6	IEX6	006bh

Die Vektoradressen liegen zu Beginn des Programmspeichers in relativ kurzen Abständen. Zwischen zwei aufeinander folgende Vektoradressen passt kein Unterprogramm als Service-Routine für eine Interrupt-Anforderung. Deshalb steht an der Vektoradresse in der Regel der Befehl LJMP zur Anfangsadresse des Unterprogramms.

Das Unterprogramm der Interrupt-Service-Routine muss mit dem Befehl RETI (Return from Interrupt) enden.