System-On-Chip zum Messen, Steuern und Regeln

Mikrocontroller mit Analogfunktionen reduziert Systemaufwand

Dr. Claus Kühnel

Microchip, bekannt durch die breite Palette an Mikrocontrollern der PICmicro®-Familie, bietet zunehmend mehr Analog- und Interface-Komponenten an. Ein Blick in die Microchip Website zeigt eine breite Produktpalette [www.microchip.com/1010/pline/analog/index.htm].

Das Beherrschen analoger und digitaler Funktionen auf dem gleichen Silizium ermöglicht aber weitergehende Produkte, wie die sogenannten „PICmicro® Advanced Analog Microcontrollers“. Diese PICmicro-Familie wurde speziell für Anwendungen der Mess-, Steuerungs- und Regelungstechnik entwickelt und kann sensor- bzw. aktornah bestimmte Verarbeitungsfunktionen übernehmen.

Ziel dieses Beitrages ist es, einige Möglichkeiten der On-Chip-Peripherie der Mikrocontroller PIC16C781 bzw. PIC16C782 aufzuzeigen und erste Erfahrungen mit diesem Chip mitzuteilen. Die On-Chip-Peripherie besteht aus den folgenden Komponenten:

- 8-Bit Analog-Digitalumsetzer
- 8-Bit Digital-Analogumsetzer
- Operationsverstärker
- zwei Komparatoren
- programmierbarer Switch-Mode-Controller
- Spannungsreferenz
- zwei Timer mit Prescaler
- programmierbare Unterspannungsdetektion

Der eigentliche Mikrocontroller-Kern ist eine normale Mid-Range-CPU der PICmicro-Familie und wird bei den Betrachtungen nicht im Vordergrund stehen.

1. Mixed-Signal-Mikrocontroller PIC16C78x

Die Hauptmerkmale der PIC16C78x wurden bereits in der Einleitung herausgestellt. Der Unterschied zwischen dem PIC16C781 und dem PIC16C782 liegt nur im verfügbaren Programmspeicher. Der PIC16C781 weist 1 K Worte (zu 14 Bit) Programmspeicher auf, während dieser beim PIC16C782 2 K Worte beträgt.

Die PIC16C78x sind in einem 20-poligen Gehäuse untergebracht, was bei den zahlreichen Funktionsgruppen zu einer Mehrfachbelegung einzelner Anschlüsse führen muss. Abbildung 1 zeigt die Anschlussbelegung mit den betreffenden Bezeichnungen. Abbildung 2 zeigt das Blockschaltbild des Mikrocontrollers PIC16C782.

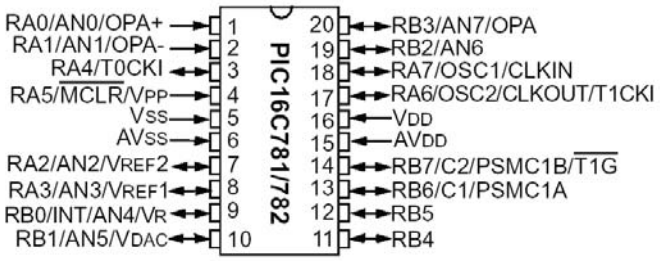

Abbildung 1 Anschlussbelegung der PIC16C78x

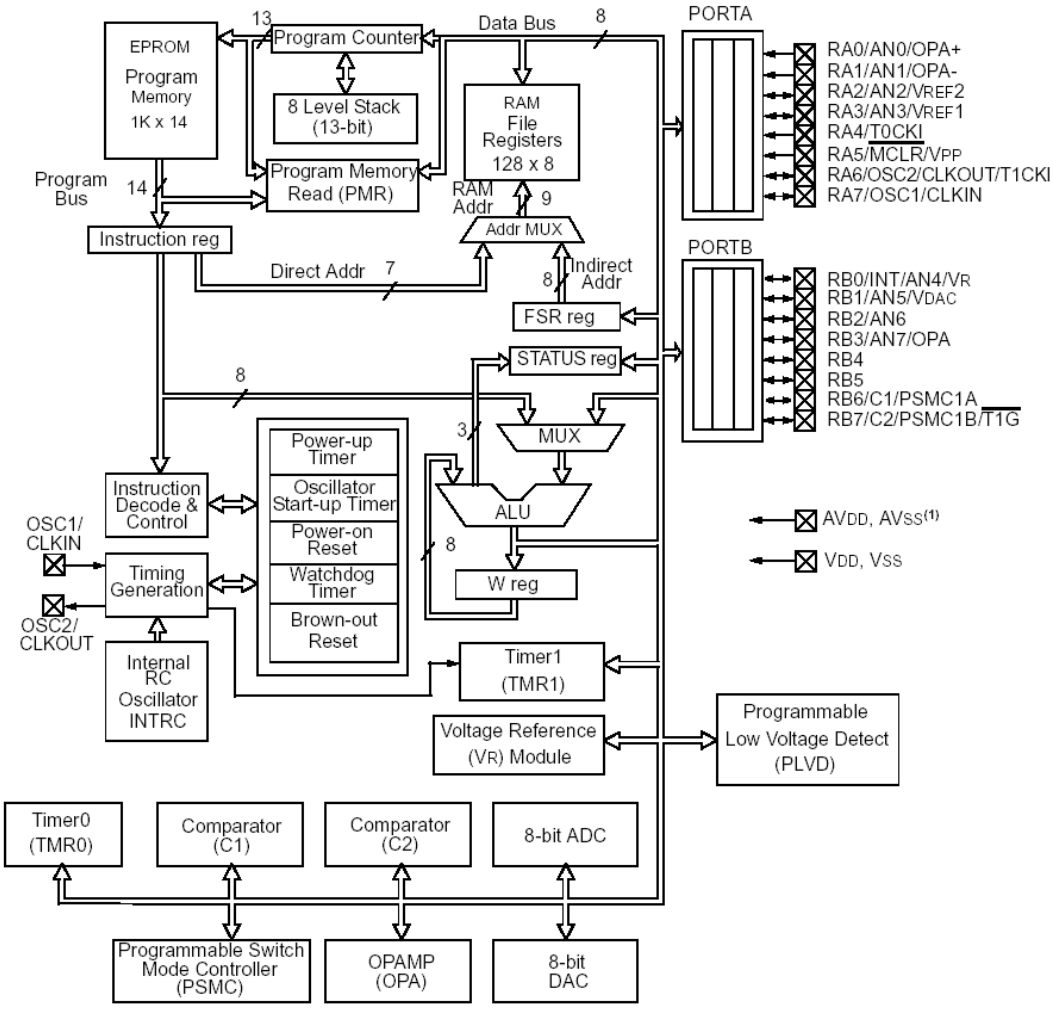

Abbildung 2 PIC16C782 Blockschaltbild

Die Zuweisung der Anschlüsse der PIC16C78x zu den gewünschten Funktionsgruppen auf dem Chip wird über Multiplexer vorgenommen. Special Function Register steuern diese Multiplexer. Abbildung 3 zeigt die analogen Funktionsgruppen des PIC16C782 einschließlich der für die Verknüpfung auf dem Chip verantwortlichen Multiplexer.

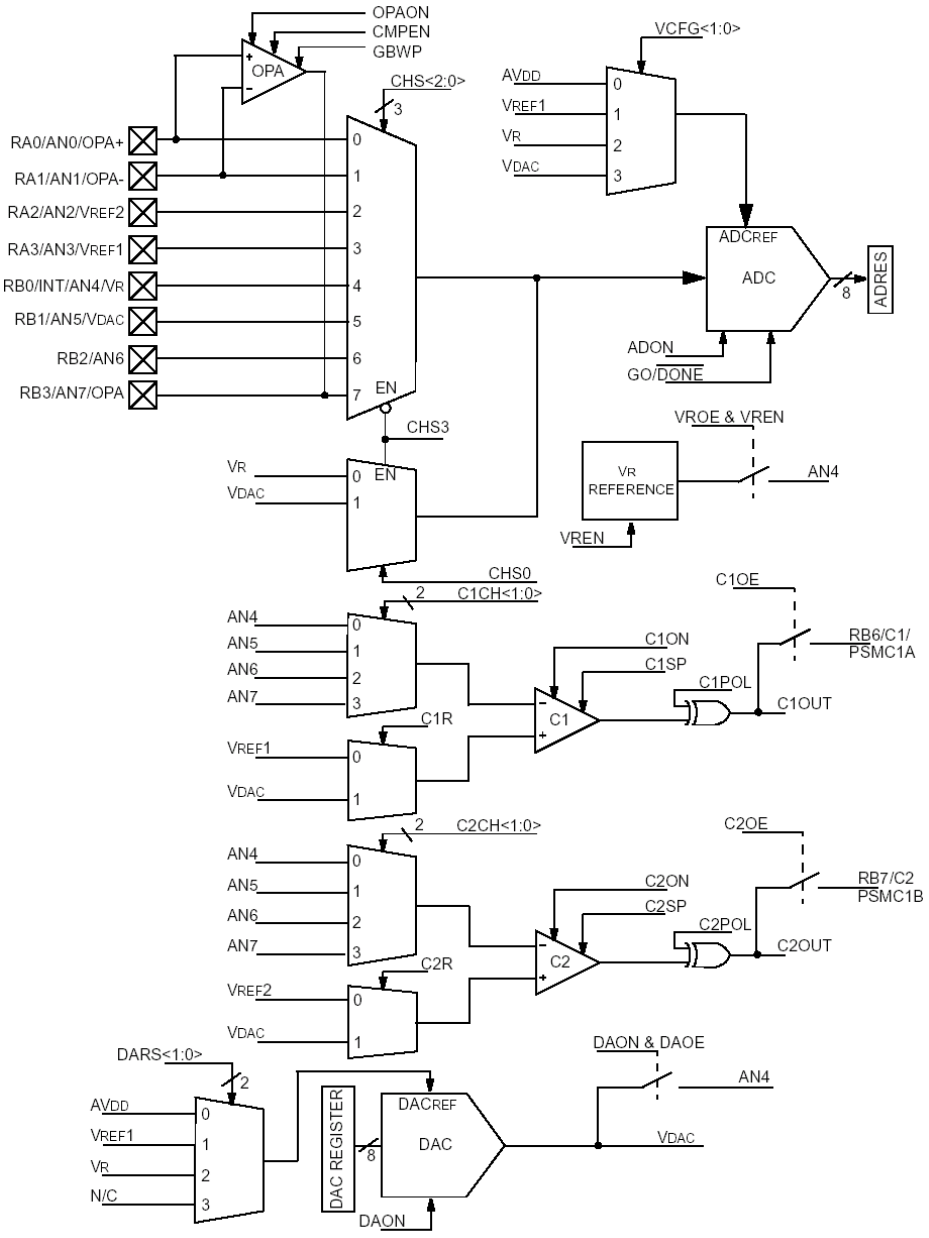

Abbildung 3 Blockschaltbild der analogen Funktionsgruppen

Damit eine fehlerfreie Initialisierung der Peripherie sowie der Multiplexer erfolgt, muss eine sorgfältige Planung vorgenommen werden.

Microchip bietet zur Einarbeitung in diese Mikrocontroller das Evaluationboard PICDEM™ MCS1 und ein grafisches Konfigurationstool (GUI) an, mit dem auch gleich der Initialisierungscode generiert werden kann. Abbildung 4 zeigt das Evaluationboard. In der Mitte befindet sich eine Pfostenleiste, auf die jeder Anschluss des Mikrocontroller geführt ist. Der eingesetzte Mikrocontroller kommuniziert seriell (RS-232) über ein Monitorprogramm mit dem GUI auf dem Entwicklungs-PC.

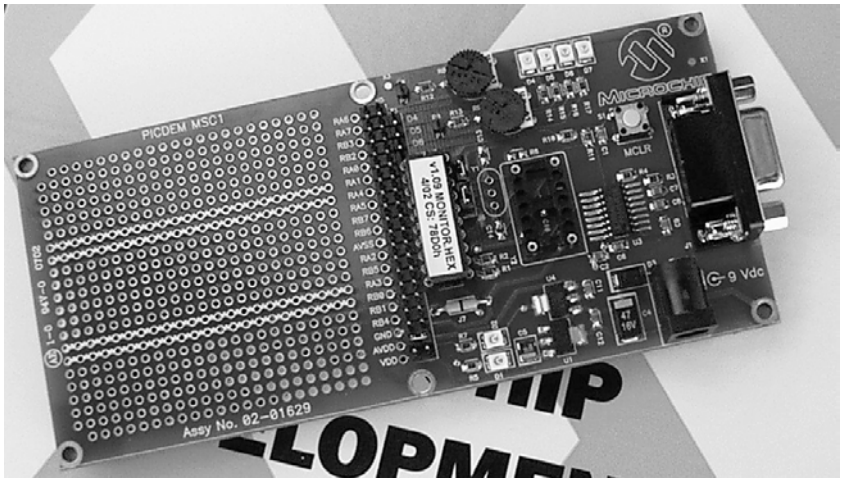

Abbildung 4 Evaluationboard PICDEM™ MCS1

2. Anwendungsbeispiele

2.1. Analog-Digitalumsetzer

Alle Komponenten zur analogen Messwerterfassung sind auf dem PIC16C78x integriert. Mit dem Operationsverstärker kann ein bandbegrenzender Tiefpass aufgebaut werden. Der Analog-Digitalumsetzer (ADU) besitzt eine vorgeschaltete Sample&Hold-Stufe. Als Referenzspannungsquelle kann die interne Bandgap-Referenz verwendet werden. Die Taktfrequenz der AD-Umsetzung wird vom Systemtakt abgeleitet. Für den zyklischen Start der AD-Umsetzung sorgt ein Timerinterrupt.

Bei der Dimensionierung des vorzuschaltenden Tiefpasses wurde von der folgenden Überlegung ausgegangen:

- Verwendung des internen Oszillator (f_{osc} = 4 MHz typ.)
- Periode der AD-Umsetzung 10 µs (bei einer Umsetzzeit von max. 6 µs)
- Grad des vorgeschalteten Tiefpasses maximal drei, da nur ein Operationsverstärker verfügbar ist

Das ideale Signal-Rauschverhältnis berechnet sich nach der Beziehung

$$\mathrm{SNR}\,[\mathrm{dB}] = 6.02 \cdot \mathrm{n} + 1.76$$

Der Parameter n bezeichnet die Auflösung der AD-Umsetzers. Demzufolge muss die Dämpfung ca. 50 dB betragen.

Mit dem Programm FilterLab® von Microchip kann die Dimensionierung des Tiefpasses sehr einfach vorgenommen werden. Abbildung 5 zeigt die Ergebnisse der Berechnung für eine Grenzfrequenz des Tiefpassfilters von 10 kHz. Bei einer Chebyshev-Charakteristik des Filters wird mit einem Filter dritten Grades eine Dämpfung von 53,7 dB erreicht.

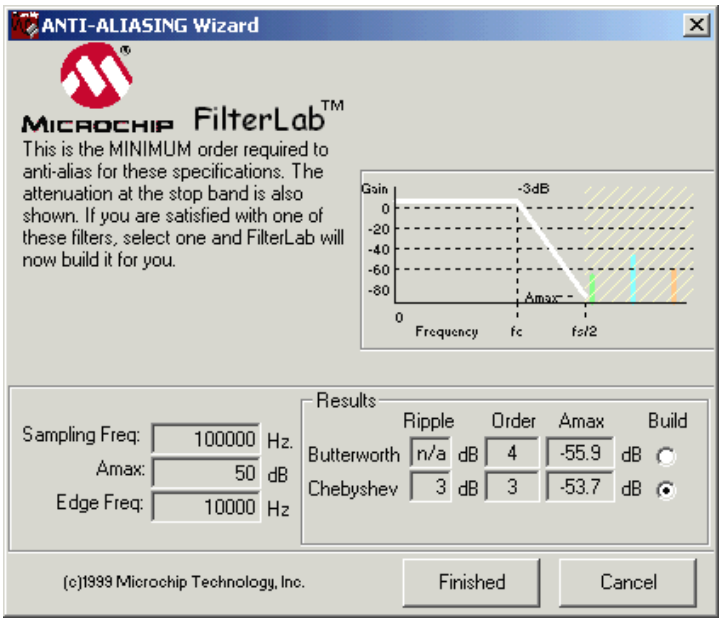

Abbildung 5 Tiefpass-Berechnung mit dem Programm FilterLab

Das Filterprogramm dimensioniert außerdem das zu entwerfende Filter. Abbildung 6 zeigt das dimensionierte Tiefpassfilter nach Sallen&Key.

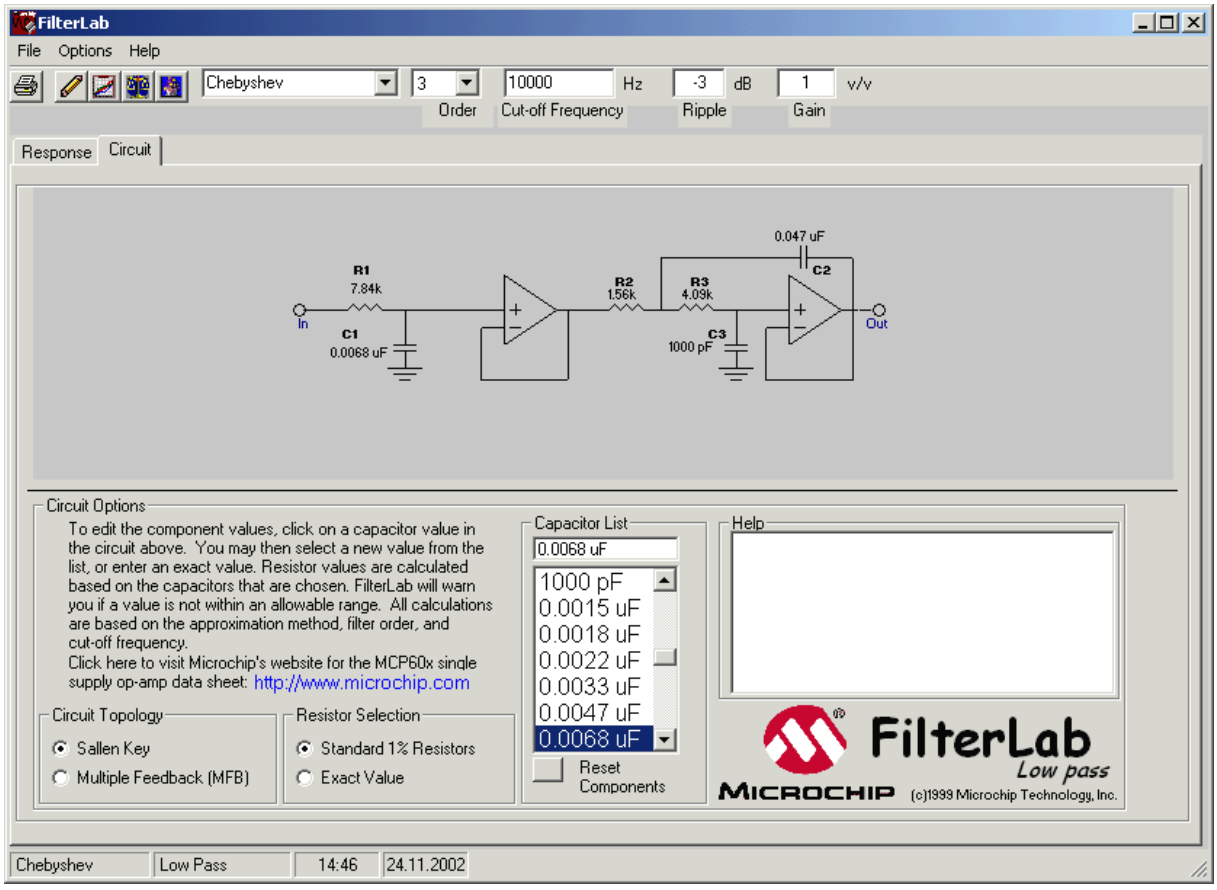

Abbildung 6 Dimensioniertes Filter nach Sallen&Key

Im PIC16C78x steht nur ein Operationsverstärker zur Verfügung. Verändert man die Anordnung der beiden Stufen, dann kann auf den Operationsverstärker beim passiven Pol verzichtet werden. Der AD-Umsetzer erwartet eine Impedanz der analogen Spannungsquelle von maximal 10 kΩ, weshalb das RC-Glied entsprechend niederohmig ausgelegt werden muss. Abbildung 7 zeigt die gesamte Funktionsgruppe zur AD-Umsetzung.

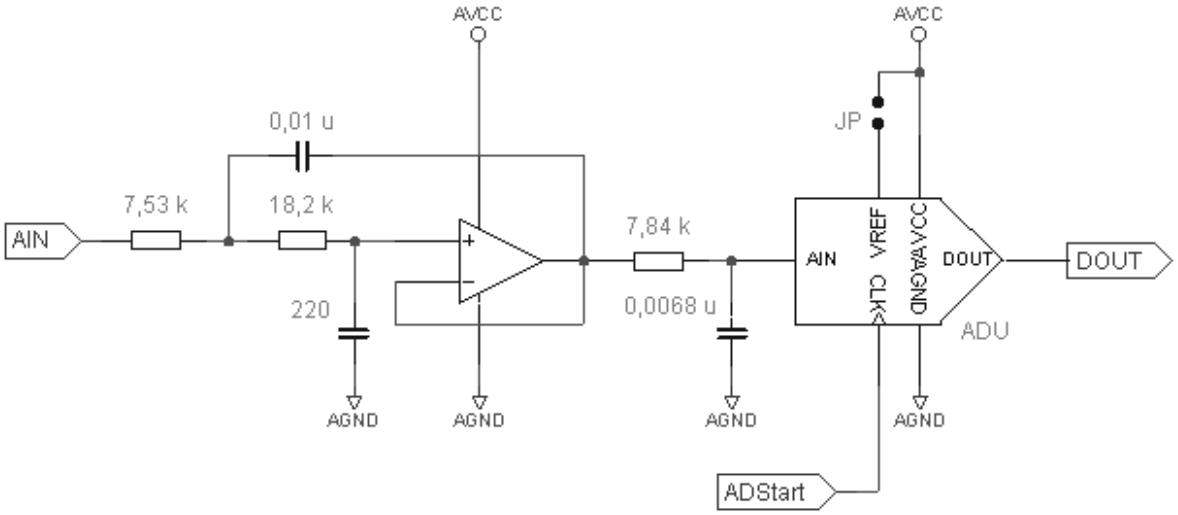

Abbildung 7 AD-Umsetzer mit vorgeschaltetem Tiefpass

Der in Abbildung 7 dargestellte AD-Umsetzer mit vorgeschaltetem Tiefpassfilter ist schließlich mit den internen Funktionsgruppen des PIC16C782 zu realisieren. Abbildung 8 zeigt eine mögliche Schaltungsrealisierung.

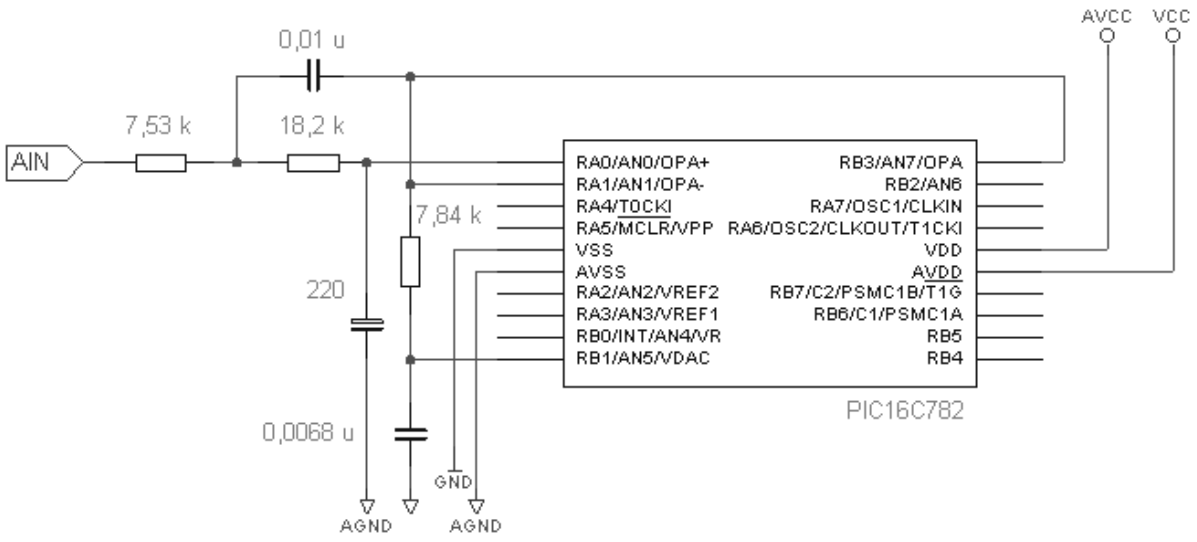

Abbildung 8 Beschaltung des PC16C782 zur AD-Umsetzung

Die Konfiguration der Multiplexer im PIC16C782 kann nun anhand des Datenbuches oder mit dem bereits erwähnten GUI erfolgen. Die nächsten Abbildungen zeigen die drei notwendigen Schritte. Der Entwicklungs-PC wird hierzu über ein serielles Kabel mit dem Evaluationboard PICDEM™ MCS1 verbunden. Auf diese Weise kann das GUI direkt mit dem PIC16C782 auf dem Board kommunizieren und die Funktion der analogen Peripherie veran-

schaulichen. Die Beschaltung der analogen Funktionsgruppen kann auf dem Lochrasterfeld des Evaluationboards vorgenommen werden.

Abbildung 9 zeigt die Zuordnung der Pins zu Operationsverstärker und AD-Umsetzer.

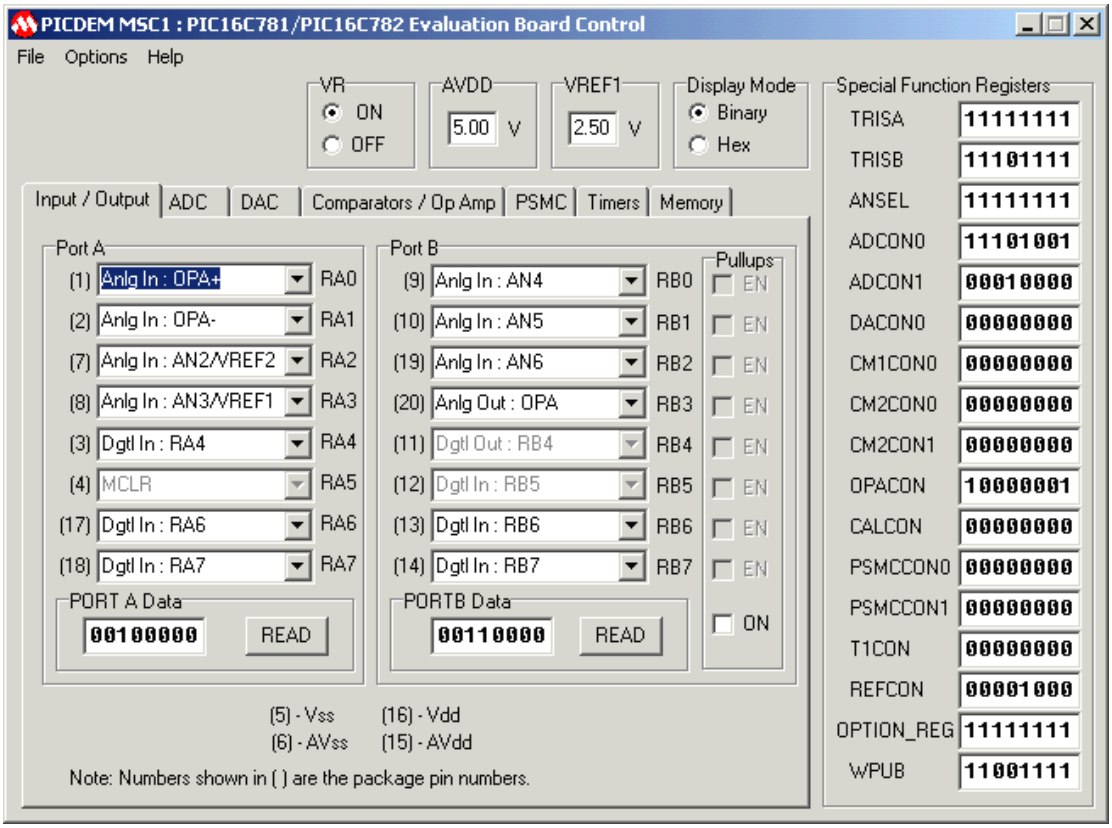

Abbildung 9 Konfiguration der I/O Pins

Für den Operationsverstärker gibt es keine Alternativen. Als Eingang für den AD-Umsetzer wird AN5 verwendet. Hier gibt es einige gleichwertige Alternativen. Wenn möglich, sollte auf AN6 (I/O Pin 19) wegen seiner Nähe zu OSC1 (I/O Pin 18) als Eingang für den AD-Umsetzer verzichtet werden.

Als Referenzspannung dient die interne Referenzspannungsquelle und der interne RC-Oszillator taktet die AD-Umsetzung. Dieses Taktsignal ist nicht mit dem Signal ADStart in Abbildung 7 zu verwechseln. Der Start der AD-Umsetzung soll über den Timer0 gesteuert werden. Abbildung 10 zeigt die Konfiguration des AD-Umsetzers gemäss den genannten Vorgaben.

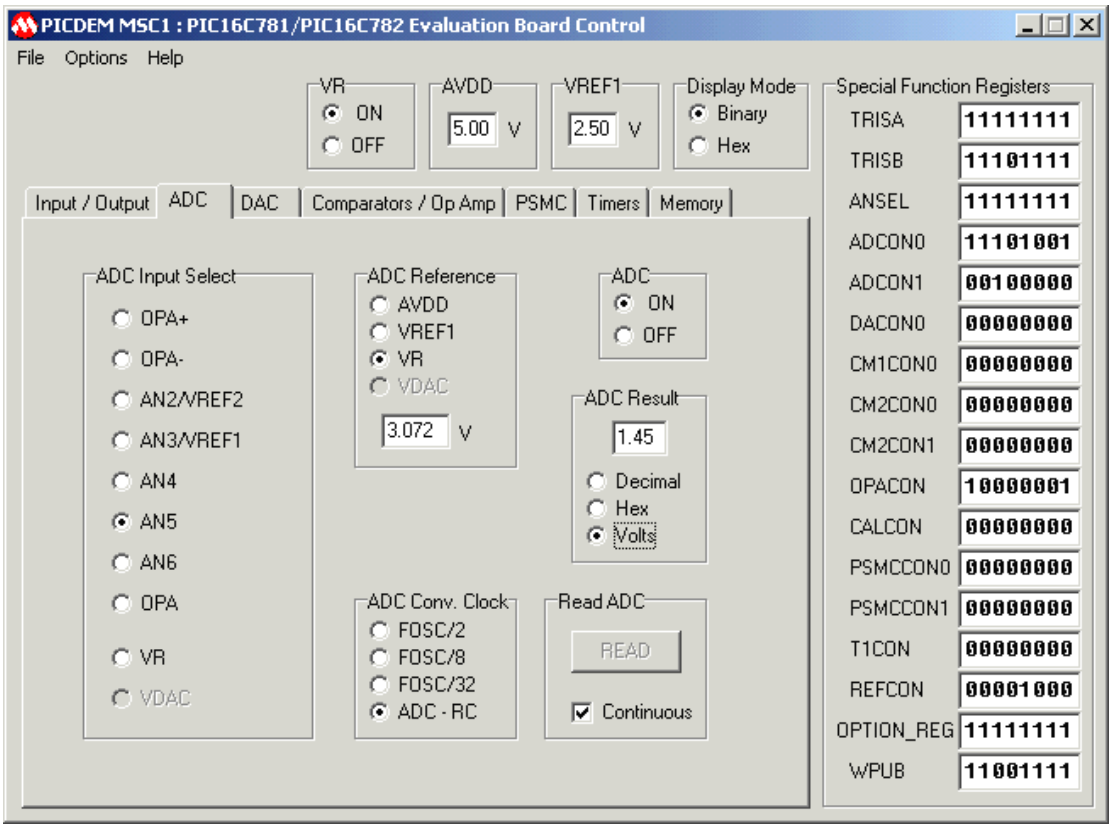

Abbildung 10 Konfiguration des AD-Umsetzers

Beim Operationsverstärker sieht die Konfiguration recht einfach aus. Wie Abbildung 11 zeigt, kann nur die Bandbreite des Operationsverstärkers selektiert werden und nach einem Enable ist er schließlich betriebsbereit.

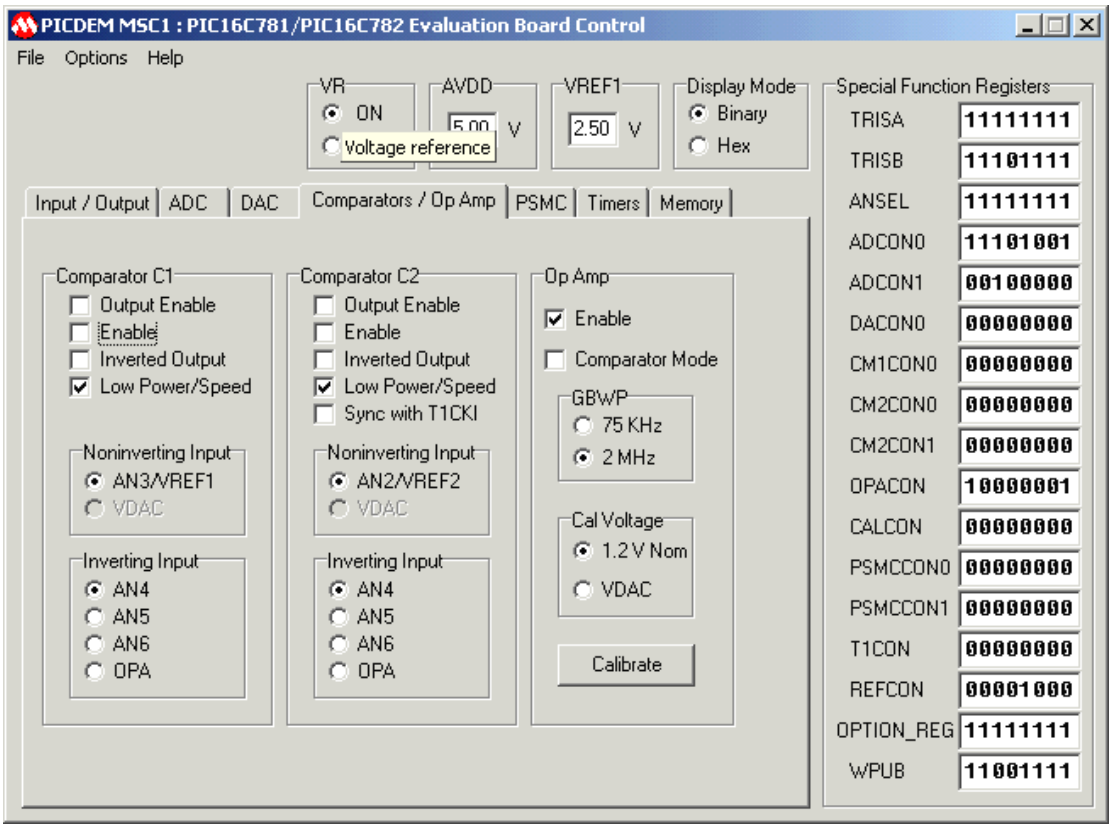

Abbildung 11 Konfiguration des Operationsverstärkers

Am Ende der vorgenommenen Konfiguration aller verwendeten Funktionsgruppen der On-Chip-Peripherie kann der Sourcecode generiert werden. Listing 1 zeigt den generierten Assemblercode, der die Basis für das Anwendungsprogramm bilden kann.

```
__CONFIG  _BODEN_OFF & _CP_OFF & _VBOR_25 & _PWRTE_ON & _MCLRE_ON & _WDT_OFF &
_INTRC_OSC_NOCLKOUT

            org     0x000
            goto    Initialize

            org     0x004
; Insert interrupt service routine here

Initialize

            BANKSEL  PORTA

            movlw   0xF0     ; Initial port value
            movwf   PORTA
            movlw   0xF0     ; Initial port value
            movwf   PORTB
            movlw   0x00     ; Gate Disabled, Prescale 1:1, LP Osc Off, Sync Off,
                             ;  Clk: Fosc/4, Timer1 Off
            movwf   T1CON
            movlw   0xE9     ; ADC clk: ADRC, ADC Input: AN5, ADC On
            movwf   ADCON0

            BANKSEL  ANSEL
```

```
            movlw   0xFF    ; Analog pins: RB(0,1,2,3), RA(0,1,2,3)
            movwf   ANSEL
            movlw   0xCF    ; PORTB Pullups: RB(0,1,2,3,6,7)
            movwf   WPUB
            movlw   0xFF    ; Inputs: RA(0,1,2,3,4,5,6,7), Outputs: RA(none)
            movwf   TRISA
            movlw   0xEF    ; Inputs: RB(0,1,2,3,5,6,7), Outputs: RB(4)
            movwf   TRISB
            movlw   0x08    ; VR: On, VR Output: Off
            movwf   REFCON
            movlw   0x20    ; ADC Reference: VR
            movwf   ADCON1
            movlw   0xDF    ; Pullups: Off, RB0/INT edge: Rising, TMR0 clk: Fosc/4,
                            ;  TMR0 edge: Falling, Prescaler: WDT(1:128)
            movwf   OPTION_REG

            BANKSEL DACON0

            movlw   0x00    ; DAC Value
            movwf   DAC
            movlw   0x00    ; DAC: Off, DAC output: Off, DAC Ref: AVDD
            movwf   DACON0
            movlw   0x00    ; C1: Off, Output: Off, Non-Inverted, Speed: Slow,
                            ;  In(+): RA3/VREF1, In(-): RB0/AN4
            movwf   CM1CON0
            movlw   0x00    ; C2: Off, Output: Off, Non-Inverted, Speed: Slow,
                            ;  In(+): RA2/VREF2, In(-): RB0/AN4
            movwf   CM2CON0
            movlw   0x00    ; C2 sync to Timer1: Off
            movwf   CM2CON1
            movlw   0x81    ; OpAmp: On, Mode: OpAmp, Speed: Fast
            movwf   OPACON
            movlw   0x00    ; OpAmp calibration source: Internal 1.2V
            movwf   CALCON
            movlw   0x00    ; PSMC Fout: Fosc/128, Min PWM DC: 0%, Max PWM DC: 50%,
                            ;  PSM DC: 12.5%
            movwf   PSMCCON0
            movlw   0x00    ; PSMC: Off, PSMC1A: High true, PSMC1B: High true,
                            ;  Slope Compensation: Off, Output: Single, Mode: PSM, C1
only
            movwf   PSMCCON1

; Insert user code here
```

Listing 1 Generiertes Konfigurationsfile

Nur wenige Ergänzungen sind am generierten Konfigurationsfile notwendig, um die eingangs erläuterte Aufgabenstellung zu erfüllen.

Timer0 soll periodisch die AD-Umsetzung auslösen. Hierzu ist der Timer noch zu initialisieren. Beginnend am Label m1 werden das Register TMR0 mit dem für die Periode der AD-Umsetzung verantwortlichen Reloadvalue geladen und anschließend der Interrupt freigegeben.

Die Interruptserviceroutine (ISR) beginnt beim PIC16C782 an Adresse 0x04 und umfasst die folgenden Aktivitäten:

1. Reload des Registers TMR0
2. Start der AD-Umsetzung
3. Reset des Timer0-Interruptflags

Listing 2 zeigt den Quelltext der gesamten Anwendung.

```
#include p16c782.inc

__CONFIG  _BODEN_OFF & _CP_OFF & _VBOR_25 & _PWRTE_ON & _MCLRE_ON & _WDT_OFF &
_INTRC_OSC_NOCLKOUT

Reload      EQU     0xFA

            org     0x000
            goto    Initialize

            org     0x004

; Insert interrupt service routine here

            movlw   Reload          ; Set TMR0 to reload value
m2          movwf   TMR0
            bsf     ADCON0,GO       ; Start AD Conversion
            bcf     INTCON,T0IF     ; Clear TMR0 Interrupt Flag
            retfie

Initialize

            BANKSEL PORTA

            movlw   0x20    ; Initial port value
            movwf   PORTA
            movlw   0x30    ; Initial port value
            movwf   PORTB
            movlw   0x00    ; Gate Disabled, Prescale 1:1, LP Osc Off, Sync Off,
                            ;  Clk: Fosc/4, Timer1 Off
            movwf   T1CON
            movlw   0xE9    ; ADC clk: ADRC, ADC Input: AN5, ADC On
            movwf   ADCON0

            BANKSEL ANSEL

            movlw   0xFF    ; Analog pins: RB(0,1,2,3), RA(0,1,2,3)
            movwf   ANSEL
            movlw   0xCF    ; PORTB Pullups: RB(0,1,2,3,6,7)
            movwf   WPUB
            movlw   0xFF    ; Inputs: RA(0,1,2,3,4,5,6,7), Outputs: RA(none)
            movwf   TRISA
            movlw   0xEF    ; Inputs: RB(0,1,2,3,5,6,7), Outputs: RB(4)
            movwf   TRISB
            movlw   0x08    ; VR: On, VR Output: Off
            movwf   REFCON
            movlw   0x20    ; ADC Reference: VR
            movwf   ADCON1
            movlw   0xDF    ; Pullups: Off, RB0/INT edge: Rising, TMR0 clk: Fosc/4,
                            ;  TMR0 edge: Falling, Prescaler: WDT(1:128)
```

```
            movwf   OPTION_REG

            BANKSEL DACON0

            movlw   0x00    ; DAC Value
            movwf   DAC
            movlw   0x00    ; DAC: Off, DAC output: Off, DAC Ref: AVDD
            movwf   DACON0
            movlw   0x00    ; C1: Off, Output: Off, Non-Inverted, Speed: Slow,
                            ;  In(+): RA3/VREF1, In(-): RB0/AN4
            movwf   CM1CON0
            movlw   0x00    ; C2: Off, Output: Off, Non-Inverted, Speed: Slow,
                            ;  In(+): RA2/VREF2, In(-): RB0/AN4
            movwf   CM2CON0
            movlw   0x00    ; C2 sync to Timer1: Off
            movwf   CM2CON1
            movlw   0x81    ; OpAmp: On, Mode: OpAmp, Speed: Fast
            movwf   OPACON
            movlw   0x00    ; OpAmp calibration source: Internal 1.2V
            movwf   CALCON
            movlw   0x00    ; PSMC Fout: Fosc/128, Min PWM DC: 0%, Max PWM DC: 50%,
                            ;  PSM DC: 12.5%
            movwf   PSMCCON0
            movlw   0x00    ; PSMC: Off, PSMC1A: High true, PSMC1B: High true,
                            ;  Slope Compensation: Off, Output: Single, Mode: PSM, C1
only
            movwf   PSMCCON1

; Insert user code here

m1          movlw   Reload          ; Set TMR0 to reload value
            movwf   TMR0

            CLRF    INTCON          ; Disable interrupts and clear T0IF
            BSF     INTCON, T0IE    ; Enable TMR0 interrupt
            BSF     INTCON, GIE     ; Enable all interrupts
;
main        goto    main
            end
```

Listing 2 Programm zur periodischen AD-Umsetzung ADU.ASM

Testet man das Programm ADU.ASM beispielsweise im MPLAB® Simulator, dann erkennt man schnell, dass nur einige wenige Prozessorzyklen außerhalb der ISR für die (leere) Endlosschleife übrigbleiben. In der Realität wird man dann mit einem höheren Prozessortakt arbeiten.

2.2. Signalgenerator

Zu Testzwecken soll ein Signalgenerator einen sin(x)/x-Impuls erzeugen. Timer0 triggert den DA-Umsetzer, der Werte aus einer im ROM abgelegten Tabelle abarbeitet.

Initialisiert man den DA-Umsetzer mit Hilfe des GUIs, dann lassen sich die Parameter für ein korrektes Timing sehr einfach ermitteln. Abbildung 12 den von einem Textfile geladenen Spannungsverlauf, der 80 Stützstellen umfasst.

Abbildung 12 Konfiguration der Parameter des Signalgenerators

Timer0 wird mit der durch Vier geteilten Oszillatorfrequenz, hier also 1 MHz, getaktet. Bei einer Wiederholrate des Pulses von 1 s erreicht der Timer bei einem Prescaler von 64 einen Wert von 195. Im PICDEM™ MCS1 kann die Funktion unseres Signalgenerator sehr gut nachvollzogen werden.

Generiert man im PICDEM™ MCS1 nun den Code, dann erhält man aber nur die Initialisierung der Special Function Register. Die Tabelle der Stützwerte und die Interruptserviceroutine für Timer0 müssen im Anwendungsprogramm konventionell codiert werden. Listing 3 zeigt das Programm WAVE.ASM, welches neben dem Initialisierungsteil nur aus einer Interruptserviceroutine und der im Includefile TABLE.INC abgelegten Tabelle mit dem zu generierenden Signalverlauf besteht.

```
__CONFIG  _BODEN_OFF & _CP_OFF & _VBOR_25 & _PWRTE_ON & _MCLRE_ON & _WDT_OFF &
_INTRC_OSC_NOCLKOUT

#include p16c782.inc
#include table.inc

Reload      EQU     0x3D            ; Reload value for 195 counts
samples     EQU     0x50            ; #samples

offset      EQU     0x20            ; Variable table offset

            org     0x000
            goto    Initialize

            org     0x004

; Insert interrupt service routine here

            movlw   Reload          ; Set TMR0 to reload value
m2          movwf   TMR0
            movlw   HIGH Table      ; load PCLATH with hi
            movwf   PCLATH
            movf    offset,w        ; load offset in w reg
            call    Table
            movwf   DAC             ; write table value to DAC
```

```
            decf    offset          ; next table value
            skpnz
            goto    wrap
            bcf     INTCON,T0IF
            retfie
wrap        movlw   samples         ; initializes table offset
            movwf   offset
            bcf     INTCON,T0IF
            retfie

Initialize

            BANKSEL PORTA

            movlw   0x20    ; Initial port value
            movwf   PORTA
            movlw   0x30    ; Initial port value
            movwf   PORTB
            movlw   0x00    ; Gate Disabled, Prescale 1:1, LP Osc Off, Sync Off,
                            ;  Clk: Fosc/4, Timer1 Off
            movwf   T1CON
            movlw   0x00    ; ADC clk: Fosc/2, ADC Input: AN0, ADC Off
            movwf   ADCON0

            BANKSEL ANSEL

            movlw   0xFF    ; Analog pins: RB(0,1,2,3), RA(0,1,2,3)
            movwf   ANSEL
            movlw   0xCF    ; PORTB Pullups: RB(0,1,2,3,6,7)
            movwf   WPUB
            movlw   0xFF    ; Inputs: RA(0,1,2,3,4,5,6,7), Outputs: RA(none)
            movwf   TRISA
            movlw   0xEF    ; Inputs: RB(0,1,2,3,5,6,7), Outputs: RB(4)
            movwf   TRISB
            movlw   0x00    ; VR: Off, VR Output: Off
            movwf   REFCON
            movlw   0x00    ; ADC Reference: AVDD
            movwf   ADCON1
            movlw   0xD5    ; Pullups: Off, RB0/INT edge: Rising, TMR0 clk: Fosc/4,
                            ;  TMR0 edge: Falling, Prescaler: TMR0(1:64)
            movwf   OPTION_REG

            BANKSEL DACON0

            movlw   0x00    ; DAC Value
            movwf   DAC
            movlw   0xC0    ; DAC: On, DAC output: RB1/VDAC, DAC Ref: AVDD
            movwf   DACON0
            movlw   0x00    ; C1: Off, Output: Off, Non-Inverted, Speed: Slow,
                            ;  In(+): RA3/VREF1, In(-): RB0/AN4
            movwf   CM1CON0
            movlw   0x00    ; C2: Off, Output: Off, Non-Inverted, Speed: Slow,
                            ;  In(+): RA2/VREF2, In(-): RB0/AN4
            movwf   CM2CON0
            movlw   0x00    ; C2 sync to Timer1: Off
            movwf   CM2CON1
            movlw   0x00    ; OpAmp: Off, Mode: OpAmp, Speed: Slow
            movwf   OPACON
            movlw   0x00    ; OpAmp calibration source: Internal 1.2V
            movwf   CALCON
            movlw   0x00    ; PSMC Fout: Fosc/128, Min PWM DC: 0%, Max PWM DC: 50%,
                                    ;  PSM DC: 12.5%
            movwf   PSMCCON0
```

```
            movlw   0x00    ; PSMC: Off, PSMC1A: High true, PSMC1B: High true,
                            ; Slope Compensation: Off, Output: Single, Mode: PSM, C1
only
            movwf   PSMCCON1

; Insert user code here

            BANKSEL PORTA

m1          movlw   Reload          ; Set TMR0 to reload value
            movwf   TMR0
            CLRF    INTCON          ; Disable interrupts and clear T0IF

            movlw   samples         ; initializes table offset
            movwf   offset

            BSF     INTCON, T0IE    ; Enable TMR0 interrupt
            BSF     INTCON, GIE     ; Enable all interrupts

main        goto    main
            end
```

Listing 3 Signalgenerator WAVE.ASM

Das Includefile ist in der für PICmicros üblichen Tabellentechnik aufgebaut. Einen kleinen Ausschnitt zeigt Listing 4.

```
            radix   dec

            org     0x320
Table
            addwf   PCL,F   ; add offset to pc to
                            ; generate a computed goto
            retlw   45
            retlw   45
            retlw   46
            ...
            retlw   46
            retlw   45
            retlw   45

            radix   hex
```

Listing 4 Stützstellen des zu generierenden Funktionsverlaufs TABLE.INC

2.3. RC-Timer

Der PIC16C782 ist mit zwei Timern ausgestattet. Der bereits verwendete Timer0 ist ein 8-Bit Timer, Timer1 hingegen ein komfortabler 16-Bit Timer.

Mit Hilfe eines externen RC-Glieds und eines internen Komparators kann ein weiterer Timer aufgebaut werden, dessen Triggerschwelle zudem mit dem internen DA-Umsetzer gesteuert werden kann. Hierzu wird das zeitbestimmende RC-Glied an einen Komparatoreingang AN4 und der Ausgang des DA-Umsetzers auf den anderen Eingang des Komparators geführt. Abbildung 13 zeigt den Schaltplan des gesteuerten RC-Timers.

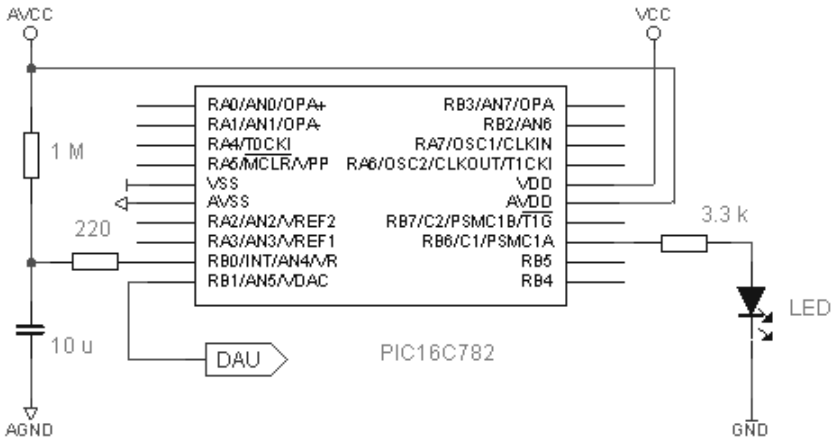

Abbildung 13 Gesteuerter RC-Timer

Über den Widerstand von 1 MΩ lädt sich der 10 µF-Kondensator auf. Die RC-Kombination bildet mit den gewählte Werten eine Zeitkonstante T von 10 s. Beide Elemente sind zeitbestimmend und sollten entsprechend hochwertig sein. Der Kondensator muss einen geringen Leckstrom aufweisen.

Erreicht die Ladespannung die durch den DA-Umsetzer bestimmt Komparatorschwelle, dann schaltet der Komparator um. Zu Kontrollzwecken ist der Ausgang des DA-Umsetzers auch nach außen geführt. Der Ausgang des Analogkomparators C1 treibt eine LED zur Statusanzeige.

Ist die Komparatorschwelle erreicht, dann kann der Ladevorgang abgebrochen und der Kondensator für den nächsten Ladevorgang entladen werden. Hierzu ist das betreffende I/O Pin von Analogeingang auf Digitalausgang mit Lo-Pegel zu schalten. Der Serienwiderstand von 220 Ω dient der Strombegrenzung beim Entladen.

Die Ladezeit bis zum Erreichen der Komparatorschwelle berechnet sich für Werte des DA-Umsetzers zwischen 0 und 255 nach folgender Beziehung:

$$t = -T \cdot \ln(1 - \frac{n}{256}) \qquad \text{für n = 0...255}$$

Abbildung 14 zeigt die resultierenden Zeiten in einer Grafik. Bei der hier verwendeten Zeitkonstante von 10 s lassen sich stabil durchaus Zeiten von bis zu 30 Sekunden erreichen. Bei höheren Komparatorschwellen wird die den Ladevorgang beschreibende e-Funktion sehr flach und es muss mit größeren Abweichungen gerechnet werden.

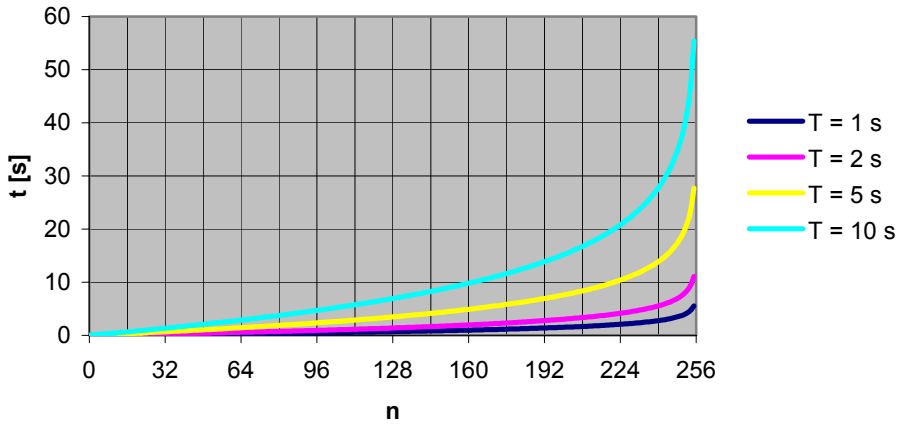

Abbildung 14 Ladezeiten in Abhängigkeit von der Komparatorschwelle

Die Register ANSEL und TRISB sind für die Konfiguration der Pins von PortB und somit für das Umschalten von analogem Eingang zu digitalem Ausgaung und umgekehrt verantwortlich. Abbildung 15 zeigt die Konfiguration von Pin 9 (RB0,...) als analogen Eingang, während Abbildung 16 die Konfiguration als digitalen Ausgang verdeutlicht. Die Belegungen der Register ANSEL und TRISB können so in ein Anwenderprogramm übernommen werden, welches beispielsweise den Komparatorinterrupt auswertet.

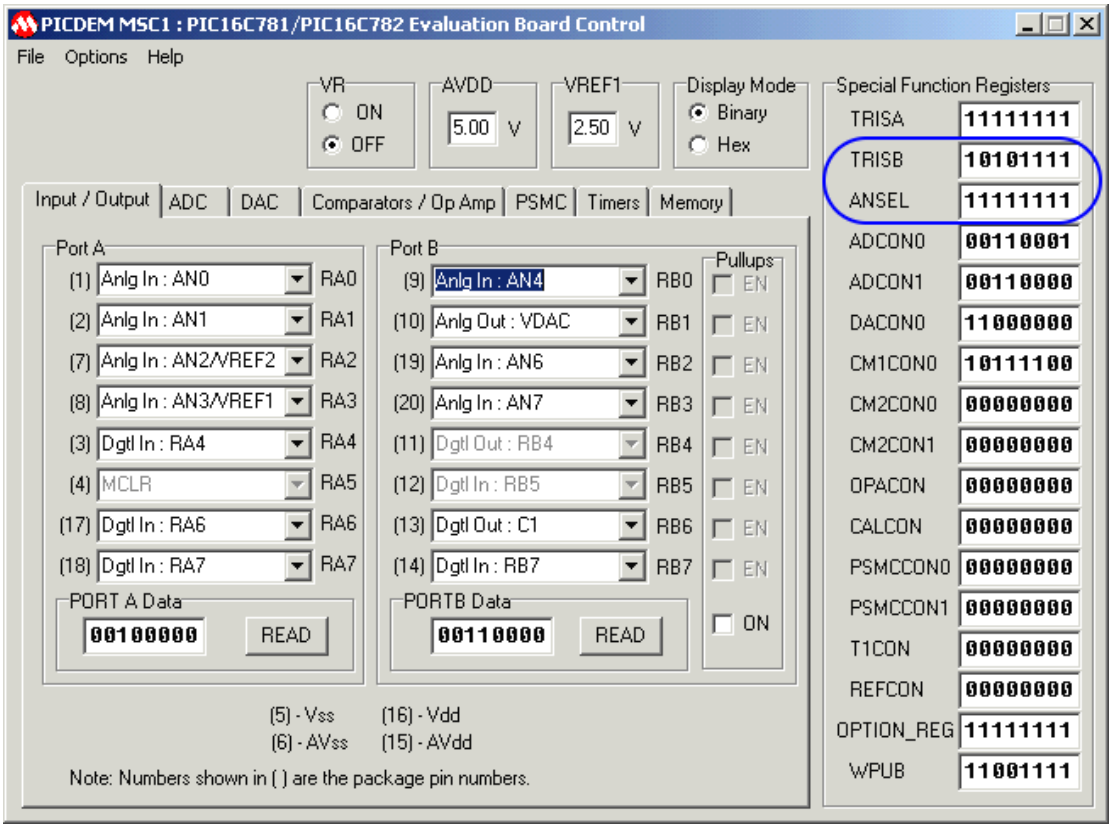

Abbildung 15 Konfiguration von Pin 9 als analogen Eingang

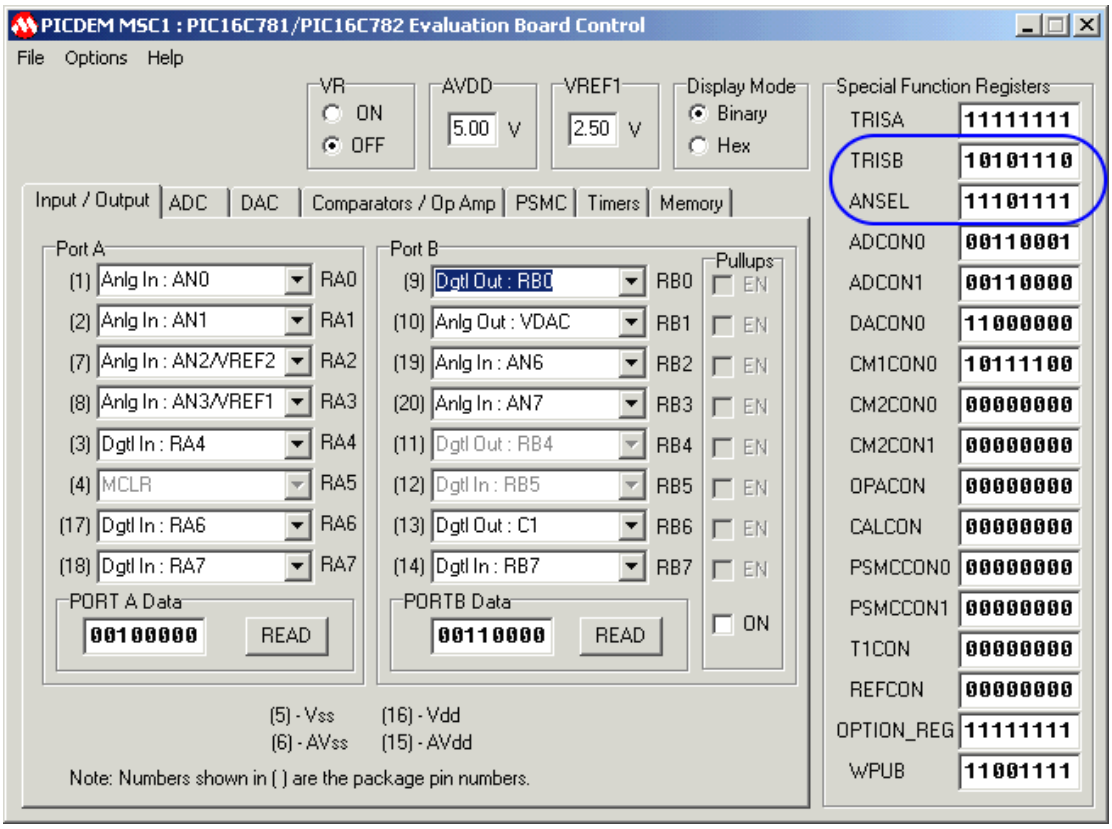

Abbildung 16 Konfiguration von Pin 9 als digitalen Ausgang

3. PICDEM™ MSC1 Evaluationboard

Ohne auf das PICDEM™ MSC1 Evaluationboard näher eingehen zu wollen, soll abschließend in Abbildung 17 dessen Schaltbild gezeigt werden. Die vorgestellten Beispiele wurde alle mit diesem Board getestet, so dass anhand der Schaltbilder ein besseres Nachvollziehen möglich ist.

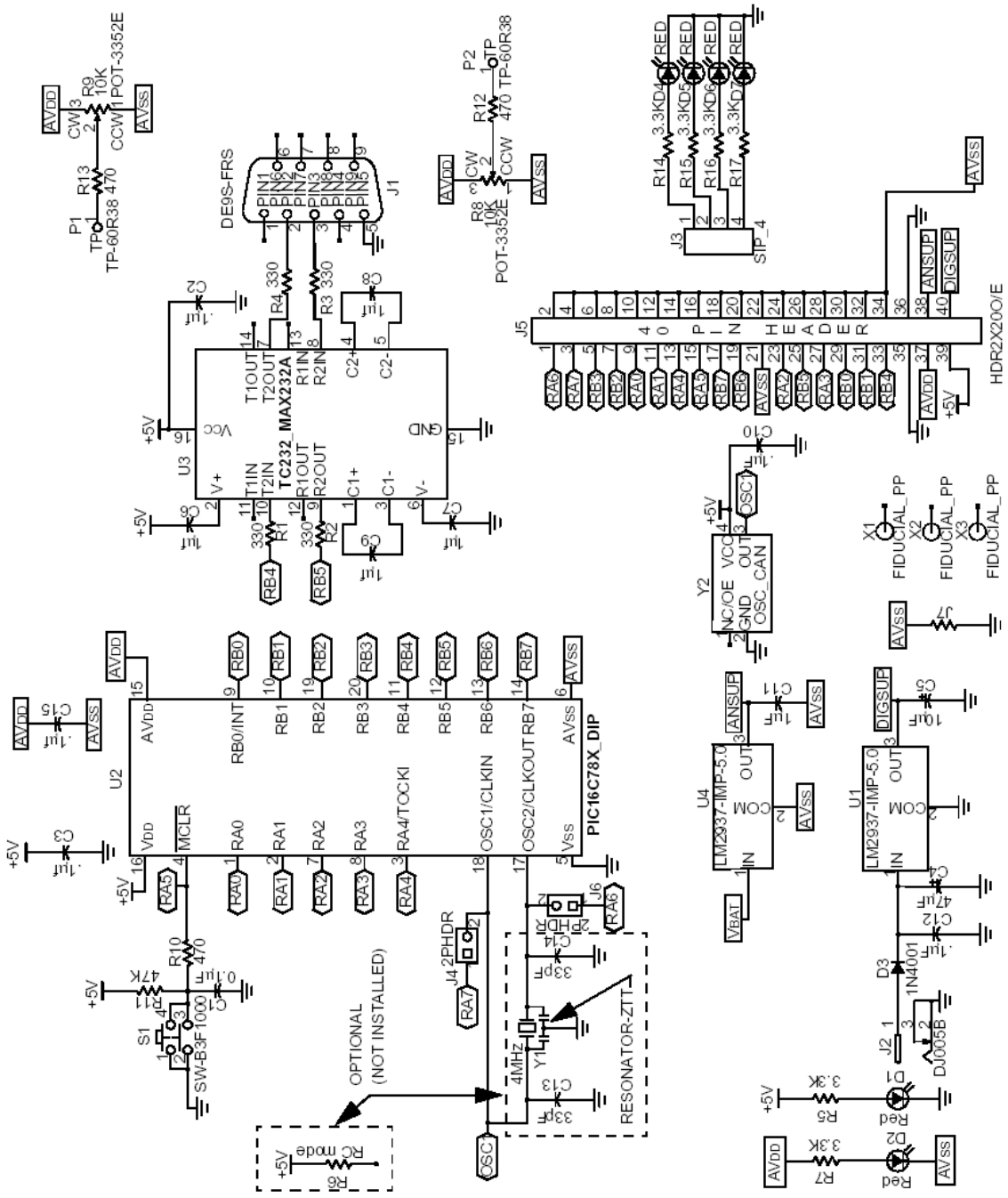

Abbildung 17 Schaltungsdetails PICDEM™ MSC1

4. Schlussbemerkung

Im Beitrag wurden einige Möglichkeiten der analogen Peripherie des PIC16C782 im Zusammenhang mit dem Evaluationboard PICDEM™ MSC1 aufgezeigt. Der Systementwurf wird durch die On-Chip Komponenten wesentlich erleichtert. Alle analogen Komponenten, die bereits auf dem Chip integriert sind, müssen nicht mehr auf einer Platine unter Beachtung von vielfältigen Störeinflüssen (Rauschen, Übersprechen etc.) platziert und verbunden werden. Das Platinenlayout dürfte sich ebenfalls vereinfachen und der Platzbedarf reduzieren.

Der PIC16C782 muss noch konventionell programmiert werden – ein Flashtyp wird von Microchip sicher vorgestellt werden. Zur Zeit kann also das PICDEM™ MCS1 nur zur Konfiguration des Chips mit anschließender automatischer Codegeneration verwendet werden. Das Anwenderprogramm, in welches der generierte Konfigurationscode eingebunden werden kann, ist dann in der jeweils bevorzugten Entwicklungsumgebung zu schreiben. Für die Beispiele hier kam Microchip's MPLAB® zum Einsatz.

5. Links

PIC16C781/782 - Data Sheet

8-Bit CMOS Microcontrollers with A/D, D/A, OPAMP, Comparators and PSMC

http://www.microchip.com/download/lit/pline/picmicro/families/16c78x/41171a.pdf

PICDEM™ MSC1 USER'S GUIDE

www.microchip.com/download/tools/picmicro/demo/pdemmsc/41178a.pdf

AN823 – Analog Design in a Digital World Using Mixed Signal Controllers

http://www.microchip.com/download/appnote/devspec/16c78x/00823a.pdf