Institut für Industrielle Informationstechnik (IIIT)
Universität (TH) Karlsruhe

Hertzstr. 16 / Geb. 06.35
76187 Karlsruhe
Tel.: 0721 608 4521
Fax: 0721 608 4500

Prof. Dr.-Ing. Uwe Kiencke
Prof. Dr.-Ing. habil. K. Dostert

Praktikum: **„Mikrocontroller und digitale Signalprozessoren“**

Versuch 2

Digitale Frequenzsynthese mit dem µC 80C517A

Inhaltsverzeichnis

1 Einleitung

Die digitale Frequenzsynthese beschäftigt sich mit der Erzeugung kontinuierlicher Signale aus diskreten Werten. Diese Aufgabenstellung hat einen durchaus realen Hintergrund. Häufig werden verschiedene elektrische Signale zum Testen von Filtern, zur Übertragung von Informationen und zum Messen von Frequenzgängen benötigt.

Je nach Anwendung sind die Anforderungen an die Signale bezüglich Form und Frequenz sehr unterschiedlich.

Da sich nicht alle Formen und Frequenzen mit angemessenem Aufwand analog erzeugen lassen, wird häufig der Weg über die digitale Frequenzsynthese beschritten. Er bietet einerseits die Möglichkeit, Signale in nahezu beliebiger Form und Frequenz zu erzeugen, andererseits ist die Langzeitstabilität der Signale nicht mehr von sich verändernden Bauteilkoeffizienten abhängig, wie es bei der analogen Erzeugung mit Hilfe von Schwingkreisen der Fall ist.

Zunächst wird ein kleiner Exkurs in die digitale Signalverarbeitung unternommen. Anschließend wird auf die Implementierung der digitalen Frequenzsynthese auf dem Mikrocontrollersystem eingangen.

2 Grundlagen

Die Synthese kontinuierlicher Signale aus diskreten Werten beruht auf der digitalen Signalverarbeitung. Die für das Verständnis notwendigen mathematischen Grundlagen sind die kontinuierliche Fouriertransformation sowie die diskrete Fouriertransformation (DFT) bzw. die schnelle Fouriertransformation (FFT), die sich leicht auf Rechnern implementieren läßt.

Eine umfassende Erläuterung der Theorie der digitalen Signalverarbeitung ist hier nicht möglich, da dies den Rahmen eines Praktikumsversuchs sprengen würde. An dieser Stelle sei deshalb auf die zahlreiche Literatur zu diesem Thema verwiesen. Für die Durchführung des Versuchs werden jedoch keine Kenntnisse in digitaler Signalverarbeitung vorausgesetzt, so daß sich das folgende Kapitel auf die wesentlichen Zusammenhänge, die für das Verständnis der Aufgaben notwendig sind, beschränkt.

Ziel dieses Abschnittes ist es, zu zeigen, wie aus diskreten Werten ein kontinuierliches Signal erzeugt werden kann. Dazu wird, aus didaktischen Gründen, zunächst einmal der umgekehrte Weg beschrieben, also vom kontinuierlichen Signal hin zu den diskreten Werten.

2.1 Zeitbereich

Ausgangspunkt sei ein kontinuierliches Signal, das zu äquidistanten Zeitpunkten abgetastet wird.

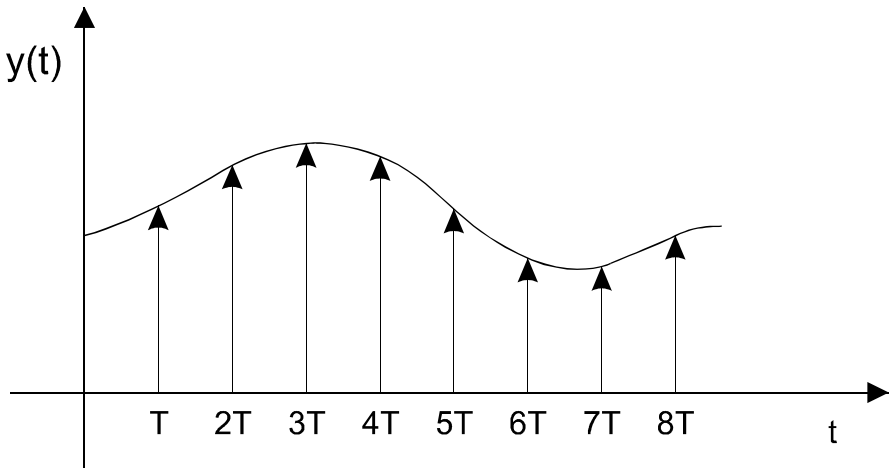


Abbildung 1 Abtastung eines kontinuierlichen Signals zu äquidistanten Zeitpunkten

Anschaulich kann man sich die Funktionswertentnahme so vorstellen: das kontinuierliche Signal wird mit einer Impulsfolge $i_T(t)$ multipliziert.

$$y_g(t) = y(t) \cdot i_T(t) \qquad \text{Gl. 1.1}$$

Dabei ist die Impulsfolge $i_T(t)$ eine Folge von Dirac-Impulsen[1], die einen zeitlichen Abstand von T besitzen.

$$i_T(t) = \sum_{n=-\infty}^{+\infty} \delta(t - nT) \qquad \text{Gl. 1.2}$$

oder in der Schreibweise diskreter Signale:

$$i_T(k) = \sum_{n=-\infty}^{+\infty} \delta(k - n) \qquad \text{Gl. 1.3}$$

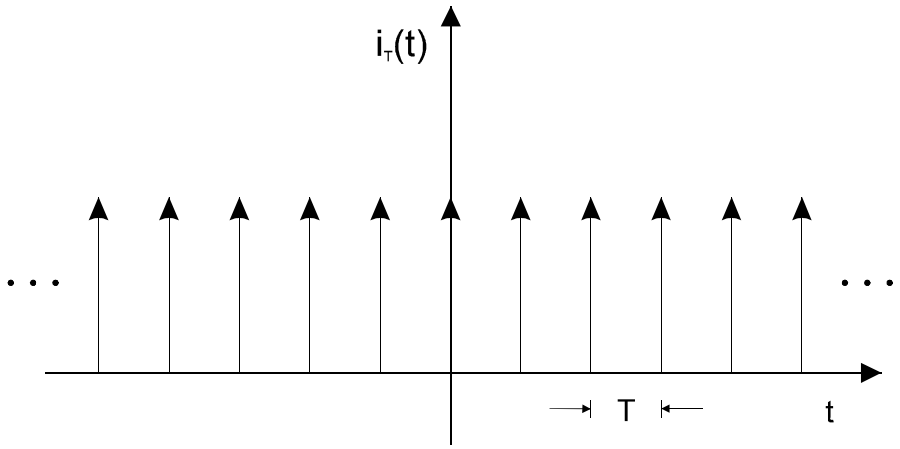


Abbildung 2 Impulsfolge

Hinweis: Für das zeitdiskrete Signal soll folgende Schreibweise verwendet werden:

$$x(t)\big|_{t=nT} = x(nT) := x(n) \qquad \text{Gl. 1.4}$$

wobei T das Abtastintervall und $f_A = 1/T$ die Abtastfrequenz bezeichnet. Der Wert n ist eine ganze Zahl im Bereich $-\infty < n < +\infty$. Da $x(n)$ für ein bestimmtes n eine feste Zahl darstellt, für laufendes n aber eine Folge von Zahlen, wäre für das diskrete Signal die

[1] Der Dirac-Impuls diskreter Systeme weist bei der Definition nicht die Probleme seines kontinuierlichen Pendants – unendlich große Amplitude, unendlich kurze Zeitdauer, Fläche vom Maß eins – auf. In diskreten Systemen lautet die Definition des Dirac-Impulses

$$\delta(k) = \begin{cases} 1 & k = 0 \\ 0 & k \neq 0 \end{cases}$$

Bezeichnung $\{x(n)\}$ angemessener. Zur Vereinfachung der Schreibweise wird im folgenden die Bezeichnung $x(n)$ verwendet. Mit Hilfe der Ausblendeigenschaft der Impulsfolge $\delta(k)$ kann das abgetastete zeitdiskrete Signal $y_g(k)$ in der Form

$$y_g(k) = \sum_{n=-\infty}^{+\infty} y(n) \cdot \delta(k-n) \qquad \text{Gl. 1.5}$$

geschrieben werden. Die Folge $y_g(k)$ besteht also aus einer unendlichen Folge äquidistanter Deltafunktionen, deren Gewichte den Funktionswerten von $y(n)$ bei $k = n$ entsprechen. Hierbei bezeichnet $y(n)$ einen einzigen Amplitudenwert, $y_g(k)$ dagegen das gesamte Signal.

2.2 Frequenzbereich

Transformiert man Gl. 1.5 mit Hilfe der Fouriertransformation in den Frequenzbereich, so erkennt man, daß das Frequenzspektrum des abgetasteten Signals um die Frequenz $f_A = 1/T$ periodisch ist.

$$\Im\{y_g(k)\} = Y_g(k) = \sum_{k=-\infty}^{+\infty} y(k) \cdot e^{-j2\pi kf/f_A} = f_A \cdot \sum_{k=-\infty}^{+\infty} Y(f - kf_A) \qquad \text{Gl. 1.6}$$

Die Wahl der Abtastfrequenz f_A ist für die weitere Verwendung der Signale von entscheidender Bedeutung. Wird das Abtastintervall T größer gewählt, verkleinert sich der Abstand $f_A = 1/T$ der einzelnen Bänder im Frequenzbereich. Das kann dazu führen, daß sich diese Bänder überlappen. Diese Verzerrung der Fouriertransformierten des Abtastsignals wird als Aliasing bezeichnet und entsteht grundsätzlich, wenn eine Zeitfunktion $y(t)$ nicht mit einer ausreichend hohen Frequenz abgetastet wird. Zur Klärung des Sachverhalts diene ein bandbegrenztes Signal $Y(f)$. Die höchste auftretende Frequenz sei $B/2$, wobei B die Bandbreite des Signals sei.

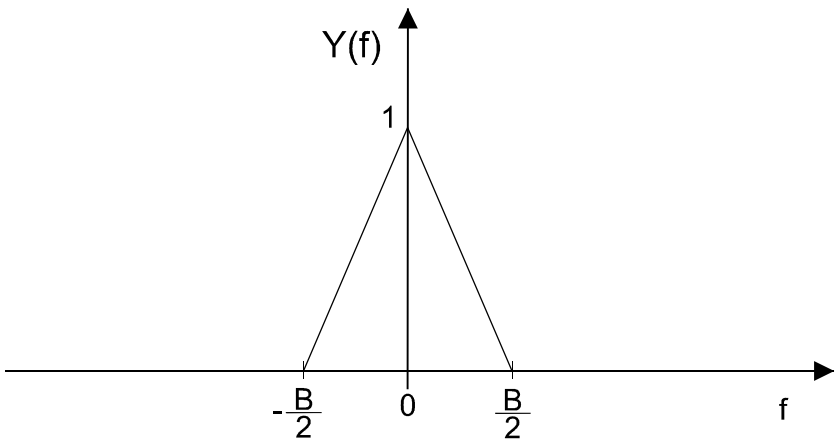


Abbildung 3 Spektrum eines bandbegrenzten kontinuierlichen Signals

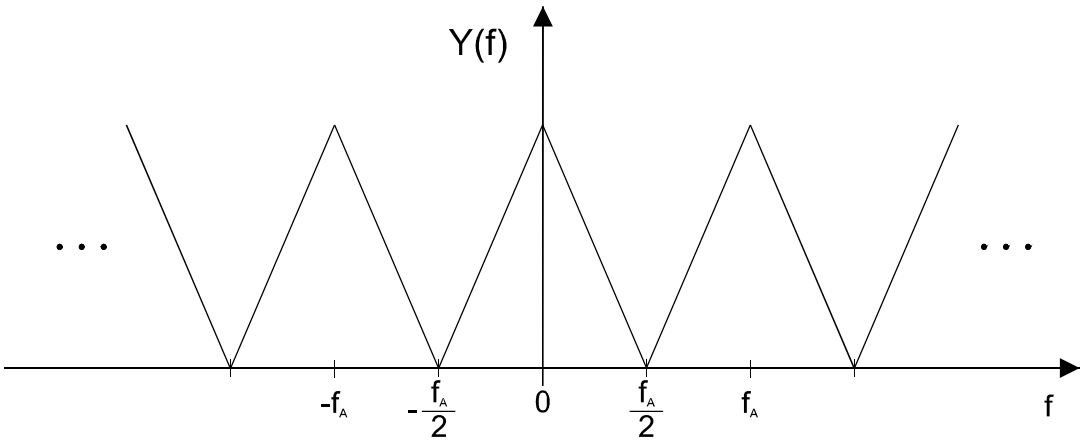


Abbildung 4 Spektrum des abgetasteten Signals. Die Abtastfrequenz ist doppelt so groß wie die höchste Signalfrequenz

Das Spektrum des abgetasteten Signals ist um die Abtastfrequenz f_A periodisch. In Abbildung 4 ist die Abtastfrequenz doppelt so groß wie die höchste Signalfrequenz. Dadurch grenzen die einzelnen Bänder aneinander. Sie überlappen sich nicht.

Abtastung mit $f_A > B$

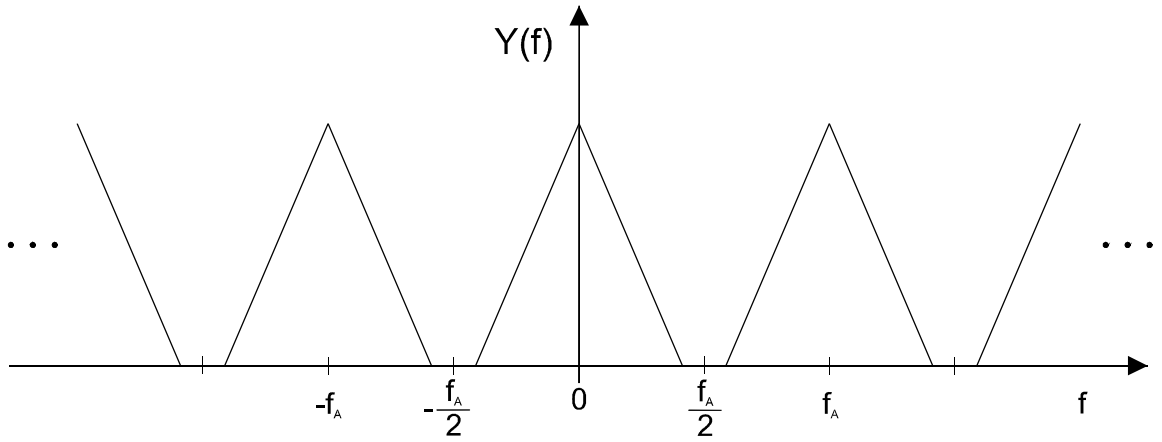


Abbildung 5 Die Abtastfrequenz ist mehr als doppelt so groß wie die höchste Signalfrequenz

Die einzelnen Bänder sind durch Abstände voneinander getrennt. Sie überlappen sich nicht. Ist die Abtastfrequenz kleiner als das Doppelte der Signalfrequenz, so bewegen sich die Frequenzbänder aufeinander zu.

Abtastung mit $f_A < B$

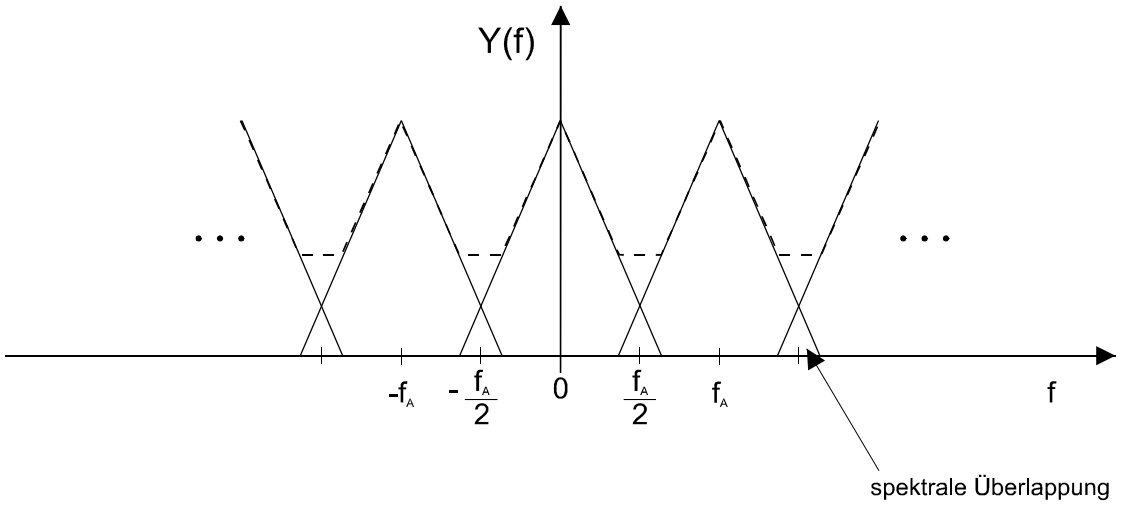


Abbildung 6 Die Abtastfrequenz ist kleiner als das Doppelte der höchsten Signalfrequenz

Die punktierte Linie ergibt sich aus der Superposition der Bänder. Es kommt zu spektralen Überschneidungen der Frequenzbänder (Aliasing), die zu Verfälschungen im interessierenden Spektralbereich führen.

Zur Rekonstruktion wird das Signal $y_g(t)$ durch einen Tiefpaß mit der Grenzfrequenz $B/2$ gefiltert. Die Frequenzen oberhalb der Grenzfrequenz werden abgeschnitten. Zurück bleibt das Signal im Bereich $f = -B/2$, ... , $f = +B/2$, das bei Erfüllung der Abtastbedingung $f_A \geq B$ bis auf einen hier nicht interessierenden Amplitudenfaktor dem Originalsignal entspricht. In Abbildung 6 ist durch das Aliasing das kontinuierliche Signal aus dem diskreten Signal jedoch nicht mehr rekonstruierbar.

2.3 Das Abtasttheorem

Das Abtasttheorem besagt folgendes:

> Ist $y(t)$ ein mit $B/2$ bandbegrenztes Signal und wird $y(t)$ mit einer Abtastfrequenz $f_A \geq B$ abgetastet, so läßt sich $y(t)$ durch die Abtastwerte eindeutig rekonstruieren. Man sagt: „Das Signal ist auf das **Nyquist-Band** begrenzt“.

Wurde das Abtasttheorem bei der Abtastung eines Signals erfüllt, so kann aus den diskreten Werten das abgetastete kontinuierliche Signal vollständig wiederhergestellt werden.

Die Frage, die sich nun stellt, ist: Wie kann die Abtastfrequenz gewählt werden, wenn das Spektrum des Signals nicht bekannt ist?

Da Signale praktisch nie wirklich bandbegrenzt sind, ist die Erhöhung der Abtastfrequenz bis an die Grenze des technisch Machbaren sicherlich keine Lösung. Es gibt zwei Möglichkeiten, mit dem Aliasing-Effekt umzugehen:

1. Man nimmt den Aliasing-Fehler als gegeben hin und berücksichtigt den dabei entstehenden Fehler in der weiteren Verarbeitung der Werte.
2. Man begrenzt das Signal mit Hilfe eines Tiefpasses auf das Nyquist-Band, um spektrale Überschneidungen zu vermeiden. Dieser Tiefpaß wird auch als Anti-Aliasing-Filter bezeichnet. Das begrenzte Signal kann abgetastet und vollständig mit Hilfe eines Rekonstruktionsfilters reproduziert werden. Der Fehler entsteht durch die Begrenzung des Signals auf das Nyquist-Band.

Wird das Abtasttheorem beachtet, so stellt die Funktionsreihe aus Gl. 1.5 das Signal dar. Nun läuft aber der Summationsindex in Gl. 1.5 über den gesamten Zeitstrang von $-\infty$ bis $+\infty$, denn über die Dauer des Zeitsignals ist keine Aussage gemacht. Das bringt für die Verarbeitung folgende Schwierigkeiten mit sich:

- Es können nur endlich lange Signale beobachtet werden. Eine Beobachtungszeit T_0 wird festgelegt.
- Die Wertemenge ist durch die Darstellung einer endlichen Anzahl von Bits beschränkt. Die Summation der diskreten Fouriertransformation muß auf endlich viele Summanden begrenzt werden.

Durch die Einführung einer Beobachtungszeit T_0 wird ein Zeitfenster über das Signal gelegt, innerhalb dessen Funktionswerte zu äquidistanten Zeitabständen T entnommen werden. Am Ende einer Beobachtungszeit liegt eine Stichprobe von $N = T_0 / T$ Elementen vor.

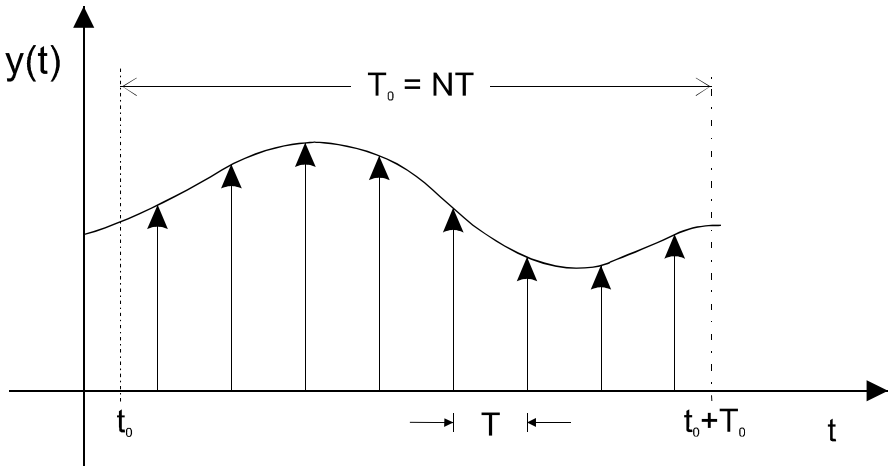


Abbildung 7 Größen der Signalerfassung im Zeitbereich

Wird der Kehrwert der Beobachtungszeit T_0 als Beobachtungsfrequenz F_0 eingeführt, so lauten die Zusammenhänge im einzelnen:

N	Umfang der Stichprobe
$T_0 = N \cdot T$	Beobachtungszeit
$F_0 = 1/T_0 = 1/(N \cdot T) = (1/N) \cdot F$	Beobachtungsfrequenz
$T = 1/N \cdot T_0$	Abtastzeit
$F = N \cdot F_0 = 1/T$	Abtastfrequenz

Wird Abbildung 7 in den Frequenzbereich transformiert, so liegen folgende Verhältnisse vor:

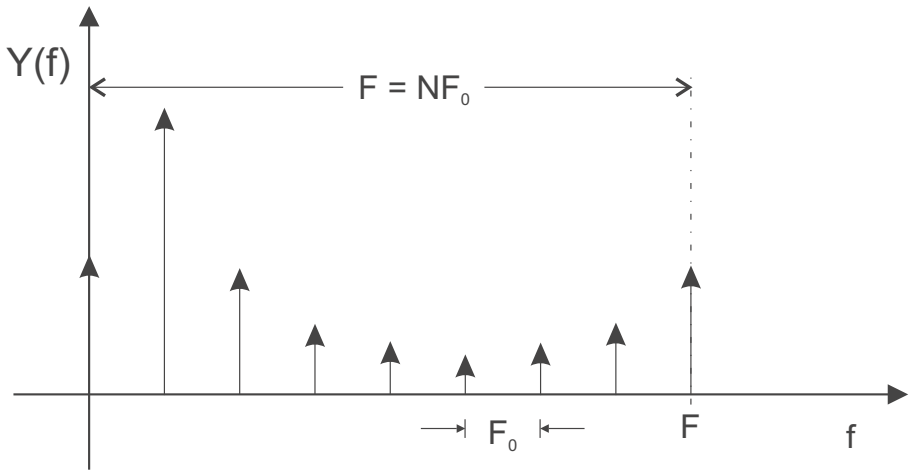


Abbildung 8 Größen der Signalerfassung im Frequenzbereich

Das zunächst analoge, kontinuierliche Signal wurde über eine Impulsfolge diskretisiert. Das diskretisierte Signal besitzt ein periodisches Spektrum. Wird mit einer zu niedrigen Frequenz abgetastet, kann dies zu einem Aliasing-Effekt führen, der eine korrekte Rekonstruktion des ursprünglichen Signals unmöglich macht. Somit ist auf die Wahl der Abtastfrequenz größte Sorgfalt zu legen.

Im Versuch selbst werden zunächst allerdings keine Signale abgetastet, sondern es sollen mit Hilfe von digitalen Werten analoge kontinuierliche Signale erzeugt werden, wobei die Rekonstruktion mittels eines D/A-Wandlers und eines Filters vorgenommen wird.

Ein D/A-Wandler interpoliert aus diskreten Werten ein kontinuierliches Signal, das aber aufgrund seiner Periodizität Frequenzanteile enthält, die in dem ursprünglichen Signal nicht enthalten waren. Sie zu beseitigen ist die Aufgabe eines nachgeschalteten Tiefpasses, dessen Grenzfrequenz bei der halben Abtastfrequenz liegt.

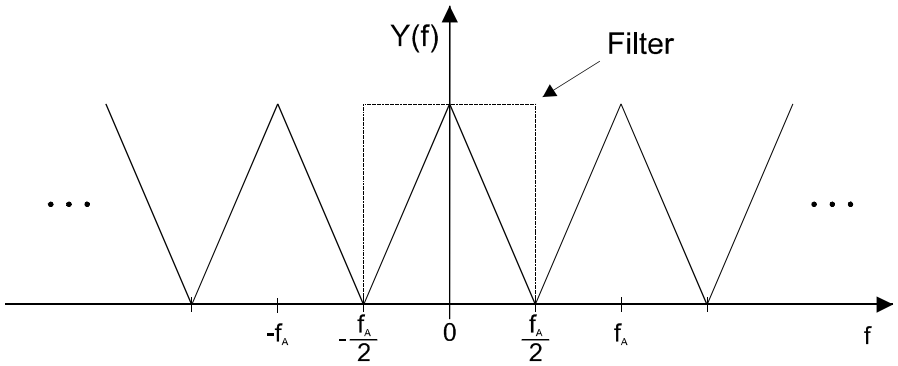


Abbildung 9 Ausgangssignal des D/A-Wandlers

So weit die Theorie. Die reale Rekonstruktion ist Gegenstand des nächsten Abschnittes.

Mehr zum Thema der digitalen Signalverarbeitung findet der interessierte Leser zum Beispiel in [KaKr89], [Föll86] und [Kron91]. In [Dost97] wird speziell auf die digitale Frequenzsynthese eingegangen.

3 Frequenzsynthese mit dem Mikrocontroller

3.1 Ausgabe der Werte über den D/A-Wandler

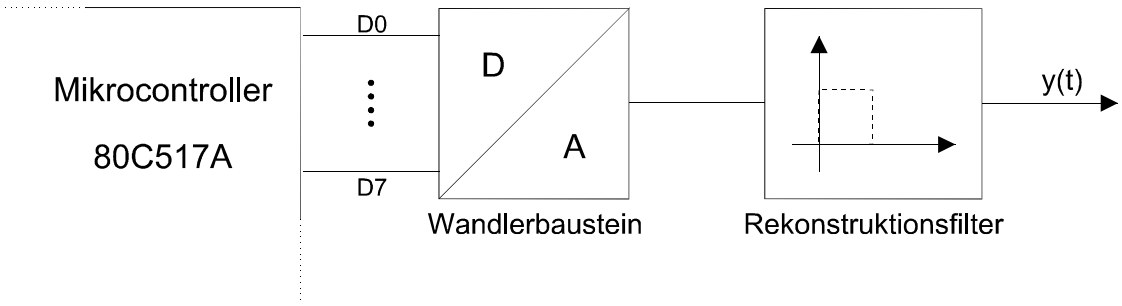


Abbildung 10 Übergang von diskreten Werten zum kontinuierlichen Signal

Obige Abbildung veranschaulicht das Prinzip der Umwandlung diskreter Werte in ein kontinuierliches Signal. Der Mikrocontroller sendet die diskreten Werte über den Daten- und Adreßbus an den D/A-Wandler. Das Ausgangssignal des D/A-Wandlers wird über einen Tiefpaß gefiltert, damit sich das rekonstruierte Signal ergibt.

Es gibt prinzipiell drei Möglichkeiten, periodische Signale auf einem Mikrocontroller auszugeben. Bei allen drei Verfahren ist die Anzahl der möglichen Werte begrenzt. Auch bei der Berechnung der Werte durch den Mikrocontroller ist die mögliche Anzahl der Werte durch den Algorithmus vorgegeben, so daß sich die Werte zyklisch wiederholen. Die dadurch generierten kontinuierlichen Signale sind periodisch. Dies bedingt einen – wie auch immer gearteten – Sprung des Programms vom letzten zum ersten Wert, bei dem es zu keiner zeitlichen Verzögerung der Ausgabe kommen darf. Daraus resultiert, daß die Zeitspanne zwischen zwei Ausgaben immer gleich lang sein muß, unabhängig von der Lage der Werte im Speicher. Um eine Aussage über die Zeitspanne machen zu können, werden in den folgenden Beispielen dieses Abschnitts die Zyklen summiert, die zur Ausgabe eines einzelnen Bytes notwendig sind. Der Mikrocontroller 80C517A benötigt für einen Zyklus zwölf Oszillatorperioden. Durch die Verwendung eines 18MHz-Quarzes entspricht dies einer Zyklusdauer von $0,\overline{6}\,\mu s$.

1.) Werte liegen innerhalb des Mikrocontrollers als Konstanten vor

Bei diesem Verfahren werden die Werte in das Programm eingebunden und von dort direkt an die Adresse des D/A-Wandlers geschrieben. Da das Signal periodisch sein soll, muß

nach der Ausgabe des letzten Wertes wieder an den Anfang gesprungen werden. Dieser Sprung muß so ausgeführt werden, daß die Ausgabe der Werte an den D/A-Wandler periodisch bleibt, d.h. der Weg vom Ende der Ausgabe zurück zum Anfang muß zeitlich so bemessen sein, daß im Signal kein unerwünschter Phasensprung entsteht.

Im folgenden Beispiel werden die Werte 80h, F9h, CBh, 34h und 06h zyklisch über die Adresse FF00h ausgegeben.

```
                              Anzahl der Zyklen
        mov  dptr,#0FF00h
ANF:
        mov  A,#80h                   1
        movx @dptr,A                  2
        nop                           1
        nop                           1
        nop                           1
        mov  A,#0F9h                  1
        movx @dptr,A                  2
        nop                           1
        nop                           1
        nop                           1
;-------- Start einer Ausgabe --------
        mov  A,#0CBh                  1
        movx @dptr,A                  2
        nop                           1
        nop                           1
        nop                           1
;-------- Ende einer Ausgabe --------- Summe = 6 Zyklen
        mov  A,#34h                   1
        movx @dptr,A                  2
        nop                           1
        nop                           1
        nop                           1
        mov  A,#06h                   1
        movx @dptr,A                  2
        nop                           1
        jmp  ANF                      2
```

Beispiel 1 Zyklische Ausgabe von als Konstanten gespeicherten Werten

In Beispiel 1 wird in jedem sechsten Zyklus ein Wert über den Adreßbus zum D/A-Wandler ausgegeben. Damit ergibt sich im obigen Beispiel, daß alle 4 µs ein neuer Wert auf den D/A-Wandler geschaltet wird. Dies entspricht einer Tastfrequenz von 250 kHz. Durch das Abtasttheorem wird die maximal generierbare Frequenz f_s auf $f_s = f_A / 2 = 125$ kHz begrenzt.

Die No-Operation-Befehle (NOPs) zwischen den einzelnen Ausgaben sind notwendig, um eine Phasenverschiebung beim kontinuierlichen Signal zu verhindern. Würden die Werte unmittelbar hintereinander ausgegeben werden, so wäre zwar eine höhere Tastfrequenz realisiert, aber durch den Rücksprung an die Adresse ANF wäre die zyklische Ausgabe unterbrochen.

Das Verfahren hat mehrere Nachteile. Zum einen muß für jedes auszugebende Signal ein nahezu gleiches Programmteil geschrieben werden, zum anderen werden die Quellcodes dadurch sehr lang. Diese Nachteile vermeidet das nächste Verfahren.

2.) Ausgabe von Tabellenwerten

Die Werte werden in Form einer Tabelle im Speicher abgelegt. Das Auslesen der Tabelle und das Ausgeben der Werte über den Adreßbus kann durch verschiedene Algorithmen erfolgen, von denen einer exemplarisch vorgestellt werden soll.

Für die Arbeit mit Tabellen bietet sich unter anderem der indirekt indizierte Befehl

```
movc  A,@A + Basisadresse
```

an. Die Basisadresse ist ein 16bit-Wert. Das erzwingt die Verwendung des DPTR-Registers. Die Basisadresse entspricht der niedrigsten Tabellenadresse. Der Inhalt des Akkus wird zur Basisadresse hinzuaddiert und die Summe als Adresse interpretiert. Der Inhalt der Adresse wird in den Akku geladen. Die Ausgabe an den D/A-Wandler erfolgt dann über die Befehle

```
mov   dptr,#0FF00h
movx  @dptr,A
```

Wendet man diese Schritte iterativ an, kann durch wiederholtes Inkrementieren des Akkus eine ganze Tabelle ausgelesen werden. Das folgende Beispiel demonstriert den Sachverhalt (es werden die gleichen Werte wie im vorangegangenen Beispiel ausgegeben):

```
                                Anzahl der Zyklen
mov  R0,#05h
;-------- Start einer Ausgabe --------
AUSGABE:
    nop                          1
    nop                          1
    nop                          1
ANF:
    nop                          1
    nop                          1
    nop                          1
    nop                          1
    nop                          1
    mov  A,R0                    1
```

```
    dec  A                      1
    mov  dptr,#TAB              2
    movc A,@A+dptr              2
    mov  dptr,#0FF00h           2
    movx @dptr,A                2
    djnz R0,AUSGABE             2
;-------- Ende einer Ausgabe ---------  Summe = 20 Zyklen
    mov R0,#05h                 1
    jmp ANF                     2
TAB:
    DB 06h
    DB 34h
    DB 0CBh
    DB 0F9h
    DB 80h
```

Beispiel 2 Auslesen einer Tabelle

Die Tabellenausgabe benötigt *einen* Code für alle im Speicher abgelegten Tabellen. Es muß nur der DPTR-Zeiger auf den Anfang der gewünschten Tabelle gesetzt und in R0 die Anzahl der auszugebenden Werte angegeben werden. Das Programm wird dadurch kurz, die Speicheraufteilung übersichtlich.

Um einen Wert auszugeben, werden 20 Zyklen benötigt. Das entspricht einer Tastfrequenz von 75 kHz. Die dadurch maximal generierbare Signalfrequenz ist somit auf 37,5 kHz beschränkt.

Die Verfahren 1.) und 2.) haben eines gemeinsam: Die Werte zur Darstellung der Signale müssen vor ihrer Verwendung errechnet und abgespeichert werden. Falls aber der Algorithmus einfach zu implementieren bzw. das Signal sehr niederfrequent ist, kann die Berechnung vom Mikrocontroller in Echtzeit durchgeführt werden.

3.) Berechnung der Werte in Echtzeit

Als einfaches Beispiel der Berechnung diene eine Rampenfunktion. Der Akkumulator wird auf 00h gesetzt und über den Adreßbus ausgegeben. Anschließend wird der Akku inkrementiert und erneut ausgegeben. Führt man diese Schritte hintereinander aus, so ändert sich der auszugebende Wert von 00h bis FFh. Das entspricht einer langsam ansteigenden Funktion mit steilem Abfall beim Übergang von FFh zu 00h.

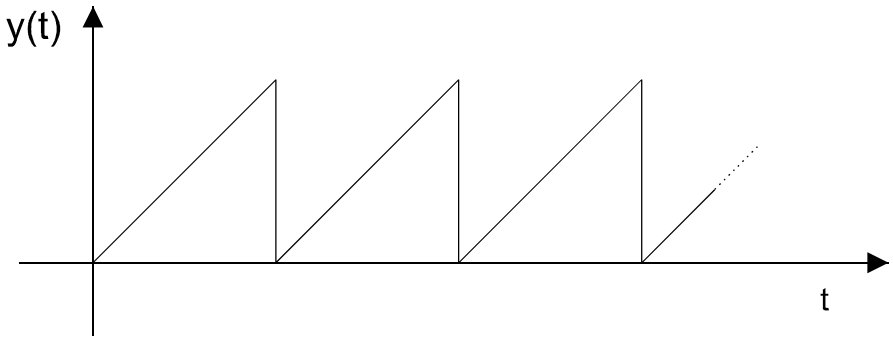


Abbildung 11 Rampen- oder Sägezahnfunktion

Das Programm könnte so aussehen:

```
ANF:
    mov A,#00h                  1
    mov dptr,#0FF00h            2
;---- Beginn der Ausgabeschleife -----
AUSGABE:
    movx @dptr,A                2
    inc A                       1
    jmp AUSGABE                 2
;----- Ende der Ausgabeschleife ------
```

Beispiel 3 Berechnung der Werte in Echtzeit

Ist der Algorithmus zur Berechnung der Werte komplizierter, so steigt die Anzahl der benötigten Zyklen stark an, während die höchste noch generierbare Signalfrequenz absinkt.

Bei den Ausgaberoutinen 1.) und 2.) wurden die diskreten Werte als gegeben vorausgesetzt. Wie man zu den Werten gelangen kann, beschreibt der nächste Abschnitt.

3.2 Errechnen der digitalen Werte

Um die digitalen Werte zu erhalten, muß nicht erst ein analoges Signal abgetastet werden. Die Werte können selbstverständlich numerisch ermittelt werden.

Die Überlegung dazu ist folgende:

Um eine zyklische Ausgabe zu erreichen, müssen sich die Werte früher oder später wiederholen. Wieviele Perioden bzw. Werte notwendig sind, hängt vom Verhältnis der zu realisierenden Frequenz f_S zur Abtastfrequenz f_A ab.

$$\frac{f_A}{f_S} = V \qquad \text{Gl. 1.7}$$

Existiert ein Verhältnis

$$N = P \cdot V \qquad \text{Gl. 1.8}$$

wobei N für die Anzahl der Werte und P für die Anzahl der Perioden steht, so ist eine Synthese möglich, vorausgesetzt, daß N und P aus dem Bereich der natürlichen Zahlen stammen. Es ist leicht einzusehen, daß eine nicht ganzzahlige Anzahl von Werten nicht realisierbar ist.

Ist die Anzahl P der Perioden nicht ganzzahlig, so kann das Signal nicht periodisch wiederholt werden.

Die Wahl der Quantisierung geht ebenfalls in die Berechnung ein. Dazu wird die Nullinie des gewünschten Funktionsverlaufs in die Mitte des Zahlenbereichs gelegt (bei einer 8bit-Quantisierung also auf 80h). Somit errechnet sich beispielsweise eine Sinusfunktion folgendermaßen:

$$A(i) = 80h * \left[1 + \sin(2\pi \frac{f_S}{f_A} * i) \right] \qquad \text{mit} \quad i = 0, .. ,N-1 \qquad \text{Gl. 1.9}$$

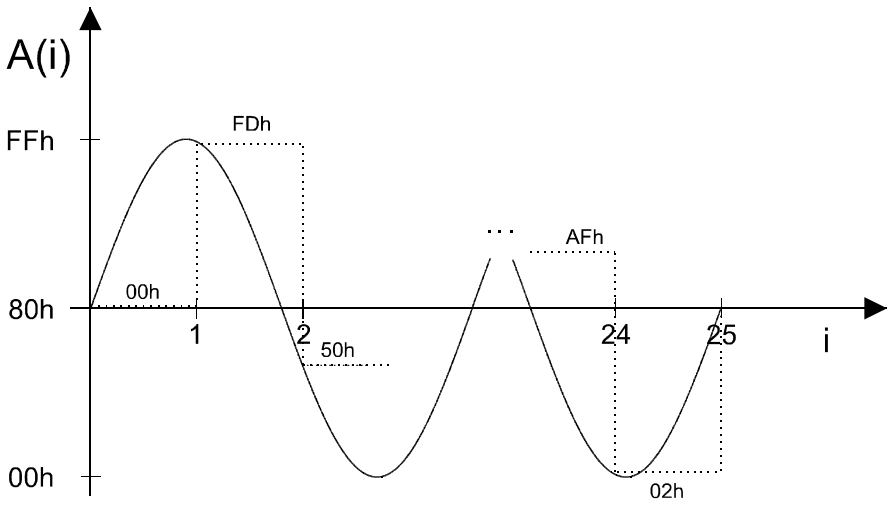


Abbildung 12 Quantisierungsbeispiel

4 Praktikumsversuch

Für den Versuch zur digitalen Frequenzsynthese mit dem Mikrocontroller 80C517A steht am Versuchstag eine komplette Entwicklungsumgebung bereit. Die Software wird auf einem PC mit Hilfe des Entwicklungstools Proview erstellt. Im folgenden wird der Versuchsaufbau beschrieben.

4.1 Versuchsaufbau

Das Gesamtsystem besteht aus einem PC und einem Mikrocontrollerboard, auf dem sich ein D/A-Wandler für die Realisierung der digitalen Frequenzsynthese befindet. Die vom Mikrocontroller generierten Werte werden über den D/A-Wandler ausgegeben. Das Ausgangssignal des Wandlers gelangt auf ein Rekonstruktionsfilter. Mit Hilfe der Flashtools kann die vom Compiler generierte Intel-Hex-Datei über die RS232-Schnittstelle in den Flashspeicher des Mikrocontrollers geladen werden.

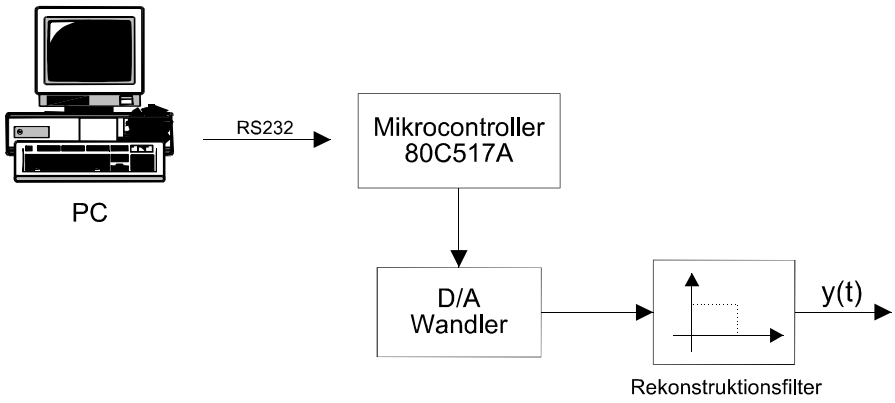


Abbildung 13 Versuchsaufbau

4.2 Der D/A-Wandler

Für die Ausgabe der Signale für die digitale Frequenzsynthese ist ein D/A-Wandler an den 8bit-Datenbus angeschlossen. Er besitzt eine Wandlungszeit von ca. 2µs, d.h. dem Wandler kann alle 2µs ein neuer Wert übergeben werden. Der Wandler wird angesprochen, indem ein 8bit-Wert an die Adresse FF00h des externen Datenspeichers geschrieben wird. An den D/A-Wandlerausgang können zwei Tiefpaßfilter angeschlossen werden, die unterschiedliche Grenzfrequenzen besitzen. Das Ausgangssignal des Wandlers und die

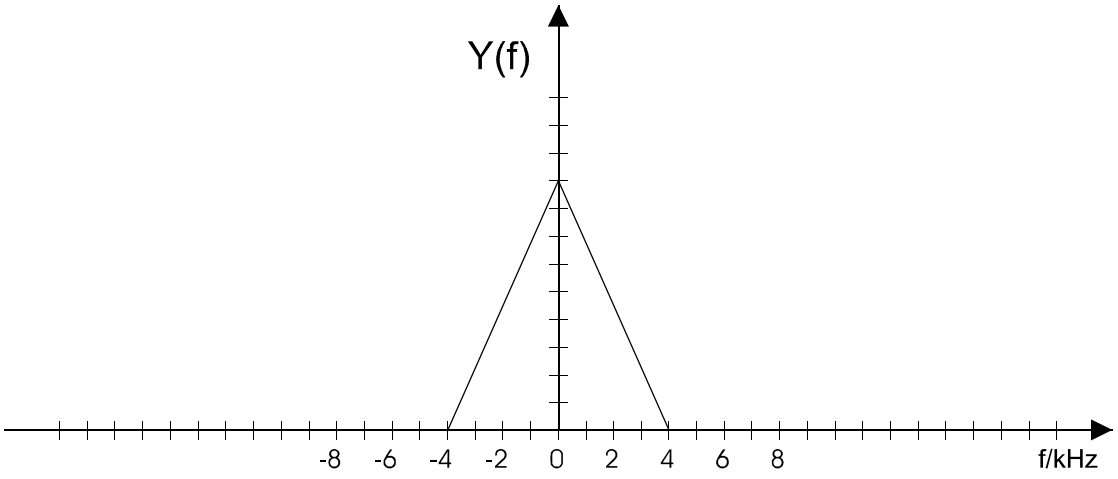
Ausgangssignale der beiden Filter können an den BNC-Buchsen der Frontplatte abgegriffen werden. Die Filter 2.Ordnung besitzen Grenzfrequenzen von 10kHz bzw. 100kHz und dienen der Rekonstruktion des Signals.

5 Aufgaben

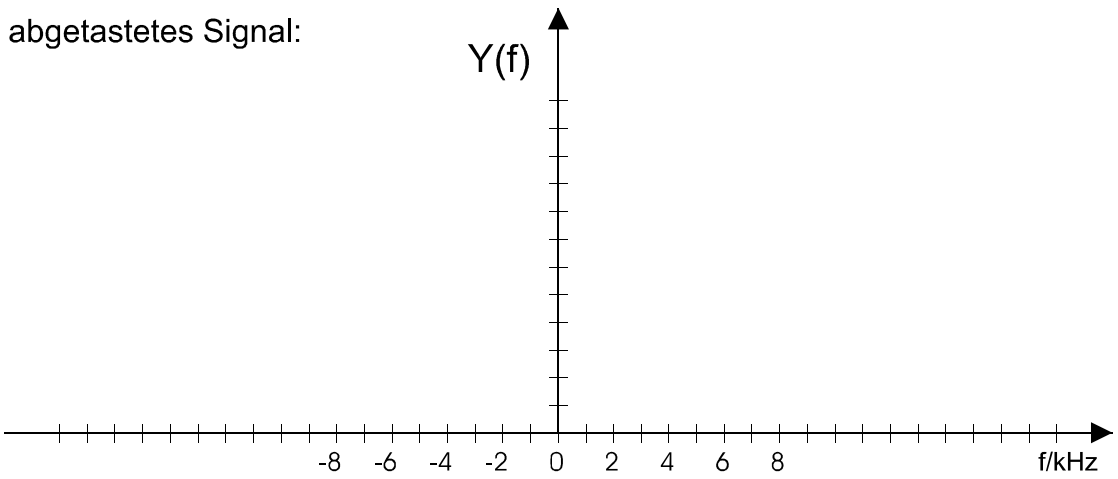
In diesem Kapitel werden einige Aufgaben gestellt, die zum Verständnis und zur Vertiefung des Stoffes dienen sollen. Sie sollten nach Möglichkeit vor dem Versuchstag bearbeitet werden. Die Aufgaben 1-3 dienen der Vorbereitung, während die Aufgaben 4-6 am Versuchstag selbst zu erledigen sind. Die Programme sollen in der Programmiersprache Assembler erstellt werden. Für jeden Aufgabenteil ist ein Projekt vorhanden, in dem die notwendigen Einstellungen zur Codegenerierung durchgeführt wurden. Bei der Bearbeitung der Aufgaben öffnen Sie diese Projekte und fügen ihnen Ihre eigenen Module hinzu. Die notwendigen Schritte der Projekterstellung sind in den Unterlagen „Einführung in den Mikrocontroller 80C517A“ anhand eines Beispielprojekts erläutert. Zur Übung sollten Sie diese Schritte vor der Bearbeitung der Aufgaben durchführen.

Aufgabe 1

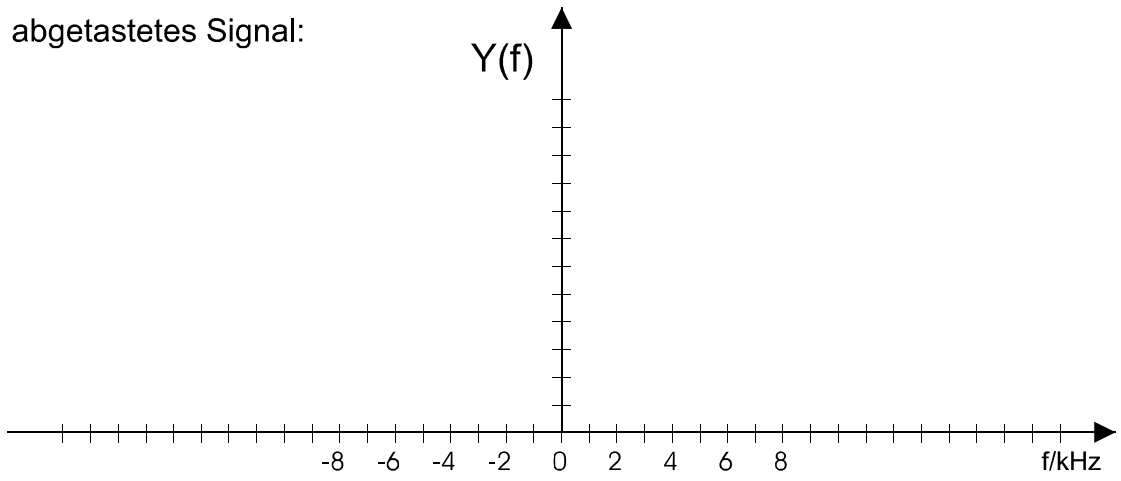
a) Ein Signal mit folgendem Spektrum werde mit einer Frequenz von 10 kHz abgetastet. Wie sieht das Frequenzspektrum des abgetasteten Signals aus?



abgetastetes Signal:



b) Nun wird das Signal aus Aufgabe 1a.) nur noch mit einer Frequenz von 6 kHz abgetastet. Wie sieht jetzt das Spektrum des abgetasteten Signals aus?



c) Wie nennt man den Effekt, der sich in 1b.) bemerkbar macht?

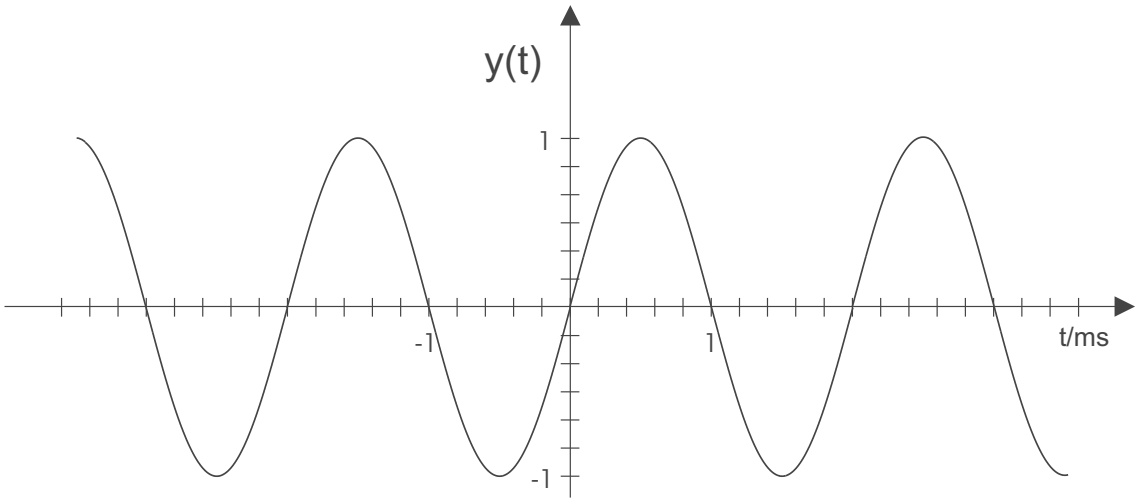
d) Was besagt das Abtasttheorem?

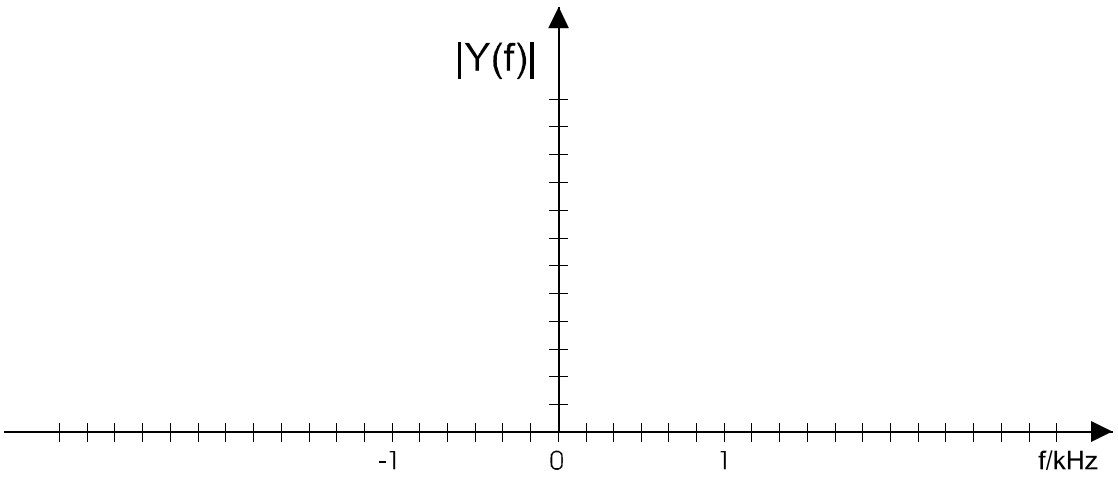
e) Genügt ein D/A-Wandler zur Rekonstruktion eines Signals? Begründen Sie Ihre Antwort.

Aufgabe 2

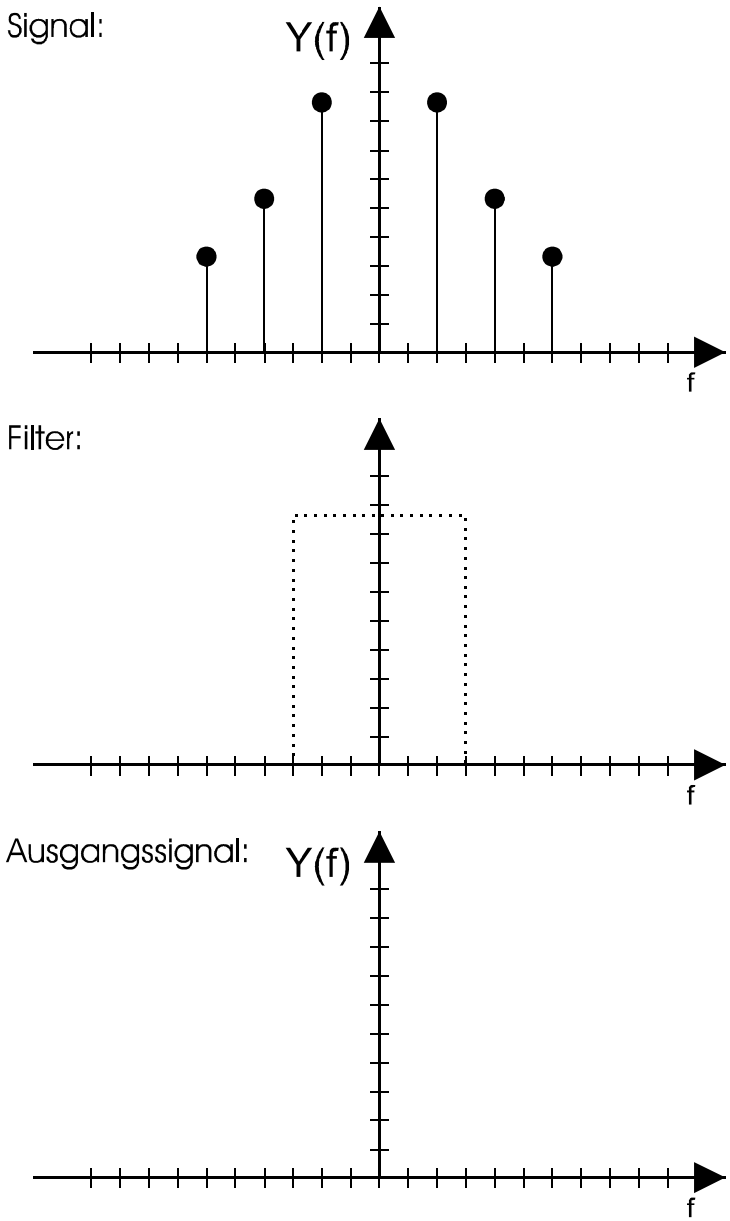
a) Wie sieht das Spektrum des folgenden sinusförmigen Zeitsignals aus?

Signal y(t):





b) Ein Signal mit folgendem Frequenzgang durchlaufe ein Filter. Wie sieht das Ausgangssignal des Filters aus?



c) Ein Signal soll generiert werden. Es wird alle 5µs ein Wert an den D/A-Wandler ausgegeben. Welcher Abtastfrequenz entspricht dies und welche Frequenz kann maximal erzeugt werden?

Aufgabe 3

Ein sinusförmiges Signal der Frequenz 10 kHz und einer Abtastfrequenz von 100 kHz soll mit 8bit quantisiert und ausgegeben werden.

a) Wie lautet die Formel zur Berechnung der diskreten Werte?

b) Wieviele Perioden und Werte müssen errechnet und abgespeichert werden?

Anzahl der Perioden:

Anzahl der Werte:

c) Errechnen Sie die Werte und tragen Sie sie in die Tabelle ein.

Index	0	1	2	3	4	5	6	7	8	9	10	11
A/hex												

Es soll ein sinusförmiges Signal mit einer Abtastfrequenz von 100 kHz und einer Signalfrequenz von 25 kHz ausgegeben werden.

d) Wieviele Perioden und Werte sind notwendig?

Periodenzahl :

Anzahl der Werte:

e) Berechnen Sie die Werte für eine 8bit-Quantisierung und tragen Sie sie in die Tabelle ein.

Index	0	1	2	3	4	5	6	7	8	9	10	11
A/hex												

f) In Kapitel 3.1, Beispiel 2 wird eine Tabelle über den Adreßbus an den D/A-Wandler ausgegeben. Warum ist die Verwendung des gezeigten Algorithmus nicht für beliebige Signaltabellen geeignet?

Aufgabe 4

a) In Aufgabe 3c) sollten Sie die für die Ausgabe eines sinusförmigen Signals mit einer Signalfrequenz von 10 kHz und einer Abtastfrequenz von 100 kHz notwendigen Werte berechnen. Formulieren Sie mit diesen Werten einen Algorithmus zur Ausgabe des Signals und setzen Sie diesen in ein Programm um. Öffnen Sie hierzu das Projekt *c:\versuch2\aufgabe4a\aufg4a.prj* und fügen Sie Ihr erstelltes Programm hinzu.

b) Betrachten Sie das Signal aus a) mit Hilfe eines Oszilloskops. Wie erklären Sie die unterschiedlichen Signale hinter dem 10 kHz-Filter und dem 100 kHz-Filter?

c) Entwerfen Sie einen Algorithmus zur Ausgabe eines rechteckförmigen Signals mit einer Signalfrequenz von 62,5 kHz und einer Abtastfrequenz von 250 kHz. Öffnen Sie das Projekt *c:\versuch2\aufgabe4c\aufg4c.prj* und verfahren Sie wie in a). Betrachten Sie das Signal und machen Sie eine Aussage über die spektralen Eigenschaften der Filter.

d) Entwerfen Sie einen Algorithmus zur Lösung der Aufgabe 3e). Die errechneten Werte sollen in Tabellenform im Speicher abgelegt werden. Verwenden Sie das Projekt *c:\versuch2\aufgabe4d\aufg4d.prj.*

e) Im folgenden sind zwei Programme gegeben, die ein Rechtecksignal auf dem D/A-Wandler ausgeben. Geben Sie jeweils die Signalfrequenz f_S und die Abtastrate f_A an.

```
start:
   mov   dptr,#0FF00h     ;Datenzeiger mit Adresse
                          ;des DAC laden
anfang:
   mov   A,#0FFh
   movx  @dptr,A
   nop
   nop
   mov   A,#00h
   movx  @dptr,A
   jmp   anfang

END
```

Beispiel 4 rect1.a51

```
start:
   mov   dptr,#0FF00h     ;Datenzeiger mit Adresse
   mov   A,#0FFh          ;des DAC laden

anfang:
   movx  @dptr,A
   cpl   A
   jmp   anfang

END
```

Beispiel 5 rect2.a51

Aufgabe 5

Bei diesem Versuch sollen die Effekte, die durch Aliasing entstehen, im Zeitbereich untersucht werden. Dazu wird ein Programm benötigt, das ein Zeitsignal mit einer festen Abtastrate f_A abtastet und wieder ausgibt. Verwenden Sie zur Bearbeitung der Aufgabe das Projekt *c:\versuch2\aufgabe5\aufg5.prj*. Schreiben Sie ein Programm, welches einen Analogwert von Pin 7.0 (A/D-Kanal 0) einliest, diesen nach einer Wartezeit von 82µs auf den Datenbus (externer D/A-Wandler) ausgibt und danach wieder von vorne beginnt. Da der A/D-Wandler für eine Konversion ca. 12µs benötigt, und die Befehle in der Schleife in ca. 6µs abgearbeitet werden, ergibt sich eine Abtastzeit von etwa 100µs. Dies entspricht einer Abtastfrequenz von 10kHz. Das zugehörige Ablaufdiagramm ist in Abbildung 14 dargestellt.

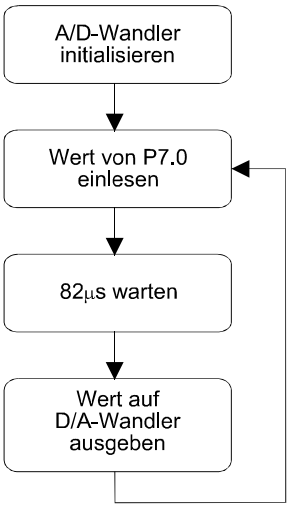


Abbildung 14 Ablaufdiagramm zu Aufgabe 5

Der Programmkopf kann z.B. folgendes Aussehen haben:

```
$NOMOD51                          ;Registerbelegung des 8051 abschalten
$include(REG517A.INC)             ;SFR Symbole von 80C517A benutzen

PROG    SEGMENT  CODE             ;Codesegment definieren
STACK   SEGMENT  IDATA            ;Stacksegment definieren
using   0                         ;Registerbank 0 auswählen

        RSEG   STACK
        DS     20h                ;32 Byte für Stapelspeicher reservieren

        RSEG   PROG
        ORG    0100h              ;Programm beginnt ab Adresse 0100h
start:
   mov   SP,#STACK-1

<Hier beginnt das Hauptprogramm>

END
```

- Initialisieren des internen A/D-Wandlers im 80C517A:
  - Der Eingangskanal 0 des A/D-Wandlers soll benutzt werden => MX0..MX2=0
  - Der A/D-Wandler soll nach jeder Messung anhalten => ADM = 0
  - Die Messung wird intern ausgelöst => ADEX = 0
  - Das Bit ADCL im SFR ADCON1 muß gesetzt sein.

- Ablauf einer Messung mit dem A/D-Wandler:
  - Start des A/D-Wandlers durch Schreiben des SFR ADDATL
  - Warten, bis die Messung abgeschlossen ist, d.h. zyklisches Abfragen des BSY-Flags solange, bis dieses von der Hardware gelöscht wird. Eine A/D-Konversion dauert ca. 12µs
  - Das Ergebnis der A/D-Konversion steht nun in den SFRs ADDATH und ADDATL

  Weitere Informationen über den A/D-Wandler im 80C517A finden Sie im User´s Manual des SAB80C517A ab Seite 5-3.

- Die Warteschleife nach dem Einlesen des Analogwertes kann durch eine Programmschleife mit einigen NOP (No Operation)-Befehlen realisiert werden. Die Zeit, die für die Ausführung benötigt wird, läßt sich durch folgende Formel berechnen:

  T_{GES} = Anzahl der Befehlszyklen * 12 / Systemtakt

  Die Anzahl der Zyklen für einen Befehl kann dem User´s Manual entnommen werden. Der Systemtakt des Mikrocontrollers ist auf 18 MHz festgelegt.

Versuchsdurchführung

Der Analogeingang des MC wird mit dem Frequenzgenerator verbunden. Dieser erzeugt ein Sinussignal von ca. 2,5V Amplitude mit einem Offset von 2,5V. Der Offset ist notwendig, da der Analogeingang des MC nur Spannungen zwischen 0V und 5V verarbeiten kann. Der Ausgang des D/A-Wandlers wird mit dem Oszilloskop und mit dem Lautsprecherverstärker verbunden. Wenn Sie nun die Frequenz des Sinussignals kontinuierlich erhöhen, so können Sie die Effekte, die durch die Periodizität des Spektrums des abgetasteten Signals entstehen, sehen und hören.

a) Wie kommen die beobachteten Effekte zustande?

b) Wie kann man die Abtastfrequenz mit dem Sinusgenerator und dem Oszilloskop ermitteln?

c) Wie groß ist die Abtastfrequenz des Programms wirklich?

d) Das abgetastete Signal soll nun mit 3 Bit quantisiert und ausgegeben werden. Führen Sie die notwendigen Änderungen im Programm durch (**Tip:** Verwenden Sie den Befehl anl zum Ausmaskieren der Bits).

Aufgabe 6

Bei diesem Versuch wird mit dem Mikrocontroller das Fourierspektrum eines Signals dargestellt. Als Eingabe dient wie bei Aufgabe 5 der Frequenzgenerator. Das analoge Signal wird mit dem im Mikrocontroller 80C517A integrierten A/D-Wandler digitalisiert. Der Betrag der vom Mikrocontroller durchgeführten 256-Punkte-FFT wird auf dem externen D/A-Wandler ausgegeben und auf einem Oszilloskop dargestellt.

Ihre Aufgabe ist es, eine Eingaberoutine, eine Ausgaberoutine und das Hauptprogramm zu programmieren. Die Routine, die aus einem Datensatz von 256 Werten die FFT berechnet, ist schon vorhanden. Binden Sie Ihr Programm und die Datei *fft.c* in das Projekt *c:\versuch2\aufgabe6\aufg6.prj* ein.

Die Eingaberoutine

Die Eingaberoutine soll Analogwerte in einen 256 Byte großen Speicherbereich ab dem Label *ad_werte* im externen Datenspeicher einlesen. Die Initialisierung des A/D-Wandlers erfolgt analog zu Aufgabe 5. Das Schreiben der Tabellenwerte kann nicht wie in Aufgabe 3 mit dem MOVC-Befehl erfolgen, da dieser auf den Programmspeicher zugreift. Die digitalisierten Meßwerte sind aber keine Programmkonstanten und müssen deshalb im externen Datenspeicher abgelegt werden.

Zugriff auf den Datenspeicher erlaubt der Befehl MOVX. Beim Schreiben lautet die Syntax:

```
movx   @DPTR,A          ;Akku -> [DPTR]
```

Dieser Befehl schreibt den Inhalt des Akkumulators an die Adresse, auf die der Datapointer zeigt.

Die Abtastfrequenz bei der Eingabe soll ca. f_A = 10 kHz betragen. Die Zeit für eine Messung beträgt also T_{Sample} = 100µs. Deshalb ist es erforderlich, nach jeder Messung eine Warteschleife von ca. 82µs einzufügen. Die restlichen 18µs ergeben sich aus der Zeit, die der A/D-Wandler für eine Messung benötigt (T_{AD} = 12µs) und der Zeit, die für die Ausführung der Assemblerbefehle in der Eingabeschleife benötigt wird (ca. 6µs).

Die Ausgaberoutine

Die Ausgaberoutine gibt die 256 Bytes, die nach dem Label *fft_betrag* im externen Datenspeicher abgelegt sind, auf dem D/A-Wandler aus. In diesem Speicher steht der Betrag der diskreten Fouriertransformation nach dem Aufruf der FFT-Routine.

Da die Berechnung einer FFT ca. 6 Sekunden dauert, ist es nicht sinnvoll, die Ausgabe nur einmal nach jeder FFT zu starten. Dies hätte zur Folge, daß nur alle 6 Sekunden eine Ausgabe auf dem Oszilloskop zu sehen wäre. Statt dessen wird die Ausgabe als Interruptroutine implementiert. Der Interrupt für die Ausgabe soll von Timer0 nach jeweils 20 ms ausgelöst werden. Dadurch ergibt sich ein stehendes Bild mit einer Wiederholrate von 50 Hz auf dem Oszilloskop.

Die Initialisierung des Timer 0

- Der Timer0 soll alle 20 ms überlaufen und einen Interrupt auslösen. Dies läßt sich am besten im Modus 1 realisieren. In diesem Modus fungiert Timer0 als 16bit-Zeitgeber. Die Moduswahl erfolgt über die Bits 1 und 0 im SFR TMOD. Das Bit 2 (C/T) im SFR TMOD muß gelöscht sein, damit der Timer die Impulse des internen Systemtaktes zählt. Bei C/T = 1 erhält der Timer0 die Zählimpulse extern von Pin T0. Das Bit 3 (Gate) muß ebenfalls gelöscht sein, da der Timer0 sonst nur bei High-Pegel am Pin INT0 gestartet werden kann. Die restlichen Bits im SFR TMOD betreffen Timer1 und sollten nicht verändert werden. Da das Byte TMOD nicht im bitadressierbaren Bereich liegt, müssen die Bits mit den Befehlen ORL/ANL gesetzt bzw. gelöscht werden.

- Der Zählerstand, der am Anfang in den SFR TH0/TL0 steht, muß so gewählt werden, daß alle 20 ms ein Überlauf (Zählerstand springt von 65535 auf 0) entsteht. Der Zähltakt beträgt 18MHz/12. Da Timer0 im Modus 1 keinen Autoreload durchführt, sondern nach einem Überlauf wieder mit 0 zu Zählen beginnt, muß der Startwert nach jedem Interrupt neu in die SFR TH0/TL0 geschrieben werden.

- Der Timer0 wird über das Bit TR0 eingeschaltet (TR0 = 1). Wird dieses Bit gelöscht, so wird er gestoppt.

  Die Timer 0 und 1 sind im User´s Manual ab Seite 101 beschrieben.

Das Hauptprogramm

Der Programmkopf sieht ähnlich aus wie in Aufgabe 5. Das Assemblerprogramm muß zusätzlich folgende Zeilen enthalten:

```
PUBLIC  ad_werte, fft_betrag      ;Diese Labels werden von der FFT-Routine
                                  ;benutzt
EXTRN  CODE(fft_init)             ;Modul fft.c
EXTRN  CODE(fft)                  ;Modul fft.c

PROG    SEGMENT  CODE             ;Codesegment definieren
STACK   SEGMENT  IDATA            ;Stacksegment definieren
DATEN   SEGMENT  XDATA            ;Datensegment im X-Speicher definieren
using   0                         ;Registerbank 0 auswählen

        RSEG   STACK
        DS     20h                ;32 Byte für Stapelspeicher reservieren

        RSEG   DATEN              ;folgende Direktiven beziehen sich auf das
        ORG    0C000h             ;Datensegment. Der externe Speicher beginnt ab
                                  ;Adresse C000h im externen RAM
ad_werte:      DS  256            ;reserviere 256 Bytes für die Meßreihe des ADC
fft_betrag:    DS  256            ;reserviere 256 Bytes für die Ausgabe der FFT

CSEG  AT  0000h                   ;Sprungadresse bei Hardware-Reset
   ljmp  start
CSEG  AT  000Bh                   ;Interruptvektoradresse für Timer0
   ljmp ausgabe_routine           ;Sprung zur Ausgaberoutine

        RSEG   PROG
        ORG    0100h              ;Programm beginnt ab Adresse 0100h
start:
   mov   SP,#STACK-1
<Hier beginnt das Hauptprogramm>
END
```

Das Hauptprogramm und die Interruptroutine besitzen folgende Struktur:

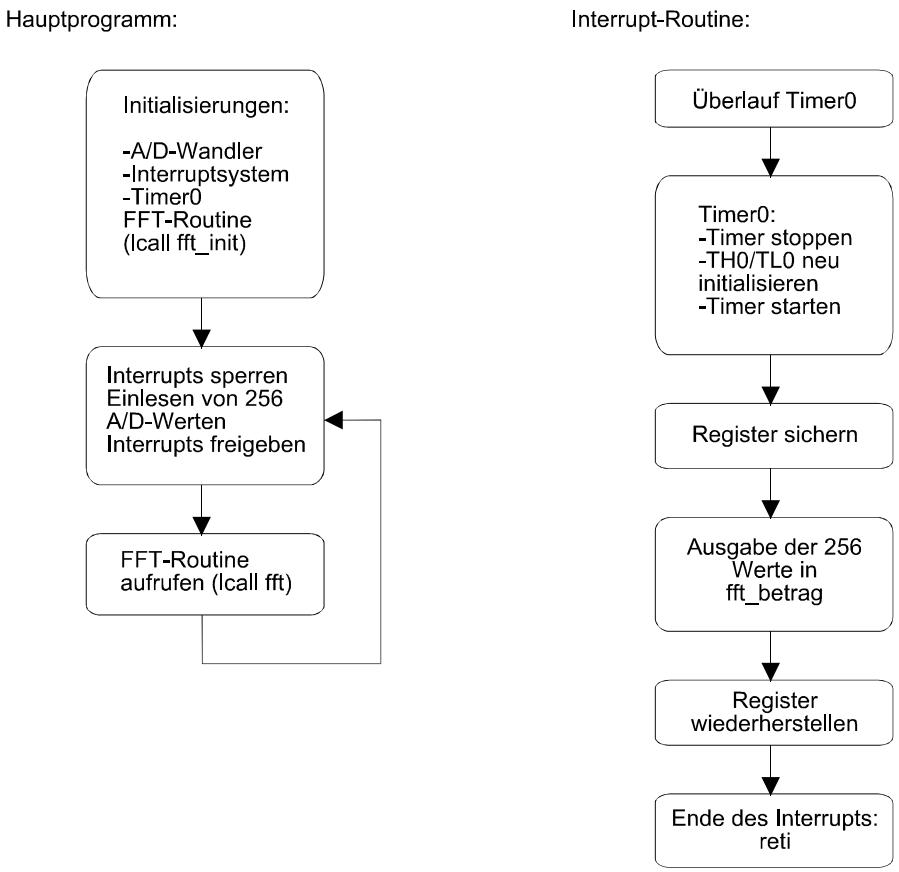


Abbildung 15 Ablaufdiagramm zu Aufgabe 6

Versuchsdurchführung

Verbinden Sie den Funktionsgenerator über das Anti-Aliasing-Filter mit dem Analogeingang P7.0 des MC. Der Eingangsspannungsbereich des Anti-Aliasing-Filters steht auf 10 V und die Grenzfrequenz wird zunächst auf ∞ gestellt, d.h. das Anti-Aliasing-Filter verändert das Signal nicht. Der Funktionsgenerator erzeugt ein sinusförmiges Signal von ca. 500 Hz. Der D/A-Wandler-Ausgang wird über das 10kHz-Filter an das Oszilloskop angeschlossen. Stellen Sie die Zeitbasis nun so ein, daß auf dem Oszilloskop ein zur y-Achse symmetrisches Bild zu sehen ist. Erhöhen Sie nun die Frequenz des Sinussignals.

a) Wie ändert sich das Betragsspektrum mit der Frequenz?

b) Ab welcher Frequenz läßt sich das wahre Spektrum nicht mehr aus dem angezeigten Spektrum rekonstruieren?

c) Wie groß ist die Abtastrate Ihres Programms?

d) Was beobachten Sie im Spektrum, wenn die Signalfrequenz größer als die halbe Abtastfrequenz ist?

Stellen Sie nun wieder eine relativ kleine Frequenz (ca. 500 Hz) ein. Beobachten Sie die Änderung im Spektrum bei anderen Kurvenformen (Dreieck, Rechteck).

e) Was ändert sich, wenn Sie das Anti-Aliasing-Filter auf eine Grenzfrequenz von 1 kHz umstellen?

6 Literaturverzeichnis

Zum Thema der Signalverarbeitung:

Kürzel	**Autor**	**Titel**	**Verlag**
[KaKr89]	Kammeyer, Kroschel	Digitale Signalverarbeitung	Teubner Studienbücher
[Kron91]	Kronmüller	Digitale Signalverarbeitung	Springer-Verlag

Zum Thema der Frequenzsynthese:

Kürzel	**Autor**	**Titel**	**Verlag**
[Dost97]	Dostert	Strukturen informationsverarbeitender Mikrorechnersysteme II	Hilfsblätter zur Vorlesung

Zur allgemeinen Mathematik:

Kürzel	**Autor**	**Titel**	**Verlag**
[BrSe85]	Bronstein, Semendjajew	Taschenbuch der Mathematik	Verlag Harri Deutsch
[Föll86]	Föllinger	Laplace- und Fourier-Transformation	Hüthig