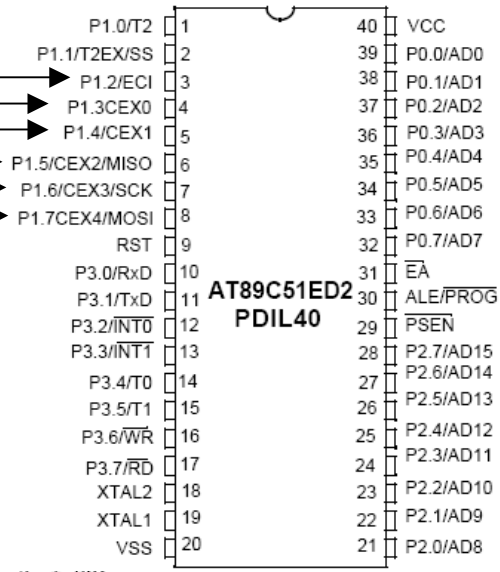
Friedrich-Ebert-Schule Esslingen	MIKROCONTROLLER	Name:
4.5.1	Programmable Counter Array (PCA)	Datum:

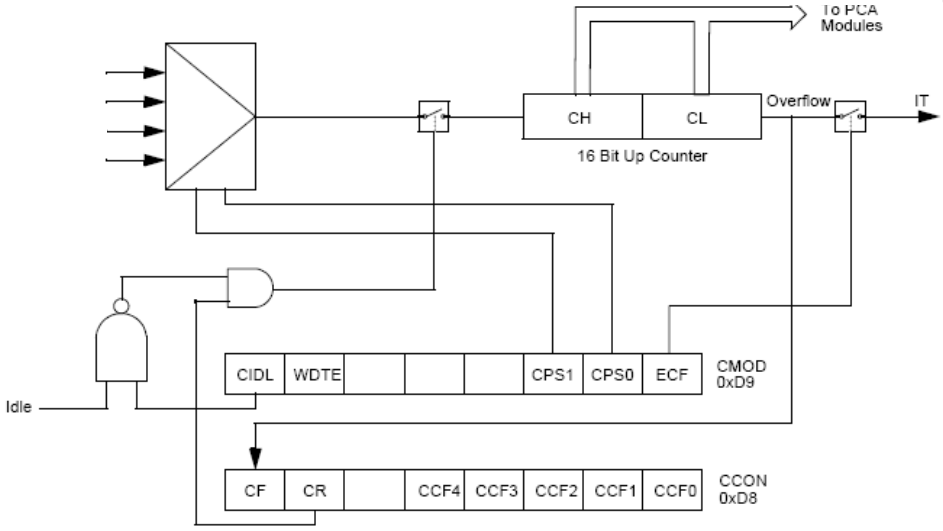
Das Programmable Counter Array besteht aus fünf PCA-Modulen, die einen gemeinsamen 16-Bit Timer/Counter als Zeitbasis verwenden.

ECI: Externer Clock-Eingang für PCA-Counter
CEX0: I/O-Pin PCA-Modul 0
CEX1: I/O-Pin PCA-Modul 1
CEX2: I/O-Pin PCA-Modul 2
CEX3: I/O-Pin PCA-Modul 3
CEX4: I/O-Pin PCA-Modul 4



16-Bit PCA Timer/Counter

Der PCA-Zähler zählt die Impulse von einer der vier Quellen: 1 MHz / 3 MHz / Timer 0 Überlauf / P1.2
Bei Überlauf wird das Überlauf-Flag gesetzt und/oder ein PCA-Interrupt ausgelöst.
Die Konfiguration erfolgt über die Register **CMOD** und **CCON**.



7	6	5	4	3	2	1	0	**CMOD** PCA Counter Mode Register (D9h)
CIDL	**WDTE**	**-**	**-**	**-**	**CPS1**	**CPS0**	**ECF**	Resetwert: **00XX X000**b
		Reserve (nicht benutzen)					**Enable Counter Overflow Interrupt** 1: CF = 1 bewirkt einen PCA-Interrupt. Einsprungadresse: 0033h 0: Es wird kein PCA-Interrupt erzeugt!	
					PCA Count Pulse Select Bits Auswahl des Takt-Eingangs für den PCA-Counter			
					0	0	Interner Takt (fosc / 12) = 1 MHz	
					0	1	Interner Takt (fosc / 4) = 3 MHz	
					1	0	Timer 0 - Überlauf	
					1	1	Externer Takt an Pin 1.2 (ECI) (maximal: fosc/8 = 1,5 MHz)	
	Watchdog Timer Enable 1: Watchdog Timer für PCA-Modul 4 aktiviert! 0: Watchdog Timer **nicht** aktiviert!							
Counter IDLE Control 1: PCA Counter im IDLE-Modus **nicht** aktiv! 0: PCA Counter im IDLE-Modus aktiv!								

7	6	5	4	3	2	1	0	**CCON** PCA Counter Control Register (D8h)
CF	**CR**	**-**	**CCF4**	**CCF3**	**CCF2**	**CCF1**	**CCF0**	Resetwert: **00X0 0000**b (bitadressierbar)
			PCA Modul Interrupt-Flags Werden bei Match oder Capture gesetzt. Müssen per Software zurückgesetzt werden!					
		Reserve						
	PCA Counter Run control bit 1: PCA Counter läuft! 0: PCA Counter läuft nicht!							
PCA Counter Overflow-Flag Wird bei Zähler Überlauf gesetzt. Muss per Software zurückgesetzt werden!								

PCA-Module

Jedes PCA-Modul verfügt über **zwei** 8-Bit Compare/Capture Register. Damit lassen sich 2 grundsätzliche Betriebszustände eines PCA-Moduls erreichen:

CCAPnH	CCAPnL

n = 0 ...4

- Im **Capture-Mode** wird der Zählwert des PCA-Counters bei einem Ereignis am externen Pin **CEXn** „eingefangen“ und in CCAPnH und CCAPnL gespeichert.

- Im **Compare-Mode** wird der Inhalt von CCAPnH und CCAPnL mit dem Zählwert des PCA-Counters verglichen. Bei Gleichheit (Match) wird ein bestimmtes Ereignis ausgelöst.

Insgesamt sind je PCA Module 7 verschiedenen Betriebsmodi möglich. Jedes der 5 Module besitzt zur Konfiguration der Modi ein eigenes Register **CCAPMn**:

7	6	5	4	3	2	1	0	**CCAPMn** PCA Module Modes Registers Modul 0: DAh – Modul 4: DEh
-	ECOMn	CAPPn	CAPNn	MATn	TOGn	PWMn	ECCFn	Resetwert: **0000 0000**b
0	0	0	0	0	0	0	0	PCA-Modul n **nicht** aktiv
0	X	1	0	0	0	0	X	16 Bit Capture bei positiver Flanke an CEXn-Pin
0	X	0	1	0	0	0	X	16 Bit Capture bei negativer Flanke an CEXn-Pin
0	X	1	1	0	0	0	X	16 Bit Capture bei pos/neg Flanke an CEXn-Pin
0	1	0	0	1	0	0	X	16 Bit Software Timer/Compare mode
0	1	0	0	1	1	0	X	16 Bit High Speed Output
0	1	0	0	0	0	1	0	**8 Bit PWM**
0	1	0	0	1	X	0	X	Watchdog-Timer (Nur Modul 4)

Reserve

Enable CCF Interrupt
1: Gibt die PCA-Modul-Interrupts frei. PCA-Interrupts werden dann über CCFn-Flag in CCON ausgelöst.

Pulse Width Modulation Mode
1: Pin CEXn als PWM-Ausgang

Toggle
1: Pin CEXn wird bei Erreichen des Compare-Wertes (Match) getoggelt!

Match
1: CCFn-Flag (Modul n Interrupt Flag) wird bei Erreichen des Compare-Wertes gesetzt!

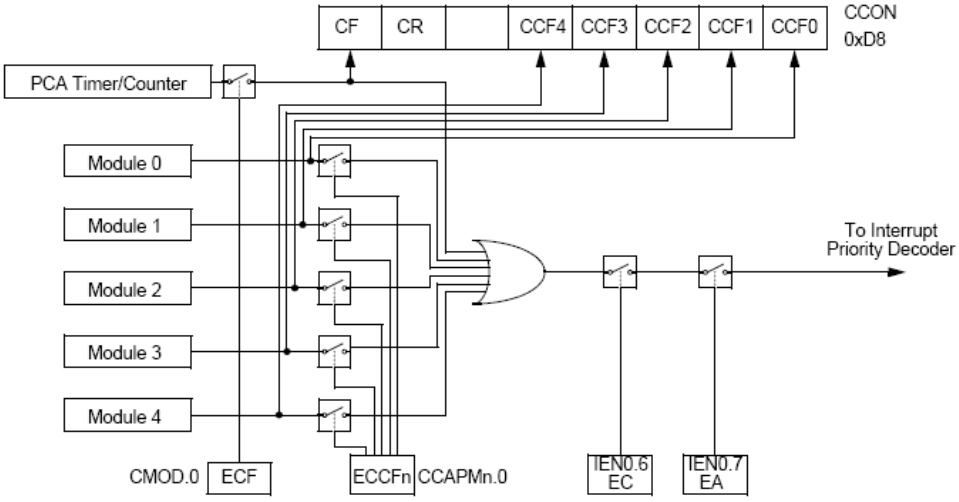
Capture Negative
1: PCA-Counter-Wert bei negativer Flanke an CEXn-Pin fangen

Capture Positive
1: PCA-Counter-Wert bei negativer Flanke an CEXn-Pin fangen

Enable Comparator
1: Freigabe der Vergleicher-Funktion

PCA-Interrupt-System

Der PCA-Counter-Interrupt und die 5-Modul-Interrupts teilen sich eine gemeinsame Interruptleitung. Die Interruptfreigabe erfolgt für den PCA-Counter mit Bit **ECF** und für die 5 PCA-Module mit den Bits **ECCFn**. Die Interruptleitung muss dann noch mit **EC** und **EA** freigegeben werden.

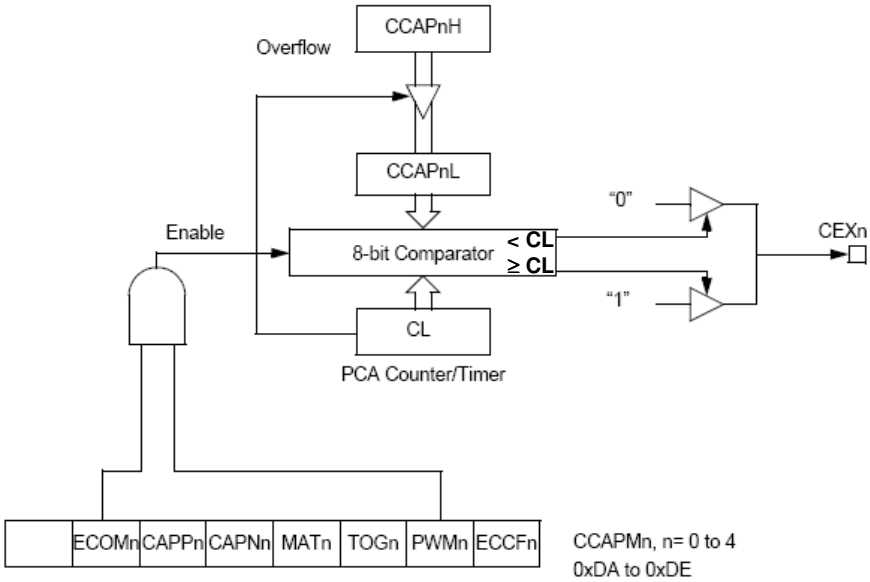


Bei einem Interrupt werden die jeweiligen Interrupt-Anforderungsflags in CCON gesetzt. Nach der Bearbeitung des Interrupts muss das Flag im Programm zurückgesetzt werden.
Die Einsprung-adresse für den PCA-Interrupt ist **0033h**.
Im PWM-Modus werden keine Modul-Interrupts erzeugt, jedoch der PCA-Counter-Interrupt.

Pulsweiten Modulation

Alle 5 PCA Module können als PWM-Ausgänge konfiguriert werden. Da alle den PCA-Counter als Zeitbasis verwenden, besitzen dann alle Ausgänge die gleiche Freuquenz. Die Pulsweite jedes Moduls kann unabhängig verändert werden, indem der Wert des CCAPnL-Registers mit dem Wert des PCA-Counter-Lowbytes (CL) verglichen wird. Abhängig vom Ergebnis wird der PWM-Ausgang CEXn auf 1 oder 0 gelegt. Es gilt:

CL < CCAPnL → CEXn = 0
CL ≥ CCAPnL → CEXn = 1



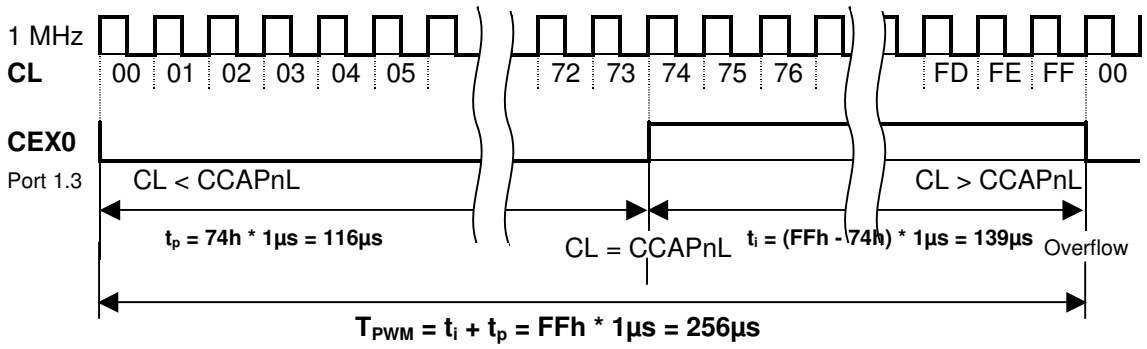
Nach 256 Zähltakten des PCA-Counters erfolgt ein Überlauf des Counter-Lowbytes CL. Dieser bewirkt, dass ein neuer Vergleichswert von CCAPnH nach CCAPnL geladen wird. Damit kann die Pulsweite programmgesteuert verändert werden.

Bsp.:

```
        include     c51rd2.inc
        code at 0

init:   mov   ccap0L,#74h            ; Für erste PWM-Periode
        mov   ccap0H,#74h            ; Nachladewert
        mov   ccapm0,#01000010b      ; PCA-Modul 0 im PWM-Mode
        mov   cmod,#00000000b        ; PCA-Clock = 1MHz
        mov   CL,#0                  ; PCA-Counter Lowbyte = 0
        setb  CR                     ; PCA-Counter starten

loop:   sjmp  loop
```



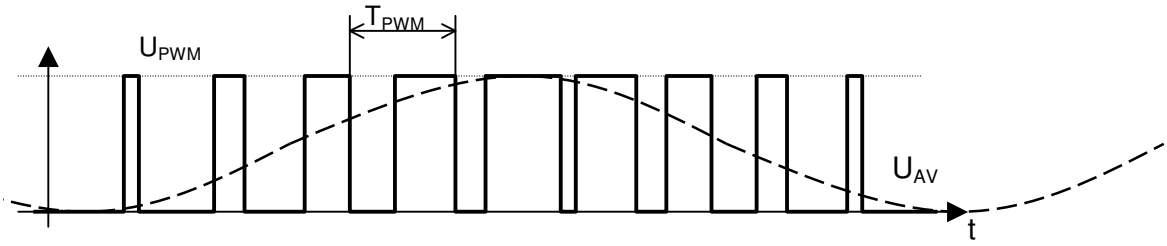
Übung:

Ändern Sie das Programm so ab, dass die Pulsweite durch Betätigen des Tasters P3.2 vergrössert wird. Mit P3.3 wird die Pulsweite wieder kleiner.

Erzeugung eines sinusbewerteten PWM-Signals

Sinusbewertete PWM-Signale werden in der Technik zum Beispiel zur Ansteuerung von Wechselrichtern verwendet. Die entstehende Spannung am Ausgang des Wechselrichters ist dann zwar Rechteckförmig, aber durch Siebung oder induktive Lasten (Motor) ist der Strom dann nahezu sinusförmig.

Prinzip:



Der Sinus wird durch mehreren Stützstellen nachgebildet. Die Spannungswerte von 0 - 5V werden dazu in den Tastgrad der PWM-Spannung umgerechnet. Der Arithmetische Mittelwert des PWM-Signals entspricht dem Sinussignal. Durch Tiefpassfilterung kann der arithmetische Mittelwert gewonnen und dargestellt werden.

	A	B	C
1	Winkel	Reloadwert	
2	0	128	,
3	15	161	,
4	30	192	,
5	45	218	,
6	60	238	,
7	75	251	,
8	90	255	,

Mit einer Excel-Tabelle können die Wert für die Stützstellen mit der folgenden Formel gefunden werden:

=RUNDEN(128+127*SIN(BOGENMASS(A2));0)

Da die PWM-Periode (T_{PWM}) 256µs dauert (1 MHz-PCA-Clock), muss nach 256µs die nächste Stützstelle als Reloadwert für den PCA-Counter geladen werden. Wird der Reloadwert erst alle 512µs aktualisiert, wird der Wert für eine Stützstelle immer 2 mal ausgegeben. Die Zeitverzögerung kann per Software (Zeitschleife) oder mit einem Timer erzeugt werden. Die Periodendauer des Sinus berechnet sich dann mit:

$$f_{Sinus} = \frac{1}{Anzahl\_Stützpunkte \cdot Zeitverzögerung}$$

Bsp.: Für 12 Stützpunkte und 256µs Zeitverzögerung ergibt das eine maximale Frequenz von f = 326 Hz

Die Sinusspannung (Mittelwert) kann mit einem Tiefpass mit fg = 330Hz sichtbar gemnacht werden. Für einen Widerstand von 100kΩ gilt:

$$C = \frac{1}{2\pi \cdot R \cdot f_g} = \frac{1}{2\pi \cdot 100k\Omega \cdot 330Hz} = 4{,}82nF$$

Übung:

- Programmieren Sie den Controller mit dem nachfolgende Programm!
- Stellen Sie die PWM-Spannung auf dem Oszilloskop dar!
- Stellen Sie den Mittelwert auf dem Skopeschirm dar. Schalten Sie gegebenenfalls 2 Tiefpässe hintereinander!
- Verdoppeln Sie die Anzahl der Stützstellen. Die Werte müssen wieder mit Excel berechnet werden. Was hat sich verändert?

```
        code at 0                       ;Programmanfangsadresse
        include c51rd2.inc              ;Registeradressen einbinden

;****************** Konstanten **********************************************************************
        anzahlwerte     equ 12          ;Anzahl der Hilfspunkte der Sinuskurve
        Trigger         equ p2.0

;****************** Programm ************************************************************************
begin:  ljmp    init                    ;Sprung zur Initialisierung

;********** Initialisierungen ***********************************************************************
init:                                   ;PCA = Programmable Counter Array
        mov     CMOD,#00000000b         ;PCA Counter Mode Register: Takt=fosz/12, kein PCA-Interrupt
                                        ;Kein Interrupt möglich bei PWM-Betrieb
        mov     CCAPM0,#01000010b       ;PCA Modules Compare/Capture Control Register 0
                                        ;Modul0 als 8-Bit PWM
        mov     dptr,#sinus             ;Adresse der Werte
        clr     a
        movc    a,@a+dptr               ;1.Wert Stützstelle aus Tabelle holen
        mov     CCAP0H,a                ;Als Compare-Wert (Beim Überlauf des PCA-Low-Bytes CL
                                        ;wird dieser Wert ins ccap0L als neuer Comparewert nachgeladen)
        mov     CCAP0L,a                ;Compare-Wert holen (Anfangswert für 1.Periode des PWM)
        mov     CL,#0                   ;PCA Counter Low-Byte
        setb    CR                      ;PCA starten
;*********** Anfangswerte ***************************************************************************
        mov     p2,#0
        mov     p0,#0                   ;LED's aus
        mov     r6,#0                   ;Zähler für Stützstellen

;!!!!!!!!!!! Hauptprogramm !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
haupt:
zeit:   djnz    r1,zeit                 ;Zeitverzögerung ca. 2µs x 256 = 512µs. In dieser Zeit
                                        ;erfolgen ca. 2 PCA-Counter-Überläufe!! Die Freuqenz des
                                        ;Sinus ist dann ca. fsin = 2 x 39Hz

        mov     a,r6                    ;Zählerstand Nr der Stützstelle holen
        movc    a,@a+dptr               ;Wert Stützstelle aus Tabelle holen
        mov     CCAP0H,a                ;ergibt nächsten Compare-Wert
        inc     r6                      ;Zählerstand erhöhen

        cjne    r6,#anzahlwerte,haupt   ;aus  der Tabelle geholt sind
        mov     r6,#0                   ;Zähler wieder null setzen
        cpl     Trigger                 ;Für Tests mit Osszi
        cpl     Trigger

        sjmp    haupt

;!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

;*********** Tabelle der Stützpunkte des Sinussignals ***********************************************

sinus:  db      128,192,238,255,238,192,128, 65, 18,  1, 18, 64,128
    ;Winkel:     0  30  60  90  120 150 180 210 240 270 300 330 360°
```

Sinusbewertete PWM mit Timer 2

```
        code at 0                       ;Programmanfangsadresse
        include c51rd2.inc              ;Registeradressen einbinden

;****************** Konstanten ************************************************
        anzahlwerte     equ 24          ;Anzahl der Hilfspunkte der Sinuskurve
        Trigger         bit p2.0        ; Triggersignal für Oszi
        Frequ_UP        bit p3.2        ; Taster zur Erhöhung der Frequenz
        Frequ_DOWN      bit p3.3        ; Taster zur Verringerung der Frequenz

;****************** Programm **************************************************
begin:  ljmp    init                    ;Sprung zur Initialisierung

        org     002Bh
;******* Timer 2 Interrupt ****************************************************
T2Int:  mov     a,r6                    ;Zählerstand Nr der Stützstelle holen
        movc    a,@a+dptr               ;Wert Stützstelle aus Tabelle holen
        mov     CCAP0H,a                ;ergibt nächsten Compare-Wert
        inc     r6                      ;Zählerstand erhöhen

        cjne    r6,#anzahlwerte,exitT2  ;Wenn letzter Wert aus Tabelle,
        mov     r6,#0                   ; dann Zähler wieder null setzen
        cpl     Trigger
        cpl     Trigger

exitT2: clr     TF2                     ; Timer 2 Interrupt-Flag zurücksetzen
        reti

;********** Initialisierungen **********************************************
init:   mov     T2MOD,#00000000b
        mov     T2CON,#00000000b        ; Timer 2 als 16 Bit-Autoreload-Timer
        mov     TL2,#00h
        mov     TH2,#0FFh
        mov     RCAP2L,#00h             ; Timer2 Reloadwert für größte Frequenz
        mov     RCAP2H,#0FFh            ;  f = 1/(12*[FFFFh-FFh]µs) = 326Hz
                                        ;PCA = Programmable Counter Array
        mov     CMOD,#00000000b         ;PCA Counter Mode Register: Takt=fosz/12, kein PCA-Interrupt
                                        ;Kein Interrupt möglich bei PWM-Betrieb
        mov     CCAPM0,#01000010b       ;PCA Modules Compare/Capture Control Register 0
                                        ;Modul0 als 8-Bit PWM
        mov     dptr,#sinus             ;Adresse der Sinus-Tabelle
        clr     a
        movc    a,@a+dptr               ;1.Wert Stützstelle aus Tabelle holen
        mov     CCAP0H,a                ;Als Compare-Wert (Beim Überlauf des PCA-Low-Bytes CL
                                        ;wird dieser Wert ins ccap0L als neuer Comparewert nachgeladen)
        mov     CCAP0L,a                ;Compare-Wert holen (Anfangswert für 1.Periode des PWM)
        mov     CL,#0                   ;PCA Counter Low-Byte

;*********** Anfangswerte *************************************************
        mov     p2,#0
        mov     p0,#0
        mov     r6,#0                   ;Zähler für Stützstelle
        setb    TR2                     ;Timer 2 starten
        setb    ET2                     ;Timer 2 Interrupt freigeben
        setb    EA                      ;globale Interruptfreigabe
        setb    CR                      ;PCA-Counter starten
;!!!!!!!!!!! Hauptprogramm !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
haupt:  sjmp    haupt

;*********** Tabelle der Stützpunkte des Sinussignals *************************
sinus:  db      128,161,192,218,238,251,255,251,238,218,192
        db      161,128,95,65,38,18,5,1,5,18,38,64,95,128

end
```

Übung:

- Geben Sie das Programm mit Zeitverzögerung durch Timer 2 ein.
- Die Zeitverzögerung soll nun durch Betätigen des Tasters P3.2 vergrössert werden. Mit P3.3 soll Sie wieder verkleinert werden. (Sinusgenerator mit variabler Frequenz)