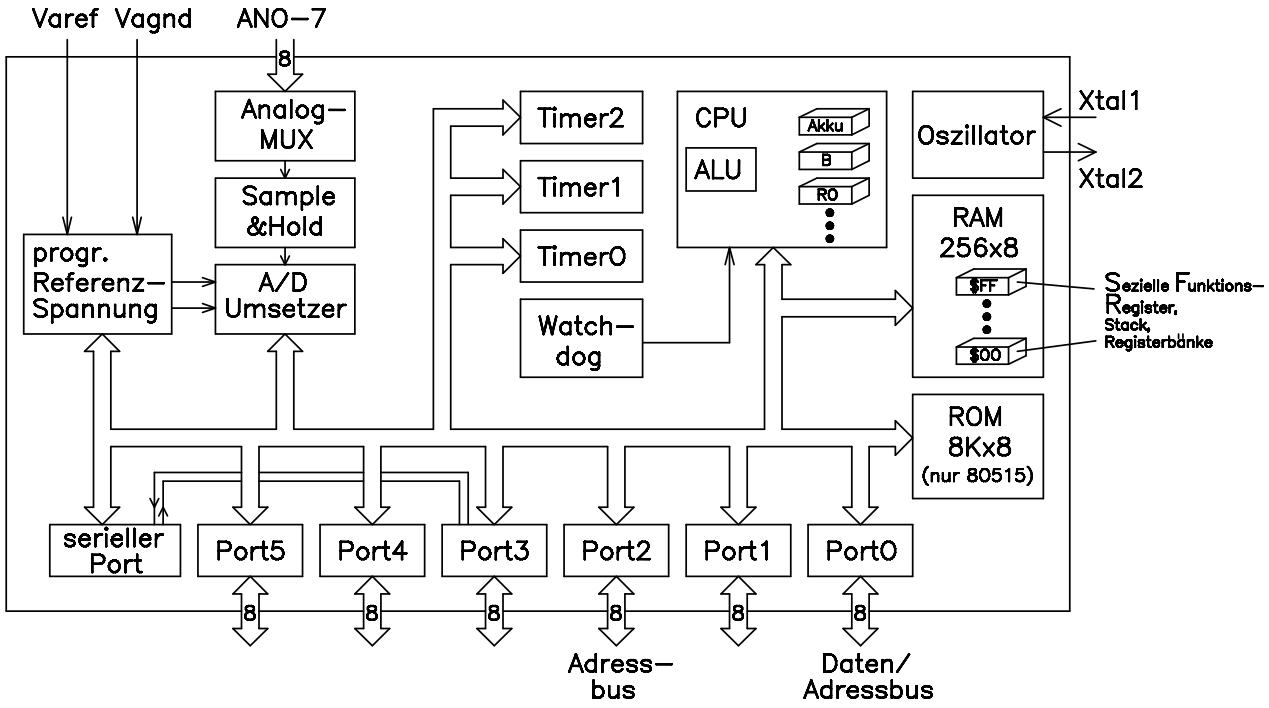
Abürzungen:
LSB: least significant Byte = niederwertiges Byte
MSB most significant Byte = höherwertiges Byte

1. Aufbau des Mikroprozessorsystems

1.1 Einführung

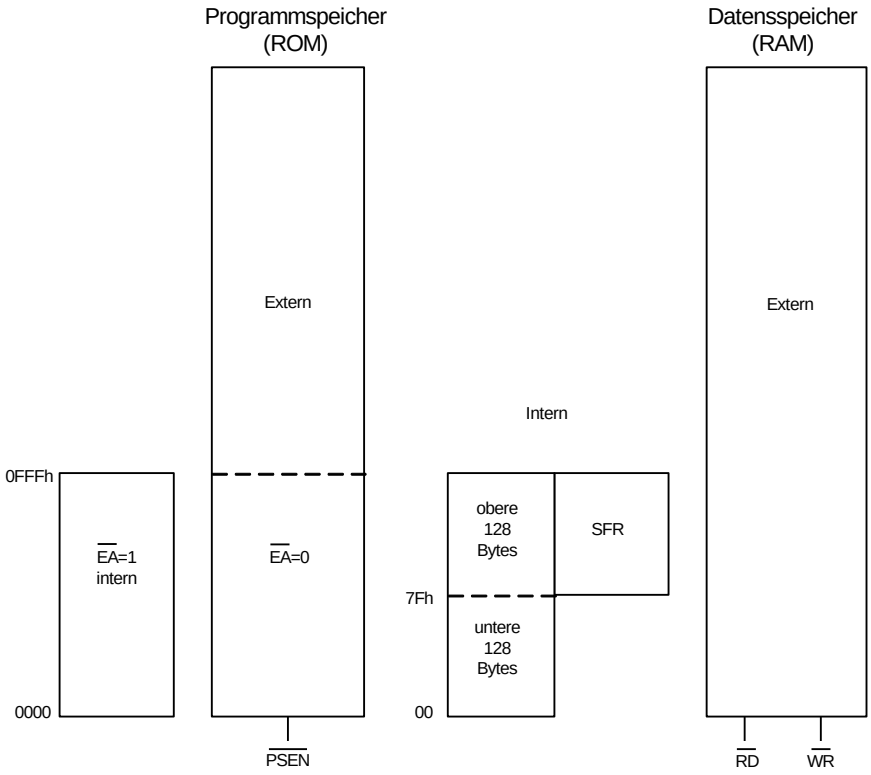
Der 80C535 von Siemens gehört zur „8051-Mikrokontroller-Familie“ von Intel. Alle Mitglieder der Familie haben die Grundfunktionen des 8051, die einzelnen Derivate unterscheiden sich durch die auf dem Chip zusätzlich vorhandene Peripherie (z. B. zusätzliche Ports, Timer, ADU, spezielle Schnittstelle wie CAN oder I^2C-Bus) und durch die Speicherarten auf dem Chip, so gibt es z.B. Versionen mit und ohne Programmspeicher (ROM) auf dem Chip (80515 / 80535), (8031 / 8051). Der 89C2051 von Atmel besitzt z.B. alle On-Chip-Komponeneten des 8051, einen zusätzlichen Flash-EPROM-Programmspeicher, hat jedoch nur 2 Ports, kann keine externen Speicher anschließen, hat dafür aber auch nur 20 statt 40 PINs.



CPU: Prozessorkern
ALU: Arithmetisch-logische Einheit zur Ausführung von Rechenoperationen / log. Verknüpfungen
Register R0-R7: Zwischenspeicher, z.B. für Zählschleifen
Akku: Wichtiges Register mit Sonderfunktionen
Port: Schnittstelle zur Außenwelt: 8 Bit breit, als Eingang und / oder Ausgang verwendbar. Port 0 u. 2 können als Adress- und Datenbus dienen, um externe Speicher anzuschließen.
Ser. Port: integrierte serielle Schnittstelle
Timer: Taktgesteuerter Zeitgeber und Zähler
Watchdog: Wachhund, kann bei Programmabsturz Reset ausführen.
ADU: Analog-Digital-Umsetzer, umschaltbar auf 8 Eingänge.
RAM: kleiner interner Datenspeicher, über die Speziellen Funktionsregister sind alle ON-Chip-Komponenten ansprechbar. (Ports, Timer, ADU, Register...)
ROM: interner Programmspeicher (nur bei der ROM-Version 80515 vorhanden)

1.2 Speicherorganisation

Die 51-er haben einen getrennten Adressbereich für Programm- und Datenspeicher (Harvard- Architektur) In der Abarbeitung von Befehlen holt sich die CPU den Befehl aus dem Programmspeicher (internes ROM / EPROM oder externes EPROM). Ergebnisse oder Messwerte (Daten) werden im sehr kleinen internen RAM oder im externen RAM (Datenspeicher) abgelegt. Die Entscheidung, ob der Programmspeicher oder Datenspeicher angesprochen wird, hängt vom Zustand der Signale PSEN (Programm Store Enable) und RD / WR (Read / Write). Mit der Leitung /EA wird zwischen internem und externem Programmspeicher umgeschaltet. Abhängig vom jeweiligen Befehl übernimmt der Controller die Aktivierung der entsprechenden Leitungen.



Für die Programmentwicklung ist es sinnvoll, den Programmcode problemlos verändern zu können. Daher verwendet unser System ein externes RAM, das gleichzeitig Programm- und Datenspeicher ist.

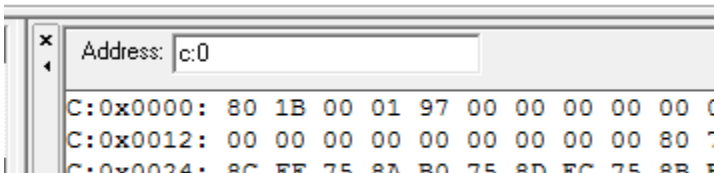
Hinweise:

- Beim Assemblieren eines Programms zeigt Keil den benötigten Speicher im internen/externen Ram und im Rom an.

```
assembling test.a51...
linking...
Program Size: data=8.0 xdata=0 code=12
creating hex file from "test"...
"test" - 0 Error(s), 0 Warning(s).
```

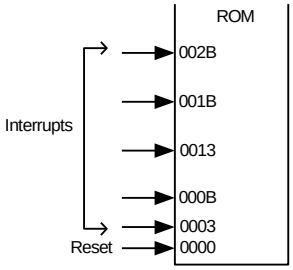
- Während des Debuggens kann in Keil der Inhalt des Speichers angezeigt werden. Hierfür muss im Debug-Modus im Menü „View“ das Memory-Window aktiviert werden. Im Adress-Feld kann mit einem der Buchstaben

d →Ram direkt
i → Ram indirekt
c →Rom (Code)
und einer Adresse der entsprechende Bereich angezeigt werden.



1.2.1 Der Programmspeicher →ROM

Der untere Teil des Programmspeichers hat spezielle Aufgaben:
An der Adresse 0000h (RESET) holt sich der Prozessor den ersten Befehl nach Anlegen der Versorgungsspannung. Es folgen die Interrupt-Einsprungadressen, hier beginnen Programmteile bei der Ausführung von Interrupts (Programmunterbrechungen bei bestimmten Ereignissen). Werden keine Interrupts verwendet, können hier auch Programme stehen.

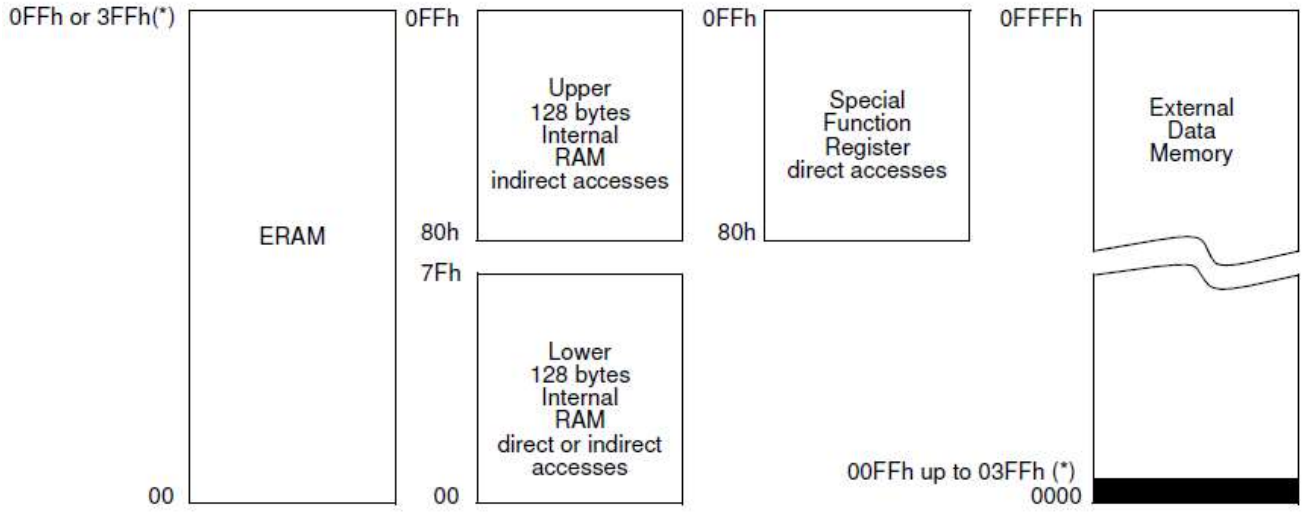


1.2.2 Der Datenspeicher → RAM

Auch beim Datenspeicher wird zwischen einem internen und externen Bereich unterschieden. Während das externe RAM bis zu 64KB groß sein kann, ist das interne RAM auf 128Bytes beschränkt. In seiner Struktur ist dieser Teil jedoch weitaus komplexer.
Der interne Datenspeicher ist in drei physikalische Bereiche geteilt:

- Die unteren 128 Bytes in denen z.B. die Registerbänke R0 bis R7, der Stack und der bitadressierbare Bereich liegen.
  direkt und indirekt adressierbar
- Mit den oberen 128 Bytes des internen RAMs werden sämtliche Funktionen des Controllers überwacht, in Gang gesetzt und abgefragt. Man spricht von **S**peziellen **F**unktions-**R**egistern (SFR).
  nur direkt adressierbar
- Daneben gibt es weitere 128Bytes im oberen Bereich.
  nur indirekt adressierbar.

1.2.3 ERAM beim AT89C5131



Entsprechend der Abbildung hat der AT89C5131 ein zusätzliches On Chip Expanded Ram mit 1024Byte. Auf diesen Speicher kann nur indirekt (R0 bis Adresse 256 bzw. DPTR alle Adressen) mit dem MOVX-Befehl zugegriffen werden. Hierzu muss im AUXR-Register (0x8E) das EXTRAM-Bit gelöscht werden.

1.2.4 Das EEPROM beim AT89S8252

Dieser Controller hat im Adressbereich von 0000h bis 7FFFh ein EEPROM das mit dem MOVX-Befehl geschrieben werden kann. Hierfür sind zwei Bit zuständig:
EEMWE (EEPROM Data Memory Write Enable Bit): Vor einem Schreibzugriff muss dieses Bit eins sein. Nach dem Schreiben sollte dieses Bit wieder gelöscht warden.
EEMEN (Internal EEPROM Access Enable): Dieses Bit muss bei einem Schreib- oder Lesezugriff auf das EEPROM eins sein.
Der Schreibvorgang dauert ca. 2,5ms und wird durch das RDY-Bit in WMCON, indem sich auch EEMWE und EEMEN befinden, angezeigt:

WMCON Address = 96H — Reset Value = 0000 0010B

Bit	7	6	5	4	3	2	1	0
	PS2	PS1	PS0	EEMWE	EEMEN	DPS	WDTRST	WDTEN

Symbol	Function
PS2 PS1 PS0	Prescaler Bits for the Watchdog Timer. When all three bits are set to "0", the watchdog timer has a nominal period of 16 ms. When all three bits are set to "1", the nominal period is 2048 ms.
EEMWE	EEPROM Data Memory Write Enable Bit. Set this bit to "1" before initiating byte write to on-chip EEPROM with the MOVX instruction. User software should set this bit to "0" after EEPROM write is completed.
EEMEN	Internal EEPROM Access Enable. When EEMEN = 1, the MOVX instruction with DPTR will access on-chip EEPROM instead of external data memory. When EEMEN = 0, MOVX with DPTR accesses external data memory.
DPS	Data Pointer Register Select. DPS = 0 selects the first bank of Data Pointer Register, DP0, and DPS = 1 selects the second bank, DP1
WDTRST RDY/$\overline{BSY}$	Watchdog Timer Reset and EEPROM Ready/$\overline{Busy}$ Flag. Each time this bit is set to "1" by user software, a pulse is generated to reset the watchdog timer. The WDTRST bit is then automatically reset to "0" in the next instruction cycle. The WDTRST bit is Write-Only. This bit also serves as the RDY/$\overline{BSY}$ flag in a Read-Only mode during EEPROM write. RDY/$\overline{BSY}$ = 1 means that the EEPROM is ready to be programmed. While programming operations are being executed, the RDY/$\overline{BSY}$ bit equals "0" and is automatically reset to "1" when programming is completed.
WDTEN	Watchdog Timer Enable Bit. WDTEN = 1 enables the watchdog timer and WDTEN = 0 disables the watchdog timer.

Bsp- für einen Schreibzugriff:

```
      mov wmcon,#00011000b    ;EEPROM zum Schreiben einrichten
      mov r7,#0
      mov a,#0
      mov dptr,#000h
loop: mov a,r7
      inc dptr

      movx @dptr,a
      mov a,wmcon
loop2:
      anl a,#00000010b  ;Prüfen ob Rdy(EEProm fertig programmiert) Bit=1
      jz loop2                ;Bit wird 1 wenn fertig
      djnz r7,loop
```

a. Der Stack (Stapel)

Der Stack speichert den aktuellen Inhalt des Programmzählers, wenn z.B. ein Unterprogramm oder eine Interruptroutine aufgerufen wird. Genauer gesagt wird der Wert des Programmzählers gespeichert der auf den nächsten Befehl zeigt, der nach dem Rücksprung aus dem Unterprogramm ausgeführt werden soll.
Der Stackpointer (SP, 1Byte) ist von der Funktion vergleichbar mit dem Programmzähler(PC, 2Byte). Er zeigt auf die Adresse, wo die Daten im RAM abgelegt werden. Nach einem Reset hat der SP standardmäßig den Wert 07h und zeigt somit auf die Adresse 08h=Beginn des „Stapels". Das liegt daran, dass der SP zunächst um eins erhöht wird, wenn ein Wert auf dem Stack abgelegt wird. Da der PC zwei Byte benötigt wird bei einem Unterprogrammaufruf und beim Ausführen einer Interruptroutine der SP um in 2 erhöht, bei einem Rücksprung (ret-Befehl) um 2 verkleinert.
Genau betrachtet wird zunächst der SP um eins erhöht, das LSB des PC an der durch den SP angegebenen Ram-Adresse abgespeichert, danach der SP nochmals um eins erhöht und das MSB des PC abgelegt. Beim Rücksprung läuft alles in umgekehrter Reihenfolge ab.
Mit dem Befehl „push" („pop") können auch direkt adressierbare Werte auf dem Stack abgelegt (gelesen) werden. Dabei wird der SP jeweils um eins erhöht (verkleinert).
Achtung: Die Befehle call- und ret sowie push und pop müssen stets paarweise verwendet werden!

b. Die Registerbänke

Insgesamt stehen vier Registerbänke mit je 8 Byte (R0-R7) zur Verfügung, die im RAM an den ersten 32 Adressen 0h-1fH liegen. Es ist jeweils nur eine Registerbank aktiv. Die Auswahl der Registerbank

erfolgt mit zwei Bits im Programmstatuswort. Die nicht verwendeten Plätze im RAM stehen zur freien Verfügung.

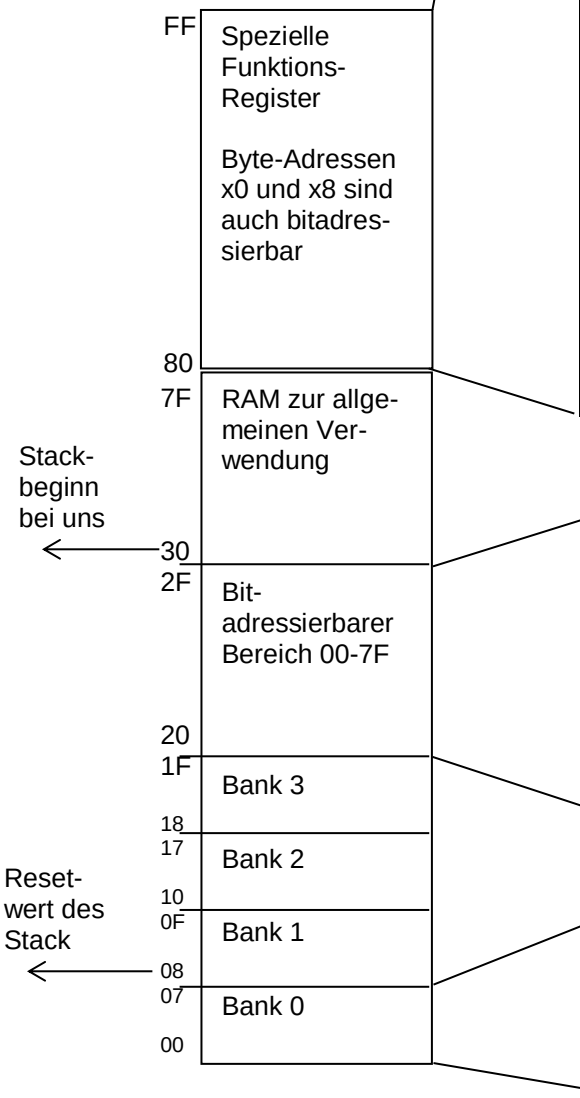
c. Die Funktionsregister SFR

Nach einem Reset haben alle SFR den Wert 0. Ausnahmen bilden lediglich die Ports die auf 0ffh, der SP, der auf 07h gesetzt wird und das Senderegister der seriellen Schnittstelle SBUF. Dieses wird nicht beeinflusst, da dies gleichbedeutend mit einem Sendebefehl wäre. Weitere Details zu den Registern werden später erläutert.

d. Der Datenpointer DPTR

Auch der DPTR ist, worauf der Name schon hindeutet, vergleichbar mit dem PC bzw. SP. Mit dem DPTR können Schreib- und Lesezugriffe auf das externe RAM bzw. Lesezugriffe auf Tabellenwerte im ROM erfolgen. Die Befehle hierzu sind „movx“ bzw. „movc“.

1.2.5 Struktur und Verwendung des internen RAMs



P0 *	Port 0	80 H
SP	Stack pointer	81 H
DPL	Data pointer, low byte	82 H
DPH	Data pointer, high byte	83 H
PCON	Power control register	87 H
TCON *	Timer control register	88 H
TMOD	Timer mode register	89 H
TL0	Timer 0, low byte	8A H
TL1	Timer 1, low byte	8B H
TH0	Timer 0, high byte	8C H
TH1	Timer 1, high byte	8D H
P1 *	Port 1	90 H
SCHON *	Serial channel control register	98 H
SBUF	Serial channel buffer register	99 H
P2 *	Port 2	0A0 H
IEN0 *	Interrupt enable register 0	0A8 H
IP0	Interrupt priority register 0	0A9 H
P3 *	Port 3	0B0 H
IEN1 *	Interrupt enable register 1	0B8 H
IP1	Interrupt priority register 1	0B9 H
IRCON *	Interrupt request ctrl register	0C0 H
CCEN	Compare/capture enable register	0C1 H
CCL1	Compare/capture register 1, low byte	0C2 H
CCH1	Compare/capture register 1, high byte	0C3 H
CCL2	Compare/capture register 2, low byte	0C4 H
CCH2	Compare/capture register 2, high byte	0C5 H
CCL3	Compare/capture register 3, low byte	0C6 H
CCH3	Compare/capture register 3, high byte	0C7 H
T2CON *	Timer 2 control register	0C8 H
CRCL	Compare/reload/capture reg. low byte	0CA H
CRCH	Compare/reload/capture reg. high byte	0CB H
TL2	Timer 2, low byte	0CC H
TH2	Timer 2, high byte	0CD H
PSW *	Program status word register	0D0 H
ADCON *	A/D converter control register	0D8 H
ADDAT	A/D converter data register	0D9 H
DAPR	D/A converter program register	0DA H
P6	Port 6	0DB H
ACC *	Accumulator	0E0 H
P4 *	Port 4	0E8 H
B *	B register	0F0 H
P5 *	Port 5	0F8 H

30

7F	7E	7D	7C	7B	..	
77	76	75	74	73		
6f	6e	6D	6C	6B		
..	..	..	..	..	.	

.....

.....

1F	1E	1D	1C	1B	..
17	16	15	14	13	
0F	0E	0D	0C	0B	
07	06	05	04	03	.

1F

07	Register R7
06	Register R6
05	Register R5
04	Register R4
03	Register R3
02	Register R2
01	Register R1
00	Register R0

1.2.6 Bitadressierbare Special-Funktion-Register

BIT- NAME								SFR- Name
7	6	5	4	3	2	1	0	Byte Adresse
P5.7	**P5.6**	**P5.5**	**P5.4**	**P5.3**	**P5.2**	**P5.1**	**P5.0**	**P5**
FFH	FEH	FDH	FCH	FBH	FAH	F9H	F8H	F8H
B.7	**B.6**	**B.5**	**B.4**	**B.3**	**B.2**	**B.1**	**B.0**	**B**
F7H	F6H	F5H	F4H	F3H	F2H	F1H	FOH	F0H
P4.7	**P4.6**	**P4.4**	**P4.4**	**P4.3**	**P4.2**	**P4.1**	**P4.0**	**P4**
EFH	EEH	EDH	ECH	EBH	EAH	E9H	E8H	E8H
A.7	**A.6**	**A.5**	**A.4**	**A.3**	**A.2**	**A.1**	**A.0**	**ACC**
E7H	E6H	E5H	E4H	E3H	E2H	E1H	EOH	E0H
BD	**CLK**	**-**	**BSY**	**ADM**	**MX2**	**MX1**	**MXO**	**ADCON**
DFH	DEH	DDH	DCH	DBH	DAH	D9H	D8H	D0H
CY	**AC**	**FO**	**RS1**	**RSO**	**OV**	**F1**	**P**	**PSW**
D7H	D6H	D5H	D4H	D3H	D2H	D1H	D0H	D8H
T2PS	**I3FR**	**I2FR**	**T2R1**	**T2R0**	**T2CM**	**T2I1**	**T2I0**	**T2CON**
CFH	CEH	CDH	CCH	CBH	CAH	C9H	C8H	C8H
EXF2	**TF2**	**IEX6**	**IEX5**	**IEX4**	**IEX3**	**IEX2**	**IADC**	**IRCON**
C7H	C6H	C5H	C4H	C3H	C2H	C1H	C0H	C0H
EXEN2	**SWDT**	**EX6**	**EX5**	**EX4**	**EX3**	**EX2**	**EADC**	**IEN1**
BFH	BEH	BOH	BCH	BBH	BAH	B9H	B8H	B8H
P3.7	**P3.6**	**P3.5**	**P3.4**	**P3.3**	**P3.2**	**P3.1**	**P3.0**	**P3**
B7H	B6H	B5H	B4H	B3H	B2H	B1H	BOH	B0H
EAL	**WDT**	**ET2**	**ES**	**ET1**	**EX1**	**ETO**	**EXO**	**IENO**
AFH	AEH	ADH	ACH	ABH	AAH	A9H	A8H	A8H
P2.7	**P2.6**	**P2.5**	**P2.4**	**P2.3**	**P2.2**	**P2.1**	**P2.0**	**P2**
A7H	A6H	A5H	A4H	A3H	A2H	A1H	A0H	A0H
SM0	**SM1**	**SM2**	**´REN**	**TB8**	**RB8**	**TI**	**RI**	**SCON**
9FH	9EH	9DH	9CH	9BH	9AH	99H	98H	98H
P1.7	**P1.6**	**P1.5**	**P1.4**	**P1.3**	**P1.2**	**P1.1**	**P1.0**	**P1**
97H	96H	95H	94H	93H	92H	91H	90H	90H
TF1	**TR1**	**TF0**	**TR0**	**IE1**	**LT1**	**IEQ**	**IT0**	**TCON**
8FH	8EH	8DH	8CH	8BH	SAH	89H	88H	88H
P0.7	**P0.6**	**P0.5**	**P0.4**	**P0.3**	**P0.2**	**P0.1**	**P0.0**	**P0**
87H	86H	85H	84H	83H	82H	81H	80H	80H

1.3 Befehlscodes

Alle Mitglieder der MCS-51 Familie haben die gleichen Befehlscodes. Der Befehlssatz ist optimiert für die Einzelbitverarbeitung und Boolsche Operationen als einem separaten Datentyp.
Eine Besonderheit der intelschen Mnemonik ist bei Mehrbytebefehlen, dass immer erst das Ziel, dann die Quelle angegeben wird. Ziel und Quellen werden auch als Operanden bezeichnet.

Opcode	Ziel ←	Quelle

Beispiel: mov P4,P5 ;kopiere die Daten von Port5 nach Port4
Der Assembler übersetzt diesen Befehl in 85 F8 E8, dabei ist 85 der Opcode und F8, e8 sind die Adressen der Ports.
In der Befehlstabelle stehen außer dem OP-Code noch

- die Wortbreite, das ist der Speicherbedarf den ein Befehl benötigt,
- die Ausführungsdauer in Maschinenzyklen, wobei ein Maschinenzyklus 1/12 der Taktfrequenz beträgt.
- Zudem werden noch die beeinflussten Flags im PSW angegeben.

1.4 Adressierungsarten

1.4.1 Adressierungsarten nach dem Hardwarezugriff

Als Ergebnis der Harvard-Architektur (Programm und Datenspeicher sind elektrisch und organisatorisch getrennt. Somit kann keine Datenmanipulation zu einer Veränderung des Programmcodes führen, was der Absturzsicherheit des Systems dient), werden Programm-, Daten und interne Speicher durch spezielle Befehle angesprochen.

FFFF 0000	Externes EPROM (Programm Konstanten)
	mov**c**

FFFF 0000	Externes RAM (Daten)
	mov**x**

FF 00	Internes RAM (SFR, Register, Daten, Stack)
	mov

- Der Programmspeicher als EPROM kann üblicherweise nur gelesen werden. Der Befehl lautet: **movc,** c steht für Code-Memory, das Ziel ist stets der Akku.

  Beispiel: Ascii-Zeichen-Text aus EPROM holen und seriell senden

```
        mov dptr,#1000h     ;Anfangsadresse des Textes in den Datenpointer
        clr a               ;Akku löschen, kein Offset
loop:   movc a,@a+dptr      ;Zeichen aus der Speicherstelle in den Akku kopieren
        jz ende             ;Sprung zur Marke ende falls Wert=0
        mov sbuf,a          ;Zeichen senden
wait:   jnb ti,wait         ;warten bis gesendet
        inc dptr            ;andernfalls Datenpointer erhöhen
        sjmp loop           ;nächsten Wert holen
```

- Der externe Datenspeicher (RAM) wird durch den Befehl **movx, x** steht für eXternal angesprochen

  Beispiel: Daten aus externen RAM holen und parallel ausgeben

```
        mov dptr,#1000h     ;Anfangsadresse in den Datenpointer
loop:   movx a,@dptr        ;Daten aus der Speicherstelle 1000h in den Akku kopieren
        jz ende             ;Sprung zur Marke ende falls Wert=0
        mov p4,a            ;Daten ausgeben
        inc dptr            ;Datenpointer erhöhen
        sjmp loop           ;nächsten Wert holen
```

- Das interne Obere und Untere Ram wird duch den Befehl **mov** gelesen und geschrieben

```
Beispiele: mov p4,p5            ;Daten von P5 nach P4 kopieren
           mov p1,#00001111b    ;Konstante (Binärzahl) an P1 ausgeben
```

1.4.2 Adressierungsarten allgemein

- **Konstanten Adressierung:** Operand wird direkt als Konstante angegeben.

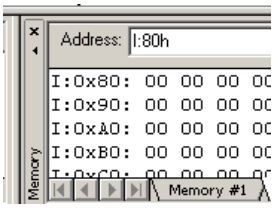
  Bsp: mov a,#12
  → der Wert 12 wird direkt in den Akku geschrieben

- **Direkte Adressierung:** Operand wird direkt durch eine 8-Bit-Adresse angegeben. Nur das interne RAM und die SFRs können direkt adressiert werden.

  Bsp: add a,8a (Ziel und Quelle werden direkt adressiert)
  → zum Akku wird der Inhalt der Zelle mit der Adresse 8A hinzuaddiert

- **Indirekte Adressierung:** Adresse des Operanden steht in einem Register. Internes und externes RAM lassen sich so adressieren.

  Bsp: movx a,@R0 (Ziel direkt, Quelle indirekt)
  → in den Akku wird der Inhalt der externen RAM-Zelle mit der Adresse, die in R0 steht, geschrieben.
  Hinweis: Die oberen 128Bytes (ab Adresse 080h) des internen RAMs können nur so angesprochen werden. In uVision kann das interne RAM im „Memory Window" durch die Eingabe von „I:Adresse" angezeigt werden.



- **Registerbefehle:** Der Operand entspricht einem Register R0-R7. Diese Befehle sind besonders kurz da die Adresse des Registers bereits ein Teil des OP-Codes ist.

  Bsp: djnz R7,Ziel

- Indizierte Adressierung: Diese Adressierung gilt eigentlich für das ROM und wurde oben schon angeführt. Sie dient dazu Tabellen aus dem ROM zu lesen, ähnelt aber der indirekten Adressierung und wird deshalb hier an einem Beispiel erläutert.

  Bsp: movc a, @a+dptr
  Adresse des Operanden ergibt sich, wenn man zum Datenpointer den Wert des Akkus hinzuaddiert, wobei das Ziel wiederum der Akku ist.

1.6 Zahlen und PSW

1.6.1 Darstellung von Zahlen

Grundsätzlich lassen sich folgende Zahlen unterscheiden:

- Vorzeichenlose Zahlen von 0..255
- vorzeichenbehaftete Zahl im Zweierkomplement: –128..+127
    - Das höchstwertige Bit ist das Vorzeichenbit.
    - Es gibt nur eine Null, und diese ist »positiv«.
    - Ist das Vorzeichenbit Null, ist das ZWK gleich der vorzeichenlosen Zahl.
    - Ist das Vorzeichenbit Eins, ist die Zahl negativ (also kleiner Null). Der Betrag ergibt sich durch Subtraktion von Null oder dem Einerkomplement (logische Negation aller Bits, »CPL«), gefolgt von Dekrement.

- ASCII-Steuerkode im Bereich 0..31, ASCII-Zeichen im Bereich 32..127, Sonderzeichen im Bereich 128..255
- Fest- und Gleitkommazahlen gibt es nicht im Befehlssatz und erfordern für deren Umsetzung sehr viel Code und Rechenzeit

1.6.2 PSW Statusregister (SFR)

das Programm Statuswort enthält Informationen über den Status der CPU.

- C: Carry Flag wird z.B. nach jeder Addition oder Subtraktion gesetzt, wenn das Ergebnis nicht im 8-Bit-Akkumulator darstellbar ist (Übertrag).
  dient zusätzlich als 1-Bit-Boolscher Accumulator, d.h. Ergebnisse logischer Bitoperationen stehen grundsätzlich im Carry-Bit. Der Befehl ORL C,P1.1 z.B. verknüpft das Carry-Bit mit dem

  Bit P1.1 (Port 1, Bit 1) ODER und schreibt das Ergebnis der Operation in das Carry-Bit zurück. Der Boolsche Akkumulator macht das sonst bei vielen Prozessoren notwendige Ausmaskieren von Bits vor deren Weiterverarbeitung überflüssig und beschleunigt damit den Programmablauf.
- AC: Auxiliary Carry Flag zeigt den Übertrag von Bit 3 und wird für BCD-Operationen verwendet.
- F0 ist ein vom Benutzer frei verfügbares Bit (Flag)
- Ov: Überlauf-Anzeige, wenn ein Abdriften in die 6. und 7. Bit auf einmal.
- P: Parität Indikator zeigt 1, wenn die Zahl im Akku ungerade ist
- Rs 0 und Rs 1: Auswahl der aktiven Registerbank.

Der 8051 enthält in seinem Programmstatuswort kein Z-Flag (Zero Flag)! Die bedingten Sprungbefehle JZ (Jump on Zero) bzw. JNZ (Jump on no Zero) werten bei ihrer Ausführung den Inhalt des Accus direkt aus.

Bsp: Abhängig von der Zahl Wert soll Led1 (Wert=0), Led2(0<Wert<50), Led3(50<=Wert<100) oder Led4(Wert>=100) leuchten.
Gesucht PAP und Assembleprogramm

2. Der Timer beim 80C535

2.1 Allgemeines

Ein Timer ist im Grunde nichts anderes als ein Zähler für die Maschinenzyklen. Da ein Maschinenzyklus aus 12 Taktperioden besteht, beträgt die Zählrate ein Zwölftel der Ozillatorfrequenz. Es ergibt sich dann bei einem Takt von 12 MHz eine Zeitbasis mit 1μs. Die Zähleingänge werden auch nach außen geführt, so daß die Timer auch tatsächlich als Zähler eingesetzt werden können. Timer und Zähler sind somit, bis auf die Herkunft ihrer Signale, identisch.
Der 80C535 besitzt drei Timer, wovon Timer 0 und Timer 1 nahezu identisch sind, während der Timer 2 recht komplexe Aufgaben wahrnehmen kann.

2.2 Register beim Timer 0 und 1

Für jeden Timer stehen zwei 8 Bit breite Zähler zu Verfügung. Zur Kontrolle der beiden Timer sind das TMOD-Register und das TCON-Register zuständig.

2.2.1 Zählregister

Zur Aufnahme von Zählwerten sind für jeden Timer zwei Register vorhanden. Diese Zählregister werden vom μC hochgezählt und können ausgelesen werden. Genauso können sie aber auch aus dem Programm heraus beschrieben werden. Die Bezeichnungen für die Register und Adressen sind wie folgt:

Timer 0 Low = TL0 08AH
Timer 0 High = TH0 08CH
Timer 1 Low = TL1 08BH
Timer 1 High = TH1 08DH

Jeder Timer kann bis maximal 2^{16} zählen. Beim Überlauf der Register kann ein Interrupt ausgelöst werden.

2.2.2 Timermodus-Register TMOD

Adresse 089H

Struktur des TMOD-Registers

	Gate	C/T	M1	M0	Gate	C/T	M1	M0
Kontrolle für Timer 1					Kontrolle für Timer 0			

Bedeutung der Bits:

- Gate: Ist das Bit 0 werden die Timer/Counter nur durch das TR-Bit in TCON ein- bzw. ausgeschaltet.
  Ist das Bit 1 erfolgt die Steuerung des Timer/Counter wenn der INT-Pin und das TR-Bit gesetzt sind.
- C/T: Umschaltung zwischen Zähler- und Timer-Funktion.
  C/T=1→ Zähler C/T=0 → Timer
- M0 M1 dienen zum Einstellen des Zählmodus

M1	M0		
0	0	Modus 0: Low-Byte des Timers dient als 5-Bit Vorteiler	
0	1	Modus 1: 16 Bit Timer/Counter	
1	0	Modus 2: 8 Bit selbstnachladender Timer/Counter. Beim Überlauf des Low-Bytes wird der Inhalt des High-Bytes in das Low-Byte übernommen	
1	1	Modus 3: Hier unterscheiden sich Timer 0 und Timer 1	
		Timer 0: Low- und High-Byte arbeiten als unabhängige Zähler	Timer 1: stoppt

Betriebsarten:

- Modus 0: Wird kam verwendet und wurde als Anpassung an vorangegangene Controller implementiert
- Modus 1: Sämtliche Bits werden zum Zählen benutzt. Beim Überlauf wird der Zähler zurückgesetzt und das Interrupt-Bit wird gesetzt. Eine Sprung in einer Interruptroutine erfolgt allerdings nur, wenn die Interrupts auch freigegeben sind.
- Modus 2: Dieser Auto-Reload-Mode ist der vielseitigste. Das Low-Byte dient als Zähler und im High-Byte steht eine konstante. Bei einem Überlauf des Low-Bytes geschieht folgendes:
- Das Interrupt-Bit wird gesetzt
- Der Inhalt des High-Byte wird in das Low-Byte kopiert und dient so als Startwert für den neuen Zählvorgang.
- Modus 3: Dieser Modus stoppt den Timer 1, wohingegen Timer 0 nun aus zwei unabhängigen Zählern besteht. Wird der Timer 1 auf einen anderen Modus eingestellt erhält man drei Timer/Counter.

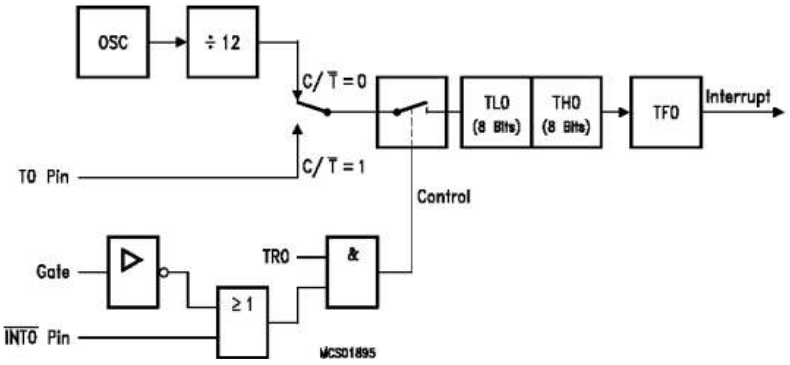


Abbildung 1: Mode 1: 16-Bit Timer/Counter

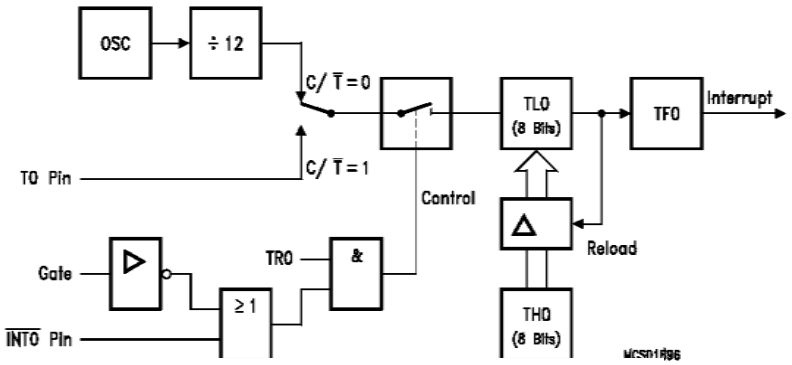


Abbildung 2: Mode 2: 8-Bit Timer/Counter mit Reload

2.2.3 Timer-Kontrollregister TCON

Adresse 088H
Struktur des TCON-Registers

	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
Bit-Nr	8F	8E	8D	8C	8B	8A	89	88
Kontrolle für Timer 1					Externe Interruptkontrolle			

Bedeutung der Bits:

- TFX: Das Überlauf-Flag wird bei jedem Überlauf automatisch gesetzt. Wird kein Interrupt verwendet kann durch Abfragen dieses Bits der Überlauf erkannt werden.
- TRX: Beim Setzen dieses Bits beginnt der Timer/Counter zu zählen, bzw. kann durch löschen wieder angehalten werden.
  Arbeitet Timer 0 in Modus 3 dient TR1 zur Steuerung des High-Bytes und TR1 steuert das Low-Byte. Timer 1 kann dann nur durch Wechseln in den Modus 3 angehalten werden.

3. Die serielle Schnittstelle

3.1 Allgemeines

Der serielle Port ist voll duplex, d.h. Daten können gleichzeitig gesendet und empfangen werden. Über den seriellen Port können Anweisungen und Daten empfangen (der Prozessor wird dann als Slave-Prozessor bezeichnet) oder ausgegeben werden können (Master-Prozessor. Es gibt auch die Möglichkeit, die Übertragungsgeschwindigkeit frei zu wählen.
Die serielle Übertragung ist im Vergleich zur Arbeitsgeschwindigkeit des Prozessors relativ langsam. Das Aussenden und Empfangen der einzelnen Bits erfolgt nach Start automatisch und ohne weiteres Zutun des Prozessors mit der eingestellten Geschwindigkeit. Während dieser Zeit erledigt der Prozessor andere Aufgaben.
Das Aussenden von Daten bedarf keines besonderen Startbefehls. Mit dem Beschreiben des seriellen Datenpuffers SBUF wird das Übertragen automatisch gestartet.
Der serielle Port ist empfangsgepuffert, was heißt, dass er mit dem Lesen eines zweiten Bytes beginnen kann, bevor das erste Byte in das Empfangsregister geschrieben wurde. Man beachte jedoch, dass ein Byte verloren geht, wenn das erste Byte nach vollendetem Empfang des zweiten Bytes von der CPU noch nicht gelesen wurde. Außerdem sind die Empfangs- und Senderegister des seriellen Ports physikalisch zwei verschiedene Register, die jedoch als Bestandteil der Spezialfunktionsregister unter denselben Namen (SBUF) und derselben Adresse (99h) anzusprechen sind: Ein Schreibbefehl an Adresse 99h lenkt die Daten in das Senderegister, ein Lesebefehl liest den Inhalt des Empfangsregisters. Es ist also unmöglich, in das Empfangsregister zu schreiben bzw. das Senderegister zu lesen.

3.2 Das Serial-Port-Control Register SCON

Adresse 98h (bitadressierbar)

Der serielle Port kann in vier Arten arbeiten. Der Arbeitsmodus und andere Kontrollbits werden inSCON eingestellt. In dem Kontrollregister finden sich ferner das 9. Datenbit für die Kommunikation von Prozessoren untereinander sowie zwei Bits, die einen Interrupt auslösen können.
Struktur des SCON-Registers

	SM0	SM1	SM2	REN	TB8	RB8	TI	RI
Bit-Nr	9F	9E	9D	9C	9B	9A	99	98

Bedeutung der Bits:

- SM2 In den Modi 2 und 3, den Multiprozessormodi, spielt es die entscheidende Rolle für die Verständigung. Ist SM2=1, wird das Empfangsinterruptflag RI gesetzt, wenn das neunte empfangene Bit RB8 eine Null ist, wenn also da gesendete Byte ein Datenbyte war. Ist SM2=0, wird immer ein Interrupt angefordert auch bei der Übermittlung von Datenbytes. Ist der Prozessor von einem anderen adressiert worden, muß SM2 auf Low gesetzt werden, damit nach jedem Übertragungsvorgang ein Interrupt zur Weiterverarbeitung der empfangenen Daten erfolgen kann. Im Modus 1 wird bei gesetztem SM2-Bit nur dann das Interruptflag RI gesetzt, wenn ein gültiges Stoppbit eingeht.
- REN Reception Enable. Es muss durch die Software gesetzt oder gelöscht werden. Nur falls REN=1 ist ein Datenempfang möglich.
- TB8 Transmit Bit 8. Es ist das 9. Datenbit, das in Modus 2 oder 3 übertragen wird. Es muss nach Wünschen des Anwenders gesetzt oder gelöscht werden und dient zur Unterscheidung von Adress- bzw. Datenbytes. Ist TB8=1 handelt es sich um ein Adressbyte, ansonsten um ein Datenbyte.
- TI Transmit Interrupt Flag. Dieses Bit zeigt der CPU an, wann der Sendevorgang beendet ist. Es wird durch die Hardware am Ende des achten Bits in Modus 0 und in den anderen Modi zu Beginn des neunten Bits (Stop Bit) gesetzt. Bei einem Sprung in die entsprechende Interruptroutine wird es nicht automatisch gelöscht, sondern muss per Befehl auf Null gesetzt werden.
- RI Receive Interrupt Flag. Dieses Bit zeigt der CPU an, wann der Empfangsvorgang beendet ist. Es wird durch die Hardware am Ende des achten Bits in Modus 0 und in den anderen Modi zu Beginn des neunten Bits (Stoppbit) gesetzt. Bei einem Sprung in die entsprechende Interruptroutine wird es nicht automatisch gelöscht, sondern muss per Befehl auf Null gesetzt werden. Eine Ausnahme hiervon ist der Beschreibung des SM2-Bits zu entnehmen.

3.3 Die Betriebsarten

SM0 und SM1 dienen zum Einstellen der Betriebsart

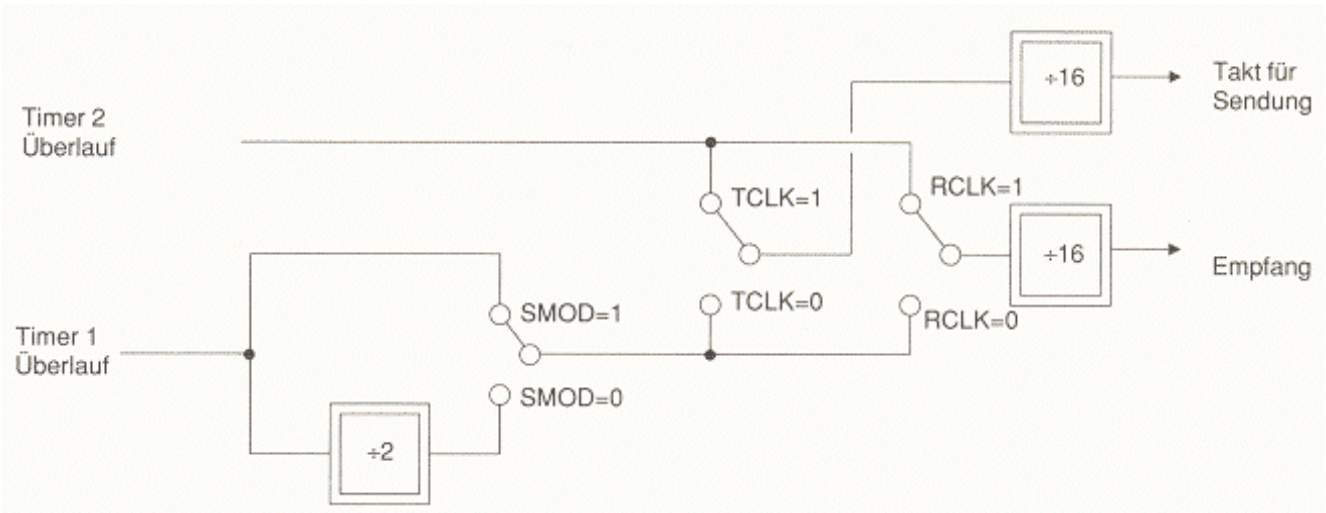
SM1	SM0	Modus	Beschreibung	Übertragungsgeschwindigkeit
0	0	0	Schieberegister	fosz/12
0	1	1	8-Bit-UART	Einstellbar
1	0	2	9-Bit-UART	fosz/32 oder fosz/64
1	1	3	9-Bit-UART	Einstellbar

3.3.1 Modus 0;

Die Daten, die empfangen oder gesendet werden, gehen über den RXD-Pin. Der TXD-Pin sendet in beiden Fällen einen Schiebetakt mit einem Zwölftel der Oszillatorfrequenz aus. Acht Bits werden ausgesendet oder empfangen, das niedrigste Bit (LSB) des übertragenen Bytes immer zuerst. Der TXD-Pin gibt für ein externes Schieberegister den Schiebetakt aus, deshalb ist in dieser Betriebsart auch kein Vollduplexmodus möglich.

3.3.2 Modus1

UART, variable Baudrate: Es werden 10 Bits an P3.0-(RxD) gesendet bzw. an P3.1-(TxD) empfangen. Die gesendeten/empfangenen Worte bestehen aus 1 Startbit (LOW), 8 Datenbits, 1 Stopbit (HIGH). Beim Empfang wird das Stopbit in RB8 (Bit2 in SCON) abgelegt. Die Baudrate wird über Timer 1 eingestellt. Die Empfangsbereitschaft ist durch REN=1 gegeben.
Es ist simultanes Senden und Empfangen möglich. Da in der Regel TXD- und RXD-Pins an Plus liegen, ist bei der Datenausgabe das erste Bit, das Startbit, eine Null, um anzuzeigen, dass mit der Datenübertragung begonnen wird. Es folgen acht Datenbits, das LSB zuerst. Zum Schluss erscheint das Stoppbit, eine Eins. Im Empfangsmodus geht das Stoppbit in das RB8-Bit von SCON.



Zur Veranschaulichung der Takterzeugung dient das folgende Bild.
Die Übertragungsgeschwindigkeit (Baudrate) wird durch das SMOD-Bits im Power Control Register PCON und die Überlaufrate des Zählers 1 oder 2 oder auch beide gleichzeitig bestimmt. Hierbei kann der eine Zählerüberlauf für die Übertragungsgeschwindigkeit und der andere für die Empfangsgeschwindigkeit verwendet werden. Dabei spielt es keine Rolle, in welchem Modus der Zähler arbeitet, z.B. als 16-Bit- Zähler, im Auto-Reload-Modus oder als externer Ereigniszähler. Der Überlauf wird jedoch nicht direkt als Takt benutzt, sondern zuvor durch 16 geteilt. Die so erzielte Baudrate berechnet sich aus:

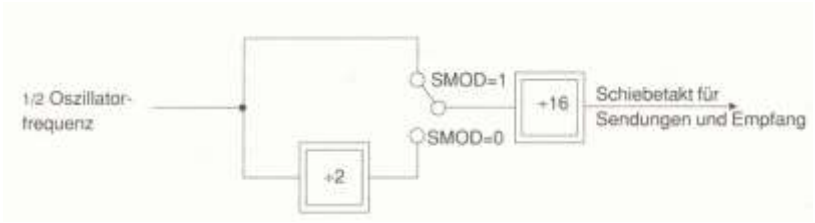
Bei gesetztem Receive Enable Bit REN wird der Datenempfang automatisch gestartet sobald eine negative Flanke am RXD-Pin entdeckt wird.
In der Praxis muss, sinnvoller weise in einer entsprechenden Interruptroutine, der serielle Puffer noch vor dem vollständigen Empfang eines zweiten Datenbytes gelesen und das Interrupttlag RI rückgesetzt werden, da sonst Informationen verlorengehen.

3.3.3 Modus 2 und Modus 3:

Es werden elf Bit gesendet über TXD oder empfangen über RXD: ein Startbit mit Wert Null, acht Datenbits, das LSB zuerst, ein programmierbares neuntes Bit und ein

Takterzeugung im Modus 1 und 3

Stoppbit mit Wert Eins. Beim Senden kann für das neunte Bit der Wert Null oder Eins gewählt werden. Beim Empfang geht das neunte Bit in das RB8-Bit des Seriellen Port Control Registers SCON.
In Modus 2 ist die Übertragungsrate zu 1/32 oder 1/64 der Oszillatorfrequenz programmierbar. Der Basistakt für das Schiebesignal ist weder der Timerüberlauf noch der Maschinentakt, sondern die halbe Oszillatorfrequenz, die mittels SMOD-Bit nochmals durch zwei geteilt werden kann. Anschließend gelangt das Signal durch den Vorteiler 16, so daß ein Schiebe- takt von 1/32 oder 1/64 der Oszillatorfrequenz entsteht.



Bei gesetztem Receive Enable Bit REN wird der Datenempfang automatisch gestartet sobald eine negative Flanke am RXD-Pin entdeckt wird.

Baudrate = 2^{SMOD} * Zählerüberlaufrate / 32

3.4 Aufgaben

1. Die serielle Schnittstelle soll mit ca. 2400 Baud betrieben werden. Verwenden Sie hierfür den Timer1 als selbstnachladender Timer und bestimmen Sie den Nachladewert für den Timer.
2. Initialisieren Sie den Timer1 vollständig. Vergessen Sie nicht S^{MOD} richtig zu setzen. Dies befindet sich im PCON-Register, das nicht bitadressierbar ist.
3. Schreiben Sie den Programmteil, der die serielle Schnittstelle folgendermaßen einstellt:
   - Betriebsart 1 (8 Bit UART)
   - Baudrate 2400 Bit/s

Entwerfen Sie ein Programm, das ein Byte auf der seriellen Schnittstelle ausgibt

4. Die Interrupts beim 80C535

4.1 Allgemeines

Wie erkennt die CPU eines PC, dass der Benutzer die Maus bewegt hat? Es gibt zwei Möglichkeiten. Entweder prüft die CPU in regelmäßigen Abständen ob die Maus bewegt wurde- die Zeitabstände dürfen natürlich nicht zu groß sein- oder die Maus „meldet sich“ selbst bei der CPU und unterbricht das aktuell bearbeitete Programm. Interrupts (I) sind also Programmunterbrechungen, die durch die Hardware ausgelöst werden (vgl. Zeitungsleser der von einer Fliege oder vom Klingeln gestört wird). So ist z.B. die mit Zählern aufgebaute Zeitschleife unsinnig, da die CPU in dieser Zählschleife keine anderen Aufgaben mehr wahrnehmen kann.

4.2 Abläufe bei der Abarbeitung des Interrupts

- Beenden des aktuellen Befehls
- Merken welcher Befehl nach der Abarbeitung des Is auszuführen wäre. Hierzu wird die Rom-Adresse dieses Befehls (PC) in einem speziellen Speicherbereich – dem Stack- gespeichert. Dies entspricht den Abläufen bei einem Unterprogrammaufruf.
- Abhängig vom Ursprung des Is wird das Programm an einer ganz speziellen Adresse fortgesetzt. D.h. der PC wird mit einem Wert beschrieben, dem I-Vektor.
- Die Programmsequenz, die nun abläuft, heißt I-Service-Routine (ISR) und muss stets mit einem „Reti“-Befehl abgeschlossen werden.
- Die bedeutet für die CPU, dass die oben abgespeicherte Rücksprungadresse vom Stack in den PC geladen wird.

Man sollte bei der ISR stets deren Ausführungsdauer, vor allem in Relation zur restlichen Laufzeit des Programms, berücksichtigen. Sonst könnte es z.B. bei einem Timer-I passieren, dass nur noch die ISR abgearbeitet wird.

Der 80C535 besitzt 12 Interrupts, von denen einige nur von einer Quelle stammen andere aber von zwei verschiedenen Ereignissen aufgelöst werden können. Löst ein Ereignis ein Interrupt aus, so wird diese Anforderung in der Regel bis zur Abarbeitung zwischengespeichert.

4.3 Bearbeitung eines Interrupts

Beim Auftreten eines Interrupts wird zunächst ein spezifisches, für den I. reserviertes Interrupt-Bit gesetzt, mit dem der I. gespeichert wird. Dies ist z.B. erforderlich, wenn gerade ein I. mit höherer Priorität abgearbeitet wird, der von dem mit niederer Priorität nicht unterbrochen werden kann. Der Zustand dieser Bits löst letztendlich den I. aus. Allerdings wird ein I. nur dann tatsächlich ausgeführt wenn er freigeschaltet wurde. Hierzu ist für jeden I. nochmals ein Bit reserviert. Die Bearbeitung eines I.s bedeutet praktisch, daß ein Sprung (LCALL) zu einer für den I. spezifischen Adresse ausgeführt wird. Diese Einsprungvektoren zeigt die folgende Tabelle. Zusätzlich ist mit angegeben in welchem Register die I. auslösenden Bits sind.

Beim Externen Interrupt 0 verzweigt das Programm zur Adresse 0003H. Hier könnte jetzt die Interruptroutine stehen, allerdings nur wenn diese nicht mehr als 8 Byte lang ist, weil ansonsten der Adressbereich des nächsten Vektors erreicht wird. Ist das Unterprogramm zu lange wird mit einem weiteren Sprung zur eigentlichen Routine verzweigt. Abgeschlossen wird ein I. stets mit dem Befehl RETI.

Name	**Vektor**	**Auslösende Bits**	**Teil von**
Externer I. 0	0003H	IE0	TCON
Timer 0 I.	000BH	TF0	TCON
Externer I. 1	0013H	IE1	TCON
Timer 1 I.	001BH	TF1	TCON
Serieller Port I.	0023H	RI oder TI	SCON
Timer 2 I.	002BH	TF2 oder EXF2	IRCON
A/D-Wandler I	0043H	IADC	IRCON
Externer I. 2	004BH	IEX2	IRCON
Externer I. 3	0053H	IEX3	IRCON
Externer I. 4	005BH	IEX4	IRCON
Externer I. 5	0063H	IEX5	IRCON
Externer I. 6	006BH	IEX6	IRCON

4.5 Interrupt-Register

Durch die relativ große Zahl an Interrupts sind auch viele Register zur Steuerung erforderlich.
Mit den beiden Interrupt-Freigaberegistern IEN0 und IEN1 wird die Ausführung der I.s gesteuert. Mit einer Eins werden die I.s jeweils freigegeben. Mit dem EAL-Bit in IEN0 können die I.s komplett aktiviert oder gesperrt werden.
Im IRCON-Register befinden sich die I. auslösenden Bits. Beim Auftreten eines I.s werden also die Bits dieses Registers gesetzt. Sie können aber auch durch einen Softwarebefehl gesetzt und so der I. praktisch simuliert werden. Einige dieser Bits werden mit dem Sprung in die I-Routine gelöscht, während andere per Softwarebefehl vom Programmierer gelöscht werden müssen.
Schließlich gibt es noch zwei Interrupt-Prioritäts-Register IP0 und IP1, mit denen die Rangfolge der einzelnen I.s eingestellt werden kann. Es existieren vier I-Ebenen, denen die I.s jeweils zugeordnet werden. Für die Zuordnung auf eine dieser vier Ebenen wären für jeden I. zwei Bits und somit insgesamt 24 Bits erforderlich. Um diese Zahl zu beschränken werden immer zwei I.s gleichzeitig einer Ebene zugeordnet.

4.5.1 Interrupt-Enable Register IEN0

Adresse A8 (bitadressierbar)

Struktur des IEN0-Registers

	EAL	WDT	ET2	ES	ET1	EX1	ET0	EX0
Bit-Nr	AF	AE	AD	AC	AB	AA	A9	A8

Bedeutung der Bits:

- EAL: Enable All
  → EAL=0 alle I. sind gesperrt
  → EAL=1 alle I.s frei, deren Freigabebits in EIN0 und IEN1 gesetzt sind.
- WDT: Watchdog Timer Reset
  WDT muß auf Eins gesetzt werden und anschließend SWDT in IEN1 um den Watchdog Timer auf Null zu setzen
- ET2: Enable Timer 2
  ET2=1 →Timer2 I. bei Überlauf des Timer2 oder beim Auftreten eines exteren Reload-Signals
- ES: Enable Serial Port
  ES=1 →die Bits TI oder RI können einen I. auslösen
- ET1: Enable Timer 1
  ET1=1 →I. beim Überlauf von Timer 1
- EX1: Enable External Interrupt 1
  EX1=1 →Low-Pegel oder negative Flanke an INT1-Pin löst I. aus
- ET0: Enable Timer 0
  ET0=1 →I. beim Überlauf von Timer 0
- EX0: Enable External Interrupt 0
  EX0=1 →Low-Pegel oder negative Flanke an INT0-Pin löst I. aus

4.5.2 Interrupt-Enable Register IEN1

Adresse B8 (bitadressierbar)

Struktur des IEN1-Registers

	EXEN2	SWDT	EX6	EX5	EX4	EX3	EX2	EADC
Bit-Nr	BF	BE	BD	BC	BB	BA	B9	B8

Bedeutung der Bits:

- EXEN2: Enable External Interrupt 2
  EXEN=1 →Reload-Impuls an Pin T2EX (P1.5) löst I. aus
  EXEN=0 →I. gesperrt, jedoch nicht der Reload-Vorgang
- SWDT: Watchdog Timer Start
  SWDT=1→ Wachtdog Timer beginnt zu zählen
- EXn: Enable External Interrupt n n=3,4,5,6
  EXn=1 → negative Flanke an Px.y löst I. aus. Dies kann ein externer Impuls, ein Capture Impuls oder im Compare-Modus die Ausgabe einer Compare-Signals sein.
  Zuordnung:
  EX6 → P1.3 EX4 → P1.1
  EX5 → P1.2 EX3 → P1.0
- EX2: Enable External Interrupt 2
  EX6=1 → Low Pegel oder negative Flanke an Pin INT2 (P1.4) löst I. aus.
- EADC: Enable A/D-Converter Interrupt
  EADC=1 → I. wird nach einer A/D-Wandlung gestartet

4.5.3 Interrupt-Kontroll Register IRCON

Adresse C0 (bitadressierbar)

	EXF2	TF2R	IEX6	IEX5	IEX4	IEX3	IEX2	IADC
Bit-Nr	C7	C6	C5	C4	C3	C2	C1	C0

Struktur des IRCON-Registers
Anmerkung: Wenn keine weiteren Angaben gemacht werden, werden die Bits beim Sprung in die I.-Routine automatisch gelöscht.

- EXF2: Timer 2 externes Relaod-Flag
  Wird im Reload-Betrieb bei einer negativen Flanke am Pin T2EX gesetzt, wenn außerdem das Bit EXEN2 im IEN1 Register eins ist. Arbeitet der Timer nicht im Reload-Betrieb kann es als weiterer externer I. 2 verwendet werden.
  EXF2 wird nicht automatisch gelöscht, sondern muss per Software-Befehl gelöscht werden.
- TF2: Timer 2 Überlauf
  Wird beim Überlauf von Timer 2, wird aber nicht automatisch gelöscht sondern per Software.
- IEXn: Externer Interrupt n n=3,4,5,6
  Werden im Compare-Betrieb gesetzt, wenn Gleichheit zwischen dem Inhalt von Timer 2 und dem CCx-Register besteht. Im Normalbetrieb kann es als externer I.n (Pin y.z) verwendet werden.
  Gelöscht durch die Hardware.
  Zuordnung:

Interrupt	Register	Externer I.-Pin
IEX 6	CC3	P1.3
IEX 5	CC2	P1.2
IEX 4	CC1	P1.1
IEX 3	CC0	P1.0

- IEX2: Externer Interrupt 2
  Wird durch eine Flanke am Pin /INT2 (P1.4) gesetzt und durch Hardware gelöscht.
- IADC: A/D-Converter Interrupt
  Wird am Ende einer A/D-Wandlung gesetzt und muß per Software gelöscht werden.

4.5.5 Interrupt-Prioritäts Register

a. IP0

Adresse A9

Struktur des IP0-Registers

---	WDTS	IP0.5	IP0.4	IP0.3	IP0.2	IP0.1	IP0.0

b. IP1

Adresse B9

Struktur des IP0-Registers

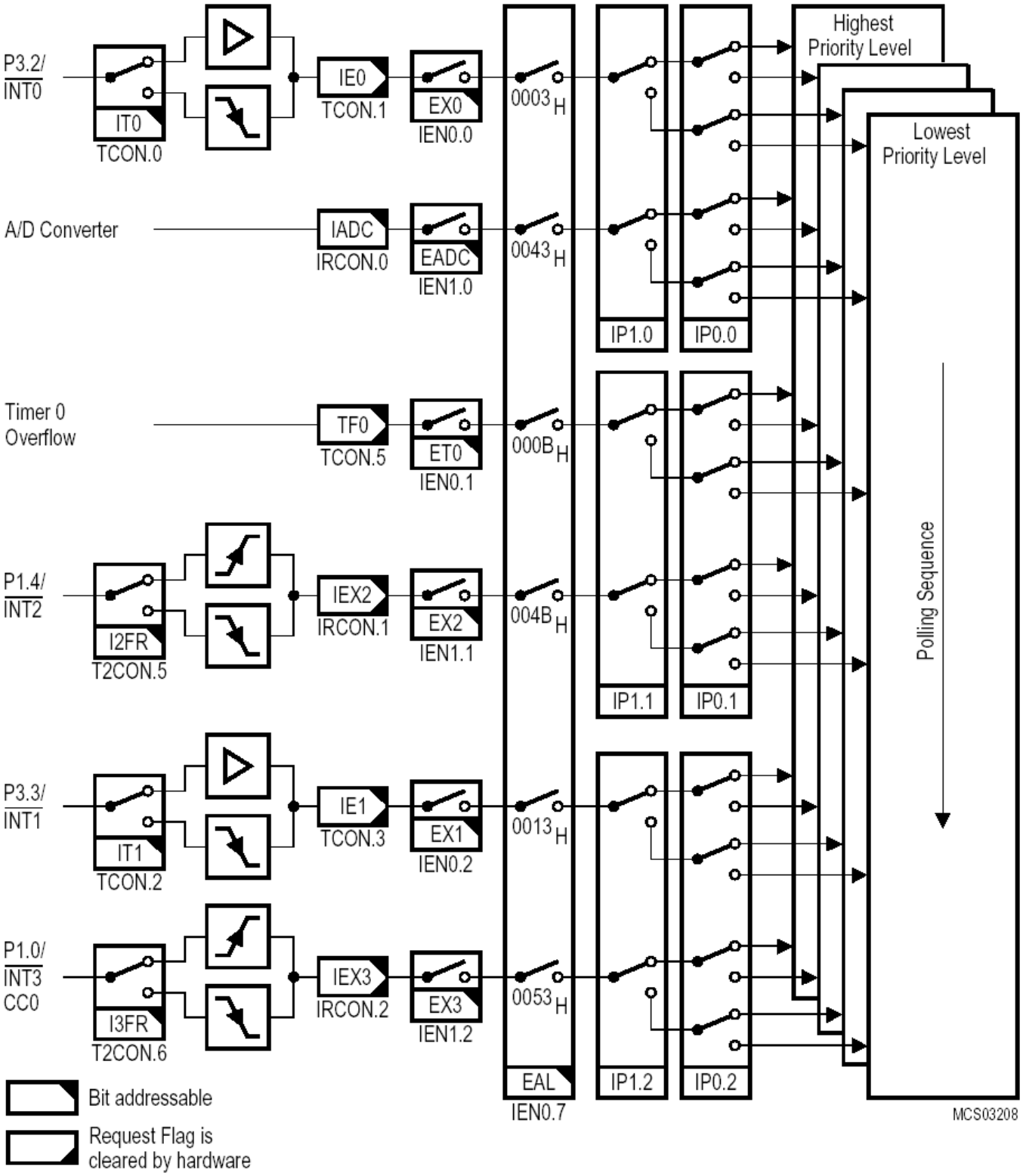
---	---	IP1.5	IP1.4	IP1.3	IP1.2	IP1.1	IP1.0

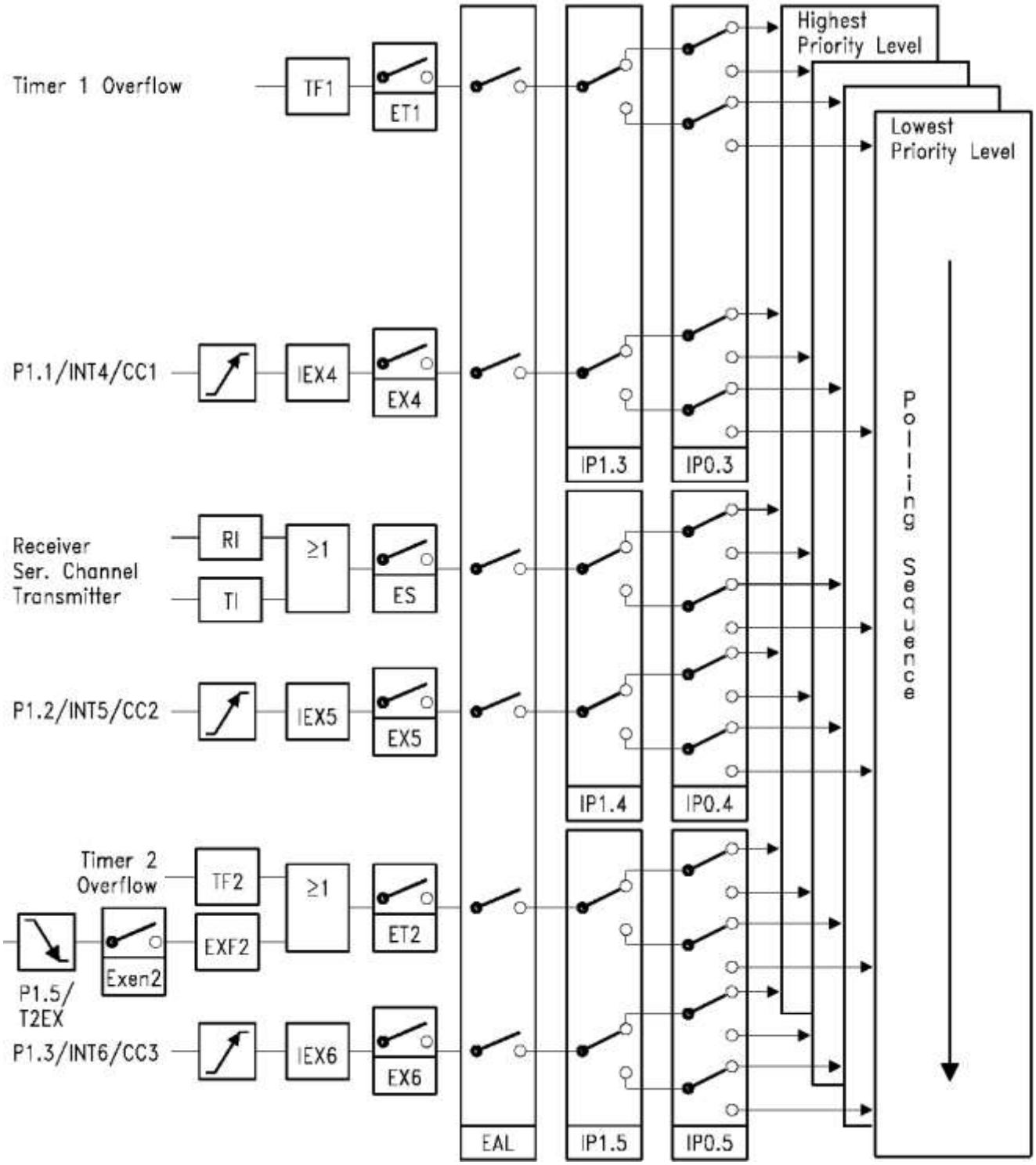
Wie bereits erwähnt werden stets zwei I.s gemeinsam einer Prioritätsebene zugeordnet. Diese I-Paare und die Bits mit denen die Zuordnung erfolgt können der folgenden Tabelle entnommen werden. Treten beide I.s eines Paares gleichzeitig auf, wird der links stehende I. zuerst ausgeführt.
Treten zwei I.s, die sich auf der gleichen Prioritätsebene befinden gleichzeitig auf so wird der in der Tabelle weiter oben stehende I. als erster ausgeführt.

	Interrupt-Paare		Prioritätsbit	
0	Externer I. 0	A/D-Wandler I.	IP1.0	IP0.0
1	Timer 0 I.	Externer I. 2	IP1.1	IP0.1
2	Externer I. 1	Externer I. 3	IP1.2	IP0.2
3	Timer 1 I.	Externer I. 4	IP1.3	IP0.3
4	Serieller Port I.	Externer I. 5	IP1.4	IP0.4
5	Timer 2 I.	Externer I. 6	IP1.5	IP0.5

Die Bestimmung der Prioritätsebene kann aus der nächsten Tabelle entnommen werden, wobei n=0-5 die Nummer des I-Paares angibt

IP1.n	IP0.n	
0	0	Ebene 0 (niedrigste Ebene)
0	1	Ebene 1
1	0	Ebene 2
1	1	Ebene 3 (höchste Ebene)


P3.2/
INT0
IT0
TCON.0
IE0
TCON.1
EX0
IEN0.0
0003H
A/D Converter
IADC
IRCON.0
EADC
IEN1.0
0043H
Timer 0
Overflow
TF0
TCON.5
ET0
IEN0.1
000BH
P1.4/
INT2
I2FR
T2CON.5
IEX2
IRCON.1
EX2
IEN1.1
004BH
P3.3/
INT1
IT1
TCON.2
IE1
TCON.3
EX1
IEN0.2
0013H
P1.0/
INT3
CC0
I3FR
T2CON.6
IEX3
IRCON.2
EX3
IEN1.2
0053H
EAL
IEN0.7
IP1.0
IP0.0
IP1.1
IP0.1
IP1.2
IP0.2
Highest
Priority Level
Lowest
Priority Level
Polling Sequence
Bit addressable
Request Flag is
cleared by hardware
MCS03208


Timer 1 Overflow
TF1
ET1
P1.1/INT4/CC1
IEX4
EX4
Receiver
Ser. Channel
Transmitter
RI
TI
≥1
ES
P1.2/INT5/CC2
IEX5
EX5
Timer 2
Overflow
TF2
EXF2
≥1
ET2
Exen2
P1.5/
T2EX
P1.3/INT6/CC3
IEX6
EX6
EAL
IP1.3
IP0.3
IP1.4
IP0.4
IP1.5
IP0.5
Highest
Priority Level
Lowest
Priority Level
Polling Sequence

5. TWI-Schnittstelle

5.1 Register

SSCON (93H:
SSCS (94H):
SSDAT (95h):
SSADR (96h):

Table 28. TWI SFRs

Mnemonic	Add	Name	7	6	5	4	3	2	1	0
SSCON	93h	Synchronous Serial Control	CR2	SSIE	STA	STO	SI	AA	CR1	CR0
SSCS	94h	Synchronous Serial Control•Status	SC4	SC3	SC2	SC1	SC0	-	-	-
SSDAT	95h	Synchronous Serial Data	SD7	SD6	SD5	SD4	SD3	SD2	SD1	SD0
SSADR	96h	Synchronous Serial Address	A7	A6	A5	A4	A3	A2	A1	A0

SSCON is used to enable the TWI interface, to program the bit rate (see Table 79), to enable slave modes, to acknowledge or not a received data, to send a START or a STOP condition on the 2-wire bus, and to acknowledge a serial interrupt. A hardware reset disables the TWI module.
SSCS contains a status code which reflects the status of the 2-wire logic and the 2-wire bus. The three least significant bits are always zero. The five most significant bits contains the status code. There are 26 possible status codes. When SSCS contains F8h, no relevant state information is available and no serial interrupt is requested. A valid status code is available in SSCS one machine cycle after SI is set by hardware and is still present one machine cycle after SI has been reset by software. to Table 85. give the status for the master modes and miscellaneous states.
SSDAT contains a byte of serial data to be transmitted or a byte which has just been received. It is addressable while it is not in process of shifting a byte. This occurs when 2-wire logic is in a defined state and the serial interrupt flag is set. Data in SSDAT remains stable as long as SI is set. While data is being shifted out, data on the bus is simultaneously shifted in; SSDAT always contains the last byte present on the bus.
SSADR may be loaded with the 7-bit slave address (7 most significant bits) to which the TWI module will respond when programmed as a slave transmitter or receiver. The LSB is used to enable general call address (00h) recognition.
Figure 51 shows how a data transfer is accomplished on the 2-wire bus.

6. Der AD-Wandler beim 80C535

6.1.1 Allgemeines

Der AD-Wandler wird an den Anschlüssen AN0-AN7 nach außen geführt. Allerdings besitzt der uC nicht acht, sondern nur einen AD-Wandler, auf den die einzelnen Anschlüsse mit einem Multiplexer aufgeschaltet werden. Realisiert sind außerdem ein Sample-and-hold-Kreis und eine programmierbare Referenzspannung. Der AD-Wandler arbeitet nach dem Prinzip der sukzessiven Approximation.

6.2 Register

Für die Kontrolle des Wandlers ist das ADCON-Register und für die Programmierung der Referenzspannung das DAPR-Register zuständig. Das Ergebnis einer Wandlung steht stets im ADDAT-Register.

6.2.1 AD-Kontroll-Register ADCON

Adresse 0D8H (Bit-adressierbar)

Struktur des ADCON-Registers:

	BD	CLK	0	BSY	ADM	MX2	MX1	MX0
Bit-Nr	DF	DE	DD	DC	DB	DA	D9	D8

Die Bits BD und CLK werden nicht für den AD-Wandler benötigt.

Bedeutung der Bits:

- BD: Baud Rate. Bei gesetztem Bit stammt die Baud-Rate des seriellen Ports in Modus 1 bzw. 3 aus dem internen Baudratengenerator
- CLK: Systemtaktfreigabe. Steht hier eine Eins wird ein Rechtecksignal mit 1/12 der Taktfrequenz an CLKOUT(Pin1.6) ausgegeben.
- 0: keine Funktion; muss immer Null sein
- BSY: Busy-Flag zeigt mit einer Eins an, dass die Wandlung noch nicht fertig ist. Sobald das Ergebnis in ADDAT steht wird BSY zu Null
- ADM: Konvertierungsmodus; mit einer Eins wird eine kontinuierliche Wandlung mit Null eine Einzelwandlung eingestellt
- MX0, MX1, MX2 — Mit diesen drei Bits wird der -Eingang gewählt

			Gewählter Eingang	Pin
0	0	0	Analogeingang 0	AN0
0	0	1	Analogeingang 1	AN1
0	1	0	Analogeingang 2	AN2
0	1	1	Analogeingang 3	AN3
1	0	0	Analogeingang 4	AN4
1	0	1	Analogeingang 5	AN5
1	1	0	Analogeingang 6	AN6
1	1	1	Analogeingang 7	AN7

6.2.2 AD-Datenregister ADDAT

Adresse 0D9H

Das Ergebnis einer Wandlung bleibt solange in diesem Register bis ein neuer Wert ermittelt wurde. Wird der AD-Wandler nicht benötigt kann dieses Register als zusätzlicher Speicher verwendet werden.

6.2.3 AD-Programierregister DAPR

Adresse 0DAH

Mit diesem Register wird sowohl die obere (V_{ref}) als auch die untere (V_{GND}) interne Referenzspannung programmiert. Hierzu kann jeweils die extern an V_{AREF} (Pin11) bzw. V_{AGND} (Pin12) angeschlossene Spannung in Schritten von 1/16 geteilt werden. Zu beachten ist, dass die Differenz der programmierten Werte immer größer als 1V ist. Überschreitet die zu messende Spannung den programmierten positven Wert liefert der AD-Wandler 0FFH als Ergebnis. Beim Unterschreiten der programmierten negativen Spannung lautet das Ergebnis 00H.
Wird eine externe Spannung von V_{AREF}=5V und V_{AGND}=0V angeschlossen ergeben sich folgende Spannungswerte für die internen Referenzspannungen.

DAPR:

7	6	5	4		3	2	1	0

				V_{ref} intern inV					V_{GND} intern in V
0	0	0	0	5V	0	0	0	0	0
0	0	0	1	-	0	0	0	1	0,3125
0	0	1	0	-	0	0	1	0	0,6250
0	0	1	1	-	0	0	1	1	0,9375
0	1	0	0	1,250	0	1	0	0	1,2500
0	1	0	1	1,5625	0	1	0	1	1,5625
0	1	1	0	1,8750	0	1	1	0	1,8750
0	1	1	1	2,1875	0	1	1	1	2,1875
1	0	0	0	2,5000	1	0	0	0	2,5000
1	0	0	1	2,8125	1	0	0	1	2,8125
1	0	1	0	3,1250	1	0	1	0	3,1250
1	0	1	1	3,4375	1	0	1	1	3,4375
1	1	0	0	3,7500	1	1	0	0	3,7500
1	1	0	1	4,0625	1	1	0	1	-
1	1	1	0	4,3750	1	1	1	0	-
1	1	1	1	4,6875	1	1	1	1	-

Eine AD-Wandlung wird immer mit dem Schreiben ins DAPR-Register gestartet, wobei eine laufende Wandlung unterbrochen wird. Die Dauer einer Wandlung beträgt 15 Taktzyklen, wenn in DAPR der Wert Null steht. Ist eine oder beide Spannungen programmiert müssen jeweils 7 Taktzyklen hinzugerechnet werden.
Vorteilhaft anwenden läßt sich die Programmierung der Referenzspannungen, wenn z.B. die Genauigkeit erhöht werden soll oder wenn die zu messenden Spannungen unterschiedliche Pegel aufweisen.

7. Leistungsreduzierung

7.1.1 Das Power-Control-Register PCON

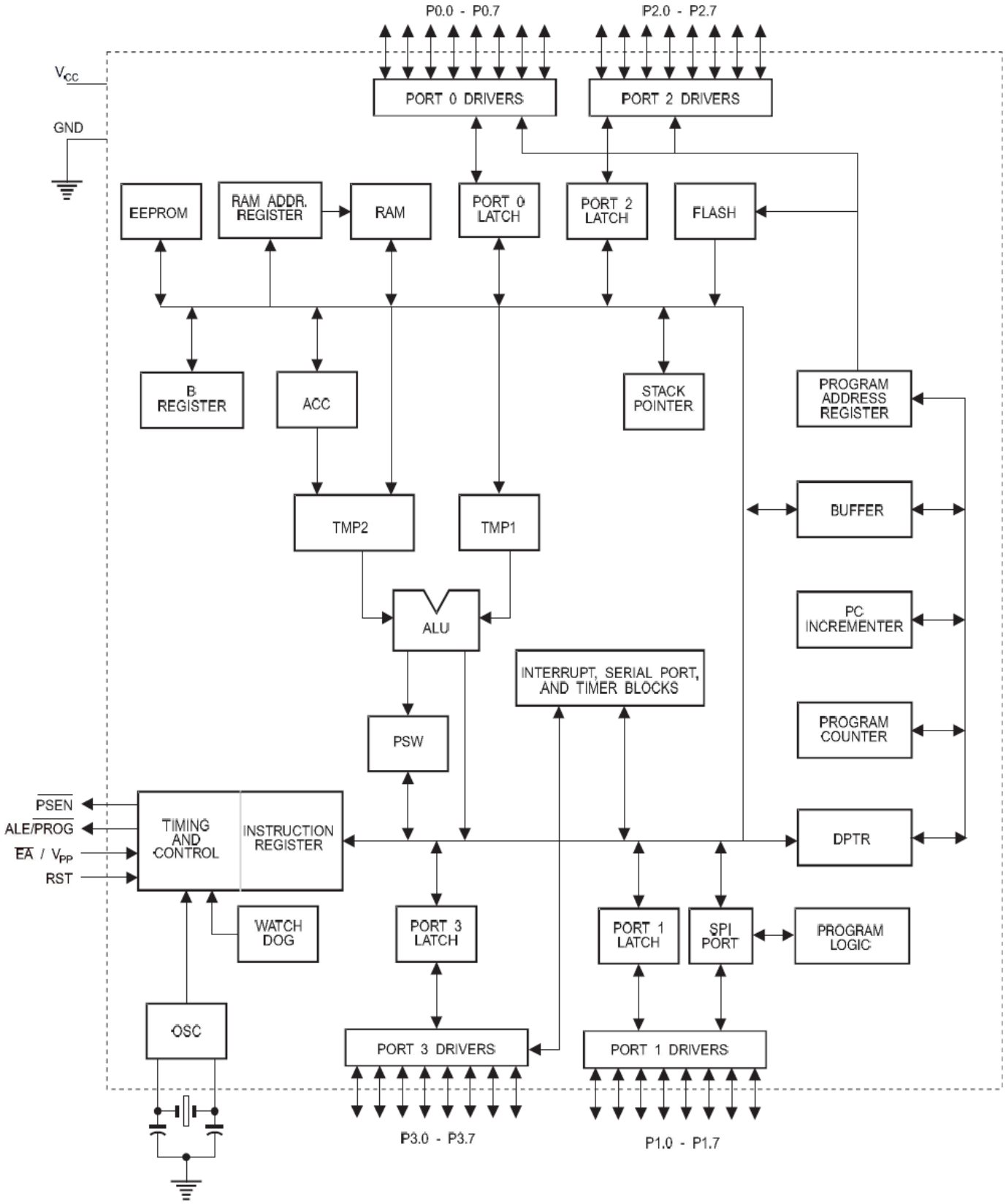
Adresse 87H
Struktur des PCON-Registers:

SMOD	--	--	--	GF1	GF0	PD	IDL

SMOD: Serial Port Modus

8. *Anhang* 1:Blockschaltbild, Befehlssatz

8.1 Blockschaltbild



9. Anhang3

Timer

	Gate	C/T	M1	M0	Gate	C/T	M1	M0
Kontrolle für Timer 1					Kontrolle für Timer 0			

M1	M0		
0	0	Modus 0: Low-Byte des Timers dient als 5-Bit Vorteiler	
0	1	Modus 1: 16 Bit Timer/Counter	
1	0	Modus 2: 8 Bit selbstnachladender Timer/Counter. Beim Überlauf des Low-Bytes wird der Inhalt des High-Bytes in das Low-Byte übernommen	
1	1	Modus 3: Hier unterscheiden sich Timer 0 und Timer 1	
		Timer 0: Low- und High-Byte arbeiten als unabhängige Zähler	Timer 1: stoppt

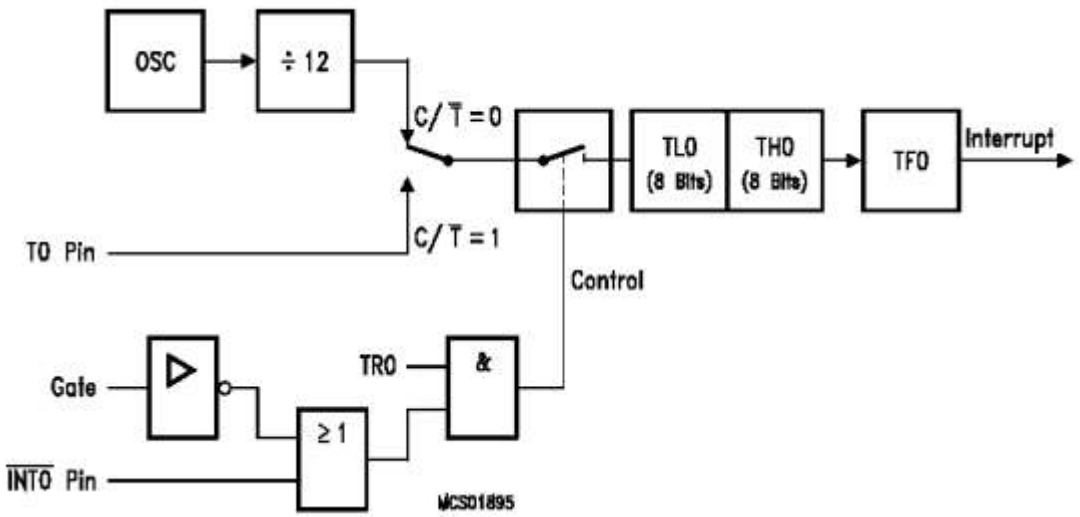


Abbildung 3: Mode 1: 16-Bit Timer/Counter

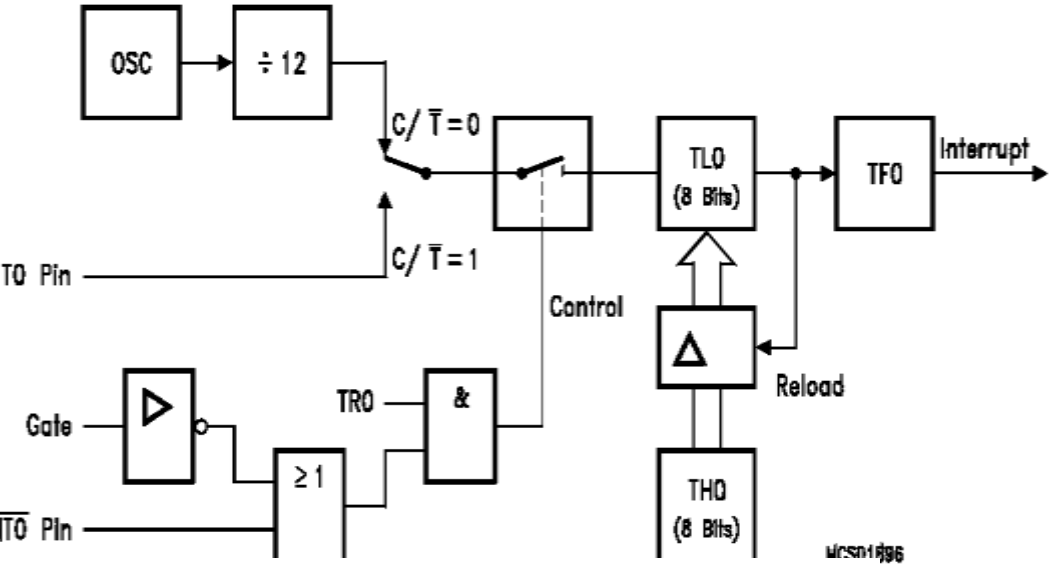


Abbildung 4: Mode 2: 8-Bit Timer/Counter mit Reload

10. Folien Interrupt

	EAL	WDT	ET2	ES	ET1	EX1	ET0	EX0
Bit-Nr	AF	AE	AD	AC	AB	AA	A9	A8

	EXEN2	SWDT	EX6	EX5	EX4	EX3	EX2	EADC
Bit-Nr	BF	BE	BD	BC	BB	BA	B9	B8

	EXF2	TF2R	IEX6	IEX5	IEX4	IEX3	IEX2	IADC
Bit-Nr	C7	C6	C5	C4	C3	C2	C1	C0

---	WDTS	IP0.5	IP0.4	IP0.3	IP0.2	IP0.1	IP0.0

---	---	IP1.5	IP1.4	IP1.3	IP1.2	IP1.1	IP1.0