

**Konfiguration eines PIC-Mikrocontrollers → s. Häckchen - Configuration Bits set in Code!!**

Address	Value	Field	Category	Setting
300001	01	OSC	Oscillator	XT
		FCMEN	Fail-Safe Clock Monitor Enable	Disabled
		IESO	Internal External Switch Over Mode	Disabled
300002	1F	PUT	Power Up Timer	Disabled
		BODEN	Brown Out Detect	Enabled in hardware, SBODEN disabled
		BODENV	Brown Out Voltage	2.0V
300003	1E	WDT	Watchdog Timer	Disabled-Controlled by SWDIEN bit
		WDTPS	Watchdog Postscaler	1:32768
300005	81	CCP2MUX	CCP2 Mux	RC1
		PBADEN	PortB A/D Enable	PORTB<4:0> configured as digital I/O on RESET
		LPT1OSC	Low Power Timer1 Osc enable	Disabled
		MCLR	Master Clear Enable	MCLR Enabled, RE3 Disabled
300006	85	STVR	Stack Overflow Reset	Enabled
		LVP	Low Voltage Program	Enabled
		XINST	Extended Instruction Set Enable bit	Disabled
300008	07	CP_0	Code Protect 00800-03FFF	Disabled
		CP_1	Code Protect 04000-07FFF	Disabled
		CP_2	Code Protect 08000-0BFFF	Disabled
300009	C0	CPB	Code Protect Boot	Disabled
		CPD	Data EE Read Protect	Disabled
30000A	07	WRT_0	Table Write Protect 00800-03FFF	Disabled
		WRT_1	Table Write Protect 04000-07FFF	Disabled
		WRT_2	Table Write Protect 08000-0BFFF	Disabled
30000B	E0	WRTC	Config. Write Protect	Disabled
		WRTB	Table Write Protect Boot	Disabled
		WRTD	Data EE Write Protect	Disabled
30000C	07	EBTR_0	Table Read Protect 00800-03FFF	Disabled
		EBTR_1	Table Read Protect 04000-07FFF	Disabled
		EBTR_2	Table Read Protect 08000-0BFFF	Disabled
30000D	40	EBTRB	Table Read Protect Boot	Disabled

**Configuration im PICKit2 anzeigen**

The screenshot shows the PICkit 2 Programmer interface. The 'PIC18F Configuration' section displays the following settings:

- Device: PIC18F2525
- Configuration: 0700 1F1F 8300 0085
- User IDs: FF FF FF FF FF FF FF FF
- Checksum: 4342
- OSCCAL: C007
- BandGap: E007

The 'Configuration Word Editor' window is open, showing a table of configuration words and their bit values:

Name	Address	Value	Bit Edit															
			15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CONFIG1	300000	0700	0	0	-	-	0	1	1	1	-	-	-	-	-	-	-	-
CONFIG2	300002	1F1F	-	-	1	1	1	1	1	-	-	-	1	1	1	1	1	1
CONFIG3	300004	8300	1	-	-	-	0	1	1	-	-	-	-	-	-	-	-	-
CONFIG4	300006	0085	-	-	-	-	-	-	-	-	1	0	-	-	-	1	-	1
CONFIG5	300008	C007	1	1	-	-	-	-	-	-	-	-	-	-	-	1	1	1
CONFIG6	30000A	E007	1	1	1	-	-	-	-	-	-	-	-	-	-	1	1	1
CONFIG7	30000C	4007	-	1	-	-	-	-	-	-	-	-	-	-	-	1	1	1

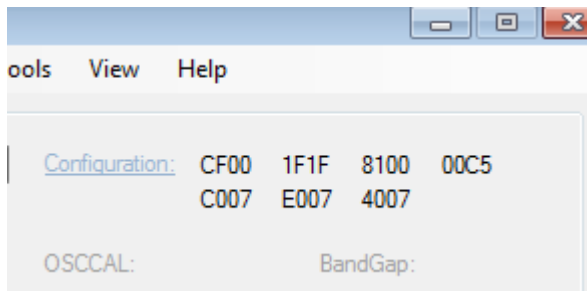
```

;*****
;
; Filename:      Aufgabe13
; Date:
; File Version:  42
;
; Author:       munz
; Company:      Gewerbeschule Loerrach FTE
;*****
;
;LIST P=18F2525, F=INHX32          ;directive to define processor
;#include <P18F2525.INC>          ;processor specific variable definitions
;*****
;Configuration bits                ;s.a.Configure | Configuration Bits
;CONFIGPBADEN = OFF                ;RB4:RB0 können als digitale Ein-Ausgänge
;                                  ;programmiert werden
;                                  ;PORTB A/D Enable - Bit
;                                  ;s. PIC18 Configuration Settings Addendum
;CONFIG      WDT = OFF              ;Watchdog Timer aus
;CONFIG      LVP = ON               ;Low Voltage ICSP (In_Circuit_Serial_Programming
;                                  ;s.a. Sprut. Es gibt auch High Voltage Programming
;                                  ;(HVP) es braucht 12V

;CONFIG      XINST = OFF            ;Extended Instruction Set Aus
;                                  ;das Mapping der ACES Bank ist ein bisschen anders
;CONFIG      OSC = LP               ;LP=Low Power - externer Takt - minimaler Stromverbrauch
;                                  ;s.a. Sprut

;*****
;

```



In diesem Kapitel werden die Möglichkeiten aufgezeigt, einen PIC18-Mikrocontroller zu konfigurieren. **Es ist sehr wichtig den PIC-Mikrocontroller entsprechend zu initialisieren.** Dieses Kapitel erläutert die verschiedenen Möglichkeiten. Es werden auch einige Eigenschaften eines PIC-Mikrocontroller definiert, die vorhanden sind. Die meisten Konfigurationen, die hier erläutert werden, werden beim Programmieren festgelegt und während des Programmablaufs nicht mehr verändert. Diese Konfigurationen werden **vorwiegend in den CPU-Konfigurationsregistern (CONFIGxL oder CONFIGxH) erledigt. Diese Konfigurationsregister können nicht per Software geschrieben werden.**

### Oszillatordoptionen

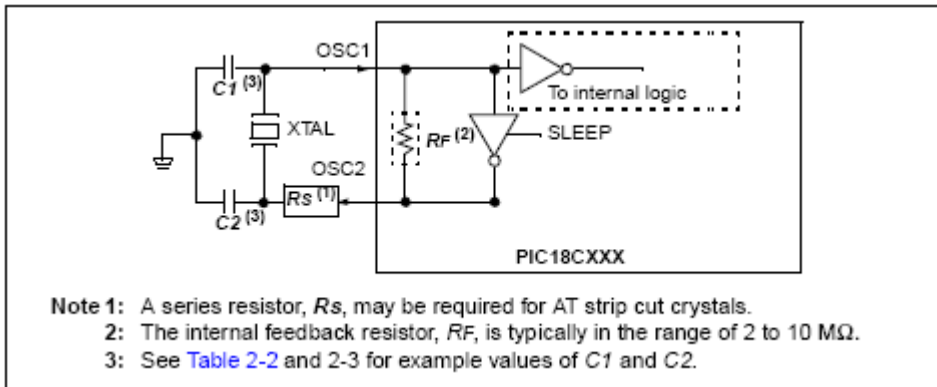
Ein Mikrocontroller braucht immer einen Arbeitstakt und die Auswahl der Art des Arbeitstaktes wird in dem PIC durch die Oszillatordoption definiert. **Am PIC18F252 kann ein externer Oszillator mit max. 40 MHz** angeschlossen werden. Es gibt mehrere Möglichkeiten einen Takt zu erzeugen, da unterschiedliche Oszillatormodi vorhanden sind. Unser PIC18F252 besitzt 8 unterschiedliche Oszillatormodi:

1. LP (Low Power Crystal) 5 - 200 KHz
- 2. XT (Crystal/Resonator) 0,1 MHz - 4 MHz**
3. HS (High-Speed Crystal/Resonator) 4 MHz - 25 MHz
4. HSPLL (High-Speed Crystal/Resonator with PLL enabled) 4 MHz - 10 MHz
5. RC (External Resistor/Capacitor with Fosc/4 output on RA6) max. 4 MHz
6. RCIO (External Resistor/Capacitor with I/O on RA6) max. 4 MHz
7. EC (External Clock with Fosc/4 output) max. 40 MHz
8. ECIO (External Clock with I/O on RA6) max. 40 MHz

### HS, XT oder LP Oszillator Modes

Bei den **Optionen LP, XT oder HS wird ein Quarz oder Keramikresonator an den Pins OSC1 und OSC2 angeschlossen**. Der Hauptunterschied der Modi ist der Frequenzbereich (siehe oben in der Auflistung). Ein Quarz muss an beiden Enden mit einem Kondensator an Masse angeschlossen sein.

Bei externer Taktfrequenz bis 4 MHz sollte man den Oszillatortyp auf XT einstellen, bei höheren Frequenzen HS.

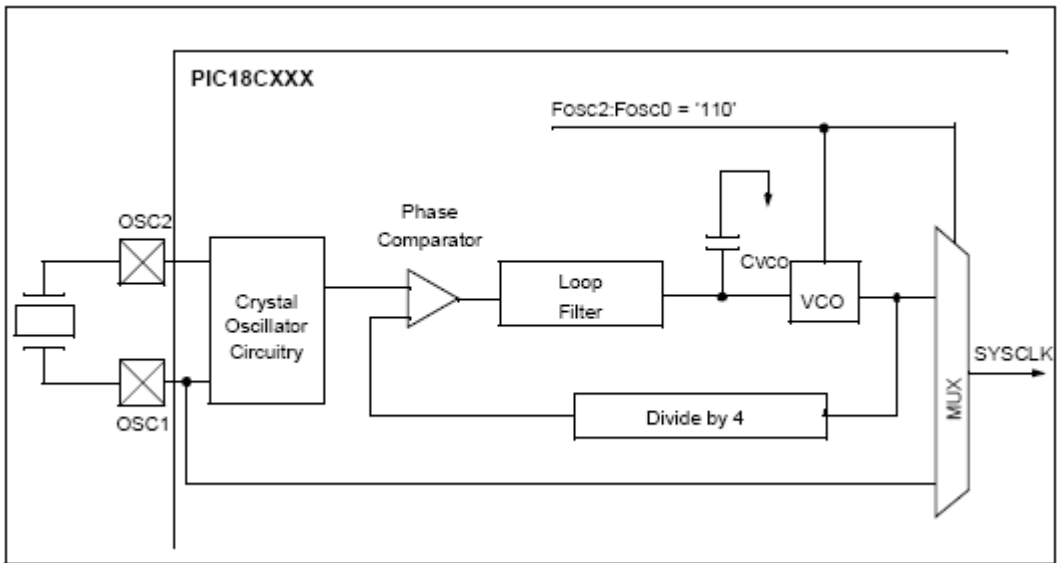


### Oszillator Typ Frequenz Typ. getestete Kondensatorwerte

		C1	C2
XT	455KHz	68-100pF	68-100pF
XT	2MHz	15-68pF	15-68pF
<b>XT</b>	<b>4MHz</b>	<b>15-68pF</b>	<b>15-68pF WIR</b>
HS	8MHz	10-68pF	10-68pF
HS	16MHz	10-22pF	10-22pF

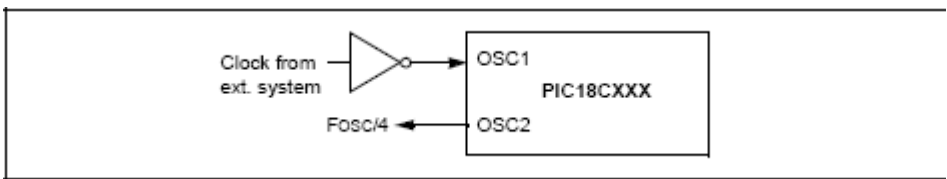
### HSPLL (HS oscillator with 4xPLL enabled)

Im HSPLL wird auch ein Quarz oder Keramikresonator an den Pins OSC1 und OSC2 angeschlossen und der PLL ist aktiviert. Mit dem **Phase Locked Loop (PLL)** im HSPLL-Modus (auch HS4 genannt) kann die Frequenz des ankommenden Quarzoszillatorsignals mit 4 multipliziert werden. Wenn z.B. eine Eingangstaktfrequenz von 10 MHz vorhanden ist, wird die interne Taktfrequenz zu 40 MHz multipliziert. Dieses ist für Leute nützlich, die mit elektromagnetischer Interferenz (EMI) wegen der Hochfrequenzquarze Probleme haben. Ein weiterer Vorteil für diesen Modus ist der Kostenfaktor, um die deutlichen teuren Quarzoszillatoren bei Frequenzen über 25 MHz nicht verwenden zu müssen. Das PLL kann nur ermöglicht werden, wenn die Oszillatorkonfigurationsbits für den Modus HSPLL gesetzt werden (FOSC2:FOSC0 = 110). Wenn irgendein anderer Modus programmiert wird, wird das PLL nicht ermöglicht und der Systemtaktgeber kommt direkt von OSC1.



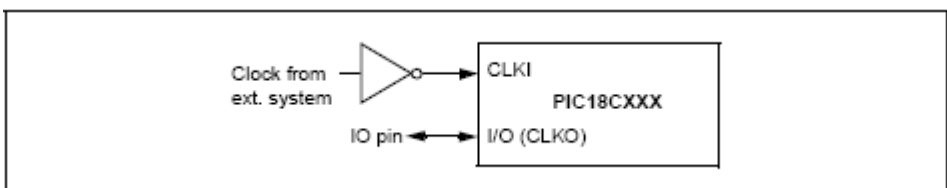
### EC Oszillator Konfiguration

Im EC-Modus wird direkt an den Pin OSC1 (CLKI) ein externes Takt-Signal angeschlossen. Am Pin OSC2 (CLKO) wird die Frequenz des ausgewählten Oszillators, der durch 4 geteilt wird ( $F_{osc}/4$ ), ausgegeben. Dieser Taktgeberausgang ist für Prüfungs- oder Synchronisierungszwecken nützlich.



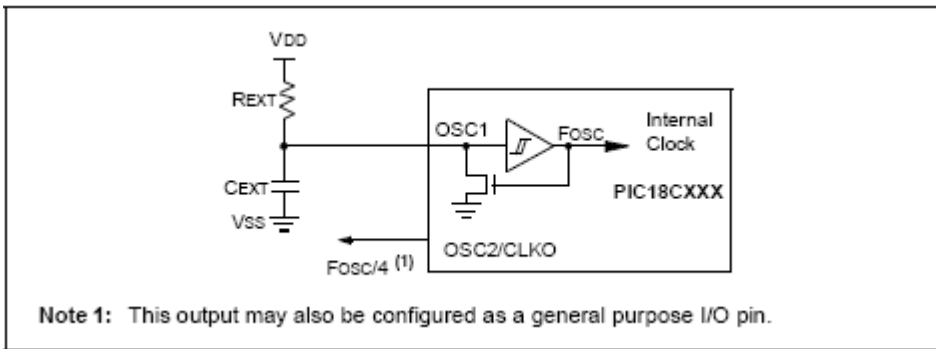
### ECIO Oszillator Konfiguration

In diesem Modus kann ein externes Takt-Signal an Pin OSC1 (CLKI) angeschlossen werden. Der Unterschied zwischen dem EC-Modus ist, das der Pin OSC2 als I/O genutzt werden kann.



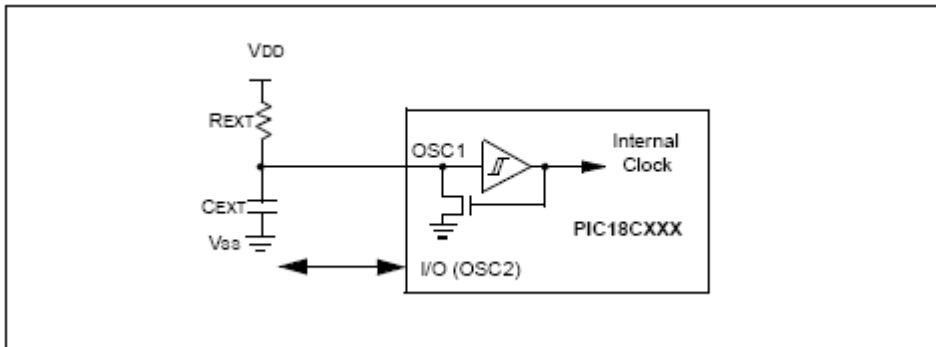
### RC Oszillator Konfiguration

Wenn man keinen stabilen Takt benötigt und Kosten sparen möchte, kann der RC-Modus benutzt werden. Es wird ein Kondensator über einen Widerstand aufgeladen. Man muss aber die Toleranz des extern benutzten Widerstand ( $R_{ext}$ ) und Kondensator ( $C_{ext}$ ) beachten. Der externe Widerstand  $R_{ext}$  darf den Wert von 2,2 kOhm nicht unterschreiten, da sonst der Oszillatorbetrieb instabil wird oder komplett aufhört zu schwingen. Für sehr hohe  $R_{ext}$ -Werte (z.B. 1 MOhm) wird der Oszillator für Geräusche und Feuchtigkeit empfindlich. Zu empfehlen sind Werte für  $R_{ext}$  zwischen 3 kOhm und 100 kOhm. Die Schaltung funktioniert auch ohne den externen Kondensator  $C_{ext}$ . Es sind aber für den externen Kondensator Werte über 20 pF aus Stabilitätsgründen zu verwenden. Am Pin OSC2 (CLKO) wird die Frequenz des ausgewählten Oszillators, der durch 4 geteilt wird ( $F_{osc}/4$ ), ausgegeben. Dieser Taktgeberausgang ist für Prüfungs- oder Synchronisierungszwecke nützlich.



### RCIO Oszillator Konfiguration

Der RCIO-Oszillatormodus arbeitet genauso wie die RC Oszillator Konfiguration. Der einzige Unterschied ist, dass der Pin OSC2 als I/O genutzt werden kann.



Umso höher die Taktgeberfrequenz, desto größer ist auch der Stromverbrauch. Der Stromverbrauch ist selbstverständlich auch von der Temperatur abhängig. In der folgenden Tabelle sind fast alle Daten bei einer Temperatur von 25°C angegeben.

### PIC-Typ typ. max Vdd Oszillatortyp Frequenz

PIC18LFXX2	0.5 mA	1 mA	2 V	XT	4 MHz
PIC18LFXX2	0.3 mA	1 mA	2 V	RC	4 MHz
PIC18LFXX2	0.3 mA	1 mA	2 V	RCIO	4 MHz
<b>PIC18FXX2</b>	<b>1.2 mA</b>	<b>1.5 mA</b>	<b>4.2 V</b>	<b>XT</b>	<b>4 MHz</b>
PIC18FXX2	1.5 mA	3 mA	4.2 V	RC	4 MHz
PIC18FXX2	0.75 mA	2 mA	4.2 V	RCIO	4 MHz
PIC18LFXX2	14 µA	30 µA	2 V	LP	32 KHz
PIC18LFXX2	40 µA	70 µA	4.2 V	LP	32 KHz
PIC18LFXX2	10 mA	25 mA	4.2 V	EC,ECIO	-

PIC18FXX2	10 mA	25 mA	4.2 V	EC,ECIO	-
PIC18LFXX2	0.6 mA	2 mA	2 V	HS	4 MHz
PIC18LFXX2	10 mA	15 mA	5.5 V	HS	25 MHz
PIC18LFXX2	15 mA	25 mA	5.5 V	HS+PLL	10 MHz
PIC18FXX2	10 mA	15 mA	5.5 V	HS	25 MHz
PIC18FXX2	15 mA	25 mA	5.5 V	HS+PLL	10 MHz
PIC18LFXX2	15 µA	55 µA	2 V	Timer1osc	32 KHz
PIC18LFXX2	-	200 µA	4.2 V	Timer1osc	32 KHz

### Der interne Oszillator

Hier werden die Vorteile und Möglichkeiten eines PICs mit **nanoWatt**-Technologie und internen Oszillator beschrieben.

Besonders die neuen PICs (PIC18F2420/2520/4420/4520) besitzen noch weitere Features und die folgenden Daten beziehen sich auf den PIC18F2520. Die neuen PICs mit nanoWatt-Technologie besitzen ein integriertes internes Oszillatormodul, welches zwei verschiedene Clocksignale generiert und somit zwei interne Oszillatoren besitzt. Der erste interne Oszillator (**INTOSC**) ist ein 8 MHz Oszillator, der einen Frequenzteiler an seinem Ausgang beschaltet hat, dadurch sind Frequenzen von 125 KHz bis 4 MHz auswählbar. Er kann auch direkt ohne Frequenzteiler mit einer Frequenz von 8 MHz nutzen. Die andere Taktgeberquelle ist der interne RC-Oszillator (**INTRC**), der nur eine Frequenz von 31 KHz liefert. Der INTRC-Oszillator läuft unabhängig vom INTOSC-Oszillator. Der INTRC kann entweder die Rolle als primärer Oszillator spielen oder nur als zusätzlicher Oszillator dienen. Der interne RC-Oszillator wird automatisch bei folgenden Möglichkeiten eingeschaltet:

- Power-up Timer
- Fail-Safe Clock Monitor
- Watchdog Timer
- Two-Speed Start-up

Wenn die Taktgeberquelle während des Programmes auf dem INTRC-Oszillator geändert wird, benötigt man eine Zeit von ca. 8 \* 32 µs = 256 µs, bis diese stabilisiert läuft. Bei der Wahl des INTOSC-Oszillator wird ein Zeit von ca. 1 ms benötigt, bis diese Taktgeberquelle stabil funktioniert. Das Programm wird trotzdem währenddessen weiter durchgeführt.

Der interne Oszillator kann auch mit PLL eingesetzt werden und kann max. eine Frequenz bis zu 32 MHz erzeugen. Dies geschieht, in dem das PLEN-Bit im OSCTUNE-Register gesetzt wird, um den PLL für den INTOSC zu aktivieren. Jedoch funktioniert dies nur bei einer internen Oszillatorfrequenz von 4 MHz oder 8 MHz. Anders als HSPLL-Modus ist das PLL durch Software kontrolliert.

Der weiterer Unterschied zwischen INTRC und INTOSC ist die Stabilität der Frequenz. Der INTRC-Mode ist nicht sehr stabil z.B. kann die Frequenz bei eingestelltem Mode 31 KHz zwischen 26,562 - 35,938 KHz schwanken. Im INTOSC-Mode beträgt die typische Frequenztoleranz +/- 1% bei einer Betriebstemperatur von 25°C. Bei niedrigster oder höchster ausgewählter Frequenz kann die Toleranz bei -2% oder +2% liegen.

Im integrierten internen Oszillatormodul können durch einen Vorteiler Frequenzen von 31KHz bis 4MHz durch das OSC TUNE-Register gewählt werden. Wird der interne Oszillator benutzt und der INTIO1 Modus initialisiert, dann wird die Taktfrequenz ( $F_{osc}/4$ ) an den OSC2-Pin ausgegeben und der OSC1-Pin (RA7) kann als digitaler Ein- oder Ausgang konfiguriert werden. Im INTIO2-Modus können die Pins OSC2 (RA6) und OSC1 (RA7) als digitale Ein- und Ausgänge benutzt werden.

## Power Management Mode

Die PICs mit nanoWatt-Technologie besitzen anstatt 2 Betriebsarten (SLEEP, Normalbetrieb) einen weiteren neuen, den IDLE-Modus. In diesem Modus wird nicht die CPU getaktet, sondern die Peripherie. Kombinationen zwischen den Betriebsarten und Oszillatorkonfigurationen bezeichnet man als Power Management Modes. Obwohl unser PIC18F252 diesen Modus nicht besitzt, möchte ich diesen Modus trotzdem erläutern.

Die PIC-Serie PIC18F2420/2520/4420/4520 besitzt sieben Operationsmodi für ein besseres und effizienter Power Management. Diese sind in drei Power Management Betriebsarten unterteilt:

- Run Modes
- Idle Modes
- Sleep Mode

Modus	OSCON Bits		getakte Module		vorhandene Taktgeberquelle
	IDLEN	SCS1:SCS0	CPU	Peripherie	
SLEEP	0	N/A	aus	aus	Keine - Alle Taktgeberquellen sind ausgeschaltet
PRI_RUN	N/A	00	getaktet	getaktet	primärer Takt - LP, XT, HS, HSPLL, RC, EC und interner Oszillatorblock (Das ist der normaler Arbeitsmodus)
SEC_RUN	N/A	01	getaktet	getaktet	sekundärer Takt - Timer 1 Oszillator
RC_RUN	N/A	01	getaktet	getaktet	interner Oszillatorblock
PRI_IDLE	1	00	aus	getaktet	primärer Takt - LP, XT, HS, HSPLL, RC, EC
SEC_IDLE	1	01	aus	getaktet	sekundärer Takt - Timer 1 Oszillator
RC_IDLE	1	1x	aus	getaktet	interner Oszillatorblock

Im Idle-Modus wird die CPU ausgeschaltet und die Peripherie ist eingeschaltet. Der Strom liegt im typischen Fall bei 5,8  $\mu$ A. Im Sleep-Modus werden die CPU und die meisten Hardwaremodule ausgeschaltet und der Strom liegt im typischen Fall bei 0,1  $\mu$ A. Nach jedem Reset befindet sich der PIC im RUN-Modus (PRI-RUN-Modus) und wird vom primären Oszillator getaktet.

Es kann auch während des Programmes zwischen zwei Taktgeberquellen gewechselt werden und man benötigt beim Wechsel 2 Befehlszyklen für die alte Taktgeberquelle und 3-4 Befehlszyklen für die neue. Nach dieser Zeit ist die neue Taktgeberquelle stabil.

Es gibt 3 mögliche Taktgeberquellen für diese PICs

- Primärer Oszillator
- Sekundärer Oszillator
- Internes Oszillatormodul

Diese Möglichkeiten werden mit den Bits SCS1:SCS0 im OSCON-Register initialisiert.

Die Bits SCS1:SCS0 sind immer nach jedem Reset auf 00 gesetzt.

OSCON-Register beim PIC18F2520:

**Bit7 Bit6 Bit5 Bit4 Bit3 Bit2 Bit1 Bit0**

IDLEN IRCF2 IRCF1 IRCF0 OSTs IOFS SCS1 SCS0

Durch die Bits IRCF2:IRCF0 kann die interne Oszillatorfrequenz gewählt werden, um damit den PIC zu betreiben.

Nach einem Reset sind die IRCF2:IRCF0 - Bits im

Default-Zustand auf 1 MHz (100) konfiguriert.

Mit den folgenden Bits können die aktuelle Taktgeberquelle und ihr Status überprüft werden:

- OSTs (OSCON-Register)
- IOFS (OSCON-Register)
- T1RUN (T1CON-Register)

Allgemein wird nur einer dieser Bits bei der Auswahl des Power Management Mode vom PIC gesetzt. Wenn das OSTs-Bit gesetzt ist, wird der primäre Taktgeber ausgewählt. Ist das IOFS-Bit gesetzt, wurde der INTOSC-Ausgang für die Taktgeberquelle gewählt. Wenn das T1RUN-Bit gesetzt ist, wurde der Timer1 Oszillator gewählt. Sollte keiner der Bits gesetzt sein, wurde die INTRC Taktgeberquelle gewählt oder die INTOSC Taktgeberquelle ist momentan instabil.

Das OSTs-Bit wird vom PIC gesetzt, sobald die Zeit des Oszillator Start-up Timer abgelaufen ist und das IOFS-Bit wird vom internen Oszillatormodul gesetzt, wenn die interne Frequenz stabil ist. Sollte die Frequenz im internen Oszillator auf 31 KHz eingestellt sein, muss noch das INTSRC-Bit im OSCTUNE-Register initialisiert werden.

### **IDLE-Modus**

Mit den Bits IDLEN, SCS1 und SCS0 wird im RUN-Modus

der Power-managed Mode ausgewählt und der SLEEP-Befehl startet

nicht nur den SLEEP-Modus, sondern alle anderen Power-managed Modes.

Die Bits IDLEN, SCS1 und SCS0 im OSCON-Register bestimmen, welcher der Power-managed Modes gestartet wird. Nach jedem Reset sind diese 3 Bits gelöscht, und der SLEEP-Befehl führt den klassischen SLEEP-Zustand aus. Damit wird der SLEEP-Befehl kompatibel mit den PICs ohne Power-Management. Der SLEEP-Befehl führt einen der 3 IDLE-Modi aus, sobald das IDLEN-Bit

gesetzt ist. Die 3 unterschiedlichen IDLE-Modi werden mit den SCS1 und SCS0 Bits gewählt. Beim Aufwachen aus dem IDLE-Modus ist ein Takt zum Zeitpunkt des Weckens vorhanden.

### **RUN-Modus**

Aus einem Power-managed RUN-Mode gelangt man entweder durch ein Reset oder ein Interrupt wieder in den PRI\_RUN-Modus zurück. Wird das T1OSCEN-Bit gelöscht während sich der PIC im SEC\_RUN-Modus befindet, dann kann ein Zurückschalten auf den PRI\_RUN-Modus. Der externe Timer 1 Oszillator läuft trotzdem noch weiter, bis der primäre Takt zur Verfügung steht. Danach wird der Timer 1 Oszillator abgeschaltet und T1RUN-Bit im T1CON-Register wird gelöscht. Beim Übergang aus dem SEC\_RUN- oder RC\_RUN-Modus zurück in den PRI\_RUN-Modus muss die CPU nicht aufwachen.

### **SLEEP-Modus**

Der SLEEP-Modus ist der klassische SLEEP-Zustand. Der primäre Takt muss zuerst aufgebaut werden, wenn der PIC aus dem SLEEP-Modus aufgeweckt wird.

Wenn der Watchdogtimer (WDT) aktiviert ist und sich der PIC im SLEEP-Modus befindet, laufen die INTRC-Taktgeberquelle und der Timer1 Oszillator weiter. Der Watchdog Timer löst im SLEEP- und IDLE-Modus keinen Reset aus, im Gegensatz zu dem RUN-Modus.

### **RESET**

Bei einem Reset startet das komplette Programm erneut und nach einem Reset oder nach dem Einschalten sind die TRIS-Register auf 0xFF gesetzt, womit alle Port-Pins zunächst Eingänge sind. In der PIC18-Serie gibt es mehrere Möglichkeiten ein RESET auszulösen:

- Power-on RESET (POR),
- MCLR Reset während beim Normalbetrieb,
- Watchdogtimer (WDT) Reset während einer Befehlsausführung,
- Programmierbarer Brown-out Reset (BOR),
- RESET - Anweisung,
- Stack Full Reset oder
- Stack Underflow Reset.

RCON (Reset Control Register) beim PIC18F252:

**Bit7 Bit6 Bit5 Bit4 Bit3 Bit2 Bit1 Bit0**

IPEN — — /RI /TO /PD /POR /BOR

Nach dem Programmstart durch ein Reset wird die Reset-Ursache durch die Bits im RCON-Register angegeben. Dies ist sehr wichtig, um im Programm überprüfen zu können, weshalb ein Reset ausgeführt wurde. Wenn ein Brown-out Ereignis (Spannungseinbruch) erfolgt ist, dann wird das /BOR-Bit auf 0 gesetzt, und im Normalbetrieb ist das Bit auf 1 gesetzt. Das /POR-Bit wird zurückgesetzt, wenn ein Power-on Reset statt fand. Das /TO-Bit wird vom PIC gelöscht, wenn der Watchdog Timer ein Reset verursacht wurde. Dieses Bit wird nach dem Power-up Timer oder bei einer CLRWDT- und SLEEP-Anweisung gesetzt. Das /RI-Bit wird gelöscht, sobald ein Reset verursacht wurde. Die Bits /RI, /POR und /BOR müssen per Software gesetzt werden.

Das RCON-Register existiert in der PIC16-Familie nicht, aber die Funktionen Brown-out und Power-on Reset befinden sich im PCON-Register.

Bei einigen PICs wird der Reset-Pin (MCLR) nicht mehr benötigt, dieser Pin kann dann als I/O definiert werden.

## Brownout

**In Anlehnung an Blackout (= Stromausfall) ist ein Brownout eine zu geringe Spannung.**

Jeder Mikrocontroller hat einen angegebenen Versorgungsspannungsbereich, z.B. 4,5 - 5,5V. Unterschreitet man diesen, dann hört er aber nicht einfach auf zu funktionieren, sondern nach und nach fallen einzelne Komponenten aus oder werden unzuverlässig. Die CPU "spielt verrückt".

Die üblichen Ursachen für einen Brownout sind leer werdende Batterien und der normale Ausschaltvorgang, wenn in der Stromversorgung ein größerer Kondensator ist (was bei Netzteilen normalerweise der Fall ist). Anders ausgedrückt: Ein Brownout ist kein exotischer Sonderfall, sondern man muss immer mit ihm rechnen.

Ob man etwas gegen einen Brownout tun muss, hängt von der Anwendung ab. Zeigt ein elektronisches Thermometer beim Ausschalten noch kurz wirre Zeichen an, dann ist das ziemlich egal; macht eine elektronische Steuerung aber dabei z.B. ein Ventil auf, dann könnte das unangenehme bis fatale Folgen haben.

Zur Lösung des Problems kann man über eine externe Schaltung am Prozessor das Reset-Signal aktivieren (solange läuft der Prozessor nämlich nicht). Dafür gibts auch fertige ICs, sogenannte Reset-Controller.

Immer wieder gerne genommene **Reset-Controller**:

- TL7705 ([Ti](#))
- MCP100 ([Microchip](#))
- MAX809 im sehr kleinen SC70 oder SOT23 Gehäuse zu haben ([Maxim](#))
- MAX6864 mit [Watchdog](#)

Neuere Mikrocontroller haben oft auch schon eine Brownout-Detection eingebaut, man muss sie dann nur noch aktivieren. Bedenken sollte man dabei, dass so eine Schaltung zusätzlich Strom benötigt. Bei den AVR-Prozessoren sind das 10-30  $\mu$ A, was bei langlaufenden Batteriegeräten nicht mehr zu vernachlässigen ist (ca. 300 mAh pro Jahr).

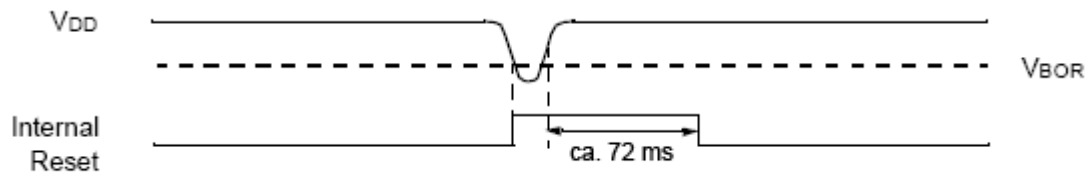
## Brown-out Reset

Ein kurzer Aussetzer in der Betriebsspannung des PIC kann zum Absturz oder zum Weiterarbeiten mit verfälschten Werten führen. Die Brown-out Reset-Option überwacht die Betriebsspannung. Falls die Betriebsspannung für 200  $\mu$ s oder länger (im Datenblatt nach TBOR Zeit nachschauen) unter eine bestimmte eingestellte Spannung (VBOR) fällt, wird sofort ein Reset ausgelöst. Sollte die Betriebsspannung für mehr als 72ms wieder im sicheren Bereich liegen, startet der PIC neu. Um das Brown-out Reset zu aktivieren, muss das Bit BOREN im Konfigurationswort (CONFIG2L-Register) gesetzt werden. Den Spannungsbereich kann man seit der PIC18-Familie selbst bestimmen, in dem man die Bits BORV1 und BORV0 entsprechend einstellt. Diese Bits befinden sich im Konfigurationswort (CONFIG2L-Register).

**PIC-Typ    BORV1:BORV0 Spannungsbereich**

	min	max
PIC18LFXX2 11	1.98 V	2.14 V
PIC18LFXX2 10	2.67 V	2.89 V
PIC18LFXX2 01	4.16 V	4.5 V
PIC18LFXX2 00	4.45 V	4.83 V
PIC18FXX2 1x	—	—
PIC18FXX2 01	4.16 V	4.5 V
PIC18FXX2 00	4.45 V	4.83 V

Wenn das Brown-out Reset aktiviert ist, wird auch automatisch der Power-up Timer aktiviert.



**Stack Full und Stack Underflow Reset**

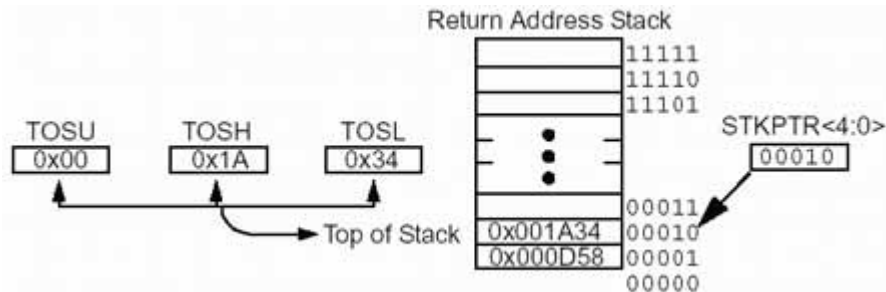
Diese Art von Reset existiert erst seit der PIC18-Familie. Für ein Reset bei einem Stack Overflow oder Stack Underflow muss das STVREN-Bit im Konfigurationsregister CONFIG4L gesetzt werden. Durch das Setzen des STVREN-Bit werden beide Stack Reset Möglichkeiten aktiviert. Bei einem Stack Overflow (Stack ist voll oder ein Überlauf) wird das STKFUL-Bit im STKPTR-Register gesetzt und bei einem Stack Underflow wird das STKUNF-Bit gesetzt. Ein Stack Underflow geschieht, wenn man z.B. versucht eine Adresse vom Stack zu holen, obwohl dieser leer ist. Nachdem eines dieser Bits vom PIC gesetzt wurde, kann auch optional ein Reset ausgelöst werden. Diese beiden Bits werden nach einem Reset nicht gelöscht, damit man nach dem Reset überprüfen kann, wodurch dieser verursacht wurde. Die STKFUL- und STKUNF-Bits müssen danach per Software gelöscht werden, sie können aber selbstverständlich nicht per Software gesetzt werden.

STKPTR für PIC18:

**Bit7    Bit6    Bit5 Bit4 Bit3 Bit2 Bit1 Bit0**

STKFUL STKUNF —    SP4 SP3 SP2 SP1 SP0

Das STKPTR-Register gibt den Status des Stacks aus. Der Stackpointers befindet sich in den Bits SP0 bis SP4. Dadurch weiß man anschließend, bei welcher Adresse der Stack Overflow oder Underflow verursacht wurde und diese Adresse wird in die Bits SP0 bis SP4 gespeichert. Der Stackpointer hat nach jedem Reset den Wert 0, sodass damit gezeigt wird, dass der Stack leer ist.



### Power-on Reset

Da bei der Betriebsspannung für den PIC-Mikrocontroller eine kleine Verzögerung existiert bis die benötigte Spannung wirklich anliegt, schaltet sich der Power-on Reset (POR) bei einer Spannung von 0,7 V an. Danach wird der Power-up Timer gestartet (wenn er aktiv ist), der Oszillator Power-up Timer. Damit ist der komplette Reset beendet und der PIC startet sein Programm.

### Watchdogtimer

Es kann passieren, dass der PIC abstürzt. Damit das ohne schlimme Folgen bleibt, löst der Watchdogtimer (WDT) in so einem Fall automatisch ein Reset des PIC aus. Der Watchdog ist ein kleiner Timer im PIC, der einen internen RC-Taktgeber besitzt, und somit von der Funktion des PIC unabhängig ist. Der WDT löst, wenn er eingeschaltet ist, in einer eingestellten Zeit ein Reset des PIC aus. Der Reset kann nur verhindert werden, wenn das im PIC laufende Programm den WDT immer wieder zurücksetzt, bevor er einen Reset auslösen kann. Dieses Zurücksetzen erfolgt mit dem Befehl clrwtd. Der Watchdogtimer kann aber weder gelesen noch geschrieben werden. Er kann nur zurück gesetzt werden. Das ordnungsgemäß laufende Programm muss also so programmiert werden, dass in kurzen Abständen (z.B. alle 5 ms ist aber abhängig von der eingestellten Zeit des WDT) der Befehl clrwtd im Programm eingefügt wird um nicht ein Reset auszulösen. Stürzt das Programm aber ab, wird der WDT nicht mehr durch das Zurücksetzen des Timers gelöscht und der PIC startet neu.

Bei den PIC18-Mikrocontrollern teilt sich der WDT die Konfiguration nicht mehr mit dem Timer0. Der WDT hat jetzt ein Vorteiler, der bis zu 1:32,768 gewählt werden kann (bei den PICs mit einem Extended Watchdog Timer). Der PIC18F252 hat ein Vorteiler bis zu 1:128. Der Watchdogtimer wird von dem internen Oszillator INTRC betrieben. Die neuen PICs (PIC18F2420/2520/4420/4520) besitzen einen erweiterten Watchdog Timer (Extended Watchdog Timer) und dieser besitzt ein Vorteiler, der ein erweiterten Timer-out Bereich von 4ms bis 131s besitzt. Die Selektion des Vorteiler geschieht in dem Register CONFIG2H, sowie die Aktivierung des WDT mit dem WDTEN-Bit. Wird der Watchdog Timer nicht durch das WDTEN-Bit aktiviert, so kann er immer noch per Software mit dem SWDTEN-Bit (Bit 0) im WDTCON-Register eingestellt werden.

Ohne besondere Notwendigkeit sollte man den WDT nicht verwenden, da er die Programmentwicklung kompliziert. Denn der Watchdogtimer dient vorwiegend zur Programmsicherheit und verhindert evtl. Programmabstürze. Der Watchdogtimer kann auch für das regelmäßige Aufwecken aus dem SLEEP-Modus genutzt werden. Dadurch kann der WDT zur Verringerung der Stromaufnahme verwendet werden.

## Power-up Delays

Es gibt zwei Möglichkeiten um das Starten des Programmes im PIC beim Einschalten zu verzögern. Da mehrere externe elektronische Schaltungen am PIC angeschlossen sind, kann es passieren das mehrere Schaltungen eine gewisse Zeit benötigen um aktiviert zu werden. Deshalb gibt es folgende Möglichkeiten um das Einschalten des PIC zu verzögern.

### Power-up Timer (PWRT)

Normalerweise beginnt der PIC beim Zuschalten der Betriebsspannung sofort mit der Abarbeitung seines Programmes. Manchmal benötigen andere Teile der elektronischen Schaltung, in die der PIC eingebaut ist, aber etwas Zeit, um ihren Anfangszustand einzunehmen. Deshalb kann der PIC auf Wunsch seine Arbeit mit einer festen Verzögerung von ca. 72ms nach dem Einschalten der Betriebsspannung aufnehmen. In dem diese Option aktiviert ist, wird sehr oft die Stabilität des Systems erhöht. Diese Option wird mit dem PWRTEN-Bit im Konfigurationswort (Register CONFIG2L) aktiviert.

### Oszillator Start-up Timer (OST)

Durch diese Option beginnt der PIC sein Programm erst, wenn der Oszillator stabil läuft. Der Oszillator Start-up Timer startet erst nach dem der Power-up Timer (nur wenn dieser aktiviert ist ) beendet ist, und durchläuft eine Zeit von 1024 Oszillator Zyklen ( $TOST = 1024 TOSC$ ). Wenn diese Zeit erreicht ist, wird der Oszillator Start-up Timer (OST) beendet, der PIC kommt aus dem Reset-Zustand und beginnt sein Programm. Der OST funktioniert und ist automatisch aktiv in den Oszillatormodis LP, HS und XT.

### Low Voltage Programming (LVP)

Die PIC18-Familie und viele PIC16 Controller können nicht nur mit 12 V gebrannt werden, sondern lassen sich alternativ auch mit nur 5 V programmieren. Beim Einsatz der 5V-Programmierung wird allerdings ein bestimmter Pin für die Programmierfunktion blockiert, steht dann nicht mehr für andere I/O-Funktion zur Verfügung. Die entsprechenden Pins sind je nach PIC unterschiedlich, schauen Sie bitte im jeweiligen Datenblatt nach den richtigen Pins. In den PIC18F-Controllern wird durch das LVP-Bit im Konfigurationswort (CONFIG4L-Register) das LVP aktiviert.

### Two-Speed-Start-up Modus

Dieser Modus befindet sich nur in den neuen PIC18F-Mikrocontrollern (z.B. PIC18F2520) und bei einigen neuen PIC16F-Controllern (z.B. PIC16F7x7) mit nanoWatt-Technologie. Mit dem Two-Speed-Start-up Modus wird das Aufwecken oder Starten des PICs beschleunigt, weil der PIC normalerweise bei einem Aufwecken mehrere Millisekunden benötigt, damit der externe Oszillator anschwingen kann. Zuerst benutzt der PIC in diesem Modus einfach den internen Oszillator und fängt damit schon mal an zu arbeiten. Nach der üblichen Verzögerungszeit schaltet er dann auf den externen Takt um. Damit das funktioniert, muss im Konfigurationswort (CONFIG1H-Register) das Bit IESO auf 1 gesetzt sein und SCS auf 0 stehen. Außerdem muss natürlich einer der Modes LP, XT oder HS ausgewählt sein.

### SLEEP

Die PIC-Mikrocontroller lassen sich mit dem SLEEP-Befehl in einen Energiesparmodus versetzen. Der Prozessor bricht dann die Befehlsverarbeitung ab, schaltet alle I/O-Pins in den Tristate-Zustand, der Oszillator steht still und der PIC wartet. Aus diesem SLEEP-Modus kann der PIC nur durch drei Dinge wieder heraustreten:

- durch einen externen Reset

- durch einen internen Reset des WDT oder
- durch einen Interrupt

Um den PIC durch ein Interrupt-Ereignis aus dem SLEEP-Modus aufzuwecken, muss das entsprechende Interrupt vor der SLEEP-Anweisung aktiviert sein.

Der PIC braucht aus dem SLEEP-Modus dann eine gewisse Zeit um normal weiter zu arbeiten. Dazu wird die Zeit von 1024 Tosc benötigt, das entspricht 256 Befehlen eines PICs. Das heißt, wenn der PIC mit einem Quarz von 4 MHz arbeitet (1  $\mu$ s Zeit für einen Befehl), benötigt er dann ca. 256  $\mu$ s um wieder in den Normalbetrieb zu kommen.

### **Fail-Safe Taktüberwachung (Fail-Safe Clock Monitor)**

Diese Eigenschaften findet man in den neuen PIC18F-Mikrocontrollern mit nanoWatt-Technologie. Mit dieser Option wird ständig die Haupttaktgeberquelle überwacht. Wenn eine Taktstörung auftritt, schaltet der Controller zum internen Oszillatorblock und läßt anhaltenden langsamen Betrieb oder eine sichere Anwendungsabschaltung zu. Durch das Bit FCMEM im Konfigurationswort (CONFIG1H-Register) wird die Fail-Safe Taktüberwachung ein- oder ausgeschaltet.