Ein digitaler
DCF77-Empfänger
mit hoher Empfindlichkeit

Diplomarbeit
Universität Hamburg - Fachbereich Informatik
Arbeitsbereich Technische Grundlagen der Informatik

von Martin Wierich

Inhaltsverzeichnis

Kapitel 1

Einleitung

Das DCF77-Signal ist ein Funksignal, das, ähnlich der Zeitansage im Telefonnetz, ständig die aktuelle Uhrzeit in codierter Form aussendet. Funkuhren können dieses Signal empfangen und dementsprechend die Uhrzeit anzeigen. Die ausgesandte Uhrzeit stammt von einer Atomuhr und ist somit sehr genau. Funkuhren haben gegenüber herkömmlichen Uhren vor allem den Vorteil, daß ein manuelles Nachstellen der Uhr nicht nötig ist. Besonders bei öffentlichen Uhren ist das manuelle Nachstellen teuer. Folgende Fälle können es notwendig machen, herkömmliche Uhren nachzustellen:

- Der Wechsel zwischen Sommer- und Winterzeit.
- Stromausfall
- Ungenauigkeit der Uhr. Der Gang von Quarzuhren beträgt immerhin etwa eine Sekunde pro Woche. Dieser Fehler kann sich auf eine Minute pro Jahr akkumulieren [BAD87].

Ein Problem mit Funkuhren ist jedoch, daß sie nicht überall funktionieren, da das DCF77-Signal an manchen Orten gegenüber dem mitempfangenen *Rauschen* und anderen Störungen zu schwach ist. Diese Empfangsfähigkeit hängt nicht zuletzt auch von der Entfernung zwischen Sender und Empfänger ab. Der Sender steht in der Nähe von Frankfurt am Main.

In dieser Arbeit wird versucht, eine Funkuhr zu entwerfen und zu implementieren, die so empfindlich wie möglich ist, die also in einem möglichst großen Abstand zum Sender noch funktioniert. Sie hat das Grundproblem zu lösen, die Nutzinformation des DCF77-Signals aus einem durch Rauschen und andere Sender gestörten Spektrum zu extrahieren

Der Funkuhr (im folgenden auch „der Empfänger“ oder „die Uhr“) liegen zwei Gedanken zugrunde:

- *Die Redundanz des Signals wird ausgenutzt.*
  In jeder Minute wird die vollständige Information über die aktuelle Uhrzeit gesendet. Diese ändert sich jedoch von Minute zu Minute kaum. Mit anderen Worten: Das DCF77-Signal wiederholt sich (fast) zu jeder Minute. Aus dieser Redundanz wird Nutzen gezogen.

- *Die Uhr geht genau, da die Taktfrequenz geregelt wird.*
  Der digitale Empfänger wird natürlich von einem Takt getrieben. Wie bei Quarzuhren auch, wird dieser Takt mit Hilfe eines Schwingquarzes erzeugt. Durch Abgleich der Taktfrequenz mit der empfangenen atomuhrgenauen Trägerfrequenz von 77.5 kHz wird der Takt auf die exakte Frequenz geregelt. „Exakt“ bedeutet hier also „atomuhrgenau“. Diese Regelung ist auch dann möglich, wenn das empfangene Signal so stark verrauscht ist, daß die Dekodierung der Zeitinformation *nicht* mehr möglich ist. Dies ist wichtig, da die Qualität des empfangenen Signals zeitabhängig ist.
  Die Idee ist folgende: Es reicht, wenn die Uhr die Zeitinformation *einmal* korrekt empfangen hat. Von diesem Zeitpunkt an kann die Signalqualität ruhig abnehmen. *Die Uhr geht dann weiterhin genau*, da ja der Takt genau ist. Die Ungenauigkeit herkömmlicher Uhren ist darauf zurückzuführen, daß deren Takt nicht exakt ist !
  Wie noch gezeigt wird, läßt sich diese Regelung durch eine *„Phase Locked Loop“ (PLL)* erzielen. Dadurch ist auch das Problem der Demodulation gelöst.

Einige Grundlagen bezüglich des DCF77-Signals und der Probleme und Verfahren beim Funkempfang sind in Kapitel 2 zusammengestellt.

In Kapitel 3 wird eine Übersicht über den Empfänger gegeben. Der Empfänger enthält im wesentlichen drei Komponenten: einen Detektor, eine Multiratenfilterkaskade und einen PLL-Regler. Die Struktur wird durch ein Blockschaltbild modelliert. In den beiden nachfolgenden Kapiteln werden diese drei Blöcke detaillierter betrachtet.

Der in Kapitel 4 vorgestellte Detektor verwirklicht das Ausnutzen der Redundanz des DCF77-Signals. Der Detektoralgorithmus erhält als Eingabe das demodulierte DCF77-Signal. Nach einer hinreichend langen Zeit liefert dieser Algorithmus als Ausgabe die aktuelle Uhrzeit. Der Algorithmus korreliert das DCF77-Signal mit verschiedenen Musterfunktionen.

In Kapitel 5 werden die Multiratenfilterkaskade und der PLL-Regler erläutert. Die Multiratenfilterkaskade filtert den Träger des DCF77-Signals aus dem empfangenen Signal heraus. Durch Vergleich mit der Trägerfrequenz kann der PLL-Regler die Taktfrequenz korrigieren. Im eingerasteten Zustand läßt sich die PLL als System zweiter Ordnung modellieren. Daraus ergibt sich, daß das Verhalten der PLL durch zwei Parameter bestimmt ist, der Eigenfrequenz und der Dämpfung. Diese Parameter sind geeignet zu dimensionieren. Am Ende dieses Kapitels wird das Zusammenwirken von Multiratenfilterkaskade und PLL erläutert, und durch eine Messung am implementierten Empfänger demonstriert.

Kapitel 6 erörtert die im Zuge dieser Arbeit durchgeführte Implementierung des Empfängers. Das Antennensignal wird zunächst analog vorverarbeitet und anschließend digitalisiert. Es wurde eine Platine für die Analogelektronik und den regelbaren Taktgenerator implementiert. Die Digitalisierung und die weitere digitale Signalverarbeitung erfolgt auf der Signalprozessor-Evaluationsplatine DSP56002EVM der Firma Motorola. Die Programmierung des Signalprozessors erfordert Kenntisse über dessen Architektur.

In Kapitel 7 wird das vorgestellte Empfängerkonzept bewertet, und ein Ausblick auf mögliche Verbesserungen gegeben.

In der Vergangenheit wurden Forschungen in bezug auf Funkuhren insbesondere an der Technischen Hochschule Darmstadt [BEL89] [BER85][WIE88] und an der Universität Stuttgart [BAD87][HET87] angestellt. Die beiden genannten Grundideen wurden bereits in [BEL89] vorgeschlagen.

Kapitel 2

Grundlagen

Ein Sender hat die Aufgabe, Informationen I in ein Funksignal zu wandeln, welches dann von einem oder mehreren räumlich entfernten Empfängern empfangen werden kann. Die Aufgabe des Empfängers ist, aus dem empfangenen Funksignal die Informationen I wieder zu extrahieren. Ein Funksignal ist eine elektromagnetische Welle, und ist stets die Folge einer Beschleunigung von elektrischer Ladung. Beispielsweise bewirkt eine an einem Draht angelegte Wechselspannung eine solche Beschleunigung von Ladung. Der Draht fungiert dann als Sendeantenne. Die elektromagnetische Welle breitet sich nun im Raum aus und kann in einem anderen Draht, der Empfangsantenne, einen Strom erzeugen. Der zeitliche Verlauf der Spannung in der Sendeantenne spiegelt sich im Verlauf der Spannung in der Empfangsantenne wider. Es kann also Information übertragen werden.

Nun gibt es aber auf der Welt mehrere Sender. Damit ein Empfänger einen bestimmten Sender selektieren kann, senden verschiedene Sender auf verschiedenen Frequenzen. Die *Selektion* geschieht dann durch Herausfiltern des zu dem gewünschten Sender gehörenden Frequenzbandes. Die beim Sender vorliegende zu übertragende Information sei modelliert durch das Nutzsignal $s(t)$, einer Funktion der Zeit in die reellen Zahlen. Das Spektrum $S(f)$ dieses Signals sei bandbegrenzt, das heißt, es existiert eine obere Grenze für die im Signal vorkommenden Frequenzen. $S(f)$ wird *Basisband* genannt. Durch *Modulation* transformiert der Sender dieses Basisband in das ihm zugeordnete Frequenzband, das RF-Band (Radio-Frequency-Band). Das RF-Band wird dann über die Sendeantenne emittiert, und vom Empfänger empfangen. Dieser selektiert das gewünschte RF-Band und transformiert es im allgemeinen durch *Demodulation* wieder in das Basisband, so daß dem Empfänger im Idealfall das Signal $s(t)$ zur Verfügung steht. Selbstverständlich sind die Frequenzen eines RF-Bandes viel höher als die eines Basisbandes. Zum Beispiel liegen sie beim UKW-Funk im Bereich von 30MHz bis 300MHz [SIE95].

In der Realität sind das gesendete und das empfangene Signal allerdings nicht gleich. Vielmehr ist das empfangene Signal additiv gestört und verzerrt. Der Empfänger hat also weiterhin die Aufgabe, aus dem empfangenen Signal das tatsächlich gesendete zu rekonstruieren. Diesen Vorgang nennt man *Signaldetektion*. Die Signaldetektion läßt sich umso besser durchführen, je mehr über die Charakteristiken des gesendeten Signals und der auftretenden Störungen bekannt ist

[KAM96]. Die Regelmäßigkeit, oder noch besser die *Regelhaftigkeit* des DCF77-Signals erleichtert dabei die Signaldetektion.

Man unterscheidet Sender und deren emittierte Signale unter anderem durch die Methodik der Modulation. Zur Modulation des Nutzsignals $s(t)$ bildet der Sender

$$r(t) = a(t)\sin(\omega_{tr}t + \varphi(t)) \tag{2.1}$$

wobei $a(t)$ und $\varphi(t)$ von $s(t)$ abhängen. Man nennt $f_{tr} = \frac{\omega_{tr}}{2\pi}$ die Trägerfrequenz. Im folgenden werden Kreisfrequenzen ω der Einfachheit halber auch als Frequenzen bezeichnet. Es werden nur reelle Signale betrachtet. Man unterscheidet verschiedene Modulationsarten je nachdem, wie die Amplitude $a(t)$ und/oder die Phase $\varphi(t)$ von $s(t)$ abhängen. In dieser Arbeit wird davon ausgegangen, daß das DCF77-Signal nur *amplitudenmoduliert* ist, das heißt, es gilt:

$$r(t) = s(t)\sin(\omega_{tr}t) \tag{2.2}$$

wobei $s(t)$ stets grösser oder gleich null ist. Die Tatsache, daß das DCF77-Signal zusätzlich zur Amplitudenmodulation auch noch phasenmoduliert ist, kann außer acht gelassen werden, vergleiche Abschnitt 2.1. Der zu entwerfende Empfänger für das DCF77-Signal fällt also in die Klasse der Empfänger für amplitudenmodulierte Signale.

Da in dieser Arbeit letztlich nur das DCF77-Signal von Interesse ist, und die Amplitudenmodulation überdies die mathematisch am einfachsten beschreibbare Form der Modulation ist, soll auf die anderen Formen der Modulation im weiteren nicht eingegangen werden. Im folgenden wird erklärt, warum die Multiplikation des Nutzsignals mit einem harmonischen Signal das Basisband in das RF-Band verschiebt:

Definition 2.1 *Den Vorgang der Multiplikation zweier Signale nennt man* Mischung.

Wenn zwei harmonische Signale unterschiedlicher Frequenzen ω_1 und ω_2 gemischt werden, dann entstehen dadurch im resultierenden Signal die Frequenzen $\omega_1 + \omega_2$ und $|\omega_1 - \omega_2|$. Nun sei das Spektrum $S(\omega)$ von $s(t)$ innerhalb der Frequenzen ω_1 und ω_2 bandbegrenzt. Durch Multiplikation von $s(t)$ mit einem harmonischen Signal der Frequenz ω_{tr} entsteht ein Spektrum, in dem das Frequenzband aus $S(\omega)$ *zweimal* vorkommt, siehe Abbildung 2.1. Dabei macht es einen Unterschied, ob ω_{tr} oberhalb oder unterhalb des Spektrums $S(\omega)$ vorkommt. Ist das Spektrum $S(\omega)$ symmetrisch um die Frequenz $\omega_{tr} = \frac{\omega_2 - \omega_1}{2}$, dann wird dieses Spektrum durch Mischung zur Frequenz null verschoben, siehe Abbildung 2.1c. Ist das Spektrum $S(\omega)$ symmetrisch um die Frequenz null, so wird es zur Frequenz ω_{tr} hin verschoben, vergleiche Abbildung 2.1d.

Durch Mischung lassen sich also Frequenzbänder beliebig verschieben. Es ist nur darauf zu achten, daß eins der beiden entstehenden Bänder, das nicht gewollte, nach der Mischung weggefiltert wird.

Da vom Nutzsignal gefordert wurde, daß es größer oder gleich null ist, besitzt das Spektrum $S(\omega)$ die Frequenz null. Da das Nutzsignal auch reell ist, ist dieses Spektrum symmetrisch um die Frequenz null. Die in Gleichung 2.2 ausgedrückte Mischung verschiebt also das Basisband des DCF77-Nutzsignals in den RF-Bereich, entsprechend Abbildung 2.1d.

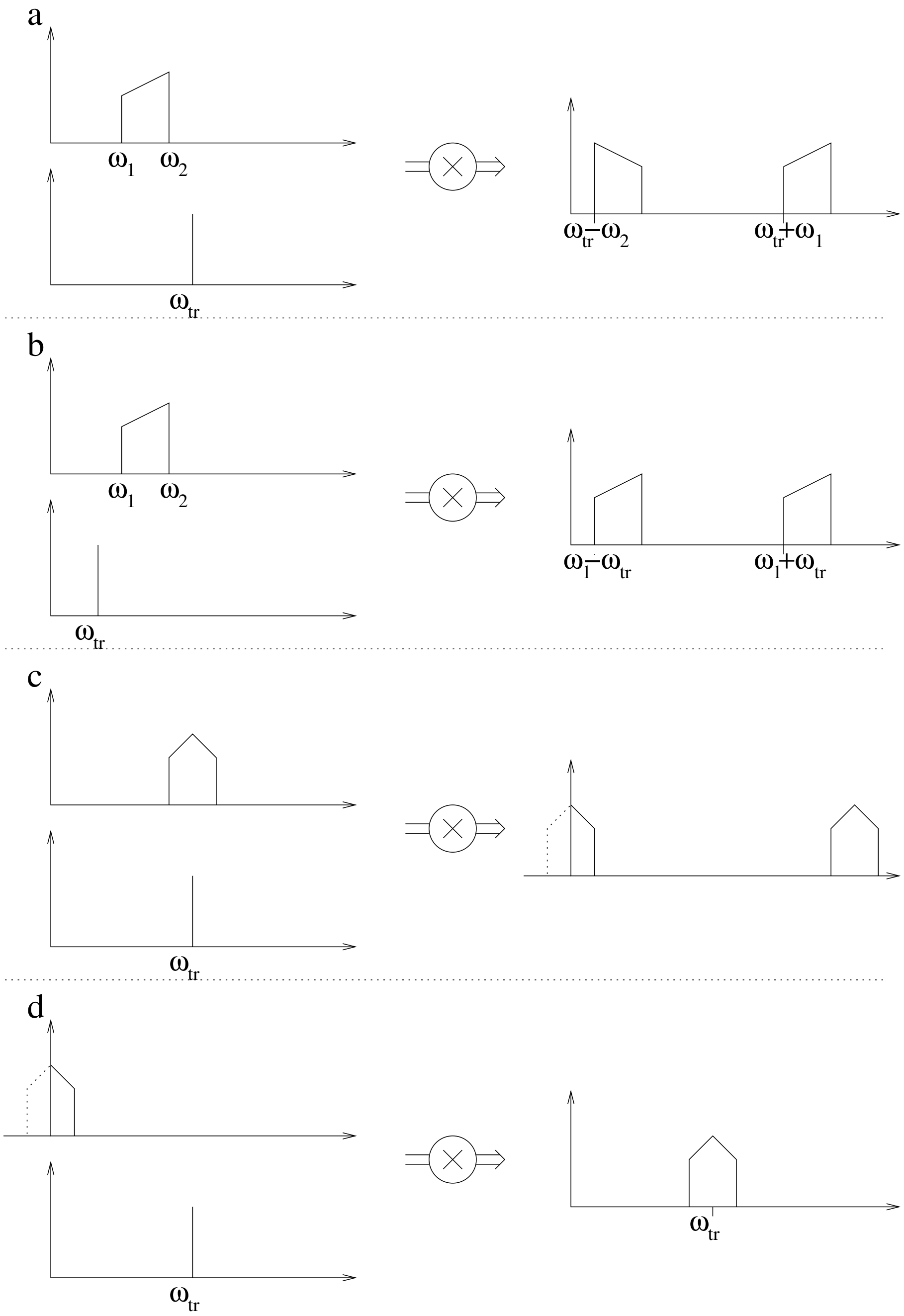

Abbildung 2.1: Mischung

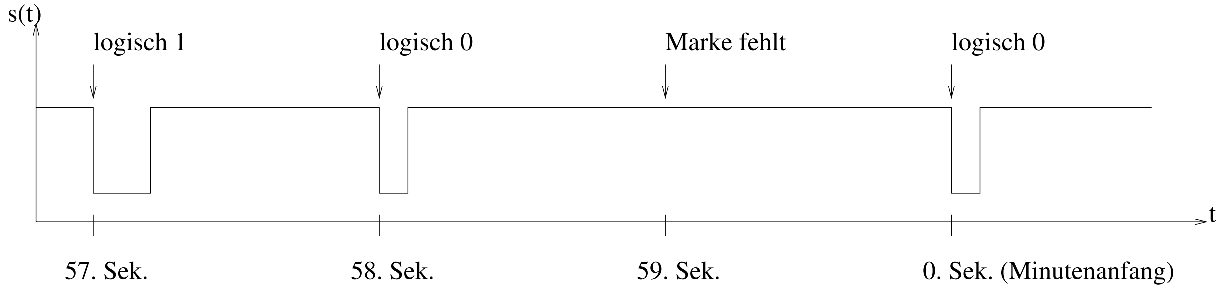

Abbildung 2.2: Das Nutzsignal $s(t)$

2.1 Das DCF77-Signal

Der DCF77-Sender sendet auf einer Trägerfrequenz von $f_{tr} = 77.5$kHz, also im Langwellenbereich. Jede Minute wird folgendes Protokoll wiederholt: Zu Beginn einer jeden Sekunde wird die Trägeramplitude für die Dauer von 0.1 oder 0.2 Sekunden auf ein Viertel abgesenkt. Eine solche Trägerabsenkung wird Marke genannt. Der Dauer einer Absenkung entspricht digitale Information. Eine 0.1-Sekunden-Marke bedeutet eine logische null, und eine 0.2-Sekunden-Marke bedeutet eine logische eins. Eine Sekunde vor einem Minutenanfang wird jedoch *keine* Marke gesendet , das heißt, der Träger wird nicht abgesenkt. Daran kann ein Empfänger den Minutenanfang, und damit den Start des Protokolls, erkennen. Abbildung 2.2 zeigt einen Auszug des Nutzsignals $s(t)$.

Die Sekunden einer Minute seien von 0 bis 59 durchnumeriert, und ebenso die Bits im Minutenprotokoll. Das Minutenprotokoll enthält in BCD-codierter Form die gesamte Information über Minute, Stunde, Datum und Wochentag, jedoch ohne die Information über das Jahrhundert. Das Protokoll enthält zusätzliche Informationen, insbesondere auch darüber, ob gerade die Sommerzeit (MESZ) oder die Winterzeit (MEZ) gilt. Es wird jeweils die Zeit der *kommenden* Minute übertragen. Tabelle 2.1 enthält das gesamte Protokoll [BEC73] [BEC77].

Die Zahlenfolgen in Klammern in der Tabelle geben die Wertigkeit der in dieser Reihenfolge übertragenen Bits an. Die Bits mit den Nummern 21 bis 24 zum Beispiel entsprechen der binären Kodierung der Einerstelle der Minuten in einer Uhr, die die Minuten in zwei Dezimalstellen anzeigt. Die Bits 25 bis 27 kodieren die Zehnerstelle der Minuten. Da diese Stelle nicht größer als fünf werden kann, genügen drei Bits.

Die Prüfbits ergänzen die Anzahl der logischen Einsen im vorhergehenden Wort auf eine gerade Anzahl.

Die Atomuhr, die der ausgesendeten Zeit zugrunde liegt, wird von der Physikalisch Technischen Bundesanstalt (PTB) in Braunschweig betrieben. Auch die Trägerfrequenz wird von dieser Atomuhr abgeleitet, und ist hochgenau. Der relative Fehler der Trägerfrequenz am Sendeort beträgt im Mittel über einen Tag weniger als 10^{-12} [HET87][BEC72].

Die vom Sender emittierte elektromagnetische Welle läßt sich in die Boden- und die Raumwelle unterteilen. Die Bodenwelle breitet sich entlang der Erdoberfläche aus, während die Raumwelle

Bitnummer	Bedeutung
0	immer logisch 0 („Minutenmarke“)
1 … 14	nicht definiert
15	genau dann logisch 1, wenn die Aussendung über eine Ersatzantenne erfolgt.
16	genau dann logisch 1 wenn zu Beginn der nächsten Stunde ein Wechsel zwischen Sommer- und Winterzeit stattfindet
17 … 18	in diesen beiden Bits ist (redundant) kodiert, ob gerade die Sommer- oder die Winterzeit gilt
19	genau dann logisch 1 wenn zu Beginn der nächsten Stunde eine Schaltsekunde eingefügt wird
20	immer logisch 1
21 … 27	Minute (Wertigkeiten:1,2,4,8,10,20,40)
28	Prüfbit für Minute
29 … 34	Stunde (Wertigkeiten:1,2,4,8,10,20)
35	Prüfbit für Stunde
36 … 41	Tag (Wertigkeiten:1,2,4,8,10,20)
42 … 44	Wochentag (Wertigkeiten:1,2,4 Montag=1 Sonntag=7)
45 … 49	Monat (Wertigkeiten:1,2,4,8)
50 … 57	Jahr (Wertigkeiten:1,2,4,8,10,20,40,80)
58	Prüfbit für Datum (Bits 36..57)
59	Marke fehlt

Tabelle 2.1: Das DCF77 Minutenprotokoll

auf dem Weg zum Empfänger in der Ionosphäre reflektiert wird. Besonders die Empfangsstärke der Raumwelle hängt vom Wetter, der Sonneneinstrahlung und der Jahreszeit ab. Die Raumwelle ist nachts besser zu empfangen als tagsüber. Während in Entfernungen von 700km zum Sender die Bodenwelle überwiegt, dominiert in Entfernungen von über 1100km die Raumwelle. In dem Bereich zwischen 700km und 1100km kann es aufgrund von Überlagerungen von Boden- und Raumwelle zu einer gegenseitigen Auslöschung kommen, da die beiden Wellen unterschiedlich lange Wege zurücklegen. Da sich die Reflexionseigenschaften der Ionosphäre stetig ändern, dauert diese Schwunderscheinung nur wenige Minuten [BEL89]. Aufgrund dieser Überlagerung kommt es am Empfangsort zu größeren Frequenzschwankungen als am Sendeort [HET87].

Der Empfang des DCF77-Signals wird maßgeblich durch Wände von Gebäuden geschwächt [BAD87][BEL89]. Deswegen ist der Empfang in Städten im allgemeinen schlechter als außerhalb der Städte.

Die ausgesendete Zeit ist die gesetzliche Uhrzeit der Bundesrepublik Deutschland. Zusammen mit etwa 100 anderen Atomuhren aus verschiedenen Ländern wird die Atomuhr der PTB zur Festlegung der internationalen Atomzeit TAI (Temps Atomique Internationale) herangezogen, aus der wiederum die koordinierte Weltzeit UTC abgeleitet wird. [KLA92]

Zusätzlich zur Amplitudenmodulation ist das DCF77-Signal auch noch phasenmoduliert. Der in dieser Arbeit vorgestellte Empfänger filtert das Signal jedoch so schmalbandig aus dem Spektrum heraus, daß diese Tatsache außer acht gelassen werden kann [HET87].

Der Sendeort ist Mainflingen bei Frankfurt am Main.

2.2 Selektion und Demodulation

Es seien noch einmal die Aufgaben eines Empfängers rekapituliert. Ein Empfänger muß aus dem Antennensignal das gewünschte Frequenzband *selektieren, demodulieren* und das so erhaltene Basisbandsignal *detektieren*. Selektion und Demodulation werden im folgenden erörtert. Das Thema Detektion wird erst nach dem Abschnitt über Digitalempfänger in einem eigenen Abschnitt behandelt.

Zur Selektion:
Ein einfaches Selektionsprinzip verwirklicht der in Abbildung 2.3a dargestellte „Geradeausempfänger“ [BER85]. Aus dem verstärkten Antennensignal wird mit Bandpaßfiltern das zum gewünschten Sender gehörende Frequenzband herausgefiltert. Dieses Verfahren hat folgende Nachteile:

- Die Bandpaßfilter müssen so schmalbandig sein, daß die im empfangenen Spektrum benachbarten anderen Sender unterdrückt werden. Außerdem müssen sie auch noch variabel sein, sofern verschiedene Sender empfangen werden können sollen. Beides macht die Bandpässe insbesondere für höhere Frequenzen aufwendig.

- Das selektierte Spektrum wird innerhalb des Empfängers verstärkt und kann im ungünstigen Fall von der Antenne wieder empfangen werden. Es kommt zu einer Rückkopplung,

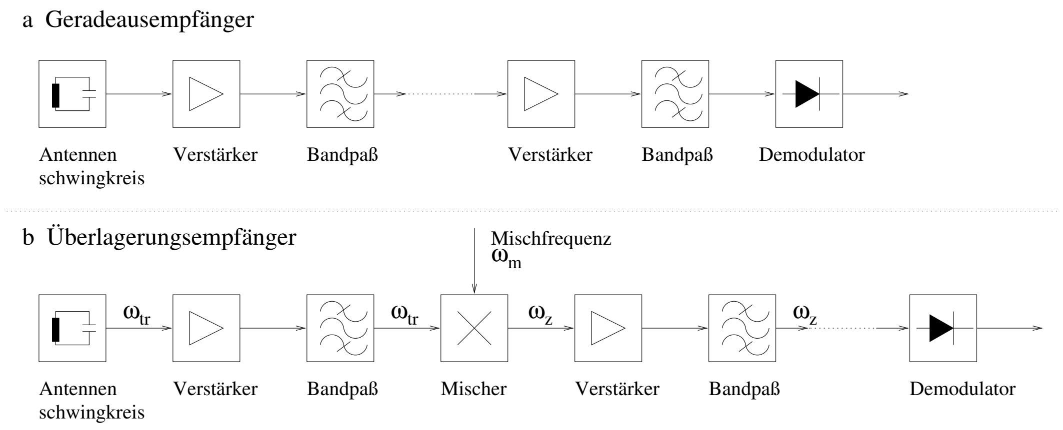

Abbildung 2.3: Empfangsprinzipien

und somit zur Anfachung von Frequenzen, die gar nicht gesendet wurden. Es ist deshalb ein sorgfältiger Aufbau notwendig.

Der in Abbildung 2.3b dargestellte „Überlagerungsempfänger“ vermeidet diese Probleme, indem das vorgefilterte Antennensignal mit einem in der Frequenz ω_m variablen harmonischen Signal auf eine Zwischenfrequenz $\omega_z = \omega_{tr} - \omega_m$ herabgemischt wird [BER85]. Das rechte Spektrum in Abbildung 2.4a entsteht auf diese Weise aus den beiden linken. Allerdings wird auch die Frequenz $\omega_m + \omega_z$ auf ω_z herabgemischt, und muß deswegen durch das Vorfilter unterdrückt werden. Abbildung 2.4b soll dies verdeutlichen. An das Vorfilter sind nun außer der Variabilität keine hohen Ansprüche zu stellen. Das die Selektion eigentlich realisierende Bandpaßfilter auf der Zwischenfrequenz kann mit wenig Aufwand realisiert werden, da die Zwischenfrequenz niedrig gewählt werden kann, und da es nicht variabel zu sein braucht.

Zur Amplitudendemodulation:
Es ist ein Bandpaßsignal in das Basisband zu verschieben. Dabei spielt es keine Rolle, ob dieses Bandpaßsignal auf der Sendefrequenz oder einer Zwischenfrequenz vorhanden ist. Eine einfache Demodulationsmethode realisiert ein Gleichrichter mit nachgeschaltetem Tiefpaß. Der Gleichrichter bildet den Betrag seines Eingangssignals. In Abbildung 2.5 ist das Signal $b(t)$ durch Multiplikation des Nutzsignals $s(t)$ mit einem Träger entstanden. Das Signal $c(t)$ ist das gleichgerichtete Signal $b(t)$. Aus diesem Signal kann (annähernd) das Nutzsignal wiedergewonnen werden, indem man es mit einem Tiefpaß filtert.

Zwei andere Methoden der Demodulation resultieren aus der Idee, die im Überlagerungsempfänger erzeugte Mischfrequenz ω_m so zu wählen, daß das empfangene Band gleich in das Basisband verschoben wird, also $\omega_m = \omega_{tr}$ und $\omega_z = 0$ zu setzen. Je nach der Art, wie das Mischsignal erzeugt wird, sind die Demodulationsprinzipien zu klassifizieren in kohärente und nicht kohärente Demodulation:

- Es sei angenommen, daß im Bandpaßsignal $r(t)$ die Trägerfrequenz vorhanden sei. Bei der kohärenten Demodulation wird über einen empfängerinternen Oszillator ein Signal

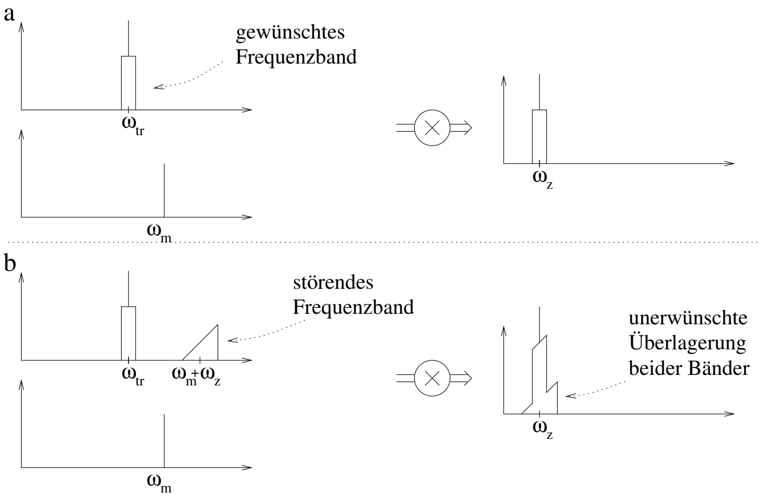

Abbildung 2.4: Mischung beim Überlagerungsempfänger

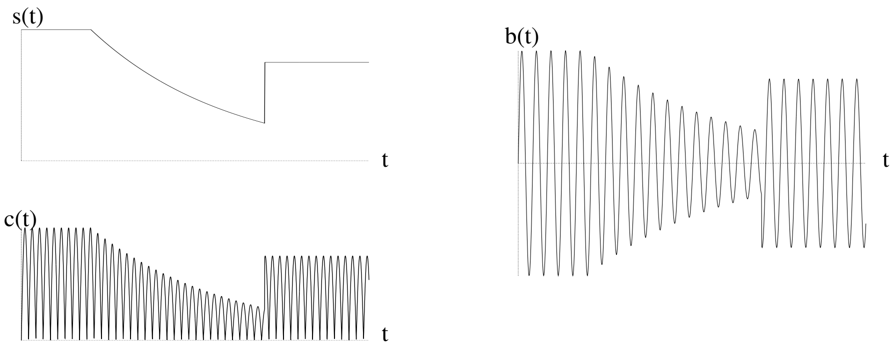

Abbildung 2.5: Demodulation durch Gleichrichtung

$o(t)$ erzeugt, das mit dem Träger in Phase und Frequenz übereinstimmt. Dieses Signal wird dann mit dem zu demodulierenden Signal gemischt. Man erhält:

$$d(t) = r(t)o(t) = s(t)\sin(\omega_{tr}t)\sin(\omega_{tr}t) \tag{2.3}$$

Durch Anwendung einer elementaren trigonometrischen Gleichung erhält man

$$d(t) = \frac{1}{2}s(t)(1 - \cos(2\omega_{tr}t)) \tag{2.4}$$

Durch einfache Tiefpaßfilterung kann der hochfrequente Anteil unterdrückt werden, und man erhält

$$d'(t) = \frac{1}{2}s(t) \tag{2.5}$$

also das ausgesendete Signal.

- Bei der nicht kohärenten Demodulation wird vom Oszillatorsignal nicht gefordert, daß es mit dem Trägersignal in Phase und Frequenz übereinstimmt. Die Demodulation kann dann durch *Quadraturmischung* stattfinden. Bei der Quadraturmischung werden durch Mischung des Bandpaßsignales mit zwei um 90^0 Grad verschobenen Oszillatorsignalen zwei Signale $q_s(t)$ und $q_c(t)$ erzeugt:

$$\begin{aligned} q_s(t) &= r(t)\sin(\omega_m t + \varphi) && (2.6)\\ q_c(t) &= r(t)\cos(\omega_m t + \varphi) && (2.7) \end{aligned}$$

Man erhält unter Anwendung einer elementaren trigonometrischen Gleichung:

$$\begin{aligned} q_s(t) &= s(t)\cos(\omega_{tr}t)\sin(\omega_m t + \varphi) \\ &= \frac{1}{2}s(t)(\sin((\omega_m - \omega_{tr})t + \varphi) + \sin((\omega_m + \omega_{tr})t + \varphi)) && (2.8) \end{aligned}$$

Durch Tiefpaßfilterung erhält man:

$$q_s' = \frac{1}{2}s(t)\sin((\omega_m - \omega_{tr})t + \varphi) \tag{2.9}$$

In entsprechender Weise läßt sich $q_c(t)$ umformen:

$$q_c' = \frac{1}{2}s(t)\cos((\omega_m - \omega_{tr})t + \varphi) \tag{2.10}$$

Letztendlich ergibt sich unter Anwendung der Gleichung $\sin^2 x + cos^2 x = 1$:

$$\sqrt{q_s'^2 + q_c'^2} = \frac{1}{2}s(t) \tag{2.11}$$

wodurch man das Nutzsignal gewonnen hat. Anders als in der kohärenten Demodulation ist es belanglos, wenn ω_m oder φ sich mit der Zeit um ihren Ruhepunkt verändern (driften). Die oben angesprochene Methode der Demodulation durch Gleichrichtung gehört auch in die Klasse der nicht kohärenten Demodulation. Den beiden angesprochenen nicht kohärenten Demodulationsmethoden ist gemeinsam, daß sie mit einer *Betragsbildung* einhergehen [KAM96]. Dies geschieht durch den Gleichrichter oder indirekt durch den Quadrierer.

2.3 Digitalempfänger

Ein Digitalempfänger tastet ein analoges Signal mit einem Analog-Digital-Wandler (A/D-Wandler) ab. Die Abtastzeitpunkte seien äquidistant, das heißt, es gibt eine Samplefrequenz f_s. Das Ergebnis der Abtastung (sampling) ist ein zeit- und wertdiskretes Digitalsignal. Die im vorherigen Abschnitt vorgestellten Empfangsprinzipien bedingen Operationen wie Multiplikation oder Radizieren. Es bietet sich an, diese Operationen digital zu realisieren, sofern die erreichbare Rechengeschwindigkeit dies zuläßt. Dann arbeiten digitale Empfänger prinzipiell mit den gleichen Verfahren wie die analogen.

Es ist festzulegen, an welchen Stellen die Grenze zwischen analog und digital zu ziehen ist. Es ist möglich, das Antennensignal direkt abzutasten, und digital zu demodulieren. Es ist aber auch möglich, das auf die Zwischenfrequenz herabgemischte Signal zu digitalisieren, oder auch erst das bereits demodulierte Signal.

Im folgenden wird die Möglichkeit der *Unterabtastung* erläutert: Die *Nyquistbedingung* lautet, daß die Samplefrequenz f_s mindestens doppelt so groß sein soll wie die höchste im abgetasteten Analogsignal vorkommende Frequenz. Um diese Nyquistbedingung einzuhalten, wird das Analogsignal mit einem Tiefpaß, dem *Aliasfilter*, vorgefiltert. Wenn die Nyquistbedingung bewußt verletzt wird, so spricht man von Unterabtastung. Die Konsequenzen der Unterabtastung für das Spektrum des Digitalsignals veranschaulicht Abbildung 2.6. Zum Analogsignal gehöre das Spektrum a. Wie man sieht, enthält es auch Frequenzen oberhalb von $\frac{1}{2}f_s$. Das resultierende Spektrum f entsteht durch Summation der vier Teilspektren b-e, die jeweils einen Abschnitt des Analogspektrums repräsentieren. Allerdings ist jedes zweite Teilspektrum zu spiegeln. Das resultierende Spektrum ist, da digital, periodisch mit der Samplefrequenz f_s und symmetrisch um die Nyquistfrequenz $\frac{1}{2}f_s$. Durch Unterabtastung erscheinen mehrere Frequenzen des Analogspektrums auf einer Frequenz des Digitalspektrums. Diesem Effekt ist durch Bandpaßfilterung des Analogsignals vorzubeugen. Dieser Bandpaß entspricht dem Aliasfilter. Abbildung 2.7 veranschaulicht dies. Das Spektrum b entsteht durch analoge Filterung aus dem Spektrum a und resultiert nach Unterabtastung im Spektrum c. Das so erhaltene Digitalsignal läßt sich nun mit den oben angesprochenen ursprünglich analogen Methoden weiterverarbeiten. So läßt sich aus den drei in Spektrum c enthaltenen Sendern durch digitale Filterung ein einziger Sender selektieren (Spektrum d).

Der große Vorteil der Unterabtastung ist, daß bei niedriger Samplefrequenz die Anforderungen an die Rechenleistung gering sind, obwohl Signale mit hohen Trägerfrequenzen verarbeitet werden können. Außerdem läßt sich durch Unterabtastung auch gleich eine Demodulation erzielen, falls

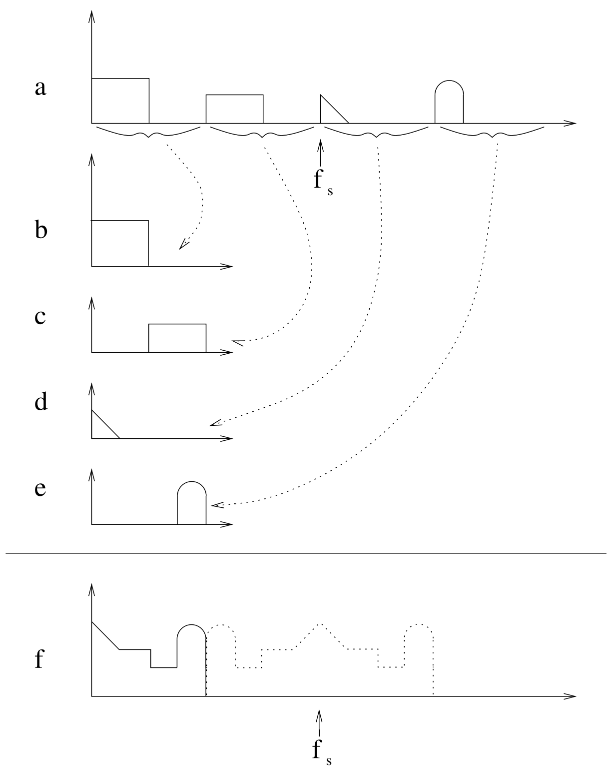

Abbildung 2.6: Unterabtastung

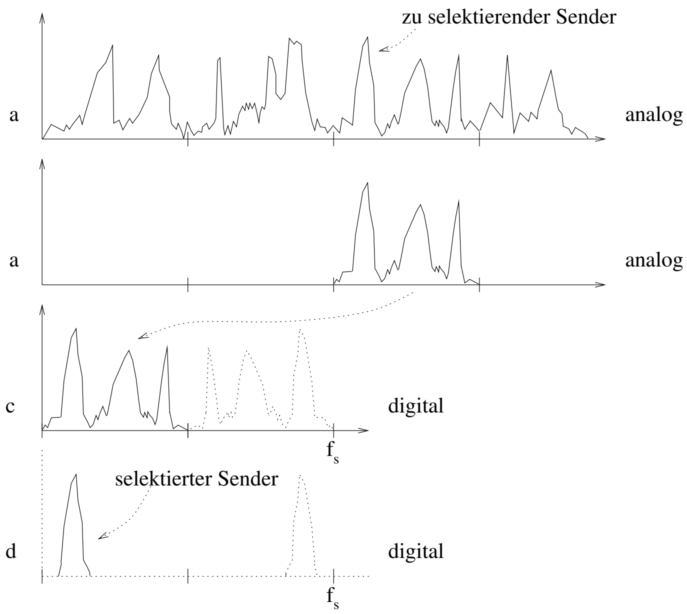

Abbildung 2.7: Selektion eines Senders mit Unterabtastung

die Samplefrequenz in der Nähe der Trägerfrequenz liegt [WIE88]. Dies ist das digitale Analogon zur oben angesprochenen Demodulation durch Mischung mit einer Mischfrequenz, die gleich der Trägerfrequenz ist.

2.4 Signaldetektion

Es folgt eine Darstellung eines klassischen Modells zur Nachrichtenübertragung [SHA69]: Ein Sender verfügt über einen endlichen Vorrat von i Zeichen t_j. In einer für den Empfänger unvorhersehbaren Weise, übermittelt der Sender dem Empfänger nacheinander verschiedene Zeichen aus seinem Vorrat. Die Übertragung dieser Zeichen verläuft durch einen gestörten Kanal, so daß beim Empfänger etwas anderes ankommt ist, als der Sender sendet.

Dieses Modell ist auf die Anwendung, um die es in dieser Arbeit geht, übertragbar. Ein Zeichen ist dabei ein Vektor $\vec{t_j}$ von nacheinander enthaltenen Samples. Es werden die durch Demodulation entstandenen Samples des Basisbandsignals betrachtet. Zunächst ist das Basisbandsignal nur ein digitaler Datenstrom, und der Empfänger hat die Aufgabe, aus diesem Datenstrom mehrere Samples zu einzelnen Zeichen zusammenzufassen. Die Lösung dieser Aufgabe heißt in diesem Zusammenhang *Synchronisation*. Man nehme an, die Zeichen $\vec{t_j}$ haben alle die gleiche Länge n. Wenn der Empfänger gerade ein demoduliertes Sample berechnet hat, besteht die Synchronisationsaufgabe also darin, zu entscheiden, zu welcher Position innerhalb von 0 bis $n-1$ dieses Sample korrespondiert.

Erst wenn der Empfänger synchronisiert ist, liegt ihm das aktuelle Zeichen in Form eines Vektors $\vec{s}$ der Länge n vor. Wegen der angesprochenen Verfälschungen wird dieser Vektor im allgemeinen keinem der möglichen Zeichen gleich sein. Der Empfänger muß sich für eine der i Möglichkeiten entscheiden. Die Methode, die im allgemeinen angewendet wird, um diese Entscheidung zu fällen heißt *Korrelation* [KAM96]. Dazu berechnet der Empfänger für jedes Zeichen $\vec{t_j}$ das Skalarprodukt aus $\vec{t_j}$ und $\vec{s}$:

$$m_j = \vec{s}\vec{t_j} \tag{2.12}$$

Es sei vorausgesetzt, daß die Beträge aller Vektoren $\vec{t_j}$ auf einen gleichen Wert normiert seien. m_j ist ein Maß für die Ähnlichkeit zwischen $\vec{t_j}$ und $\vec{s}$. Das Skalarprodukt m_j ist proportional zum Cosinus des Winkels zwischen diesen beiden Vektoren im n-dimensionalen Raum. Also ist m_j umso größer, je mehr diese beiden Vektoren in die gleiche Richtung zeigen. Der Empfänger entscheidet sich demnach für das Zeichen $\vec{t_j}$, das m_j maximiert.

Man nennt die Zeichen $\vec{t_j}$ Musterfunktionen. Man kann einen Vektor interpretieren als eine Funktion von $\mathbb{N}$ nach zum Beispiel $\mathbb{R}$. Abbildung 2.8 veranschaulicht die Korrelationsmethode an einem Beispiel. In der linken Spalte sind die beiden Kandidaten $\vec{t_1}$ und $\vec{t_2}$ als fünfelementige Vektoren dargestellt, die jeweils mit dem empfangenen Signal $\vec{s}$ verglichen werden sollen. In der rechten Spalte wurden für $j = 1$ und $j = 2$ die Komponenten der Vektoren $\vec{s}$ und $\vec{t_j}$ multipliziert. Das Ähnlichkeitsmaß m_j ist gleich der Fläche unter der betreffenden Kurve. Die Entscheidung fällt zugunsten von $\vec{t_1}$. Intuitiv ist das empfangene Signal $\vec{s}$ dem Zeichen $\vec{t_1}$ ähnlicher als $\vec{t_2}$.

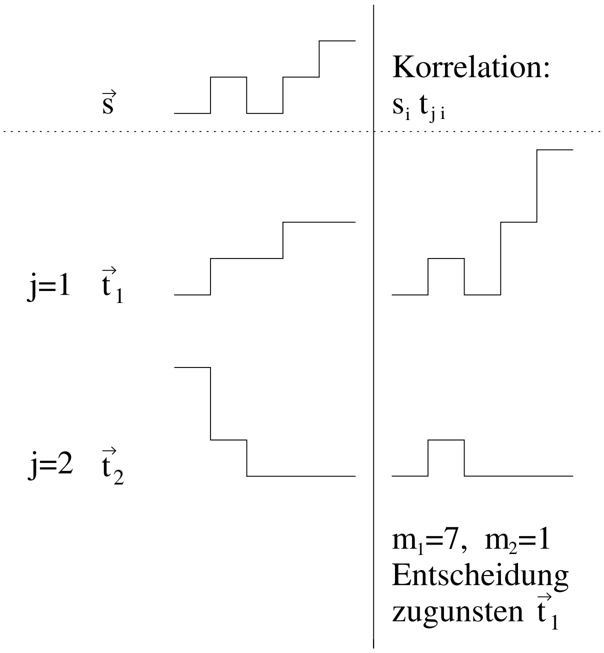

Abbildung 2.8: Korrelation an einem Beispiel

Kapitel 3

Übersicht über den Empfänger

Dieses Kapitel gibt eine Übersicht über den Empfänger. Aus den beiden in der Einleitung erwähnten Grundideen folgen Konsequenzen, aus denen wiederum eine Struktur der Uhr abgeleitet werden kann. Der Empfänger läßt sich demnach zunächst grob in verschiedenene Funktionseinheiten (Blöcke) aufteilen. Daher wird die Uhr durch ein Blockschaltbild modelliert. In einem Blockschaltbild sind verschiedene Funktionseinheiten durch Signalleitungen miteinander verbunden.

Im weiteren Verlauf dieser Arbeit wird „Top-Down" vorgegangen. In diesem Kapitel werden im Sinne einer Spezifikation die *Aufgaben* der einzelnen Blöcke und ihre Beziehung zueinander dargelegt. Es ist in diesem Kapitel weniger wichtig, *wie* diese Blöcke ihre Aufgabe algorithmisch bewerkstelligen. Detailliertere Darstellungen enthalten die Kapitel 4 und 5.

Um Mißverständnissen vorzubeugen sei vorneweg gesagt, daß nicht daran gedacht ist, die im Digitalteil der Uhr vorgestellten Blöcke als (räumlich getrennte) Hardware zu implementieren. Eine solche Strukturierung soll lediglich dazu dienen, die funktionalen Zusammenhänge zwischen digitalen Signalen darzustellen. Tatsächlich hat diese Modellierung ihren Ursprung in der analogen Signalverarbeitung. Die gesamte digitale Datenverarbeitung kann ohne Probleme durch Software realisiert werden. Eine dedizierte Hardware ist nicht notwendig.

Die in der Einleitung eingeführten Grundideen seien noch einmal rekapituliert:

- Die Redundanz des Signals wird ausgenutzt.

- Die Uhr geht genau, da die Taktfrequenz geregelt wird. Durch Abgleich der Taktfrequenz mit der empfangenen atomuhrgenauen Trägerfrequenz von 77.5 kHz wird der Takt auf die exakte Frequenz geregelt. Diese Regelung soll auch dann möglich sein, wenn das Signal stark verrauscht ist.

Der erste Punkt führt zu folgenden Konsequenzen:

Stufe i	f_{si} [Hz]	Bandbreite [Hz]
0	9600	4800
1	1920	960
2	384	192
3	76,8	38,4
4	15,36	7,68
5	3,072	1,536
6	0,6144	0,3972
7	0,12288	0,06144
8	0,024576	0,012288

Tabelle 3.1: Parameter der Stufen der Multiratenfilterkaskade

- Aus dem empfangenen Spektrum ist ein Signal inf (wie Information) zu extrahieren, das einerseits zwar so schmalbandig ist, daß Störungen und Rauschen möglichst gut weggefiltert werden, das andererseits aber noch so breitbandig ist, daß die im Signal vorhandene Nutzinformation (die Uhrzeit) noch erkennbar ist. [BAD87] gibt an, daß sich der wesentliche Anteil des Spektrums des DCF77-Signals im Bereich von 77480Hz bis 77520Hz befindet. Würde man die Bandbreite des Signals inf durch Filterung auf wesentlich weniger als diese 40Hz herabsetzen, so würden die Sekundenflanken sehr flach werden, so daß die Sekundenmarken im Extremfall nicht mehr zu erkennen sind.

  Das demodulierte Signal inf ist einer Funktionseinheit *„Detektor“* zuzuführen. Die Aufgabe des Detektors ist, die übertragene Uhrzeit zu ermitteln, unter Ausnutzung der im Signal enthaltenen Redundanz.

Der zweite Punkt führt zu folgender Konsequenz:

- Aus dem empfangenen Spektrum ist ein Signal tr (wie Träger) herauszufiltern, das nur den Träger des DCF77-Signals enthält. Die Frequenz dieses Signals wird zwecks Regelung mit der Taktfrequenz abgeglichen. Die Filterung kann extrem *schmalbandig* sein, da das Spektrum nach dem Filtern ja nur *eine* Frequenz enthalten soll. Der Anteil des Rauschens im Signal tr ist umso kleiner, je schmaler das verwendete Bandpaßfilter ist. Auf diese Weise wird der obigen Forderung Rechnung getragen, daß die Taktregelung auch dann noch möglich sein soll, wenn das empfangene Signal stark verrauscht ist.

Die Schwierigkeit liegt in der geforderten extremen Schmalbandigkeit des Signales tr. Ein einfaches FIR-Filter zum Beispiel könnte hier nicht eingesetzt werden, da dieses zu viele Koeffizienten hätte, so daß der dementsprechende Rechenaufwand zu groß wäre. Das Problem wird durch eine *Multiratenfilterkaskade* [FLI93] gelöst, vergleiche Kapitel 5. Diese besteht, dem Namen entsprechend, aus einer Kaskade einzelner *Multiratenfilter*, und sie realisiert auf ökonomische Weise einen schmalbandigen Bandpaß. Mit jeder Stufe dieser Kaskade verringert sich sowohl die Bandbreite als auch die Datenrate des Signals um den Faktor *fünf*. Tabelle 3.1 enthält die

Datenraten f_{si} und Bandbreiten am Eingang jeder Stufe. Die Bedingung, daß das Signal tr sehr schmalbandig sein soll, wird durch hinreichende Tiefe der Kaskade erfüllt.

Die Bezeichnung „Multiratenfilter“ rührt daher, daß die Datenraten am Ein- beziehungsweise Ausgang des Filters unterschiedlich sind.

Nun bleibt zu klären, wie die Regelung der Taktfrequenz realisiert werden soll. Wie schon erwähnt, wird diese Regelung durch eine „Phase Locked Loop“ (PLL) erzielt [GAR66][BES93][PRE81]. Eine PLL ist ein Regelkreis. Als Eingabe erhält dieser Regelkreis ein Signal mit einer Referenzfrequenz. Zu einer PLL gehört ein regelbarer Oszillator. Dies ist in diesem Fall der Taktgenerator. Die Aufgabe einer PLL besteht darin, die vom Oszillator erzeugte Frequenz so zu regeln, daß gilt:

- Die Oszillatorfrequenz ist gleich der Referenzfrequenz.
- Die Phase zwischen den beiden Signalen ist nahezu konstant und bekannt.

Hat die PLL den Zustand erreicht, daß beide Bedingungen erfüllt sind, so sagt man, sie hat *gefangen*.

Der zweite Punkt muß näher erläutert werden: Aufgrund von Rauschen kann es zu Abweichungen der Phase vom Sollwert kommen. Dies ist aber nicht weiter schlimm. Wenn die PLL gefangen hat, so bedeutet dies, daß das Referenzsignal und das Oszillatorsignal in beliebig großen Zeiträumen exakt die gleiche Anzahl von Schwingungen durchführen. Geringfügige Abweichungen der Phase zwischen den beiden Signalen ändern daran nichts. Beide Frequenzen sind dennoch gleich.

Genauer gesagt müssen nicht die beiden genannten Frequenzen gleich sein, sondern nur die am Eingang des Regelteils der PLL anliegenden Signale. Frequenzteiler zum Beispiel können Signale so herunterteilen, daß die Signale an diesem Eingang gleich sind, obwohl Oszillator- und Referenzfrequenz verschieden sind. Wird beispielsweise das Oszillatorsignal um den Faktor n geteilt, so wird der Oszillator auf die n-fache Frequenz des Referenzsignals geregelt.

Zur Veranschaulichung sei zunächst angenommen, daß Oszillatorsignal und Referenzsignal die gleiche Frequenz haben sollen, und daß die PLL die Phase auf null Grad regeln soll. Abbildung 3.1a zeigt den Zustand, in dem die Signale die beiden Bedingungen erfüllen, in Abbildung 3.1b hingegen sind die Frequenzen unterschiedlich, und in Abbildung 3.1c ist die Phase nicht null. Die letzteren beiden Konstellationen sind deswegen nicht stationär und würden ausgeregelt werden.

Es bietet sich nun an, das Taktsignal zur im Kapitel 2 eingeführten kohärenten Demodulation zu benutzen. Dies ist die Anwendung des Prinzips des PLL-Empfängers. Die Lösung der Probleme „schmale Bandpaßfilterung“ und „Taktregelung“ bewältigt die im Kapitel 2 angesprochenen grundlegenden Aufgaben „Selektion“ und „Demodulation“.

Wir sind nun soweit, eine detaillierte Struktur für den Empfänger anzugeben, siehe Abbildung 3.2. Die Uhr enthält folgende Blöcke:

- einen regelbaren Taktgenerator

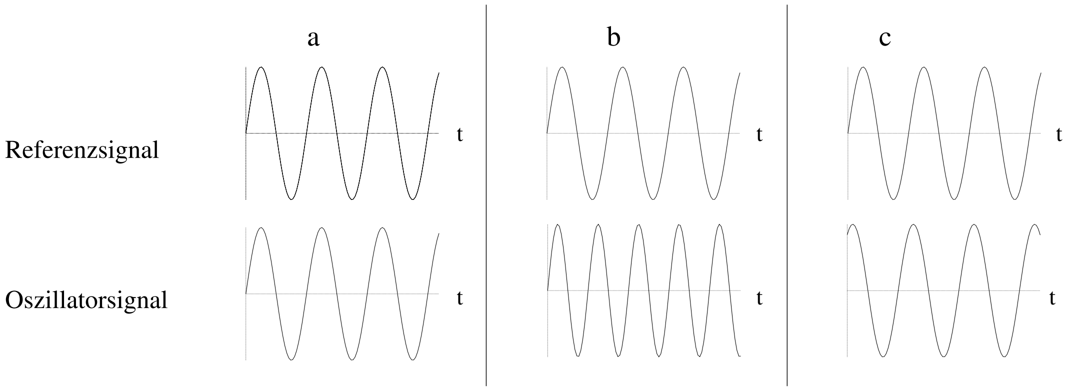

Abbildung 3.1: stationäre und instationäre Zustände der PLL

- einen analogen Frequenzmischer
- die Multiratenfilterkaskade (in der Abbildung die *linke* Kaskade)
- einen PLL-Regler
- einen Demodulator
- einen Detektor
- eine Ausgabeeinheit
- eine Taktteilerkaskade (in der Abbildung die *rechte* Kaskade)
- einen DDS-Block (Direct Digital Synthesis)
- einen Bandpaß (BP) vor dem Demodulator

Die Uhr ist in einen Analogteil und einen Digitalteil geteilt. Diese beiden Teile sind durch A/D- beziehungsweise D/A-Wandler verbunden. Diese Wandler seien mit Aliasfiltern ausgestattet, so daß die Nyquistbedingung in jedem Fall eingehalten wird.

Zunächst sei der Analogteil erläutert:

Das Antennensignal passiert zunächst ein Bandpaßfilter, welches nur aus einer Spule und einem Kondensator besteht. Das auf diese Weise vorselektierte DCF77-Signal wird dann auf die Zwischenfrequenz von 2400Hz herabgemischt[1] und vom A/D-Wandler gesampelt. Der Sampletakt kommt vom regelbaren Taktgenerator. Die Samplefrequenz f_{s0} muß lediglich innerhalb eines kleinen Bereiches um die Mittenfrequenz von 9600Hz variiert werden können. Die Regelspannung für den Taktgenerator wird von dem D/A-Wandler erzeugt und im Digitalteil berechnet.

[1]Das ist nicht ganz richtig. Tatsächlich wird das Signal zunächst analog auf eine Frequenz von 4420Hz herabgemischt. Anschließend wird das Signal noch einmal digital auf die Frequenz 2400Hz herabgemischt. Doch soll die Darstellung zunächst einfach gehalten werden, vergleiche Kapitel 6.

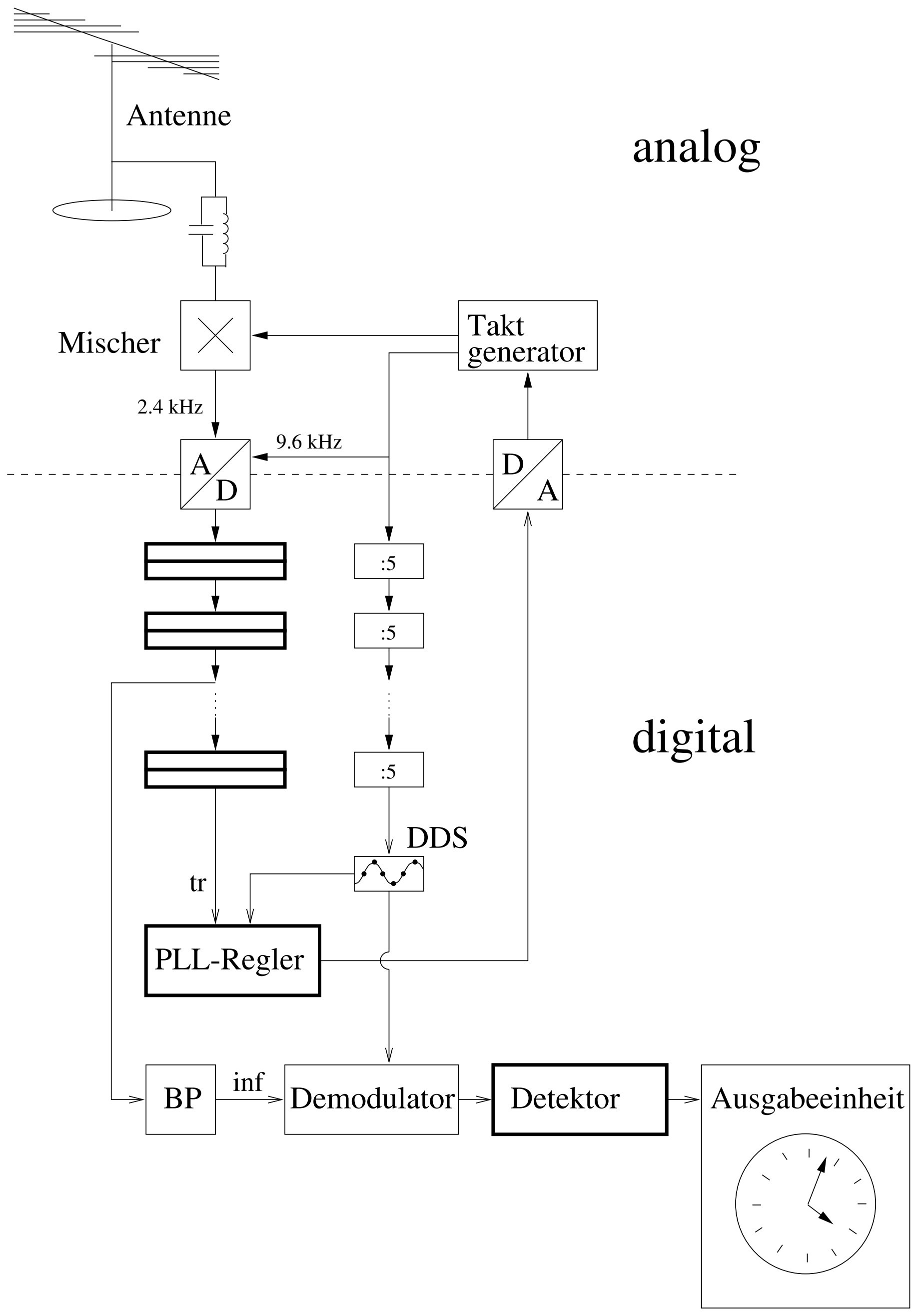

Abbildung 3.2: Grundstruktur des Empfängers

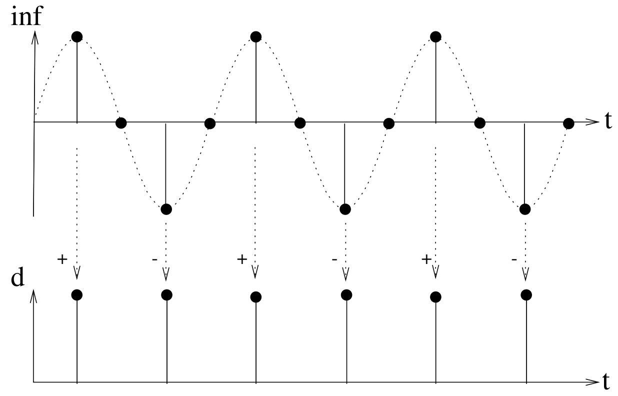

Abbildung 3.3: kohärente Demodulation des Signals inf

Die Samplefrequenz bestimmt die Geschwindigkeit, mit der der Digitalteil arbeitet, und somit die Ganggenauigkeit der Uhr.

Nun zum Digitalteil:

Das Signal inf wird nach der zweiten Stufe der Multiratenfilterkaskade abgegriffen, und hat dort eine Datenrate von 384 Samples pro Sekunde. Allerdings ist die Bandbreite dieses Signals noch zu breit, so daß vor den Demodulator noch ein Bandpaß geschaltet ist, welcher die Bandbreite zusätzlich auf etwa 40Hz herabsetzt. In bezug auf die Bandbreite wäre es günstiger, das Signal erst nach der dritten Stufe abzugreifen. Dann würde die Bandbreite 38.4Hz betragen, und der zusätzliche Bandpaß könnte weggelassen werden. Aber leider wäre die zugehörige Samplefrequenz von 76.8Hz ungeeignet. Der Detektor kann einfacher gestaltet werden, wenn die Anzahl der zu verarbeitenden Samples pro Sekunde ganzzahlig ist.

Die Tiefe der Filterkaskade ist *variabel*. Der PLL-Regler vergleicht das Signal tr aus der Filterkaskade mit der Frequenz, die vom Taktgenerator kommt. Diese muß genau wie die Frequenz des Signals tr pro Stufe um den Faktor fünf herabgesetzt werden. Das Herabsetzen der Frequenz vom Taktgenerator leistet die Taktteilerkaskade. In Kapitel 5 wird begründet, warum die Veränderlichkeit der Kaskadentiefe sinnvoll ist.

Zum Demodulator:
Abbildung 3.3 veranschaulicht das Demodulationsverfahren, welches der in Abschnitt 2.3 eingeführten kohärenten Demodulation entspricht. Das Signal inf ist in unverrauschter Weise dargestellt, so daß man die Trägerfrequenz sehen kann, die ein Viertel der Samplefrequenz beträgt. Durch die PLL wird die Phase dieses Signals so geregelt, daß die Lage der Maxima der Elongation bekannt ist. Das demodulierte Signal d entsteht durch Auswählen jedes zweiten Samples aus dem Signal inf, wobei die Samples in den negativen Positionen invertiert werden. Wie man sieht, entspricht d der Amplitude des Signals inf.

Zum Detektor:
Der Detektor erhält das demodulierte DCF77-Nutzsignal mit einer Datenrate von $\frac{384}{2}$Hz $=$ 192Hz Dieses Signal ist wahrscheinlich stark verrauscht. Die Aufgabe des Detektors ist, die

enthaltene Information über die Uhrzeit zu extrahieren, zu speichern, und über eine zu vereinbarende Schnittstelle an die Ausgabeeinheit weiterzureichen. Um die im Signal enthaltene Redundanz auszunutzen, ist das Signal über einen längeren Zeitraum zu beobachten, notfalls stundenlang. Der Detektor muß also Information über die Vergangenheit des Signals speichern. Weil die empfangene Information verrauscht ist, ist sie unsicher. Dem Detektor fällt die Aufgabe zu, zu entscheiden, ab wann diese unsichere Information eine relativ gesicherte Aussage über die Uhrzeit zuläßt. Wahrscheinlich wird ein Schwellwert verwendet werden. Kapitel 4 enthält die Details.

Zur Ausgabeeinheit:
Es sind diverse Möglichkeiten der Realisierung einer Anzeige der Uhrzeit vorstellbar. In dieser Diplomarbeit soll nicht weiter auf die Ausgabeeinheit und die Schnittstelle zum Detektor eingegangen werden, da dies mit der eigentlichen Aufgabenstellung nichts zu tun hat.

Zum DDS-Block:
Dieser Block erzeugt ein Sinussignal, dessen Frequenz genau ein Viertel der Taktfrequenz beträgt. Im PLL-Regler wird dieses Signal mit dem Signal aus der Multiratenfilterkaskade verglichen. Der DDS-Block läßt sich sehr einfach konstruieren: Er erzeugt nur die Zahlenfolge $0,1,0,-1,0,1,0,-1,\ldots$

3.1 Zusammenfassung

Die Grundidee der Redundanzausnutzung verwirklicht der Detektor. Die Grundidee der Taktregelung verwirklicht der PLL-Regelkreis. Hierzu werden zwei Signale gebildet: ein Signal inf für den Detektor, und ein sehr schmalbandiges Signal tr für den PLL-Regler. Die hohe Schmalbandigkeit des Signals tr vermindert das darin enthaltene Rauschen und wird durch die Kaskadierung mehrerer Multiratenfilter erreicht. Jede Stufe der Kaskade vermindert die Bandbreite des selektierten Frequenzbandes um den Faktor fünf. Die Verwendung einer PLL ermöglicht eine einfache Demodulation.

Die folgenden beiden Kapitel erläutern den Detektor, die Multiratenfilterkaskade und die PLL-Regelung im Detail.

Kapitel 4

Der Detektor

In diesem Kapitel wird der Detektor detailliert betrachtet. Dieser verwirklicht das Ausnutzen der Redundanz des DCF77-Signals. Als Eingabe erhält er das demodulierte DCF77-Signal mit einer Datenrate von 192Hz. Es wird ein Algorithmus angegeben, der nach einer hinreichend langen Zeit als Ausgabe die aktuelle Uhrzeit liefert.

Der Detektor hat nacheinander folgende Aufgaben zu lösen:

- Die Synchronisation auf den Sekundentakt
- Die Synchronisation auf den Minutentakt
- Das Lesen der Zeitinformation

Diesen drei Aufgaben entsprechen drei Funktionseinheiten, aus denen der Detektor besteht. In den Abschnitten 4.1 bis 4.3 wird jede Funktionseinheit einzeln erläutert. In Abschnitt 4.4 wird der Detektoralgorithmus in Form eines imperativen Programms vorgestellt.

Der Prozeß des Detektierens soll erst starten, wenn die PLL gefangen hat, also erst wenn der Demodulator ein korrektes Signal liefern kann. Deswegen erhält der Detektor ein binäres Signal „erfolgreich“, das genau dann auf eins geht, wenn die PLL gefangen hat. Die Synchronisation auf den Minutentakt hingegen kann erst starten, wenn der Empfänger sich auf den Sekundentakt einsynchronisiert hat. Genauso kann das Lesen der Zeitinformation erst beginnen, wenn die Synchronisation auf den Minutentakt abgeschlossen ist. Daraus resultiert die in Abbildung 4.1 dargestellte Struktur des Detektors. Die drei Funktionseinheiten sind *stufenweise* hintereinandergeschaltet. Zwei aufeinanderfolgende Stufen sind mit zwei Signalen verknüpft: dem Sample-Datenstrom, und einem Signal „erfolgreich“, das angibt, ob die vorhergehende Stufe synchronisiert ist, beziehungsweise ob die PLL gefangen hat.

Jede Stufe löst ihre Aufgabe durch Korrelation des demodulierten Signals mit verschiedenen Musterfunktionen.

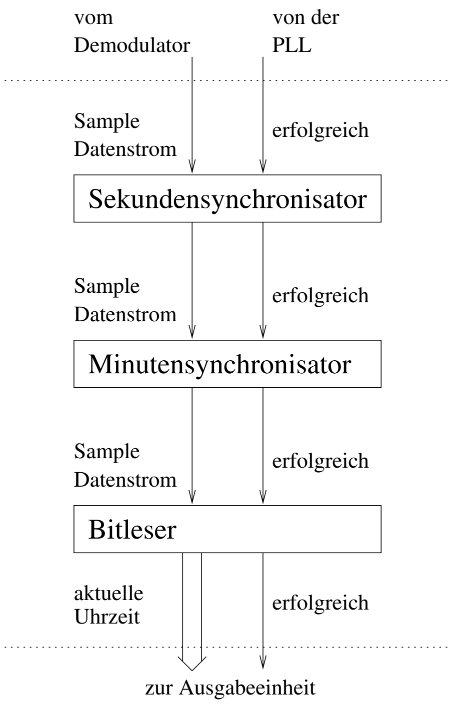

Abbildung 4.1: Struktur des Detektors

4.1 Die Synchronisation auf den Sekundentakt

Im Detektor wird mit jedem Eintreffen eines neuen Samples ein modulo-192-Zähler z_0 inkrementiert. Die Aufgabe der Sekundensynchronisation besteht darin, zu entscheidenen, welchen Wert der Zähler beim Eintreffen einer abfallenden Sekundenflanke hat. Also korreliert der Sekundensynchronisator sein Eingangssignal mit 192 Musterfunktionen und entscheidet eine von ihnen als die richtige. Die Stufe sei *erfolgreich*, wenn es in ihr zu einer solchen Entscheidung gekommen ist. Die 192 Musterfunktionen sind jeweils um eine Position verschoben. Abbildung 4.2 zeigt die zu den Zählerständen null und vierundzwanzig gehörenden Musterfunktionen. Die Amplitude des empfangenen Signals ist im Zeitabschnitt von 0.1 bis 0.2 Sekunden nach jedem Sekundenanfang entweder abgesenkt oder nicht. Deswegen befindet sich der Wert der Musterfunktionen an dieser Stelle in der Mitte zwischen dem Wert für die logische Null und Eins.

Anders als in Kapitel 2 sind die Musterfunktionen potentiell unendlich lang. Bei der Korrelation wird zur Berechnung des Ähnlicheitsmaßes m eine Summe gebildet, zu der ein jedes Sample nach Verknüpfung mit der Musterfunktion einen Beitrag leistet. Mit jedem Eintreffen eines Samples kann also für jede Musterfunktion ein neuer Wert in einen Akkumulator hineinaddiert werden. Anschließend ist jeweils zu entscheiden, ob schon eine Aussage zugunsten einer Musterfunktion $\vec{t_k}$ gemacht werden kann. Wenn dies der Fall ist, und wenn der modulo-192-Zähler z_0 den Wert k annimmt, dann gehört das gerade empfangene Sample zu einem Sekundenanfang. Das Gesagte sei im folgenden formalisiert.

Abbildung 4.2: Musterfunktionen bei der Sekundensynchronisation

Zunächst werden die Musterfunktionen definiert:

$$t_{0,i} = \begin{cases} 0.25 & \text{für } 0 \leq i \bmod 192 < 19 \\ 0.625 & \text{für } 19 \leq i \bmod 192 < 38 \\ 1 & \text{für } 38 \leq i \bmod 192 < 192 \end{cases} \tag{4.1}$$

$$t_{j,i} = t_{0,i-j+192} \text{ für } 0 < j < 192 \tag{4.2}$$

Mit dem Eintreffen jedes i-ten Samples s_i sind zu den 192 Akkumulatoren m_j neue Werte hinzuzuaddieren, gemäß den Reihen

$$m_{j,i} = \sum_i s_i t_{j,i} \tag{4.3}$$

Es ist also das folgende Programmstück auszuführen:

```
for j:=0 to 191 do
  m[j]:=m[j]+s*t(j,i)
i:=i+1
z0:=(z0+1) mod 192
```

Man sagt, es wird auf 192 *Kanälen* korreliert.

Die Akkumulatoren m_j seien alle durchnumeriert.

Der Empfänger entscheidet sich für die Musterfunktion, deren zugehöriger Akkumulator maximal ist. Im folgenden soll begründet werden, warum dieses Verfahren auch bei hohem Rauschen die richtige Entscheidung fällt, sofern es über eine hinreichend lange Zeit angewandt wurde. Dazu sei angenommen, daß bisher schon n Samples zur Korrelation herangezogen wurden, und daß das empfangene Signal $\vec{s}$ der Länge n mit genau einer der 192 Musterfunktionen $\vec{t_r}$ identisch ist, bis auf daß es auch noch additiv durch Rauschen $\vec{r}$ gestört ist, also

$$\vec{s} = \vec{t_r} + \vec{r} \tag{4.4}$$

Dann lautet das Ähnlichkeitsmaß

$$m_j = \vec{s}\vec{t_j} = \sum_i s_i t_{ji} = \sum_i t_{ri} t_{ji} + r_i t_{ji} \tag{4.5}$$

wobei i von null bis $n-1$ läuft. Weiterhin sei angenommen, daß der Erwartungswert $E[r_i]$ einer jeden Komponente aus $\vec{r}$ gleich null ist. Die durch das Rauschen verursachte Abweichung des empfangenen Signals von $\vec{t_r}$ ist also für manche Samples positiv und für andere negativ. Dann folgt für den Erwartungswert von m_j

$$E[m_j] = \sum_i E[t_{ri} t_{ji}] + E[r_i t_{ji}] = \sum_i t_{ri} t_{ji} = \vec{t_r}\vec{t_j} \tag{4.6}$$

Bei Betrachtung der Gleichung 4.5 legt die Intuition schon nahe, daß der Einfluß des Rauschens für große n verschwindet. Da die verschiedenen r_i symmetrisch um null verteilt sind, heben sie sich gegeneinander in der Summe nahezu auf. Diese Intuition soll nun formal begründet werden. Es soll die Varianz $\sigma^2_{mj} = V[m_j]$ berechnet werden. Es wird im folgenden vorausgesetzt, daß die verschiedenen r_i alle stochastisch unabhängig seien, obwohl dies gerade beim Langwellenempfang nur eingeschränkt gilt. Unter dieser Voraussetzung ist $\sigma_r^2 := V[r_i]$ unabhängig von i und es gilt [HIN91]:

$$V[r_i + r_k] = V[r_i] + V[r_k] \tag{4.7}$$

σ_r^2 ist übrigens proportional zur Rauschleistung des Signals. Unter Anwendung von Gleichung 4.7 ergibt sich:

$$V[m_j] = V\left[\sum_i t_{ri} t_{ji} + r_i t_{ji}\right] = \sum_i V[r_i t_{ji}] = \sum_i {t_{ji}}^2 V[r_i] = \sigma_r^2 \sum_i {t_{ji}}^2 \tag{4.8}$$

$$= \sigma_r^2 \vec{t_j}^{\,2} \tag{4.9}$$

Nun wird untersucht, wie sich diese Größen langfristig entwickeln, also für große n. Zunächst halte man sich vor Augen, daß die Musterfunktionen $\vec{t_j}$ sich alle $x = 192$ Elemente wiederholen. Also wiederholt sich auch das Skalarprodukt $\vec{t_k}\vec{t_j}$ zweier Musterfunktionen alle x Elemente. Es sei im folgenden vorausgesetzt, daß die Vektorlänge n ein Vielfaches von x sei, also $n = yx$. Dann sind alle x Elemente langen Teilabschnitte von $\vec{t_k}\vec{t_j}$ konstant:

$$K_{kj} := \sum_{i=mx}^{mx+x-1} t_{ki} t_{ji} \quad \text{für } 0 \leq m < \tfrac{n}{x} \tag{4.10}$$

Also gilt:

$$\vec{t_k}\vec{t_j} = \sum_{i=0}^{n-1} t_{ki} t_{ji} = \frac{n}{x} K_{kj} \tag{4.11}$$

Man kann nun Erwartungswert und Varianz des Ähnlichkeitsmaßes mit Hilfe von K_{kj} als Funktion von n ausdrücken:

$$E[m_j] = \vec{t_r}\vec{t_j} = \frac{n}{x}K_{rj} \tag{4.12}$$

$$V[m_j] = \sigma_r^2 \vec{t_j}^2 = \frac{n}{x}\sigma_r^2 K_{jj} \tag{4.13}$$

Die Standardabweichung $\sigma_{mj} := \sqrt{V[m_j]}$ wächst mit zunehmenden n langsamer als $E[m_j]$. Man kann dies am Variationskoeffizienten erkennen:

$$\lim_{n\to\infty} \frac{\sigma_{mj}}{E[m_j]} = \lim_{n\to\infty} \frac{\sqrt{\frac{n}{x}\sigma_r^2 K_{jj}}}{\frac{n}{x}K_{rj}} = \lim_{n\to\infty} \frac{\sqrt{x\sigma_r^2 K_{jj}}}{\sqrt{n}K_{rj}} \to 0 \tag{4.14}$$

Der Variationskoeffizient ist ein Maß für den Einfluß des Rauschens auf die getroffene Entscheidung. Er ist damit auch ein Maß der Unsicherheit dieser Entscheidung. Daß diese Unsicherheit gegen null strebt, ist der Grund dafür, daß das Korrelationsverfahren auch bei hohem Rauschen das richtige Ergebnis bringt. Aus diesem Grund ist die PLL das Element, welches die Reichweite dieser Funkuhr begrenzt, nicht der Detektor.

Dennoch soll der Korrelationsalgorithmus schon nach endlichem n ein Ergebnis bringen. Der Sekundensynchronisator hat also nicht nur eine richtige aus 192 Musterfunktionen zu finden. Er hat auch zu entscheiden, ab welchem n dieses Ergebnis einigermaßen verläßlich ist. Es ist also ein Kriterium zu finden, das zuverlässig entscheidet, ob der Akkumulator mit dem maximalen Wert auch der richtige ist. Dieses Kriterium heiße Signifikanz.

Der Sekundensynchronisator beobachtet die zeitliche Änderung der Akkumulatorwerte. Bei jedem Eintreffen des i-ten Samples können sich alle Akkumulatorwerte verändern. Es wird dann jeweils das Maximum v_i der Änderung über alle Akkumulatoren berechnet. Signifikanz tritt dann ein, wenn der Abstand a_i des größten Akkumulators zum zweitgrößten hinreichend größer als v_i ist. „Hinreichend größer" kann formalisiert werden durch

$$a_i > cv_i \tag{4.15}$$

wobei c die Rolle eines Schwellwertes zukommt. Weiterhin wurde im Sekundensynchronisator eine Mittelwertbildung von v über mehrere aufeinanderfolgende i durchgeführt, um den Einfluß des Rauschens auf die Berechnung dieses Kriteriums zu mindern.

Zwei Akkumulatoren oder deren zugehörige Musterfunktionen seien „benachbart", genau dann wenn sich ihre Indizes um eins unterscheiden. Dies soll auch modulo der Anzahl der Akkumulatoren gelten, das heißt, der Akkumulator mit dem größten Index ist benachbart zum Akkumulator mit dem Index null. Wenn 192 Musterfunktionen zur Sekundensynchronisation verwendet werden, dann kann es lange dauern, bis sich eine Signifikanz einstellt. Das liegt daran, daß einige Musterfunktionen sich stark ähneln. Benachbarte Musterfunktionen unterscheiden sich nur in drei von 192 Stellen pro Sekunde. Aus diesem Grunde unterscheiden sich auch benachbarte Akkumulatorinhalte wenig voneinander, und daher ist die Signifikanz schwer feststellbar. Dieses

Problem wird dadurch gelöst, daß statt 192 nur 20 Musterfunktionen verwendet werden. Bei 20 Akkumulatoren mit 20 gleichmäßig über eine Sekunde verteilten Musterfunktionen prägen sich Differenzen zwischen benachbarten Akkumulatorinhalten schneller aus.

Ein Problem ist das folgende: Wie ist die Situation zu handhaben, in der nacheinander *verschiedene* Akkumulatoren signifikant werden ? Es muß immer eine Behandlung dieses Falls vorgesehen werden, da das empfangene Signal prinzipiell von stochastischer Natur ist. Die Strategie ist wie folgt: Nach dem Einschalten ist der Sekundensynchronisationszustand „undefiniert“. Sobald das Signifikanzkriterium erfüllt ist, wechselt dieser Zustand auf den Wert „signifikant mit Index k“. Das „erfolgreich“- Signal wechselt von 0 nach 1. Der Wert k berechnet sich aus dem Index des signifikanten Akkumulators. Wie bereits erwähnt, erkennt der Empfänger den Sekundenanfang durch Testen auf Gleichheit von k und dem modulo-192-Zähler z_0. Die Uhr verändert weiterhin mit jedem Eintreffen eines Samples die Inhalte der 192 Akkumulatoren. Falls irgendwann kein Signifikanzkriterium mehr erfüllt ist, so wird der Zustand „signifikant mit Index k“ dennoch beibehalten. Man beachte, daß dieser Zustand auch dann beibehalten wird, wenn ein anderer Akkumulator zwar größer als m_k wird, er aber nicht signifikant wird. Das Verlieren der Signifikanz der Akkumulorinhalte kann zum Beispiel aufgrund von sich ändernden Empfangsverhältissen eintreten. Wenn zum Beispiel das DCF77-Signal zeitweilig ganz aussetzt, so wird die Uhr aufgrund der Taktregelung, nicht übermäßig falsch weiterlaufen. Wird das DCF77-Signal später wieder empfangen, so stellt sich wahrscheinlich wieder eine Signifikanz mit demselben Index k ein. Es kann aber auch sein, daß im Zustand „signifikant mit Index k_1“ eine Signifikanz mit einem anderen Index k_2 eintritt. In diesem Fall wird in den Zustand „signifikant mit Index k_2“ gewechselt. Das „erfolgreich“-Signal geht in diesem Moment einmalig auf 0 und führt zu einem Reset aller nachfolgenden Stufen.

Ein weiteres Problem ist folgendes: Es ist möglich, daß keiner der Akkumulatoren jemals signifikant wird. Dies muß gar nicht am mitempfangenen Rauschen liegen. Dieser Fall kann auch eintreten, wenn das empfangene Signal absolut rauschfrei ist. Das liegt daran, daß der Sekundensynchronisator versucht, dem tatsächlichen Sekundenanfang einen Zeitpunkt in einem Zeitraster mit einer Auflösung von $\frac{1}{20}$ Sekunde zuzuordnen. Es ist nun möglich, daß der tatsächliche Sekundenanfang so ungünstig innerhalb dieses Rasters liegt, daß zwei benachbarte Akkumulatoren gleich schnell wachsen. In diesem Fall ist das empfangene Signal zu zwei benachbarten Musterfunktionen gleich ähnlich. Damit irgendwann also doch einer der Akkumulatoren als signifikant klassifiziert werden kann, sind seine beiden Nachbarn bei der Berechnung der Differenz a auszuklammern. Aufgrund des Rauschens kann es aber nun passieren, daß einmal der eine und einmal der andere Akkumulator den maximalen Wert annimmt. Der oben angeführten Strategie zufolge würde dies zu einem Zustandswechsel und damit zu einem Reset der nachfolgenden Synchronisationsstufen führen. Dies wird vermieden, indem die oben angeführte Zustandsübergangsfunktion wie folgt verändert wird: Ein Zustandswechsel vom Zustand „signifikant mit Index k_1“ in den Zustand „signifikant mit Index k_2“ führt nicht zu einem Nullsetzen des „erfolgreich“-Signals, wenn k_1 und k_2 benachbart sind.

Der Sekundensynchronisator wählt eine Position seines Zeitrasters als die „richtige“, oder besser gesagt „naheliegendste“ aus. Durch einen Indexwechsel zwischen benachbarten Indizes verschiebt sich diese Position des Sekundenanfangs um $\frac{1}{20}$ Sekunde. Dies sollte jedoch beim Lesen

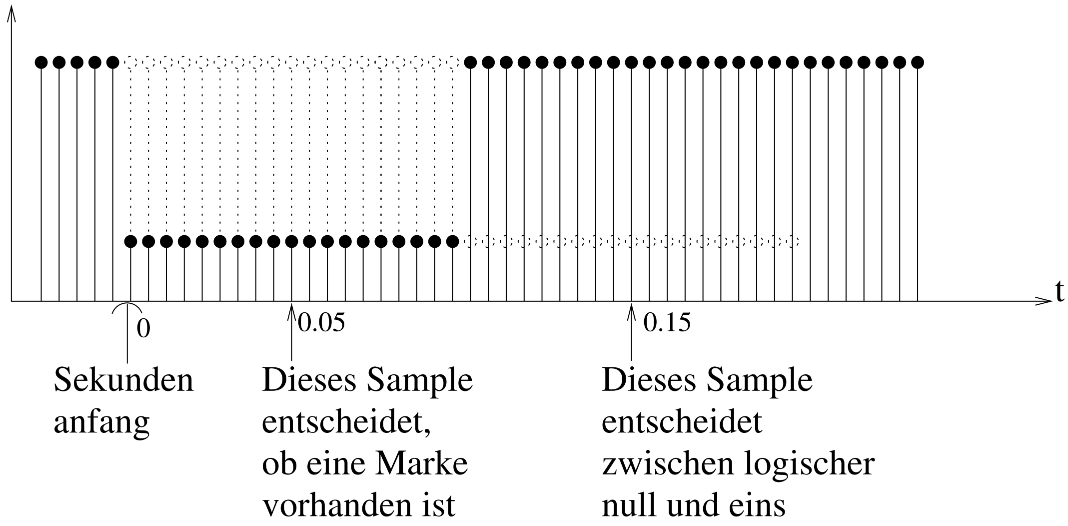

Abbildung 4.3: Eine Sekundenmarke

der einzelnen Bits des Minutenprotokolls nicht zu Problemen führen, da diese Information in einem Zeitraum von $\frac{1}{10}$ Sekunde zur Verfügung steht.

4.2 Die Synchronisation auf den Minutentakt

Zunächst ist festzustellen, welche Merkmale des Minutenprotokolls zum Finden des Minutenbeginns herangezogen werden können: Ein Minutenanfang ist im DCF77-Signal dadurch gekennzeichnet, daß eine Sekunde vorher keine Marke ausgesendet wird. Außerdem sind zwei Bits im Minutenprotokoll auf konstante Werte festgelegt: Das Bit Nummer null hat stets den logischen Wert null, und das Bit Nummer zwanzig hat stets den logischen Wert eins, vergleiche Abschnitt 2.1. Es ist weiterhin noch möglich, die undefinierten Bits im Minutenprotokoll mit den Nummern eins bis vierzehn zur Minutensynchronisation heranzuziehen. Tatsächlich wird derzeit in diesen undefinierten Bits ein konstanter logischer Wert übertragen. Diese Möglichkeit sei aber ausgeschlossen, da sich das von der Physikalisch Technischen Bundesanstalt festgesetzte Protokoll in Zukunft in diesen Bitpositonen noch ändern kann.

Nur die fehlende Minutenmarke weist eindeutig auf einen Minutenanfang hin. Das Muster in den Bitpositionen null und zwanzig wiederholt sich zwar alle sechzig Sekunden, allerdings kann dieses Muster in einem aktuell übertragenem Bitstrom auch um einige Sekunden verschoben an anderer Stelle vorkommen. Zum Beispiel könnte dieses Muster auch an den Bitpositionen 38 und 58 auftauchen, und das auch noch über mehrere Tage. Auch an vielen anderen Bitpositionen könnte dieses Muster über längere Zeit vorkommen. Aus diesem Grund soll einzig und allein das Fehlen der Marke zur Minutensynchronisation herangezogen werden.

Das Verfahren zur Minutensynchronisation ähnelt dem zur Sekundensynchronisation. Die zugehörige Funktionseinheit erhält vom Sekundensynchronisator das „erfolgreich“-Signal und einen Datenstrom mit ausgewählten Samples. In jeder Sekunde wird nur das Sample an den Minutensynchronisator weitergegeben, welches 0.05 Sekunden nach dem Sekundenanfang empfangen wurde, vergleiche Abbildung 4.3. Dieses Sample enthält die Information darüber, ob die aktuelle Marke fehlt oder nicht. Es wäre auch möglich, pro Sekunde mehrere zeitlich nebeneinander liegende Samples zur Korrelation heranzuziehen, zum Beispiel alle Samples zwischen 0.03 und 0.07 Sekunden nach dem Sekundenanfang. Man beachte, daß Abbildung 4.3 ja nur einen unverrauschten Idealfall darstellt. In der Realität haben aufeinanderfolgende Samples nicht alle denselben Wert. Eine Mittelwertbildung mehrerer Samples könnte also ein genaueres Ergebnis liefern. Dennoch soll zur Minutensynchronisation nur ein Sample pro Sekunde herangezogen werden, da bereits durch den Bandpaß vor dem Demodulator eine Art von Mittelwertbildung aufeinanderfolgender Samples geleistet wird. Die zeitliche Unsichertheit des $\frac{1}{20}$-Sekunden Zeitrasters führt dazu, daß die Position des einen benutzten Samples unsicher im Bereich von 0.025 bis 0.075 Sekunden nach dem Sekundenanfang ist. Dies ist ein weiterer Grund dafür, nur ein Sample zu benutzen, insbesondere wenn man zusätzlich noch bedenkt, daß die Flankensteilheit begrenzt ist. Beim Verwenden mehrerer Samples pro Sekunde könnten die ersten aufgrund der zeitlichen Unsicherheit noch zur abfallenden Flanke gehören, und das wäre nicht erwünscht.

Der Minutensynchronisator enthält einen modulo-60-Zähler z_1. Mit dem Datenstrom von 60 Samples pro Minute werden 60 Musterfunktionen korreliert, ähnlich wie bei der Sekundensynchronisation. Die Definition der Musterfunktionen ist:

$$t_{0,i} = \begin{cases} 1 & \text{für } i \bmod 60 = 59 \\ 0 & \text{sonst} \end{cases} \tag{4.16}$$

$$t_{j,i} = t_{0,i-j+60} \quad \text{für } 0 < j < 60 \tag{4.17}$$

Es werden 60 Akkumulatoren verwendet, von denen sich wie beim Sekundensynchronisator einer irgendwann als signifikant herauskristallisieren sollte. Der Minutenanfang findet zu dem Zeitpunkt statt, in dem der Index k_1 des signifikanten Akkumulators gleich dem Wert z_1 des modulo-60-Zählers ist. Auch das „erfolgreich“- Ausgangssignal entspricht seinem Pendant im Sekundensynchronisator.

Aus den Werten z_1 und k_1 ergibt sich die Bitnummer n_2 des aktuellen Bits im Minutenprotokoll durch

$$n_2 = (z_1 - k_1) \bmod 60 \tag{4.18}$$

Im vorigen Abschnitt wurde begründet, warum das Korrelationsverfahren nach hinreichend langer Zeit das korrekte Ergebnis liefert. Dies gilt natürlich auch für die Minutensynchronisation, da das Verfahren dasselbe ist. Allerdings wurde im vorigen Abschnitt vorausgesetzt, daß die in den zur Korrelation herangezogenen Samples enthaltenen Rauschanteile stochastisch unabhängig voneinander seien. Diese Voraussetzung ist beim Langwellenempfang nicht gegeben, wenn die Samplefrequenz 192Hz beträgt. Bei der Minutensynchronisation hingegen wird nur eines dieser 192 Samples pro Sekunde zur Korrelation herangezogen. Die stochastische Abhängigkeit beziehungsweise Kovarianz [HIN91] der Rauschanteile der Samples die zur Minutensynchronisation herangezogen werden, wird wesentlich geringer sein als die stochastische Abhängigkeit der Rauschanteile der Samples für die Sekundensynchronisation. Der Grund dafür ist einfach der, daß die zur Minutensynchronisation herangezogenen Samples zeitlich 192-mal weiter auseinanderliegen. Die Begründung dafür, daß das Korrelationsverfahren gut arbeitet ist also im Falle der Minutensynchronisation gegeben. Beim Lesen der Zeitinformation beträgt der Abstand der benutzten Samples eine Minute, so daß die Kovarianz noch geringer sein wird, vergleiche Abschnitt 4.3.

Die Sekundensynchronisation ist viel weniger kritisch als die Minutensynchronisation. Schließlich wird in jeder Sekunde eine Sekundensynchronisationsmarke ausgesendet, aber nur in jeder Minute eine fehlende Marke zur Minutensynchronisation.

Die im vorigen Abschnitt hergeleitete *quantitative* Analyse der zeitlichen Entwicklung des Variationskoeffizienten der Akkumulatorinhalte scheint ohne die gemachte Voraussetzung der stochastischen Unabhängigkeit undurchführbar.

4.3 Das Lesen der Zeitinformation

Auch die Funktionseinheit „Bitleser“ erhält vom Sekundensynchronisator ein Sample pro Sekunde. In jeder Sekunde wird das Sample an den Bitleser weitergegeben, welches 0.15 Sekunden nach einem Sekundenanfang empfangen wurde. Dieses Sample entspricht dem logischen Wert des aktuellen Bits im Minutenprotokoll, vergleiche Abbildung 4.3. Außerdem kennt der Bitleser die Bitposition n_2 des aktuellen Bits im Minutenprotokoll.

Im folgenden wird gezeigt, wie der Grundgedanke „Ausnutzen der Redundanz“ umgesetzt wird: Die Detektion eines gesendeten Zeichens funktioniert umso besser, je mehr Information über die Gestalt dieses Zeichens bekannt ist. Die wesentliche Erkenntnis ist, daß ein gesendetes „Zeichen“ aus der Bitinformation mehrerer aufeinanderfolgender Minuten zusammengesetzt sein kann. Die Bits mit den Nummern 21 bis 24 im Minutenprotokoll enthalten die Einerstelle der Minuten. Die Einerstelle der Minuten zählt ständig von null bis neun. Am Anfang des Leseprozesses ist nichts über die aktuelle Uhrzeit bekannt. Das einzige, was dem Empfänger über die empfangene Information bekannt ist, ist, daß von Minute zu Minute das Bit Nummer 21 seinen Wert ändert. Man sagt, es toggelt. Dieses Bit ist das „least significant“-Bit in der Kodierung der Uhrzeit. Der Empfänger betrachtet zunächst nur jeweils das Bit Nummer 21 einer jeden Minute. Dann kann man sagen, daß der Sender nur zwei verschiedene potentiell unendlich lange Zeichen sendet, entsprechend den Folgen 0,1,0,1,0,1,... und 1,0,1,0,1,0,.... Durch Korrelation mit den beiden Musterfunktionen

$$t_{0,i} = \begin{cases} 0 & \text{für } i \text{ gerade} \\ 1 & \text{sonst} \end{cases} \tag{4.19}$$

$$t_{1,i} = t_{0,i+1} \tag{4.20}$$

und Anwendung des Signifikanzkriteriums kann entschieden werden, ob das gerade empfangene Bit Nummer 21 eine eins oder eine null war, auch wenn das aktuelle Sample für dieses Bit so stark verrauscht ist, daß es für sich alleine betrachtet keine Information hergibt.

In Analogie zum Detektor ist auch der Bitleser stufenweise aufgebaut. Jede Stufe kann nur dann arbeiten, wenn alle ihre Vorgängerstufen erfolgreich sind. Die erste Stufe im Bitleser ist für die Detektion des Bits Nummer 21 zuständig. Die nächste Stufe detektiert die Bits 22 bis 24.

Nachdem der Empfänger erkannt hat, wann das least-significant Bit von eins auf null wechselt, ist ihm auch bekannt, wann die Bits mit den Nummern 22 bis 24 ihren Wert verändern. Auch diese Bits folgen einem sich immer wiederholenden Muster: Sie zählen ständig von null bis vier. Entsprechend der binären Kodierung der Zahlen null bis vier lassen sich folgende Musterfunktionen aufstellen:

$$\vec{\vec{t}}_0 = \left\langle \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, \ldots, \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \ldots \right\rangle \tag{4.21}$$

$$\vec{t}_{j,i} = \vec{t}_{0,i-2j+10} \text{ , für } 0 < j < 5 \tag{4.22}$$

Man beachte, daß die Elemente der Vektoren $\vec{\vec{t}}_j$ wiederum Vektoren sind. Die Korrelationsmethode ist auch in diesem Fall anwendbar. Die fünf Akkumulatoren ergeben sich entsprechend Gleichung 4.3 zu

$$m_{j,i} = \sum_i \begin{pmatrix} s_{24,i} \\ s_{23,i} \\ s_{22,i} \end{pmatrix} \vec{t}_{j,i} \tag{4.23}$$

Der linke Vektor in dieser Gleichung wird gebildet aus den Samples der Bits 22 bis 24 der i-ten Minute. In jeder Minute sind also die Skalarprodukte dieses Vektors mit den zugehörigen Vektoren der fünf Musterfunktionen $\vec{\vec{t}}_j$ zu bilden und in die entsprechenden Akkumulatoren hineinzuaddieren. Es sollte sich irgendwann wieder eine Signifikanz ergeben, so daß eine der Musterfunktionen als die richtige entschieden werden kann, und somit die Bits 21 bis 24 bestimmt werden können.

Nachdem die Bits 22 bis 24 erfolgreich gelesen wurden, kann mit dem Lesen des Bits 25 fortgefahren werden. Dieses Bit toggelt alle zehn Minuten. Es ist also wieder mit zwei Musterfunktionen zu korrelieren. Das weitere Vorgehen beim Lesen der Zeitinformation ist offensichtlich. Der Werteverlauf anderer Bitgruppen besteht aus regelmäßigen Wiederholungen. Für jede Bitgruppe existiert eine Stufe, die eine Korrelation durchführt, in Analogie zu den bisher erläuterten Bitgruppen. Am Ende dieses Prozesses sind alle Bits für die Minuten- und Stundenanzeige der Uhr erfolgreich gelesen. Tabelle 4.1 enthält die Aufteilung in die sechs Bitgruppen. Warum die Sekunden- und Minutensynchronisationsstufen mit den Stufennummern null und eins auch in dieser Tabelle enthalten sind, wird weiter unten erläutert.

Stufe i	Bitnummer	$\hat{z}_i$	$\hat{m}_i$	Wiederholungs-zeitraum	w_i	Bedeutung
0		192	20	1 sek		Sekundensynchronisation
1		60	60	1 min	1	Minutensynchronisation
2	21	2	2	2 min	2	Minute Wertigkeit 1
3	22…24	5	5	10 min	10	Minute Wertigkeiten (2,4,8)
4	25	2	2	20 min	20	Minute Wertigkeit 10
5	26…27	3	3	60 min	60	Minute Wertigkeiten (20,40)
6	29	2	2	2 h	120	Stunde Wertigkeit 1
7	30…34	12	12	24 h	1440	Stunde Wertigkeiten (2,4,8,10,20)

Tabelle 4.1: Parameter der Stufen im Detektor

Der Vorgang des Lesens der Bits 22 bis 24 darf nur in synchronisierter Weise mit dem inzwischen bekannten Verlauf des Bits 21 stattfinden. Nur wenn dieses Bit von eins nach Null wechselt, kann ja ein Wechsel der Bits 22 bis 24 eintreten. Zur Erläuterung, wie diese Forderung eingehalten werden kann, muß etwas näher auf Implementationsaspekte des Detektors eingegangen werden:

So wie die beiden Stufen zur Sekunden- und Minutensynchronisation besitzen auch alle anderen Stufen des Bitlesers einen modulus-Zähler z_i. Dieser Zähler zählt immer von null bis zum in Tabelle 4.1 angegebenen modulus-Wert $\hat{z}_i$ minus eins. Die Zähler zählen jedoch unterschiedlich schnell. w_i ist der Zeitraum eines vollen Zähldurchgangs in Minuten, siehe Tabelle 4.1. Jede Zählstufe i erhält einen Wert n_i von ihrer Vorgängerstufe, sofern diese erfolgreich ist. Dieser Wert n_i gibt an, wieviele Minuten schon vergangen sind, seitdem der Wiederholungszeitraum w_{i-1} der Stufe $i-1$ angefangen hat. Genau dann, wenn die Stufe i den Wert $n_i = 0$ erhält, wird ihr modulus-Zähler z_i inkrementiert.

Beispiel 4.1 *Der Wiederholungszeitraum der Stufe Nummer drei ist zehn Minuten lang. Ein solcher Abschnitt beginnt um 11:20. Der Wert n_4, den diese Stufe an die nächste um 11:25 weitergibt, ist 5. Um 11:30 ist n_4 gleich null, und der modulus-Zähler z_4 der Stufe Nummer vier wird inkrementiert. z_4 wird also zum Beginn jeder vollen zehn Minuten inkrementiert. Dieser Zähler toggelt immer nur zwischen null und eins. Entsprechend wird der modulus-Zähler z_5 dreimal pro Stunde inkrementiert, nämlich zur vollen Stunde, um zwanzig nach und um vierzig nach.*

Dieses Prinzip ähnelt übrigens dem Zeitzählverfahren in mechanischen Uhren. Die Berechnung des Terms $\vec{t}_{j,i}$ in Gleichung 4.23 hängt auf diese Weise nur von dem Zähler z_i und den Musterfunktionen der betreffenden Stufe ab. In Stufe drei (für die Bits 22 bis 24) zum Beispiel:

$$m_{j,i} = \sum_i \begin{pmatrix} s_{24,i} \\ s_{23,i} \\ s_{22,i} \end{pmatrix} \vec{t}_{(z_3 - j + 5) \bmod 5} \tag{4.24}$$

mit

$$\vec{t} = \left\langle \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \right\rangle \tag{4.25}$$

Aus den Werten n_i kann die Uhrzeit abgeleitet werden und zur Ausgabeeinheit übertragen werden.

Die Bits im Minutenprotokoll, die die Information über das Datum beinhalten, sind nicht so regelmäßiger Natur. Die Monate haben jeweils eine unterschiedliche Anzahl von Tagen, und außerdem hängt die Anzahl der Tage im Februar auch noch vom Jahr ab. Im derzeit gültigen Gregorianischen Kalendersystem wiederholt sich der Verlauf der Tage erst alle vierhundert Jahre (nicht jedes vierte Jahr ist ein Schaltjahr [HAH89]). Wollte man diese Regelmäßigkeit in der Uhr ausnutzen, dann müßte man auf $303 \cdot 356 + 97 \cdot 357 = 142497$ Kanälen korrelieren . Von diesem Aufwand soll hier abgesehen werden. Die Datumsinformation kann anstatt durch Korrelation durch Mittelwertbildung gewonnen werden. Nachdem alle Stufen im Bitleser erfolgreich geworden sind, kennt der Empfänger die Zeitdauer bis zum Beginn des nächsten Tages. Innerhalb dieser Zeit ist die Information über Wochentag, und Datum konstant. Es wird über diese Zeit für die betreffenden Bitpositionen der arithmetische Mittelwert der Samples gebildet. Die verschiedenen Samples, die zu einer Bitposition gehören, sind durch Rauschen additiv überlagert. Diese Abweichungen vom gesendeten Signal sind manchmal positiv und manchmal negativ, so daß das Rauschen im Mittelwert gegenüber dem Nutzsignal abgeschwächt ist. In der bei der Mittelwertbildung ausgeführten Summation heben sich die durch das Rauschen verursachten positiven und negativen Abweichungen gegenseitig auf. In Analogie zur Korrelation strebt auch bei der Mittelwertbildung der Variationskoeffizient gegen null. Der so erhaltene Mittelwert wird für jedes betreffende Bit einmal pro Minute neu berechnet. Zur Entscheidung, ob diese Bits null oder eins sind, können die Mittelwerte der Bits mit den Nummern null und zwanzig herangezogen werden. Diese Bits haben die konstanten logischen Werte null und eins. Genauso wie in den verschiedenen Korrelationsstufen kann auch bei der Mittelwertbildung ein Maß für die Varianz gebildet werden, indem die zeitliche Änderung des Mittelwertes verwendet wird.

4.4 Der Detektoralgorithmus

Man kann die verschiedenen Stufen im Detektor *generalisieren*. Dann hat die generalisierte i-te Stufe folgende Eigenschaften:

- Es gibt drei Verbindungen zur Vorgänger- und Nachfolgerstufe: den Datenstrom mit den Samples, das „erfolgreich“-Signal und das Signal n_i.

- Es gibt einen Zähler z_i und einen modulus-Wert $\hat{z}_i$. Der Zähler wird modulus $\hat{z}_i$ hochgezählt, wenn n_i gleich null ist

- Es gibt $\hat{m}_i$ Akkumulatoren und Musterfunktionen. Für $i > 0$ ist $\hat{m}_i = \hat{z}_i$, siehe Tabelle 4.1. Die Musterfunktionen unterscheiden sich nur durch eine Verschiebung in der Abszisse. Die Samples aus dem Eingangsdatenstrom werden mit den Musterfunktionen korreliert. Die Stufe ist erfolgreich, wenn einer der Akkumulatoren signifikant ist. Dementsprechend hat die Stufe die $\hat{m}_i + 1$ Zustände „undefiniert" und „signifikant mit Index k_i". Die in Abschnitt 4.1 vorgestellte Strategie legt die Abhängigkeit des „erfolgreich"-Ausgangssignals von den Zustandsübergängen fest. Hat das „erfolgreich"-Eingangssignal den Wert null, so ist ein Reset durchzuführen, das heißt, die Akkumulatoren und das „erfolgreich"-Ausgangssignal sind auf null zu setzen. Dies führt also auch zu einem Reset aller nachfolgenden Stufen. Nicht alle Samples des Eingabedatenstroms müssen auf den Ausgabedatenstrom gegeben werden.

- Ab der Stufe Nummer zwei benötigen die Stufen die Information darüber, zu welcher Bitposition im Minutenprotokoll das gerade erhaltene Sample korrespondiert. Diese Information enthält n_2. Deswegen kann jede Stufe auf diesen Wert zugreifen.

Es besteht ein Unterschied zwischen der ersten Stufe und den anderen. Die Sekundensynchronisationsstufe erzeugt nicht einen sondern zwei Ausgangsdatenströme: einen für die Minutensynchronisationsstufe, und einen für die Stufe Nummer zwei, vergleiche Abbildung 4.3 auf Seite 30.

Man gewinnt mit dieser generalisierten Sicht eine Regelmäßigkeit, die sich bei der Implementierung als Programm als vorteilhaft erweist.

Der Algorithmus in Abbildung 4.4 implementiert den Detektor. Die Konstanten $\hat{z}_i$, $\hat{m}_i$ und w_i sind in dieser Abbildung noch einmal tabellarisch aufgeführt. Der Algorithmus wird nun erläutert. Zunächst sei angenommen, daß die Sekundensynchronisationsstufe noch nicht erfolgreich sei. Der Programmzähler befinde sich gerade in Zeile 3. Nach dem Eintreffen des nächsten Samples werden in den Zeilen 6 bis 12 folgende Aktionen ausgeführt:

- Der modulo-192-Zähler z_0 wird inkrementiert (Zeile 6).

- Die Werte der 20 Akkumulatoren dieser Stufe werden durch Korrelation mit den Musterfunktionen verändert (Zeile 8 und 9). Die Variable i zählt in der WHILE-Schleife die Detektorstufen durch. Da sie den Wert 0 hat, liefert die Funktion t (Zeile 9) die Musterfunktionen für die Sekundensynchronisationsstufe entsprechend Abbildung 4.2 auf Seite 26. Dabei spielt der vierte Parameter n_2, der die aktuelle Bitposition im Minutenprotokoll enthält, keine Rolle. n_2 ist noch undefiniert. In der FOR-Schleife wird auch das Maß für die Varianz v_i berechnet (Zeile 10). v_i ist die maximale Änderung eines Akkumulatorinhaltes über alle j, vergleiche Abschnitt 4.1.

- Da sich die Inhalte der Akkumulatoren der Sekundensynchronisationsstufe geändert haben, kann sich auch der Zustand dieser Stufe geändert haben. In den Zeilen 11 und 12 wird der neue Zustand berechnet. Falls unter Berücksichtigung des Varianzmaßes v einer der Akkumulatoren signifikant ist, so wird in Zeile 11 die Boolesche Variable „signifikant"

```
Initialisierung
LOOP forever
        $s$ :=WarteAufSample()
        $i := 0$; $n_i := 0$; weiter:=true
        WHILE weiter
                if $n_i = 0$ then $z_i := (z_i + 1)$ mod $\hat{z}_i$
                if $i < 2$ or „$n_2$ ist Bitposition, die in Stufe $i$ korreliert wird“
                        FOR j:=0 to $(\hat{m}_j - 1)$
                                $m_{ij} := m_{ij} + s \cdot t(i, z_i, j, n_2)$
                                „Berechne Maß $v_i$ für Varianz“
                        ($k_i$,signifikant):=BerechneSignifikanz($m_i$,$v_i$)
                        (zustand$_i$,erfolgreich$_i$):=Folgezustand(zustand$_i$,$k_i$,signifikant)
                if erfolgreich$_i$
                        if    $i = 0$ then $n_1$     := 0
                                           weiter :=„aufhören oder fortfahren mit
                                           i      := Stufe $1$ ($0.05$ sek.) oder $2$ ($0.15$ sek.) “
                        elsif $i = 1$ then $n_2$     := $(z_1 - k_1)$ mod $60$
                                           weiter := false
                        elsif $i < 7$ then $n_{i+1}$ := $((z_i - k_i)$ mod $\hat{z}_i)w_{i-1}$ + (if $(i = 2)$ $0$ $n_i)$
                                           weiter := true
                                           i      := $i + 1$
                        elsif $i = 7$ then weiter :=false
                else
                        „Setze Stufen $i + 1$ bis $7$ zurück“
                        weiter:=false
```

Stufe i	0	1	2	3	4	5	6	7
$\hat{z}_i$	192	60	2	5	2	3	2	12
$\hat{m}_i$	20	60	2	5	2	3	2	12
w_i		1	2	10	20	60	120	1440

Abbildung 4.4: Der Detektionsalgorithmus

auf „wahr“ gesetzt, und k erhält den betreffenden Index. Die Zustandsübergangsfunktion (Zeile 12) ist in Abschnitt 4.1 spezifiziert.

Solange die Korrelation nicht zum Erfolg führt, werden diese drei Schritte mit jeweils neuen Samples wiederholt. Die WHILE-Schleife führt dann immer nur eine Iteration aus. Irgendwann führt die Korrelation aber zu einer Signifikanz und damit zum Erfolg. Dann wird zum ersten mal der positive Zweig der „if“-Anweisung mit $i = 0$ betreten (Zeilen 14 bis 22). Es ist nun bekannt, welches der 192 Samples pro Sekunde auf einen Sekundenanfang fällt. In 190 von 192 Fällen wird die Variable „weiter“ auf „false“ gesetzt (Zeile 15). Nur in zwei Fällen wird „weiter“ auf „true“ gesetzt, vergleiche Abbildung 4.3. Im ersten Fall entscheidet das aktuelle Sample s darüber, ob eine Marke vorhanden ist, oder nicht. Im zweiten Fall entscheidet das aktuelle Sample zwischen logischer null und logischer eins. Der Empfänger erkennt beide Fälle an den Werten von z_0 und k_0. Die beiden Fälle sollen getrennt betrachtet werden:

- Das aktuelle Sample wurde 0.05 Sekunden nach einem Sekundenanfang empfangen: Dieses Sample ist zur Minutensynchronisation heranzuziehen. Mit dem Wert $i = 1$ wird die WHILE-Schleife ein zweitesmal iteriert. Dies geschieht einmal pro Sekunde. Da n_1 immer null ist, wird der modulo-60-Zähler z_1 immer inkrementiert (Zeile 6). In den Zeilen 7 bis 12 wird die Korrelation mit den Musterfunktionen der Minutensynchronisationsstufe durchgeführt, und der Zustand dieser Stufe neu berechnet. In Abhängigkeit vom ersten Parameter, der Stufennummer, enthält die Funktion t alle vorkommenden Musterfunktionen. Bei erfolgreicher Minutensynchronisation wird in Zeile 17 die aktuelle Bitnummer im Minutenprotokoll n_2 berechnet. Da das aktuelle Sample s in den nachfolgenden Bitlesestufen nicht verwendet werden kann, wird „weiter“ auf jeden Fall auf „false“ gesetzt, so daß die WHILE-Schleife terminiert.

- Das aktuelle Sample wurde 0.15 Sekunden nach einem Sekundenanfang empfangen: Nur wenn die Minutensynchronisation erfolgreich ist, darf in diesem Fall „weiter“ auf „true“ und $i = 2$ gesetzt werden. Die WHILE-Schleife wird dann mit dem Wert $i = 2$ betreten. Das Lesen der Zeitinformation beginnt also. Man betrachte Zeile 6. Nur wenn die aktuelle Bitnummer im Minutenprotokoll n_2 gleich null ist, wird der Zähler z_2 inkrementiert, oder besser gesagt, zwischen null und eins getoggelt. Dies findet nur zum Minutenanfang statt, also einmal pro Minute. Man nehme nun aber an, daß die Bitposition im Minutenprotokoll nun gerade 21 sei, so daß s gerade dem zur Korrelation in dieser Stufe benötigten least-significant-Bit entspricht. Nur in diesem Fall wird der positive Zweig der „if“-Anweisung in den Zeilen 8 bis 12 ausgeführt, der die Korrelation berechnet. Ist diese Stufe nicht erfolgreich, so terminiert die WHILE-Schleife. Andernfalls werden „weiter“ auf „true“ und $i = 3$ gesetzt (Zeilen 20 und 21). Vorher wird aber noch der Wert n_3 berechnet. Da $i = 2$ und $w_2 = 1$ sind, wird n_3 auf den Wert $((z_2 - k_2) \bmod 2)$ gesetzt, also entsprechend der Signifikanz in den beiden Akkumulatoren dieser Stufe auf null oder eins.

  Wenn die Stufe Nummer zwei erfolgreich war, so wird eine Sekunde später (Bit Nummer 22) die WHILE-Schleife viermal iteriert, so daß sie mit $i = 3$ betreten wird. Das Sample s wird dann in Stufe zwei nicht zur Korrelation herangezogen, wohl aber in Stufe drei (Zeile 7).

> In der WHILE-Schleife kann eine Stufe also nur dann korrelieren, wenn alle Vorgängerstufen erfolgreich sind. Sind zum Beispiel alle Stufen erfolgreich, so wird die WHILE-Schleife einmal pro Sekunde für jede Stufe einmal durchlaufen. Ein Sample kann nur in einer der sechs Bitlesestufen zur Korrelation herangezogen werden (Zeile 7). Tatsächlich können aber alle modulo-Zähler der acht Stufen mit nur einem neuen Sample inkrementiert werden.

Beim Lesen der Bits werden in diesem Algorithmus alle Samples jeweils nur in *einer* Stufe zur Korrelation verwendet, und zwar in der Stufe, die nicht erfolgreich ist, aber deren Vorgängerstufe erfolgreich ist. Der Algorithmus kann verbessert werden, indem alle diese Samples schon während der Minutensynchronisation *gespeichert* werden. Wenn die Vorgängerstufe der Stufe Nummer i erfolgreich geworden ist, so können dann in Stufe i nicht nur die zukünftig kommenden Samples zur Korrelation herangezogen werden, sondern auch die bereits gespeicherten. Auf diese Weise kann die Zeit, die zum Lesen der Bits benötigt wird, reduziert werden. Allerdings wurde diese Methode im Rahmen dieser Arbeit nicht weiter verfolgt.

Kapitel 5

Die Multiratenfilterkaskade und die PLL

In diesem Kapitel werden die Multiratenfilterkaskade und der PLL-Regler erläutert. Diese beiden Blöcke werden in den Abschnitten 5.1 und 5.2 zunächst getrennt erörtert. Ein sehr interessanter Aspekt des Empfängers ist aber das Zusammenspiel dieser beiden Blöcke, auf das in Abschnitt 5.3 eingegangen wird.

5.1 Die Multiratenfilterkaskade

Die Multiratenfilterkaskade hat die Aufgabe, aus dem empfangenen Spektrum das möglichst schmalbandige Signal tr zu gewinnen. Das Grundprinzip einer solchen Kaskade ist die in Kapitel 2 eingeführte Unterabtastung. Dort wurde allerdings die Unterabtastung eines analogen Signals erläutert. Die Unterabtastung digitaler Signale nennt man *Dezimation*. Unterabtastung und Dezimation haben ähnliche Auswirkungen auf die resultierenden Spektren. Ein digitales Signal ist letztendlich ein Datenstrom von Digitalzahlen, mathematisch gesprochen also eine Zahlenfolge x_i. Durch Dezimation mit dem Faktor n entsteht aus dem Signal x_i das Signal y_i, in dem nur jeder n-te Wert aus x_i auftaucht, also

$$y_i = x_{n \cdot i} \tag{5.1}$$

Beispiel 5.1 *Durch Unterabtastung des Signals 0,1,5,4,2,0,-1,0,-3,2,... mit dem Faktor 3 entsteht das Signal 0,4,-1,2,.... Die restlichen Werte werden einfach „vergessen“.*

Durch Unterabtastung verringert sich die Datenrate beziehungsweise die Samplefrequenz um den Faktor n. Wie schon erwähnt, wurde für n der Wert *fünf* gewählt.

Zu jeder Stufe der Filterkaskade gehört ein sogenanntes Dezimationsfilter. Dieses filtert aus dem Eingangsspektrum zunächst zwei Bänder heraus. Anschließend folgt die Dezimation. Das als Beispiel gewählte Spektrum a in Abbildung 5.1 resultiert nach dem Herausfiltern der zwei Bänder in Spektrum b. Spektrum c entsteht aus der anschließenden Unterabtastung. Wie man sieht, hat Spektrum c bis auf den Skalierungsfaktor fünf die gleiche Gestalt wie Spektrum a. Der

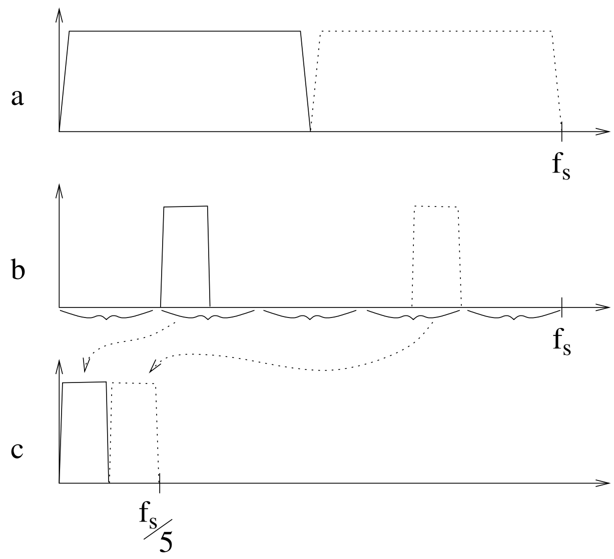

Abbildung 5.1: Dezimation durch 5

Ausgang einer solchen Filterstufe kann also gleich zur nächsten Stufe geleitet werden. Wegen der Form des Dezimationsfilters wird gefordert:

> *Die Mittenfrequenz des herausgefilterten Frequenzbandes muß ein Viertel der Samplefrequenz betragen.*

Jede Stufe dieser Kaskade schmälert das Eingangsfrequenzband um den Faktor fünf. Also realisiert eine Kaskade von n Stufen einen Bandpaß mit einem 5^n-tel der Bandbreite des Eingabespektrums. Abbildung 5.2 verdeutlicht diesen Sachverhalt. Das Spektrum c in dieser Abbildung ist durch zweistufiges Dezimieren aus Spektrum a hervorgegangen, und enthält nur ein sehr schmales Band aus Spektrum a. Die Bedingung, daß das Signal tr sehr schmalbandig sein soll, wird durch hinreichende Tiefe der Kaskade erfüllt.

Das Dezimationsfilter muß relativ zur Samplefrequenz *nicht* sehr schmalbandig sein, und kann deswegen als FIR-Filter mit wenigen Koeffizienten implementiert werden. In dieser Anwendung ist es wichtig, die Anzahl der Koeffizienten zu minimieren, vergleiche Abschnitt 5.3. Der Betragsfrequenzgang aus Abbildung 5.3a ist besonders geeignet für die Multiratenfilterung mit dem Dezimationsfaktor fünf. Man stelle sich vor, daß das zu filternde Spektrum aus einem „Nutzanteil“ und einem „Störanteil“ besteht. Der Nutzanteil sei ein sehr schmaler Bereich mit Frequenzen in der Nähe von einem Viertel der Samplefrequenz, der Störanteil enthalte alle anderen Frequenzen. Die Multiratenfilterkaskade soll den Nutzanteil möglichst gut durchlassen, und den Störanteil möglichst gut wegfiltern. Man stelle sich ein Signal vor, in dessen Spektrum alle Frequenzen gleichmäßig verteilt vorkommen. Dieses Signal wird nun durch eine Stufe der Multiratenfilterkaskade geschickt, das heißt, es wird zunächst mit dem Dezimationsfilter vorgefiltert,

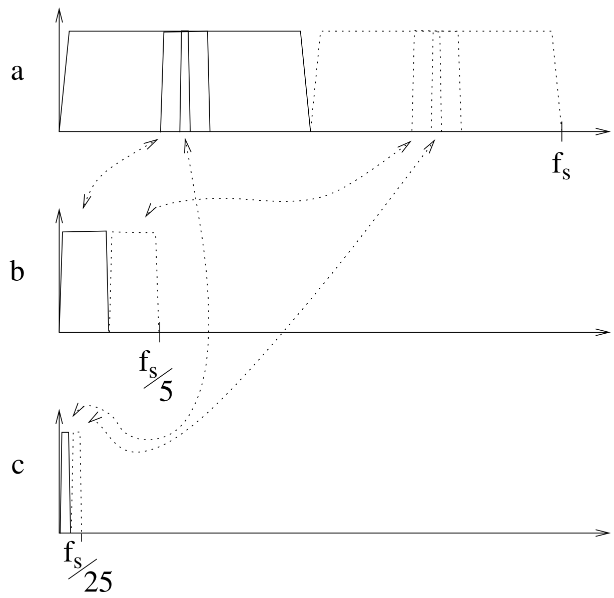

Abbildung 5.2: Schmalbandige Filterung durch Kaskadierung mehrerer Stufen

und anschließend dezimiert. Die fünf Teilstücke, aus denen das resultierende Spektrum zusammenaddiert wird, sind in den Abbildungen 5.3b bis 5.3f dargestellt. Der schraffierte Teil soll dabei jeweils den Nutzanteil darstellen. Man sieht, daß in der Summe dieser fünf Teilspektren der Störanteil dort minimal ist, wo der Nutzanteil auftritt. Es wird also möglichst vermieden, Nutzanteil und Störanteil untrennbar zu vermischen. Das Dezimationsfilter sollte also entsprechend Abbildung 5.3a an den Stellen $0.05f_s$,$0.15f_s$,$0.35f_s$,$0.45f_s$ und an den Spiegeln an der Nyquistfrequenz das Eingangsspektrum maximal unterdrücken. Abbildung 5.3g zeigt den Betragsfrequenzgang des Dezimationsfilters der folgenden Stufe der Multiratenfilterkaskade, welches weitere Störanteile wegfiltert.

In Abbildung 5.4 ist jeweils der Betragsfrequenzgang der Multiratenfilterkaskade mit verschiedenen Stufentiefen aufgetragen. Das verwendete Dezimationsfilter hat nur dreizehn Koeffizienten und entstand mit Hilfe eines CAD-Programmes. Die Anzahl der Koeffizienten des Dezimationsfilters ist so gering, weil vom CAD-Programm nur gefordert werden mußte, daß die Frequenzen um die oben genannten Bereiche möglichst gut unterdrückt werden sollen, und daß der Durchlassbereich bei einem Viertel der Samplefrequenz liegt. Es ist nur der nicht redundante Teil des Frequenzgangs dargestellt, also von der Frequenz null bis zu einem Viertel der Samplefrequenz. Abbildung 5.4a zeigt den Frequenzgang einer Filterkaskade, die nur aus einer Stufe besteht. Die Dämpfung im Sperrbereich beträgt nicht einmal 40dB. In den Abbildungen 5.4b und 5.4c beträgt die Anzahl der Stufen 2 beziehungsweise 5. Wie erwartet ist der Durchlaßbereich der zweistufigen Filterkaskade fünfmal schmaler als der Durchlaßbereich der einstufigen Kaskade. Der Durchlaßbereich der fünfstufigen Kaskade ist bereits schon so schmal, daß er in Abbildung 5.4c gar nicht mehr zu erkennen ist. Der erkennbare Teil des Frequenzgangs liegt bereits im Sperrbereich mit einer Dämpfung von über 80dB. Der Durchlaßbereich der fünfstufigen Kaskade ist in

Abbildung 5.4d zu erkennen. Dort wurde ein sehr kleiner Frequenzbereich dargestellt.

Wie man sieht, wird das Ziel des Herausfilterns eines sehr schmalen Frequenzbandes gut erreicht.

Zum Ende dieses Abschnittes soll noch der Rechenaufwand für eine Multiratenfilterkaskade analysiert werden. Der Rechenaufwand entsteht durch die wiederholte Berechnung des Dezimationsfilters. k sei die Anzahl der Koeffizienten dieses Filters. Ein einmaliges Durchrechnen des Dezimationsfilters für k Samples benötigt k „MACs". Ein MAC (Multiply-Accumulate) ist eine Operation, die zwei Zahlen multipliziert und das Produkt in einen Akkumulator hineinaddiert [AND86]. n sei der Dezimationsfaktor, und die Tiefe der Kaskade sei unendlich. Die Anzahl A der MACs, die benötigt werden, um s Samples zu filtern ergibt sich für $s = n^x$ zu

$$A = \sum_{i=1}^{x} \frac{s}{n^i} k \tag{5.2}$$

Nun wird der Rechenaufwand pro Sample betrachtet:

$$\lim_{x\to\infty} \frac{A}{s} = \lim_{x\to\infty} \sum_{i=1}^{x} \frac{1}{n^i} k = \lim_{x\to\infty} k \left(-1 + \sum_{i=0}^{x} \frac{1}{n^i}\right) = k \left(-1 + \frac{1}{1-\frac{1}{n}}\right) = \frac{k}{n-1} \tag{5.3}$$

Zur Berechnung der Filterkaskade mit $k = 13$ und $n = 5$ werden durchschnittlich also *nur* 3.25 MACs pro Sample ausgeführt. Der Rechenaufwand ist also außerordentlich gering.

5.2 Die PLL

Die PLL verwirklicht die dem Empfänger zugrundegelegte Idee, daß die Uhr genau geht, wenn der Takt durch Vergleich mit der Trägerfrequenz des DCF77-Signals auf die korrekte Frequenz geregelt wird.

An dieser Stelle seien die Aufgaben einer PLL noch einmal rekapituliert.

Die Aufgabe einer PLL besteht darin, die vom Oszillator erzeugte Frequenz so zu regeln, daß gilt:

- Die Oszillatorfrequenz ist gleich der Referenzfrequenz.
- Die Phase zwischen den beiden Signalen ist nahezu konstant und bekannt.

Die Phase zwischen beiden Signalen kann aufgrund von Rauschen schwanken. Hat die PLL den Zustand erreicht, daß beide Bedingungen erfüllt sind, so sagt man, sie hat *gefangen* beziehungsweise sie ist *eingerastet*.

Zur Einführung wird im folgenden zunächst eine *analoge* PLL-Schaltung erläutert.

Abbildung 5.5 zeigt die Struktur. Die Größen u_r,u_o,u_d und u_f seien zeitabhängige Spannungen. Der Ausgang u_d des Phasendetektors PD ist ein Maß für die Phasendifferenz ϑ zwischen Referenzsignal u_r und Oszillatorsignal u_o. Man nennt ϑ den *Phasenfehler*. Das Schleifenfilter SF ist

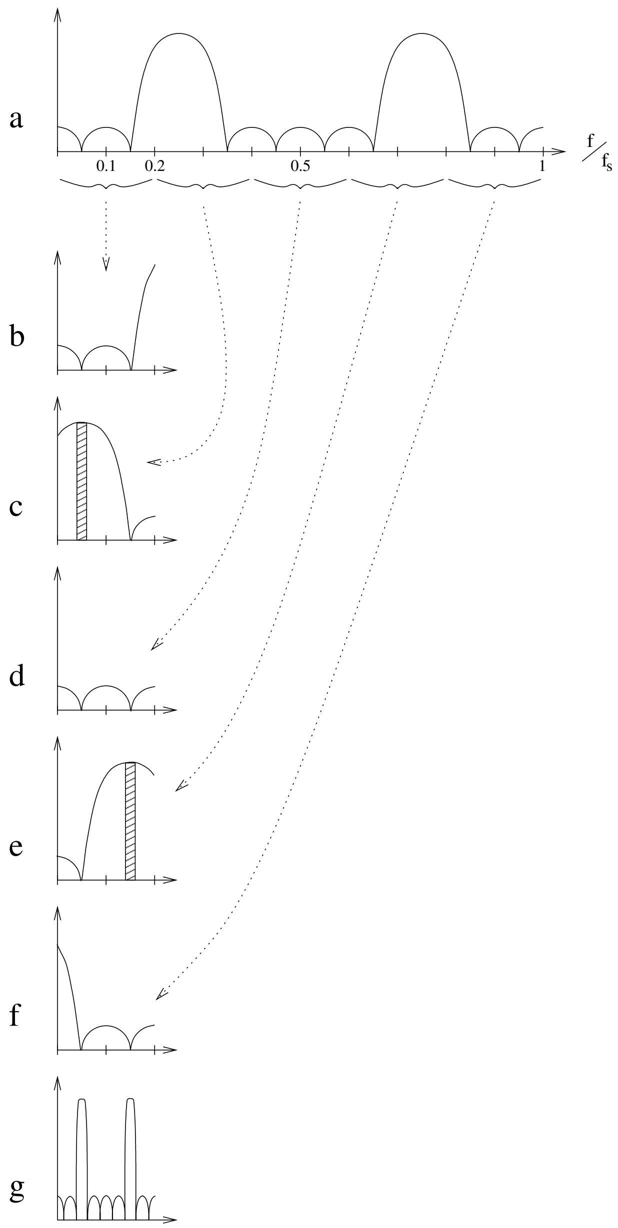

Abbildung 5.3: Ein Filter mit günstigem Betragsfrequenzgang für die Multiratenfilterung

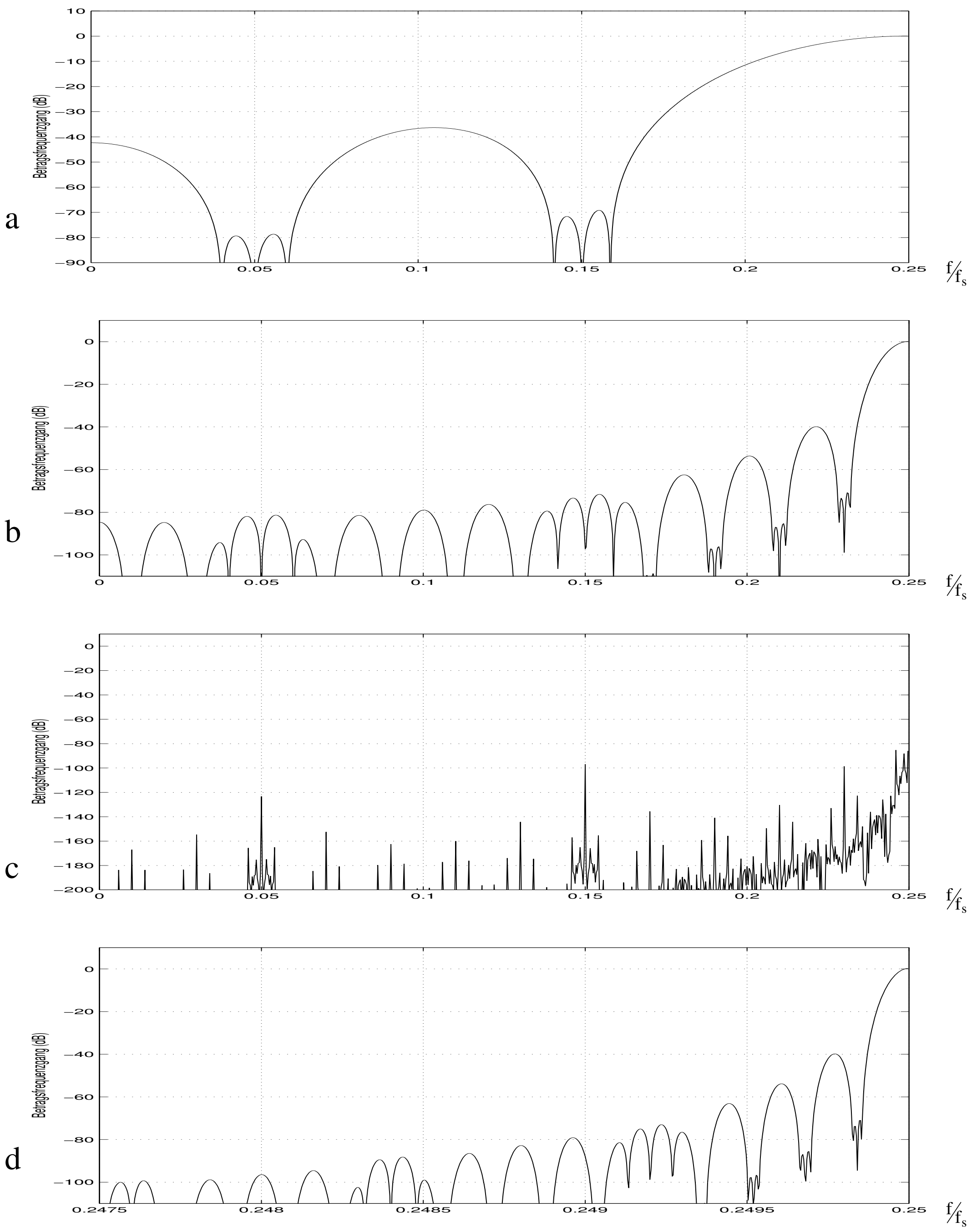

Abbildung 5.4: Frequenzgang der Filterkaskade mit verschiedenen Tiefen

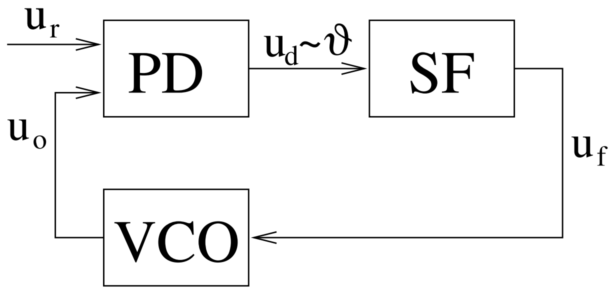

Abbildung 5.5: Struktur einer PLL

ein PI-Glied [CRE88], also ein Tiefpaß. Der Ausgang u_f dieses Filters steuert den spannungsgesteuerten Oszillator VCO (Voltage Controlled Oscillator), und variiert somit die Frequenz von u_o. In diesem Fall ist der VCO der Taktgenerator. Das Referenzsignal u_r sei eine harmonische Schwingung.

Wie bei jedem Regelkreis, sorgt die PLL dafür, daß die Regelabweichung u_d zu null wird. Der Phasenfehler ϑ wird also minimiert.

Abbildung 5.6 zeigt die Reaktion einer PLL auf einen *Frequenzsprung* im Referenzsignal. Der Frequenzsprung bewirkt einen Anstieg der zur Phasendifferenz zwischen u_r und u_o proportionalen Spannung u_d. Dies führt zum Ansteigen der Regelspannung u_f, sodaß sich die Frequenz ω_o von u_o erhöht, und somit das Signal u_o das Signal u_r „einholt“. Das heißt, daß die Phase ϑ zwischen diesen beiden Signalen mit der Zeit wieder abnimmt, und schließlich zu null wird. Der Frequenzsprung ist dann ausgeregelt. Wie man diesem Szenario entnehmen kann, folgt die Oszillatorfrequenz der Referenzfrequenz. Dies gilt allerdings nur in gewissen Grenzen.

Man erinnere sich nun, daß der Empfänger mit digitalen Signalen arbeitet. Es soll zunächst geklärt werden, welches Signal der Empfänger als Referenzsignal auffaßt, und welches als Oszillatorsignal. Wie aus Abbildung 3.2 auf Seite 21 ersichtlich ist, wird die Mischfrequenz von dem geregelten Taktgenerator abgeleitet. Dies geschieht durch Frequenzteilung. Eine Veränderung der Frequenz des Taktgenerators bewirkt über die Änderung der Mischfrequenz eine Veränderung der Zwischenfrequenz bei 2400Hz. Wie in Abschnitt 6.1 noch erläutert wird, ist diese Änderung viel größer als die Änderung der Taktfrequenz. Für den Empfänger ist also das DCF77-Signal, wie er es sieht, in der Frequenz veränderlich, obwohl die DCF77-Trägerfrequenz ja atomuhrgenau ist. Also ist die Taktfrequenz als die Referenzfrequenz anzunehmen, und das empfangene Signal als die geregelte Oszillatorfrequenz. Dies würde auch für den Fall gelten, in dem das DCF77-Signal gar nicht erst auf eine Zwischenfrequenz gemischt würde, sondern direkt gesampelt würde. In diesem Fall würde dem Empfänger ein Ansteigen der Taktfrequenz so vorkommen, als würde die Frequenz des empfangenen DCF77-Signals abnehmen. Um Verwechselungen zu vermeiden wird das DCF77-Signal im folgenden als o-Signal bezeichnet und das Taktsignal als r-Signal.

Im folgenden wird auf den Phasendetektor und das Schleifenfilter näher eingegangen:

- Zum Phasendetektor:

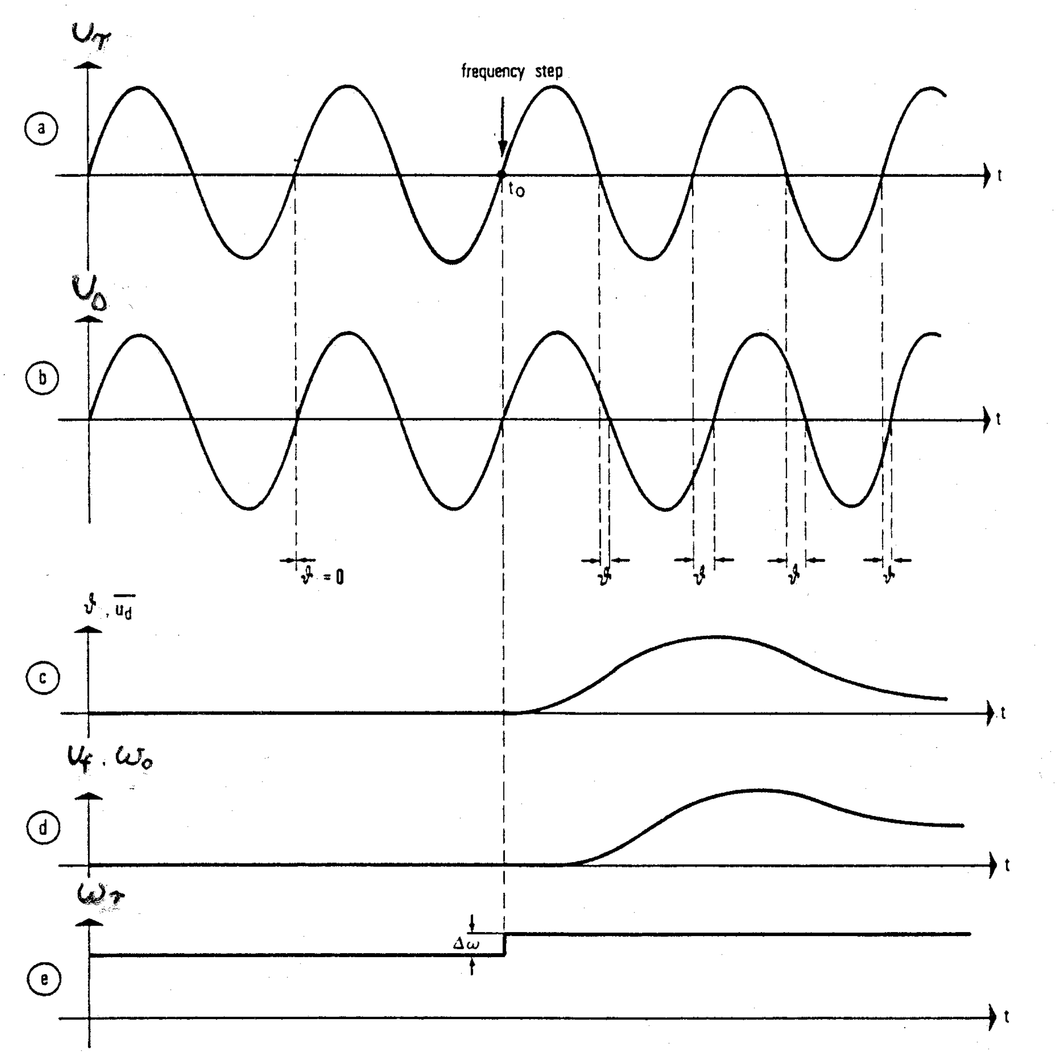

Abbildung 5.6: Reaktion einer PLL auf einen Frequenzsprung, aus [BES93] (leicht geändert)

Es ist das digitale Äquivalent des o-Signals $u_o(t)$ zu betrachten. Dieses enstammt dem Ausgang der Multiratenfilterkaskade, siehe Abbildung 3.2 auf Seite 21. Das vom DDS-Block erzeugte r-Signal erscheint auf einem Viertel seiner Taktfrequenz, vergleiche auch Abbildung 3.3 auf Seite 22. Da dieses Signal die Zahlenfolge $0,1,0,-1,0,1,0,-1,\ldots$ ist, ist die Phasenlage dieses Signals leicht zu erkennen. Man nehme zunächst an, das o-Signal habe die gleiche Frequenz wie das r-Signal. Dann beträgt auch diese Frequenz ein Viertel der Samplefrequenz. Zwischen zwei aufeinanderfolgenden Samples des r-Signals, o_1 und o_2, vergeht dann die Phase $\frac{\pi}{2}$. φ sei die Phase des o-Signals. Es gilt:

$$o_1 = \sin(\varphi) \tag{5.4}$$

$$o_2 = \sin(\varphi + \frac{1}{2}\pi) = \cos(\varphi) \tag{5.5}$$

Dividiert man o_1 und o_2, dann erhält man:

$$\frac{o_1}{o_2} = \tan(\varphi) \tag{5.6}$$

und somit

$$\varphi = \arctan \frac{o_1}{o_2} \tag{5.7}$$

Die Arcustanges-Funktion liefert nur Werte zwischen $-\frac{\pi}{2}$ und $\frac{\pi}{2}$. Die Phase im gesamten Vollkreis zwischen $-\pi$ und π läßt sich unter Berücksichtigung der Vorzeichen von o_1 und o_2 berechnen. Um die Phasendifferenz ϑ zu erhalten, ist noch die bekannte Phase des r-Signals abzuziehen. Diese nimmt zyklisch die Werte $0,\frac{1}{2}\pi,\pi,\frac{3}{2}\pi,0,\frac{1}{2}\pi,\ \pi,\frac{3}{2}\pi,\ldots$ an.

- Zur digitalen Realisation des Schleifenfilters: Das integrierende Verhalten des Schleifenfilters wird durch folgende Gleichung beschrieben (vergleiche [PRE81]):

$$u_{fi} = mu_{di} + mx \sum_{j=0}^{n-1} u_{dj} \tag{5.8}$$

Hierbei entsprechen den Spannungen u_d und u_f die digitalen Werte u_{di} und u_{fi} im i-ten Takt. m und x bestimmen das Verhalten des Schleifenfilters. Softwaremäßig läßt sich diese Gleichung implementieren, indem in jedem Regeltakt folgendes Programmstück ausgeführt wird:

u_f := $m \cdot u_d$ + akku
akku :=akku + $m \cdot x \cdot u_d$

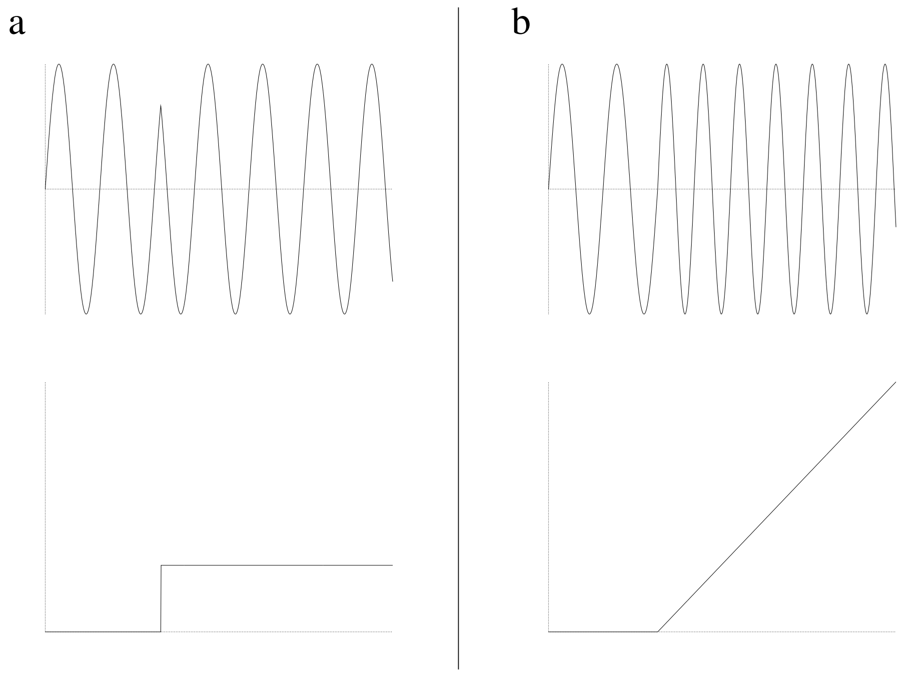

Abbildung 5.7: Phasensignale

Um das Verhalten einer PLL zu beschreiben, unterscheidet man zwischen dem Verhalten der PLL im *eingerasteten* Zustand und dem Verhalten der PLL im *ausgerasteten* Zustand. Im eingerasteten Zustand läßt sich die PLL als LTI-System beschreiben. Das Verhalten ist demnach vollständig durch eine *Übertragungsfunktion* $H(s)$ beschreibbar. In diesem Fall handelt es sich um ein System zweiter Ordnung. Ein solches System ist durch zwei reelle Parameter festgelegt. Die Aufgabe des Entwurfs besteht in der Dimensionierung dieser beiden Parameter.

Die beiden folgenden Abschnitte erörtern das Verhalten der PLL im eingerasteten und im ausgerasteten Zustand.

5.2.1 Die PLL im eingerasteten Zustand

Es folgt eine Definition des Begriffs „Phasensignal“:

Definition 5.1 *Sei $u(t)$ ein oszillierendes Signal mit konstanter Amplitude:*

$$u(t) = \hat{u}\sin(\omega t + \varphi(t)) \tag{5.9}$$

hier sei die Größe ω konstant. Man nennt $\varphi(t)$ das Phasensignal *von $u(t)$.*

In Abbildung 5.7a ist der Zusammenhang zwischen u und φ im Falle eines Phasensprungs dargestellt. Abbildung 5.7b zeigt den Zusammenhang im Falle eines Frequenzsprungs. Die sprunghafte Veränderung der Frequenz bewirkt ein lineares Ansteigen der Phase.

In diesem Unterabschnitt wird die Übertragungsfunktion der PLL im eingerasteten Zustand beschrieben. Diese Übertragungsfunktion $H(s)$ entspricht nicht etwa dem Verhältnis von u_o und u_r, sondern dem Verhältnis der zugehörigen *Phasensignale* φ_o und φ_r:

$$H(s) = \frac{\Phi_o(s)}{\Phi_r(s)} \tag{5.10}$$

Dabei sind $\Phi_o(s)$ und $\Phi_r(s)$ die Laplace-Transformierten von $\varphi_o(t)$ und $\varphi_r(t)$. Nach wie vor sei $\vartheta(t)$ die Differenz dieser beiden Signale.

Wie schon erwähnt, ist die PLL im eingerasteten Zustand ein LTI-System zweiter Ordnung. Damit verhält sie sich genau so wie andere Systeme, die dieser Klasse angehören. Zum Beispiel zeigt ein Federpendel, das zu Schwingungen angeregt wird, das gleiche Verhalten wie die PLL. Das zeitliche Verhalten dieser Systeme ist durch eine lineare Differentialgleichung zweiten Grades beschreibbar [CRE88]. Diese Systeme haben die Eigenschaft, daß sie zwei „speichernde Elemente" enthalten. Im Falle des Federpendels sind die speichernden Elemente die kinetische und die potentielle Energie. Die speichernden Elemente der PLL sind das Schleifenfilter und der VCO. Es folgen Anmerkungen zu den drei Blöcken der PLL:

- Zum Phasendetektor:
  Der Ausgang des Phasendetektors ist proportional zur gemessenen Phasendifferenz. Damit ist der Phasendetektor ein LTI-System nullter Ordnung. Der Proportionalitätsfaktor sei K_d:

  $$K_d = \frac{u_d}{\vartheta} \tag{5.11}$$

  Aus Abschnitt 6.2 ist ersichtlich, daß es sinnvoll ist, das Ergebnis der Berechnung von ϑ auf den Wertebereich von -1 bis 1 zu normieren. Dann ist die digitale Größe u_d gleich eins, wenn $\vartheta = \pi$ ist. Also gilt

  $$K_d = \frac{1}{\pi} \tag{5.12}$$

- Das Schleifenfilter ist ein LTI-System erster Ordnung. Es zeigt integrierendes Verhalten. In dem oben angegebenen Algorithmus, der das Schleifenfilter implementiert, ist die Variable „akku" das speichernde Element.

- Zum VCO:
  Die Frequenzabhängigkeit des VCO sei linear:

  $$\omega_o = \omega_{mitte} + K_o u_f \tag{5.13}$$

  Zu betrachten ist die *Phasen*übertragungsfunktion. Die *Phase* φ_o des Signals u_o ist durch das Integral der Kreisfrequenzabweichung $K_o u_f$ über die Zeit gegeben, vergleiche Abbildung 5.7.

  $$\varphi(t) = \int_{\tau=0}^{t} K_o u_f(\tau) d\tau \tag{5.14}$$

Genau wie das Schleifenfilter ist der VCO also ein Integrationsglied erster Ordnung (für Phasensignale).

Es gibt also vier Parameter, von denen das Verhalten der PLL abhängt: m,x,K_d und K_o. Nach [PRE81] hängt das Verhalten der PLL nur von den beiden Parametern M und x ab, wobei

$$M := mK_dK_oT \tag{5.15}$$

T ist hier die Zeit, die zwischen zwei Regeltakten vergeht. Diese Zeit ist abhängig von der Stufentiefe.

Aus den Übertragungsfunktionen der drei Blöcke läßt sich eine Übertragungsfunktion für das Gesamtsystem gewinnen. Diese Funktion enthält die Information über das gesamte Verhalten der PLL. Nach [BES93] lautet das Ergebnis

$$H(s) = \frac{2\zeta\omega_n s + \omega_n^2}{s^2 + 2s\zeta\omega_n + \omega_n^2} \tag{5.16}$$

Die Parameter ω_n und ζ heißen *Eigenfrequenz* und *Dämpfung*. Diese Parameter (ω_n, ζ) sind mit den Parametern (M, x) verknüpft. Zur Veranschaulichung der Bedeutung der Parameter ω_n und ζ dient Abbildung 5.8. Die Abbildung zeigt Reaktionen einer PLL auf einen Phasensprung im Referenzsignal unter Variation der Parameter ω_n und ζ. Wie man sieht entspricht ω_n etwa der Kreisfrequenz der Schwingung des Phasenfehlers ϑ. Ist die Dämpfung ζ klein, so dauert es lange, bis die Schwingung des Phasenfehlers abgeklungen ist. Je größer die Dämpfung ist, desto weniger Schwingungen macht ϑ. Im aperiodischen Grenzfall reagiert die PLL ohne Schwingungen des Phasenfehlers. Bei zu großer Dämpfung kann es aber auch sehr lange dauern, bis der Phasenfehler ausgeregelt wird. [GAR66] gibt einen hinsichtlich der Ausregelzeit optimalen Dämpfungsfaktor an. Dementsprechend sei festgelegt, daß

$$\zeta = \frac{1}{\sqrt{2}} \approx 0.707 \tag{5.17}$$

Zur weiteren Veranschaulichung soll das Beispiel des Federpendels dienen [KNE57]. Die Eigenfrequenz ist die Resonanzfrequenz des Systems. Wird das Federpendel in Analogie zum Phasensprung impulsartig angeregt, so schwingt es mit der ihm eigenen Frequenz auf und ab. Je nach der Dämpfung des Pendels klingt diese Schwingung nach und nach ab. Das Pendel wird durch innere Reibung in der Feder gedämpft. Stärkere Dämpfung erreicht man zum Beispiel indem man das Pendel in eine zähe Flüssigkeit taucht.

Zum Abschluß dieses Abschnittes soll noch erörtert werden, unter welchen Umständen die eingerastete PLL in den ausgerasteten Zustand übergeht [BES93]:

- Die PLL könnte ausrasten, wenn die Referenzfrequenz Sprünge ausführen würde, die größer sind als der sogenannte „pull-out“-Bereich $\Delta\omega_{PO}$. Die Referenzfrequenz führt aber in dieser Anwendung keine Sprünge aus.

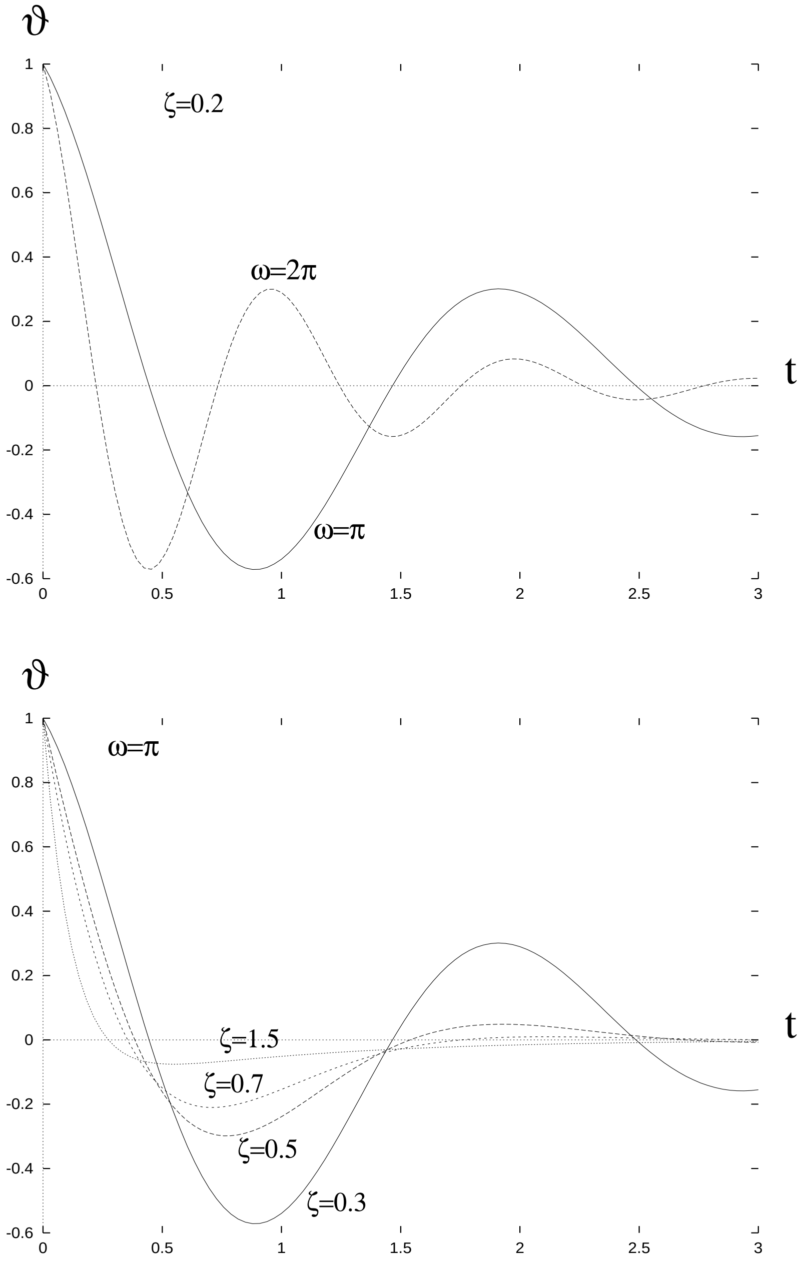

Abbildung 5.8: Variation von ω_n und ζ

- Rauschen kann zum Ausrasten der PLL führen, vergleiche Abschnitt 5.2.3.
- Die PLL kann ausrasten, wenn die Änderungsgeschwindigkeit der Referenzfrequenz $\dot{\omega}_r = \frac{d\omega_r}{dt}$ größer als das Quadrat der Eigenfrequenz ω_n ist. Dieser Fall kann eintreten, da der Taktgenerator *Störungen* ausgesetzt ist. Zum Beispiel können Temperaturschwankungen eine Änderung der Referenzfrequenz bewirken. Aus drei Gründen ist die PLL in diesem Punkt sehr empfindlich:
  - $\dot{\omega}_r$ nimmt nicht mit zunehmender Kaskadentiefe ab, wohl aber ω_n, vergleiche Abschnitt 5.3
  - ω_n ist bei hohem Rauschen im empfangenen Signal sehr klein zu wählen, vergleiche Abschnitt 5.2.3.

5.2.2 Die PLL im ausgerasteten Zustand

Man muß annehmen, daß Oszillatorfrequenz und Referenzfrequenz nicht von Anfang an gleich sind. Also „startet“ die PLL zunächst im ausgerasteten Zustand. Das Korrigieren der Oszillatorfrequenz soll dann zum Einrasten führen. Man nehme an, daß die Referenzfrequenz zu Beginn um den Betrag $\Delta\omega_{start}$ von der Mittenfrequenz des Oszillators ω_{mitte} abweiche. Die PLL wird dann u_f so verändern, daß die Oszillatorfrequenz gleich der Referenzfrequenz wird. Folgende Fragen sind zu klären:

- Unter welchen Umständen rastet die PLL ein ?
- Wie lange dauert dieser Vorgang ?

Der Vorgang des Einrastens läßt sich unterteilen in zwei nacheinander stattfindende Phasen [BES93]:

- den *Ziehvorgang*,
- und den *Fangvorgang*.

Abbildung 5.9 zeigt beispielhaft einen Einrastvorgang. Während des Ziehvorgangs wird ω_o langsam „hochgepumpt“. Anschließend findet der Fangvorgang statt. Dieser führt viel schneller zu einer Regelung von ω_o auf den Sollwert. Leider kann der Fangvorgang nur stattfinden, wenn sich ω_o schon nahe am Sollwert befindet, also wenn die Differenzfrequenz $\Delta\omega = \omega_r - \omega_o$ gering genug ist. Der *Fangbereich* $\Delta\omega_F$ ist definiert durch die maximale Abweichung $\Delta\omega$, innerhalb derer der Fangvorgang anstatt des Ziehvorganges stattfindet. T_Z und T_F seien die Zeiten, die Zieh- bzw. Fangvorgang dauern. [BES93] gibt Formeln für $\Delta\omega_F$,T_Z und T_F an:

$$\Delta\omega_F \approx 2\zeta\omega_n \tag{5.18}$$

$$T_Z = \frac{\pi^2}{16}\frac{\Delta\omega_{start}^2}{\zeta\omega_n^3} \tag{5.19}$$

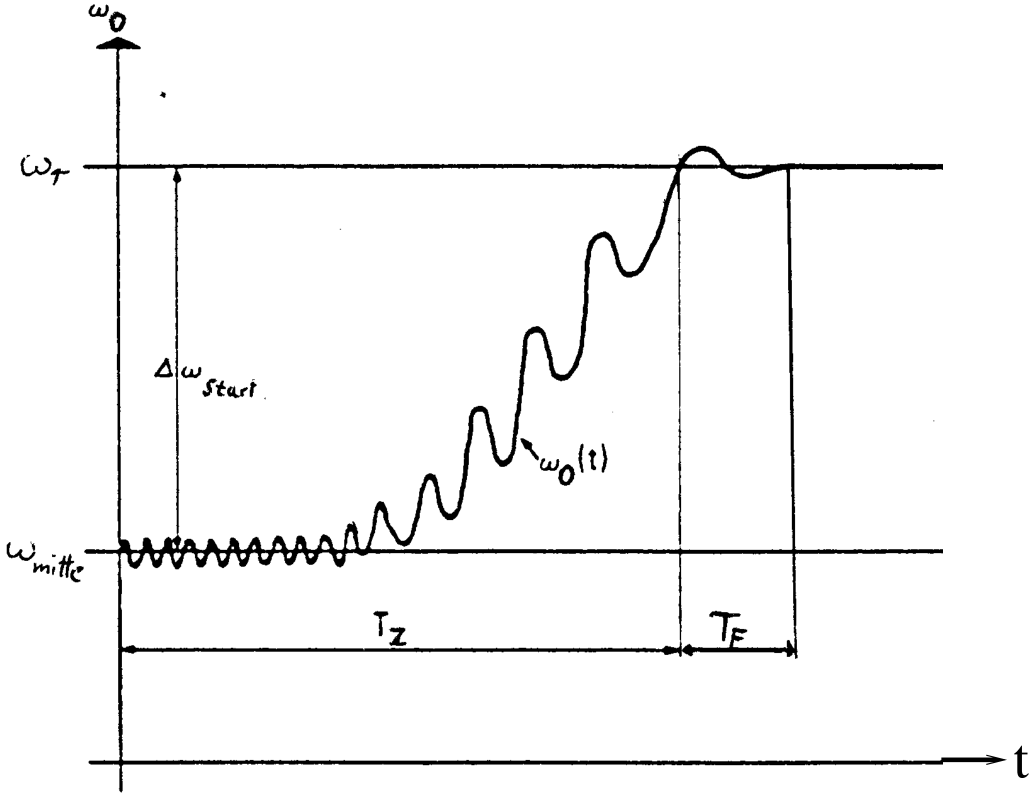

Abbildung 5.9: Zieh- und Fangvorgang bei der PLL, aus [BES93] (leicht geändert)

$$T_F \approx \frac{2\pi}{\omega_n} \tag{5.20}$$

Es ist anzunehmen, daß das empfangene Signal stark verrauscht ist. Wie noch gezeigt wird, macht dies eine kleine Eigenfrequenz ω_n notwendig. Dies führt einerseits zu einem kleinen Fangbereich, andererseits aber auch zu einer langen Ziehzeit. Dies kann unangenehm werden, wenn die Ziehzeit der PLL die Zeit, die zum Dekodieren der Zeitinformation benötigt wird, übersteigt.

5.2.3 PLL und Rauschen

In den bisherigen Ausführungen wurde angenommen, daß das Referenzsignal u_r ein unverrauschtes Signal sei. Natürlich empfängt die Uhr aber ein stark verrauschtes Signal. Dieses wird zunächst mit einem Bandpaß der Bandbreite B, der Multiratenfilterkaskade, vorgefiltert und dann als Signal tr dem PLL-Regler zugeführt. Auch das Signal tr wird, zwar weniger stark, immer noch verrauscht sein. Dadurch verschieben sich die Nulldurchgänge dieses Signals. Das kann man auch so interpretieren, daß die Phase dieses Signals Schwankungen unterliegt, dem sogenannten Phasenrauschen. Man beschreibt verrauschte Signale durch das Signal-Rauschverhältnis SNR (Signal to Noise Ratio), auch Störabstand genannt. Das SNR ist der Quotient aus der Leistung des Signals ohne Rauschen, und der Leistung des Rauschens. Das Signal-Rauschverhältnis des *Phasensignals* des Oszillatorsignals sei SNR_{po}. Das Signal-Rauschverhältnis des Signals tr sei SNR_{tr}. SNR_{tr} ist nicht der Störanstand eines Phasensignals, sondern der des Signals u_r. [BES93] gibt einen Zusammenhang zwischen dem SNR_{po} und SNR_{tr} an:

$$SNR_{po} = SNR_{tr} \frac{B}{\omega_n(\zeta + \frac{1}{4\zeta})} \tag{5.21}$$

Nun wird der Begriff der normierten Kreisfrequenz eingeführt:

Definition 5.2 *Ein Sinussignal mit der Kreisfrequenz ω werde mit der Frequenz f_s abgetastet. Dann ist $T = \frac{1}{f_s}$ die Zeit zwischen zwei Abtastzeitpunkten. Die zugehörige normierte Kreisfrequenz $\breve{\omega}$ wird definiert durch:*

$$\breve{\omega} = \frac{\omega}{f_s} = \omega T \tag{5.22}$$

Die normierte Kreisfrequenz nimmt Werte zwischen 0 und 2π an, sofern die Abtastrate höher ist als die Frequenz des Sinussignals.

Gleichung 5.23 ist für analoge Signale gedacht. Durch Erweiterung mit der Abtastrate erhält man:

$$SNR_{po} = SNR_{tr} \frac{BT}{\breve{\omega}_n(\zeta + \frac{1}{4\zeta})} \tag{5.23}$$

Eine Daumenregel besagt [BES93], daß die PLL nur dann stabil arbeitet, wenn gilt:

$$SNR_{po} \geq 4 \tag{5.24}$$

5.2.4 Dimensionierung der PLL

Die zu bestimmenden Größen sind m,x und K_o. Die Größe K_o ist durch den Aufbau des Analogteils des Empfängers vorgegeben, vergleiche Kapitel 6. Die Größen m und x werden im Regelalgorithmus verwendet, und sind im Digitalteil des Empfängers zu speichern. Sie ergeben sich durch Wahl der Eigenfrequenz ω_n wie folgt: Es ist ein Zusammenhang zwischen $(\breve{\omega}_n, \zeta)$ und (M, x) bekannt. Wenn ω_n frei gewählt wird, und gemäß Gleichung 5.17 $\zeta = \frac{1}{\sqrt{2}}$ gewählt wird, können auf diese Weise M und x bestimmt werden. Schließlich erhält man m aus Gleichung 5.15. In Gleichung 5.15 geht auch die Größe T ein, die dem zeitlichen Abstand zwischen zwei Regeltakten entspricht. Diese Größe hängt von der Anzahl der Stufen der Filterkaskade ab. Also ist die Größe m für jede Stufenanzahl zu berechnen und zu speichern.

Bei der Wahl der Eigenfrequenz ω_n ist darauf zu achten, daß sie hinreichend klein ist, um gemäß Gleichung 5.24 dem Rauschen zu genügen.

[PRE81] gibt den Zusammenhang zwischen $(\breve{\omega}_n, \zeta)$ und (M, x) wie folgt an:

$$\breve{\omega}_n = \sqrt{\phi^2 + (\ln|z_\infty|)^2} \tag{5.25}$$

$$\zeta = \frac{1}{\sqrt{1 + \left(\frac{\phi}{\ln|z_\infty|}\right)^2}} \tag{5.26}$$

wobei die Werte für z_∞ und ϕ wie folgt definiert sind [1]:

$$|z_\infty| = \sqrt{Mx - M + 1} \tag{5.27}$$

$$\phi = \arccos\frac{2 - M}{2|z_\infty|} \tag{5.28}$$

Bei falscher Dimensionierung können Regelkreise *instabil* sein. Ein instabiler Regelkreis erzeugt von alleine Schwingungen seiner Ausgangsgröße und ist deshalb nicht zu gebrauchen. [PRE81] gibt eine notwendige und hinreichende Bedingung für die Stabilität des Regelkreises an. Demnach müssen folgende zwei Gleichungen erfüllt sein:

$$0 < x < 1 \tag{5.29}$$

$$0 < M < \frac{4}{2 - x} \tag{5.30}$$

Abbildung 5.10 stellt den erlaubten Bereich für die Größen M und x dar. In diesem Diagramm sind Kurven konstanter normierter Eigenfrequenz und Kurven konstanter Dämpfung eingetragen. In dieser Anwendung wird nur der Bereich unterhalb der Diagonalen verwendet. Die Gleichungen 5.25 bis 5.28 gelten auch nur innerhalb dieses Bereichs. Oberhalb der Diagonalen ist

[1] Anmerkung: $|z_\infty| \exp^{\pm j\phi}$ sind die beiden konjugiert komplexen Polstellen der Übertragungsfunktion in der z-Ebene. Allerdings verwendet [PRE81] die *Frequenz*übertragungsfunktion und nicht die hier verwendete Phasenübertragungsfunktion.

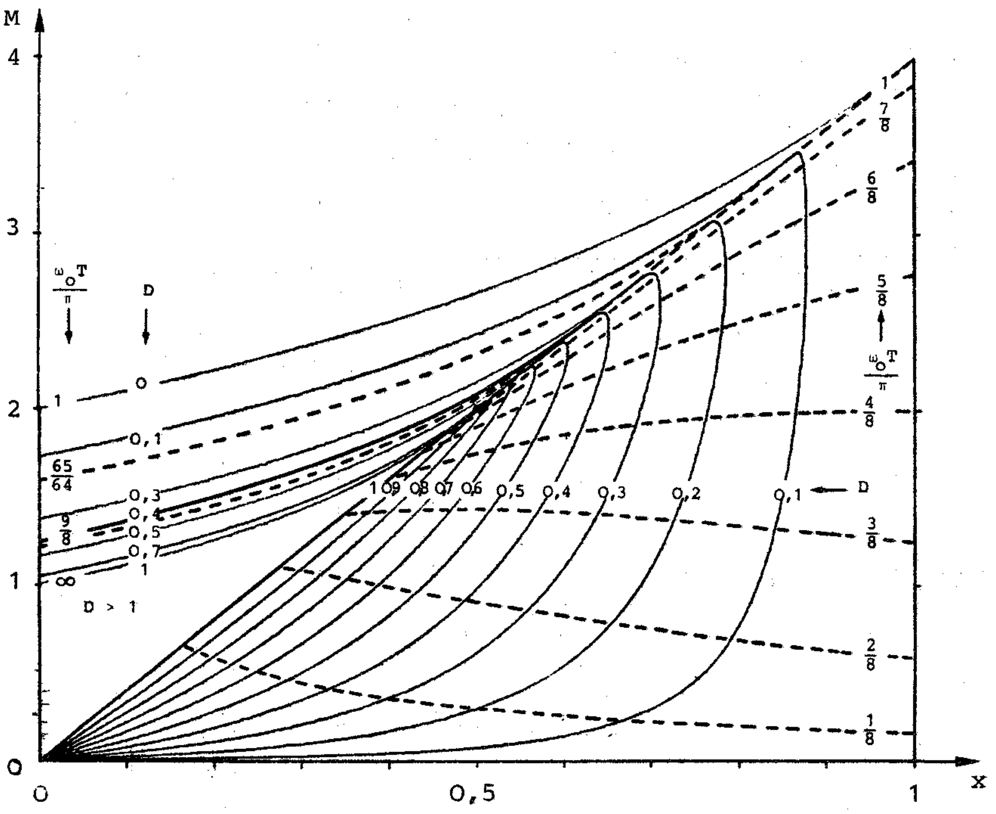

Abbildung 5.10: Einschränkung von x und M, aus [PRE81]

die relative Eigenfrequenz größer als π, die Eigenfrequenz ist also höher als die halbe Abtastrate. Man erinnere sich, daß die Eigenfrequenz der Zeit entspricht, mit der Störungen ausgeregelt werden. Es ist nicht notwendig, Störungen in weniger als zwei Takten auszuregeln. Wie gleich im nächsten Abschnitt noch gezeigt wird, ist die relative Eigenfrequenz viel kleiner als π zu wählen. Der Bereich für die Dimensionierung von M und x liegt deswegen in der linken Ecke unterhalb der Diagonalen.

5.3 Zusammenwirken von Multiratenfilterkaskade und PLL

Außer Gleichung 5.24 gibt es noch eine weitere Einschränkung bezüglich der Wahl der Eigenfrequenz ω_n. Bisher wurde die PLL in dieser Anwendung als ein System zweiter Ordnung betrachtet. Sie ist aber kein System zweiter Ordnung. Das liegt daran, daß die Multiratenfilterkaskade ein Teil der Regelschleife ist. Im System zweiter Ordnung würde sich eine Änderung der Regelspannung u_f *sofort* am Phasendetektor bemerkbar machen. Im Empfänger jedoch muß das durch die Mischung in seiner Frequenz veränderte empfange Signal erst die Multiratenfilterkaskade durchlaufen, bevor sich diese Änderung am Phasendetektor bemerkbar macht. In der Regelschleife liegt also eine zusätzliche Verzögerung. Diese Verzögerung kann zu Instabilität des Regelkreises führen, wenn nicht geeignete Maßnahmen ergriffen werden. Eine geeignete Maßnahme ist, die Eigenfrequenz ω_n so klein zu wählen, daß ihr Kehrwert groß gegen die Verzögerung ist. Die Veränderung der Oszillatorfrequenz macht sich dann am Phasendetektor bemerkbar bevor sich die Regelgröße u_f merklich verändern konnte. Da dieses Problem analytisch schwer zu handhaben ist, ist man bei der Wahl der Eigenfrequenz auf das Experiment angewiesen. Es hat sich gezeigt, daß sich die PLL bei einer Wahl der relativen Eigenfrequenz durch

$$\breve{\omega}_n \leq \frac{1}{8k} \tag{5.31}$$

wie gewünscht verhält. Dabei ist k die Anzahl der Koeffizienten der Dezimationsfilter.

Aus der Forderung, daß das Signal tr durch sehr schmalbandige Bandpaßfilterung des empfangenen Spektrums gewonnen werden soll, ergibt sich folgendes Problem: Nach dem Einschalten des Gerätes wird die Frequenz des Taktgenerators nicht von Anfang an die korrekte sein. Dies führt wegen der Mischung im Analogteil zu einer Frequenzabweichung des Signals am Eingang der Filterkaskade. Die Frequenz des dort erscheinenden Trägers des DCF77-Signals sei im folgenden mit f_e bezeichnet. Der Sollwert dieser Frequenz beträgt 2400Hz. Bei großer Tiefe der Multiratenfilterkaskade ist die Bandbreite dieses Bandpasses sehr gering. Es muß verhindert werden, daß der Fehler von f_e größer ist als die Bandbreite der Kaskade. Wäre dies der Fall, so würde die Trägerfrequenz des DCF77-Signals gar nicht am Ausgang der Filterkaskade erscheinen. Das gewünschte Signal läge außerhalb des herausgefilterten Frequenzbandes.

Im folgenden sei mit i die Tiefe der Filterkaskade bezeichnet. Ausgehend von der maximalen Frequenzabweichung beim Einschalten ist die Anfangstiefe i_0 der Filterkaskade zu bestimmen. Für i_0 wurde der Wert vier festgelegt. Nach dem Einschalten regelt die PLL die Frequenz f_e in die Richtung des Sollwertes. Der Fehler Δf_e vermindert sich, so daß f_e auch im Frequenzband des Bandpasses liegt, dem die Filterkaskade mit der Tiefe fünf entspricht. Der Empfänger inkrementiert die Stufenanzahl der Kaskade, nachdem die PLL gefangen hat. Dieser Prozeß kann fortgesetzt werden. Wenn die PLL mit der Tiefe i korrekt arbeitet, dann wird die Tiefe inkrementiert. Auf diese Weise gewinnt der Empfänger ein mehr und mehr schmalbandiger werdendes Signal tr, und je schmalbandiger dieses Signal ist, desto geringer ist der darin enthaltene Rauschanteil.

Dieser Mechanismus bewirkt eine *Hysterese*: Die PLL kann nicht bei beliebig schlechten Empfangsbedingungen korrekt arbeiten. Es ist also möglich, daß die PLL bei einem stark gestörten Empfang überhaupt nicht fängt. Die Empfangsbedingungen unterliegen aber Schwankungen. Man stelle sich folgendes Szenario vor: Der Empfang wird irgendwann ausreichend gut, so daß die PLL nun fängt. Dies bewirkt eine Korrektur der Frequenz des Taktgenerators und eine Erhöhung der Tiefe der Multiratenfilterkaskade. Dadurch erhöht sich auch das Signal-Rauschverhältnis des Signals tr. Wenn jetzt die Empfangsbedingungen wieder so ungünstig werden wie am Anfang, dann kann die Qualität des Signals tr aufgrund der nun schmalbandiger gewordenen Filterung trotzdem noch ausreichen, um die PLL-Regelung aufrecht zu erhalten. Der Empfänger funktioniert also nun bei Empfangsbedingungen, bei denen er anfangs nicht funktionierte. Dieses Verhalten konnte durch Simulation bestätigt werden.

Der Tiefe i der Kaskade ist bedingt durch die Instabilität des Taktgenerators eine Grenze gesetzt. Im nicht geregelten Fall unterliegt die Frequenz des Quarz-Schwingkreises einer Drift. Diese Drift ist der Grund für Ungenauigkeiten bei Quarzuhren. Die Regelung dieser Frequenz soll diese Drift verhindern. Je großer die Stufentiefe der Kaskade ist, desto langsamer findet aber auch die Regelung statt. Die Datenrate, mit der die Regelgröße verändert werden kann entspricht der Datenrate des Signals tr. Wie aus Tabelle 3.1 auf Seite 18 ersichtlich ist, beträgt die Datenrate am Ausgang der Stufe Nummer acht etwa 0.025Hz. Dementsprechend kann sich die Regelspannung u_f nur alle 40 Sekunden verändern. Entsprechend der notwendigerweise geringen relativen Eigenfrequenz der PLL wird es außerdem noch viele Regeltakte dauern, bis eine Abweichung der Quarzfrequenz ausgeregelt werden kann. Wenn die PLL mit dem Nachregeln nicht mehr mit der Drift der Quarzfrequenz mithalten kann, so wird dies zwangsläufig zu einem Anstieg des Phasenfehlers ϑ führen. Wenn dieser einen Schwellwert überschritten hat, dann vermindert der Empfänger die Tiefe der Kaskade um eine Stufe. Da der Phasenfehler dann immer noch über dem Schwellwert liegt, ist dieser Mechanismus für die Zeit außer Kraft zu setzen, die die nun *schneller* gewordene PLL zum Ausregeln dieses Fehlers benötigt. Ansonsten würde der Empfänger die Kaskadentiefe noch weiter vermindern. Die Zeit zum Ausregeln bemißt sich an der Eigenfrequenz.

Nun soll gezeigt werden, wie sich der Fehler Δf_e der Frequenz f_e sich bei großen Tiefen der Kaskade bemerkbar macht: Da die Kaskade auf Unterabtastung beruht, setzt sich Δf_e auf allen Stufen als absoluter Fehler fort. Ist beispielsweise $\Delta f_e = 1$Hz, so ist der relative Fehler gleich $\frac{1}{2400} \approx 4 \cdot 10^{-4}$. Hat die Kaskade die geringste Stufenanzahl vier, so beträgt die Datenrate am Ausgang der Kaskade 15.36Hz, vergleiche Tabelle 3.1. Das DCF77-Signal erscheint dort auf der Frequenz $(\frac{15.36}{4} - 1)$Hz $= 2.84$Hz, obwohl es bei 3.84Hz erscheinen sollte. Innerhalb eines Regeltaktes wächst der Phasenfehler ϑ demnach um 23^0. Der Phasenfehler wächst viel schneller als er ausgeregelt werden kann. Würde stattdessen auf die Filterkaskade verzichtet, und das gesampelte Signal direkt an den PLL-Regler geführt, dann würde dieser mit einer Frequenz von 9600Hz arbeiten. Zwischen zwei Regeltakten würde der Phasenfehler nur um 0.6^0 anwachsen.

Es ist noch zu klären, wie der Empänger erkennen soll, ob die PLL gefangen hat, damit er dann die Kaskadentiefe inkrementieren kann. [BES93] beschreibt einen „In-Lock-Detektor“. Dieser bildet das Produkt p aus dem Referenzsignal und dem Oszillatorsignal. Ist die Phase zwischen beiden Signalen längerfristig etwa null, so hat p einen positiven Mittelwert. Sind andernfalls die beiden Frequenzen so unterschiedlich, daß der Phasenfehler, der Schwebung entsprechend, ständig von null bis 360 Grad im Kreis herumwandert, so ist der Mittelwert von p null. Also kann über den Mittelwert von p entschieden werden, ob die PLL gefangen hat.

Beim Empfänger wurde eine andere Entscheidungsmethode implementiert. Der Empfänger mißt die Zeit, die der Phasenfehler ϑ unterhalb eines Schwellwertes bleibt. Wird diese Zeit größer als die Fangzeit (Gleichung 5.20), so hat die PLL gefangen, siehe hierzu auch Abbildung 5.8. Dieses Verfahren hat gegenüber dem zuerst vorgestellten Verfahren den Vorteil, daß ein Ausrasten der PLL *schneller* erkannt werden kann. Dies ist für das Dekrementieren der Stufenanzahl von Bedeutung. Dieser Zusammenhang wird im folgenden Abschnitt verdeutlicht.

Ein weiteres Problem entsteht durch Schwunderscheinungen des empfangenen Signals, vergleiche Abschnitt 2.1. Der Empfänger sollte bei einem kurzfristigen Ausbleiben des Eingangssignals in seinem Zustand bleiben, das heißt, sowohl die Regelspannung als auch die Stufentiefe sollten *nicht* verändert werden. Der Quarzoszillator läuft dann ohne Nachregelung weiter. Wenn das empfangene Signal nach einiger Zeit wieder stärker wird, so sollte der Phasenfehler nicht stark gestiegen sein. Nun kann das Nachregeln wieder einsetzen. Im Rahmen dieser Arbeit wurde dieses Problem allerdings nicht gelöst.

5.3.1 Messungen des zeitlichen Verlaufs von Parametern der PLL

Mit dem implementierten Empfänger wurden die Steuergröße u_f, der Phasenfehler ϑ und die Stufenanzahl i gemessen. u_f ist hier die Digitalzahl, die über den D/A-Wandler ausgegeben wird, und kann Werte zwischen -1 und 1 annehmen Die Abbildungen 5.11a und 5.11b zeigen die zeitliche Entwicklung dieser Größen. Dabei ist Abbildung 5.11b die Fortsetzung von Abbildung 5.11a. Man beachte, daß die Zeit in Abbildung 5.11b gegenüber Abbildung 5.11a gestaucht ist.

Nach dem Einschalten erreicht die Regelgröße u_f innerhalb von 40 Sekunden den Wert von etwa -0.1. Die Frequenz des Taktgenerators wird dadurch auf den korrekten Wert geregelt. Der mit einem empfangsbedingten Rauschen behaftete Phasenfehler ϑ wird in dieser Zeit zu null geregelt. Die PLL fängt also sofort. Nach 47 Sekunden ist die Zeitspanne abgelaufen, innerhalb derer der Betrag des Phasenfehlers ϑ kleiner ist als die vorgegebene Grenze von 0.25π. Die Stufenzahl i wird um eins erhöht. Wie man sehen kann, verringern sich dadurch sowohl die Datenrate als auch der Anteil des Rauschens im Phasenfehler ϑ. Der Phasenfehler bleibt innerhalb der 50-ten und der 220-ten Sekunde innerhalb der vorgegebenen Grenzen. Allerdings weicht er merklich von null ab. Nach der 220-ten Sekunde kann die Filterkaskade wieder um eine Stufe verlängert werden, so daß sie jetzt 6 Stufen tief ist. u_f wird nur noch alle 6.5 Sekunden verändert. Die Frequenz des Quarzoszillators driftet nun schneller, als sie durch den Regelkreis korrigiert werden kann. Dies führt zu einem Ansteigen des Phasenfehlers, so daß die Grenze von 0.25π überschritten wird. Im folgenden wird die Driftgeschwindigkeit D der Mischfrequenz in diesem Zeitraum abgeschätzt. Angenommen die Mischfrequenz ändere sich im Zeitraum $\Delta t = t_1 - t_0$ mit einer konstanten Geschwindigkeit um den Betrag Δf:

$$D := \frac{\Delta f}{\Delta t} = \frac{\Delta\omega}{2\pi\Delta t} \tag{5.32}$$

Nun gilt für die Änderung $\Delta\vartheta$ des Phasenfehlers in diesem Zeitraum

$$\Delta\vartheta = \int_{t_0}^{t_1} \Delta\omega dt \tag{5.33}$$

Der Phasenfehler wächst innerhalb von $t_0 = 211\text{s}$ bis $t_1 = 270\text{s}$ von 0.0562π auf 0.315π. Also folgt für die Driftgeschwindigkeit D der Mischfrequenz:

$$D = \frac{\Delta\vartheta}{\pi\Delta t^2} = \frac{0.315 - 0.0562}{(59\text{s})^2} \approx 7.43 \cdot 10^{-5}\frac{\text{H}z}{\text{s}} \approx 0.268\frac{\text{H}z}{\text{h}} \tag{5.34}$$

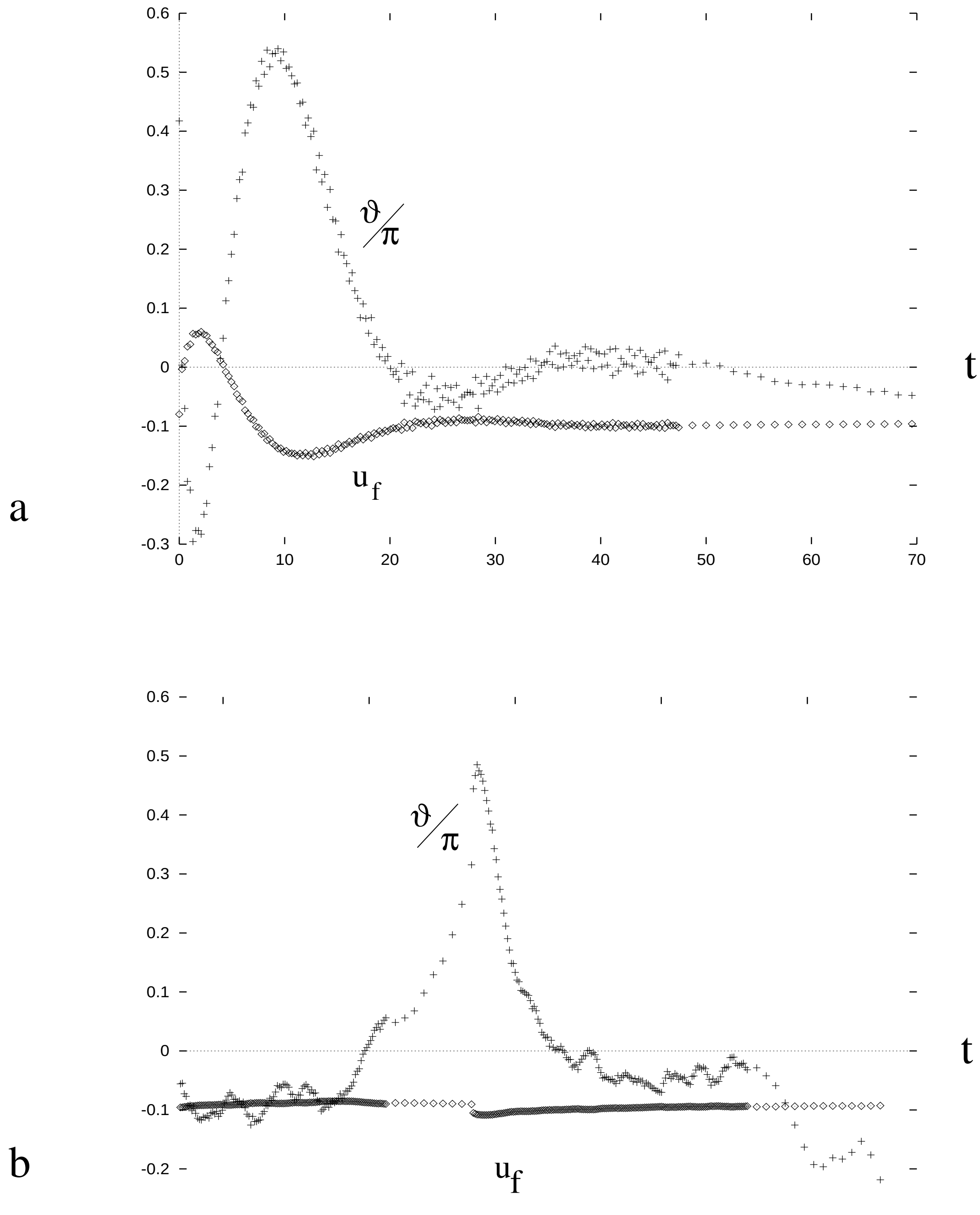

Abbildung 5.11: Messung des Phasenfehlers ϑ und der Steuergröße u_f

Diese Drift würde zum Ausrasten der PLL führen. Der Empfänger verkürzt die Filterkaskade jedoch in dem Moment auf fünf Stufen, in dem die Toleranzgrenze für den Phasenfehler von 0.25π überschritten wird. Dies geschieht in der 270-ten Sekunde. Die Regelrate ist nun wieder um den Faktor fünf größer geworden, und der Phasenfehler kann gut ausgeregelt werden. Allerdings steigt der Phasenfehler noch bis über den Wert 0.5π. Bei einem so großen Phasenfehler arbeitet der Demodulator nicht mehr korrekt. Das Signal wird nicht mehr im Scheitel der Sinuskurve abgegriffen, sondern genau im Nulldurchgang, vergleiche Abbildung 3.3 auf Seite 22. Der gewählte Wert für die Toleranzgrenze des Phasenfehlers ist also in diesem Sinne zu groß.

Nach der 460-ten Sekunde wiederholt sich das Verhalten: Die Stufentiefe wird wieder auf sechs erhöht und der Phasenfehler kann nicht schnell genug ausgeregelt werden. Im weiteren Verlauf, der hier nicht mehr dargestellt ist, wechselt die Stufentiefe immer zwischen den Werten 5 und 6.

Kapitel 6

Implementierung des Empfängers

Der Empfänger besteht aus einem Analogteil und einem Digitalteil. Der Analogteil realisiert den Taktgenerator und das Herabmischen des RF-Signals auf eine Zwischenfrequenz. Das herabgemischte Signal wird anschließend gesampelt und digital weiterverarbeitet. Die digitale Signalverarbeitung leistet ein Programm, welches auf einem digitalen Signalprozessor (DSP) abläuft. Der Signalprozessor befindet sich auf einer Entwicklungsplatine, die über eine Schnittstelle mit einem Steuercomputer verbunden ist. Für den Steuercomputer gibt es Programme, die das Assemblieren und Laden der DSP-Programme und die Fehlersuche unterstützen.

Die folgenden beiden Abschnitte erläutern den Analog- und den Digitalteil. Der Abschnitt über den Digitalteil enthält Ausführungen über die Architektur des verwendeten Signalprozessors. Der Empfängeralgorithmus wurde in der Programmiersprache C kodiert. In Abschnitt 6.3 werden diverse Funktionen vorgestellt, die als Softwareschnittstelle zwischen dem DSP und dem C-Programm fungieren. Diese Funktionen sind in Assembler geschrieben, so daß sie dem sie aufrufenden C-Programm die speziellen Eigenschaften der DSP-Architektur zur Verfügung stellen können. In Abschnitt 6.4 wird eine Übersicht über die Struktur des C-Programms gegeben.

6.1 Der Analogteil

Die im Zuge dieser Arbeit entwickelte Analogschaltung zeigt Abbildung 6.1. Das Herabmischen der Samplefrequenz geschieht mit dem Analogmultiplizierer AD835. Dieser kann das Produkt aus zwei Eingangsspannungen zwischen -1 und 1 Volt bilden. Die beiden Hochpässe (HP) sollen mögliche Gleichspannungsanteile der Eingangssignale herausfiltern. Der Tiefpaß (TP) am Ausgang des Multiplizierers unterdrückt die bei der Mischung entstehenden hochfrequenten Anteile. Das Antennensignal wird mit einem Bandpaß (BP) vorselektiert, und anschließend mit einem Operationsverstärker verstärkt.

Ein regelbarer Quarzoszillator war nicht erhältlich, so daß dieser aus diskreten Bauteilen aufgebaut werden mußte. Der VCO enthält einen Schwingquarz der Frequenz 24576kHz. Durch Übersteuerung erhält man aus dem harmonisch schwingenden Signal am Schwingquarz ein Rechtecksignal. Die Variation der Oszillatorfrequenz geschieht mittels einer Kapazitätsdiode [TIE93].

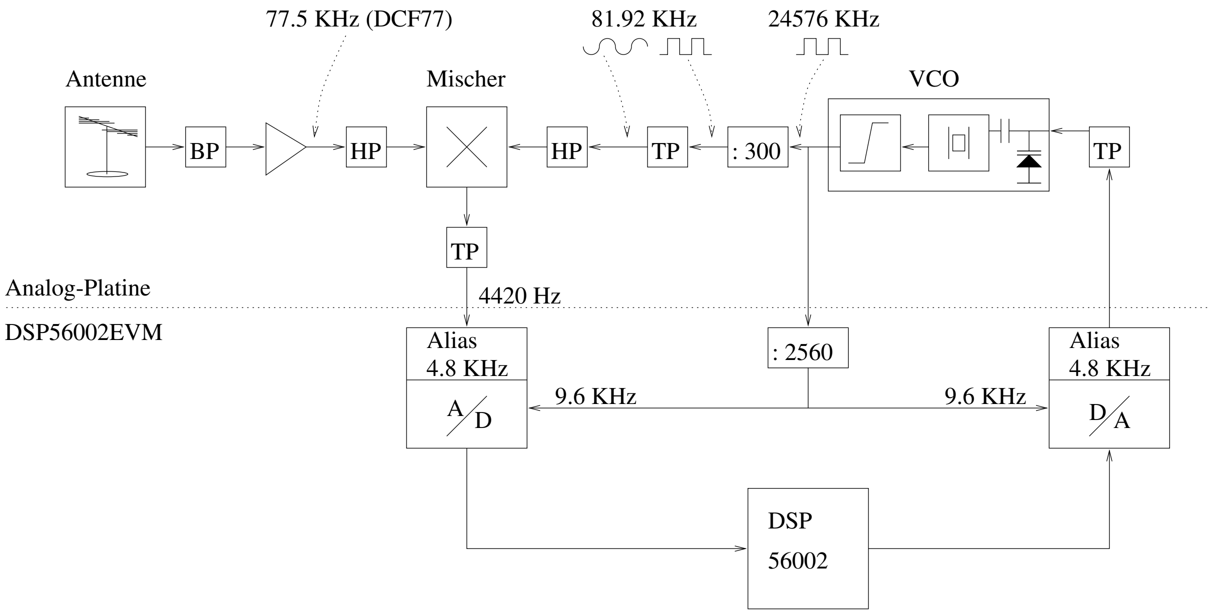

Abbildung 6.1: Analog- und Digitalplatine

Diese hat eine Kapazität, die von der angelegten Spannung abhängt. Die Kapazität wiederum beeinflußt in geringem Maße die Frequenz des Schwingkreises. Der Oszillator wurde so aufgebaut, daß die maximale Frequenzabweichung etwa ± 200Hz betrug.

Um Störungen bei der Übertragung der Regelspannung vom D/A-Wandler zur Kapazitätsdiode zu unterdrücken, wurde der Kapazitätsdiode ein Tiefpaß vorgeschaltet. Das digitale Ausgangssignal des VCO wird nun einerseits der DSP-Platine zugeführt, und andererseits zur Generierung der Mischfrequenz verwendet. Das Mischsignal sollte möglichst sinusförmig sein, also hauptsächlich die Grundfrequenz enthalten. Das Rechtecksignal am Ausgang des Taktteilers enthält jedoch Oberwellen. Diese werden durch das nachgeschaltete Tiefpaßfilter unterdrückt. Dem kommt zugute, daß die erste Oberwelle bei 163.84kHz bei Rechtecksignalen nicht vorkommt [LAG87]. Der Taktteiler wurde aus einfachen Zählerbausteinen aufgebaut. Durch Teilung der Oszillatorfrequenz durch den Faktor 300 erhält man die Frequenz 81.92kHz. Wird diese mit dem DCF77-Signal bei 77.5kHz gemischt, so erscheint das DCF77-Signal bei 4420Hz. Da gefordert wurde, daß die Frequenz des Eingangssignals der Filterkaskade ein Viertel der Samplefrequenz betragen soll, muß das DCF77-Signal im Digitalteil noch weiter von 4420Hz auf 2400Hz herabgemischt werden.

Nun kann der für die Dimensionierung der PLL notwendige Wert K_o berechnet werden. K_o gibt die lineare Abhängigkeit der Kreisfrequenz des o-Signals von der Steuergröße u_f wider, siehe Gleichung 5.13. Das o-Signal ist das auf 4420Hz herabgemischte DCF77-Signal. Da der Teilungsfaktor 300 viel kleiner ist als der Teilungsfaktor zur Ableitung der Samplefrequenz, kann zur Vereinfachung davon ausgegangen werden, daß die Samplefrequenz konstant sei, und daß nur die Variabilität der Mischfrequenz die Frequenz des o-Signals verändert. Dann entspricht einer maximalen Frequenzabweichung von ± 200Hz im Quarzoszillator eine maximale Frequenzabweichung von $\Delta f_o = \frac{200}{300}$Hz in der Mischfrequenz und im o-Signal. Die maximale Frequenzabweichung tritt ein, wenn $u_f = \pm 1$. Also kann K_o wie folgt berechnet werden:

$$K_o = \frac{\Delta\omega_o}{1} = 2\pi\Delta f_o = 2\pi\frac{200}{300}\text{Hz} = \frac{4}{3}\pi \tag{6.1}$$

Alle Hoch-, Band- und Tiefpässe wurden mit einfachen RC-Gliedern implementiert.

6.2 Der DSP-Teil

Die Anforderungen an die Rechenleistung des Digitalteils sind sehr gering. In Abschnitt 5.1 wurde hergeleitet, daß zur Berechnung der Filterkaskade pro Sample durchschnittlich 3.25 MACs notwendig sind. Der Rechenaufwand zur Korrelation ist noch geringer. Die vom verwendeten Signalprozessor zur Verfügung gestellte Rechenleistung wird gar nicht in diesem Umfang benötigt. Wahrscheinlich würde die Leistung eines billigen 8-Bit-Microcontrollers für diese Anwendung ausreichen.

Der verwendete Signalprozessor ist der Prozessor DSP56002 der Firma Motorola [AND86] [MOT95,1] [MOT95,2]. Diese Firma stellt auch die hier verwendete Entwicklungsplatine DSP56000EVM her. Diese Platine unterstützt das Produkt DSP56002 und ist gedacht für Entwickler von Anwendungen, die vom DSP56002 Gebrauch machen sollen. Auf der Platine befinden sich außer dem Signalprozessor auch noch Speicher und der Wandlerbaustein CS4215. Dieser Baustein enthält jeweils zwei A/D- und D/A-Wandler. Die Verbindung zum Steuercomputer bewerkstelligt eine RS232-Schnittstelle.

Die Verbindung zum Analogteil ist in Abbildung 6.1 dargestellt. Der Wandlerbaustein benötigt einen Takt von etwa 24576kHz. Diese Frequenz wird intern durch den Faktor 2560 geteilt, so daß sich die Samplefrequenz von 9.6kHz ergibt. Auch andere Samplefrequenzen sind einstellbar. Sowohl das Ein- als auch das Ausgangssignal passieren jeweils ein Aliasfilter, also einen Tiefpaß, bei dem die Grenzfrequenz die halbe Samplefrequenz beträgt. Aus diesem Grunde ist es nicht möglich, das DCF77-Signal unterabzutasten. Es war problemlos, das Taktsignal an einen Pin des Wandlerbausteins heranzuführen. Über den einen freien Ausgabekanal konnten zur Fehlersuche interne Signale im Empfänger ausgegeben und auf einem Oszilloskop sichtbar gemacht werden.

Die Programmierung eines DSPs erfordert Hintergrundwissen über dessen Architektur. Der Rest dieses Abschnittes befaßt sich mit dieser Architektur.

Ein DSP ist ein *dediziertes System*, das heißt, er ist auf eine bestimmte Klasse von Anwendungen, die Signalverarbeitung, zurechtgeschnitten. Diesem Verlust an Allgemeinheit steht ein Effizienzgewinn gegenüber. Anwendungen der Signalverarbeitung zeichnen sich durch folgende Eigenschaften aus:

- Die Algorithmen sind von einfacher Natur: Sie werden meist direkt in der Assemblersprache kodiert und benötigen wenig Speicher. Die Schachtelungstiefe von Unterprogrammaufrufen ist gering. Dynamische Speicherverwaltung wird ebensowenig benötigt wie ein Betriebssystem.

- Eine typische benutzte Wortbreite ist 24 Bit.
- Eine häufig auszuführende Operation ist die Bildung des Skalarproduktes zweier Vektoren, zum Beispiel bei der Filterung. Für die „Fast Fourier Transformation" ist die Operation der Bitumkehr eines Wortes wichtig. Bei der Bitumkehr werden die höherwertigsten Bits die niedrigwertigsten und umgekehrt. Funktionen mit hohem Rechenaufwand, zum Beispiel trigonometrische Funktionen, werden durch Tabellen realisiert.

Im folgenden wird ein Einblick in die Architektur des DSPs 56002 gegeben:

Der DSP unterscheidet verschiedene Datentypen. Ein Datentyp enthält alle Speicheradressen und ein anderer enthält alle Samples von Signalen. Der letztere Datentyp sei hier mit „fract" benannt, als Abkürzung von „fraction". Jedes Datum dieses Typs repräsentiert einen Wert s, mit

$$-1 \leq s < 1 \tag{6.2}$$

Die Wortbreite ist 24 Bit. Die Kodierung der Werte dieses Datentyps ist im bekannten Zweierkomplement. Allerdings haben die 24 Bits nicht die Wertigkeiten 0 bis 23, sondern die Wertigkeiten -22 bis 1. Mit anderen Worten: Das Komma befindet sich zwischen den zwei höchstwertigen Bits. Zum Beispiel repräsentiert das Datenwort 0110, gefolgt von weiteren zwanzig Nullen, den Wert $2^{-1} + 2^{-2} = 0.75$.

Der DSP besitzt zwei 56 Bit lange Akkumulatoren, den A- und den B-Akkumulator. Diese Akkumulatoren sind zum Aufaddieren mehrerer Werte gedacht. Das Komma befindet sich zwischen der neunthöchsten und der zehnthöchsten Binärstelle. Dementsprechend kennt der DSP also einen weiteren Datentyp „Akkumulator". Die Arbeitsweise des Akkumulators soll an einem Beispiel erläutert werden:

Beispiel 6.1 *Es ist der Wert $0.7^2 + 2 \cdot 0.9^2$ zu berechnen. Die Zahlen 0.7 und 0.9 seien in den fract-Registern* `x0` *beziehungsweise* `y0` *schon vorhanden. Zunächst wird der A-Akkumulator mit der Anweisung*

```
clr a
```

auf null gesetzt. Mit der „Multiply Accumulate"-Anweisung

```
mac x0,x0,a
```

wird das 48 Bit lange Produkt $0.7 \cdot 0.7$ zum Akkumulatorinhalt hinzuaddiert. Dieses Zwischenergebnis 0.49 ist in hexadezimaler Schreibweise:

```
003EB851EB851E
```

Durch die Anweisungen

```
mac y0,y0,a
mac y0,y0,a
```

wird nun $2 \cdot 0.9^2$ *zum Akkumulatorinhalt hinzuaddiert. Das Ergebnis ist nun*

```
010E147AE147AE
```

und damit etwa gleich 2.11. *Damit ist es außerhalb des Wertebereiches des fract-Datentyps. Wenn nun der Akkumulatorinhalt wieder in ein fract-Register oder in den Speicher geschrieben wird, so wird das Ergebnis „abgeschnitten", das heißt, es wird der Wert* `7FFFFF` *entsprechend der Zahl* 0.999999 *in das Register geschrieben.*

Dieses Abschneiden von Werten, die größer als eins sind, ist ein wesentlicher Unterschied zur konventionellen modulo-Arithmetik, bei der das Ergebnis einer Addition zweier positiver Zahlen negativ sein kann. Wenn der Überlauf eines Samples durch Abschneiden behandelt wird, dann hat dies weniger katastrophale Folgen für das Spektrum des Signals, zu dem dieses Sample gehört. Akustisch gesprochen entspricht dem Abschneiden eine Verzerrung des Schallsignals, während ein Überlauf in der modulo-Arithmetik gleich ein lautes Knacken hervorruft.

Der DSP hat eine Harvard-Architektur. Er kann gleichzeitig auf maximal drei verschiedene Speicher zugreifen, den P- den X- und den Y-Speicher. Zu den meisten Befehlen können gleichzeitig zwei Adreßberechnungen parallel ausgeführt werden.

Der DSP unterstützt auch eine spezielle Adreßarithmetik, die die Verwaltung von Puffer- oder Ringspeichern erleichtert. Die Größe eines solchen Speicherbereiches muß dann eine Zweierpotenz sein, und die Startadresse des Speicherbereiches muß ein vielfaches der Größe betragen. Durch Setzen einiger Register auf spezielle Werte kann der DSP dazu gebracht werden, daß das Ergebnis von Adreßberechnungen niemals außerhalb des vorgesehenen Bereiches liegt.

Beispiel 6.2 *Für einen Ringspeicher mit* 64 *Elementen sei der Speicherbereich mit den Adressen von* 192 *bis* 255 *im X-Speicher reserviert. Ein Zeiger in einem Adreßregister* `r0` *zeige gerade auf die Adresse* 255. *Wenn das modulo-Register* `m0` *den Wert* 63 *hat, dann bewirkt die Anweisung*

```
move x:(r0)+,x0
```

daß der Inhalt des X-Speichers an der Adresse 255 *in das Register* `x0` *geladen wird, und daß der Zeigerwert im Register* `r0` *um ein Element im Ringspeicher verschoben wird. Da der Wert in* `r0` *am Ende des Speicherbereiches angelangt war, wird er auf* 192 *zurückgesetzt. Ansonsten wäre er einfach inkrementiert worden.*

Das folgende Beispiel enthält eine effiziente Befehlsfolge zur Berechnung des Skalarproduktes.

Beispiel 6.3 *In den Ringspeicher aus Beispiel 6.2 werden fortlaufend alle Samples, die vom A/D-Wandler kommen, eingespeichert. Im Y-Speicher befindet sich eine Folge von* n *Filterkoeffizienten, die an einer frei wählbaren Adresse* k *beginnt. Zu irgendeinem Zeitpunkt habe der Zeiger in* `r0` *einen Wert, der auf das zuletzt gewandelte Sample zeigt, und* `r4` *habe den Wert* k. *Die zugehörigen modulo-Register* `n0` *und* `n4` *seien entsprechend gesetzt. Folgende Befehlsfolge berechnet das nächste Sample am Ausgang des FIR-Filters:*

```
clr  a
move            x:(r1)+,x1  y:(r4)+,y1
rep  #n
 mac x1,y1,a    x:(r1)+,x1  y:(r4)+,y1
```

Die `move`*-Anweisung lädt parallel ein Sample und einen Filterkoeffizienten. Die* `rep`*-Anweisung bewirkt, daß der folgende* `mac`*-Befehl n mal ausgeführt wird. Mit jeder dieser* `mac`*-Ausführungen wird eine Multiplikation, eine Addition, zwei Speicherzugriffe und das Inkrementieren beziehungsweise Zurücksetzen der beiden Zeiger geleistet.*

6.3 Die Softwareschnittstelle für den DSP

Es wurde die Programmiersprache C zur Kodierung des Empfängeralgorithmus' verwendet [KER77]. Diese Programmiersprache ist die einzige, für die ein Compiler für den DSP56002 existiert. Zwar hat es sich als etwas mühsam herausgestellt, mit den verfügbaren Mitteln ein C-Programm auf dem DSP zum Laufen zu bringen, aber diese Mühe hat sich auf jeden Fall gelohnt, denn die Vorteile einer Programmierung in C gegenüber der Assemblerprogrammierung sind immens: Die Fehlersuche in Assemblerprogrammen kann sehr aufreibend sein, und C-Programme sind leichter zu ändern. Weiterhin erweist sich die Portabilität der Sprache C als günstig, denn die Algorithmen können auf einer anderen Plattform entwickelt und ausprobiert werden. In diesem Fall entspringt das Eingangssignal nicht einer Antenne, sondern wird vom Programm selbst erzeugt, also wird die Realität *simuliert*. Auf diese Weise können Parameter des zu verarbeitenden Signals beeinflußt werden.

Leider nimmt der zur Verfügung stehende Compiler auf die Besonderheiten des DSPs keine Rücksicht. Tatsächlich verwirklicht er eine modulo-Ganzzahlarithmetik. Aus diesem Grund wurden verschiedene Unterprogramme direkt in Assembler geschrieben, die die speziellen Fähigkeiten des DSPs zur Verfügung stellen. Diese Routinen können vom C-Programm aufgerufen werden, und definieren damit eine Softwareschnittstelle zur DSP-Hardware.

Da der DSP verschieden Datentypen kennt, ist es zweckmäßig, auch auf der Hochsprachenebene verschiedene Datentypen zu definieren. Es wurden drei Datentypen definiert:

- Der Datentyp `fract`, siehe oben.
- Der Datentyp `fract48` ähnelt dem Datentyp `fract`, da die Elemente dieses Datentyps auch nur Zahlen zwischen -1 und 1 repräsentieren. Auch die Arithmetik, die auf diesem Datentyp zur Anwendung kommt ist analog zur `fract`-Arithmetik. Der Unterschied besteht in der größeren Wortbreite von 48 Bit. Dieser Datentyp ist nicht zu verwechseln mit dem Prozessor-Datentyp „Akkumulator", denn dessen Elemente sind 56 Bit breit.
- Um die Möglichkeiten der schnellen Berechnung des Skalarproduktes und der speziellen Adreßarithmetik auszunutzen, wurde der Datentyp `Buffer` definiert. Dieser dient zur Implementierung von Ringspeichern.

Für den Datentyp `fract` wurden Funktionen in Assembler programmiert, die grundlegende arithmetische Funktionen wie Addition oder Multiplikation mit der DSP-eigenen Überlaufarithmetik implementieren.

Das Schleifenfilter im PLL-Regler ist ein Integrator. Algorithmisch wird dieses integrierende Verhalten umgesetzt, indem in jedem Regeltakt zu einem Akkumulator ein Teilbetrag der Gesamtsumme hinzuaddiert wird, vergleiche den Algorithmus auf Seite 48. Für den Akkumulator wird eine Genauigkeit von 48 Bit verwendet. Die auf dem Datentyp `fract48` ausführbaren Funktionen, die benötigt werden, sind:

```
void  Clear48(fract48 *f48);
void  MAC48(fract f1,fract f2,fract48 *akku);
fract High24(fract48 f48);
```

Die erste Funktion setzt eine Variable dieses Datentyps auf null. Die Funktion `MAC48` addiert das 48 Bit breite Produkt der ersten beiden Parameter zu einer `fract48`-Variable hinzu. Die letztgenannte Funktion dient der Typkonvertierung, und liefert das auf 24 Bits gerundete `fract48`-Argument.

Auf dem Datentyp `Buffer` sind folgende Operationen ausführbar:

```
void  InitBuffer(int size,fract *freeMemory,Buffer *buffPtr);
void  InsertBuffer(fract f,Buffer *buffPtr);
fract SProd(int noOfCoefs,Buffer *buffPtr,fract coefs[]);
```

Die erste Funktion erwartet als Argumente die Größe des Ringpuffers und einen Zeiger auf einen freien Speicherbereich. Über den „call-by-reference“-Mechanismus initialisiert sie eine Instanz des Datentyps `Buffer`. Die Funktion `InsertBuffer` fügt ein neues Sample in den Ringspeicher ein. Die Funktion `SProd` erwartet Zeiger auf einen Ringspeicher und auf die Koeffizenten und deren Anzahl. Diese Funktion berechnet das Skalarprodukt.

Zwei wichtige Funktionen sind noch zu erwähnen. Wie jedes vernünftige Programm benötigt auch der Empfängeralgorithmus eine Eingabe, um aus ihr eine Ausgabe zu produzieren. Die Schnittstelle zu den A/D- beziehungsweise D/A-Wandlern leisten die Funktionen

```
fract WaitForSample();
void  OutputSample(fract);
```

Der Empfängeralgorithmus arbeitet in einer Endlosschleife, in der stets ein empfangenes Sample verarbeitet wird, und anschließend pro Ausgabekanal ein Sample ausgegeben wird, also

```
while true
  { s=WaitForSample();
    ProcessSample(s,&uf);
    OutputSample(uf);
  };
```

Die Funktion `ProcessSample` implementiert den Empfängeralgorithmus. `uf` sei dabei die Größe, die den Taktgenerator regelt. Die Rechenzeit, die eine Ausführung der Funktion `ProcessSample` in Anspruch nimmt, variiert von mal zu mal stark. Die Berechnung des Multiratenfilters erfordert in den meisten Fällen nur ein Hineinschieben des aktuellen Samples in einen Ringspeicher. Alle 5^n Takte jedoch müssen alle Dezimationsfilter bis zur n-ten Stufe berechnet werden. Ein Problem entsteht, wenn die Rechenzeit, die eine Ausführung der Funktion `ProcessSample` in Anspruch nimmt, einmal länger als $\frac{1}{9600}$ Sekunde ist, so daß der A/D-Wandler *während* dieser Ausführung ein neues Sample produziert. Dieses Problem wurde wie folgt gelöst: Der A/D-Wandler initiiert in dem Moment, in dem ein neues Sample zur Verfügung steht, eine Unterbrechung im DSP. Die zugehörige Unterbrechungsbehandlungsroutine schreibt das so empfangene Sample zunächst in einen Puffer, von wo es später von der Routine `WaitforSample` abgeholt wird. Mit dieser nebenläufigen Methode wurde das klassische Erzeuger-Verbraucher-Problem gelöst. Da die Unterbrechungsbehandlungsroutine selber nicht wieder unterbrochen werden kann, kommt man bei der Implementierung dieser Methode ohne Semaphoren oder den Algorithmus von Dekker aus [ARI85].

6.4 Die Struktur des Programmes

Der Empfänger ist ein endlicher Automat, das bedeutet, er besitzt zu jedem Zeitpunkt einen *Zustand*, der mit jedem Empfang eines neuen Samples mittels einer Zustandsübergangsfunktion verändert wird, wobei auch eine Ausgabe berechnet wird. Der Zustand ist, den drei Hauptteilen entsprechend, in den drei statischen Variablen `mrfKaskade`, `detektor` und `pll` gespeichert. Die Verbundvariable `mrfKaskade` enthält im wesentlichen für jede Filterstufe eine Instanz des `Buffer`-Datentyps, damit das Dezimationsfilter darauf berechnet werden kann. Die Variable `detektor` enthält für jede Stufe des Detektors die nötigen Komponenten wie Akkumulatoren, Zähler und Zustand. Die Variable `pll` enthält die Komponenten „u_d"„u_f" und „akku", entsprechend dem Programmstück auf Seite 48. Die Zustandstransformation leistet die Funktion `ProcessSample`:

```
void ProcessSample(fract sample,fract *uf)
  { fract mrfOut[maxDepth];
    int   l;
    MrfKaskade(sample,&mrfKaskade,&mrfOut,&l);
    PLL(mrfOut,l,&pll);
    Detektor(mrfOut,l,&detektor);
    *uf=pll.uf;
  };
```

Die Funktion `MrfKaskade` berechnet die Multiratenfilterkaskade unter Hinzufügung eines neuen Samples. Dabei verändert sich ihr Zustand `mrfKaskade`. Als Ausgabe liefert diese Funktion eine Liste variabler Länge. Diese Liste wird durch das Feld `mrfOut` implementiert, wobei `l` die Länge der Liste enthält. Die Elemente der Liste sind die Samples an jedem Ausgang einer Stufe der Kaskade. Die Länge der Liste variiert, da die Datenraten an den Ausgängen der verschiedenen Stufen mit zunehmender Tiefe abnehmen. Dies sei an einem Beispiel erläutert:

Beispiel 6.4 *Die Dezimationsrate ist fünf. Das bedeutet, daß die ersten vier Samples, die nach dem Einschalten empfangen wurden, zwar in einem Puffer für die erste Stufe der Filterkaskade gespeichert werden, daß aber noch keine Berechnung des Skalarproduktes mit den Koeffizienten des Dezimationsfilters stattfindet. Dies geschieht erst mit dem Eintreffen des fünften Samples. Entsprechend ist die Länge* `l` *in obigem Programmstück viermal null und einmal eins. Die Größe* `l` *ist nur jedes fünfte mal größer als null. Der Ausgang der zweiten Stufe liefert nur jedes fünfundzwanzigste mal ein Sample. Dementsprechend ist die Größe* `l` *nur jedes fünfundzwanzigste mal größer als eins.*

Der Detektor greift sein Eingangssignal nach der zweiten Stufe der Filterkaskade ab. Wenn `l` größer als eins ist, dann ist das entsprechende Sample das zweite Element der Liste `mrfOut`. Nur in diesem Fall führt die Funktion `Detektor` überhaupt eine Zustandsänderung durch. Die Größe `l` dient also dazu, die Funktionseinheiten mit ihren verschiedenen Datenraten zu *synchronisieren*.

Nachdem dieses Synchronisationsproblem gelöst wurde, ist die Kodierung der Funktionen `PLL` und `Detektor` als C-Funktionen einfach.

6.5 Der Einfluß der Begrenztheit der Wortbreite

Daß der Empfänger mit einer begrenzten Wortbreite arbeitet hat einige Konsequenzen:

Alle Samples und Koeffizienten, mit denen die Filterkaskade berechnet wird, sind 24 Bit breit. Daraus ergibt sich ein digitales Grundrauschen. Die Samplewerte sollten wesentlich größer sein als das Digitalrauschen. Deswegen ist es notwendig, die Signale zwischen verschiedenen Stufen der Filterkaskade geeignet zu verstärken. Bei sehr schmaler Bandbreite könnten diese Signale ansonsten zu schwach werden.

Bei der Korrelation im Detektor wird eine Summe aus beliebig vielen Summanden gebildet. Diese Summe kann jede Grenze überschreiten und zum Überlauf führen. Um dieses Problem zu umgehen, kann man zum Beispiel die voraussichtliche Arbeitsdauer des Empfängers begrenzen. Ist die Wortbreite des Akkumulators nur 48 Bit größer als die Wortbreite der Summanden die aufaddiert werden, so kann es bei einer maximalen Datenrate von 192Hz im Detektor erst nach etwa 23000 Jahren zum Überlauf kommen. Doch wurde ein anderer Weg beschritten:

Sei m_i der Wert eines Akkumulators zum Zeitpunkt i. Zum Zeitpunkt $i+1$ soll nun ein neuer Wert a_i hinzuaddiert werden. Statt der einfachen Addition wird jedoch der Wert

$$m_{i+1} = \gamma a_i + (1-\gamma)m_i \tag{6.3}$$

gebildet, wobei γ klein gegen eins ist. Diese Methode verhindert das Überlaufen.

Kapitel 7

Bewertung

In diesem Kapitel soll der vorgestellte Empfänger bewertet werden. Dabei sei das wichtigste Ergebnis zuerst genannt: Der Empfänger funktioniert ! Sowohl die PLL-Regelung als auch die Detektion der Information verhalten sich wie erwartet, so daß der Empfänger die richtige Uhrzeit anzeigt.

Für eine tiefergehende Analyse sind verschiedene Parameter in Betracht zu ziehen. Diese Parameter sind die Zeitdauern vom Fangen der PLL bis zur Vollendung der Sekunden- beziehungsweise Minutensynchronisation, die Zeitdauer bis zur Anzeige der korrekten Uhrzeit und die maximal erreichte Stufenanzahl der Filterkaskade. Diese Parameter wurden gleichzeitig gemessen mit dem Verlauf der Steuergröße u_f und des Phasenfehlers ϑ aus Abbildung 5.11.

Die PLL hat nach 47 Sekunden gefangen. Von nun an dauert es nur 4 Sekunden bis zum Einsynchronisieren auf den Sekundentakt. Die anschließende Synchronisation auf den Minutentakt benötigt 5 Minuten und 31 Sekunden. Das Lesen der Zeitinformation ist hingegen 4 Minuten und 21 Sekunden später abgeschlossen. Es fällt auf, daß die Minutensynchronisation relativ lange dauert. Das liegt zum einen daran, daß von den 60 Samples, die pro Minute bei der Minutensynchronisation zur Korrelation herangezogen werden, nur eines Information über die Lage des Minutenanfangs enthält. Ein anderer Grund könnte der zeitweilig hohe Phasenfehler während der Minutensynchronisation sein, vergleiche Abbildung 5.11. Die Minutensynchronisation dauert von der 51-ten bis zur 382-ten Sekunde.

Die maximal erreichte Stufenzahl ist 6. Die Bandbreite der Filterkaskade betrug also zeitweilig 0.39Hz.

Die Zeitinformation konnte so schnell gefunden werden, da das empfangene Signal in diesen Messungen mit wenig Rauschen behaftet war. Leider ist es nicht gelungen, die Qualität des Signals noch weiter zu verschlechtern. Eine aussagekräftige Abschätzung der Leistungsfähigkeit des Empfängers läßt sich nur unter schlechten Empfangsbedingungen machen. Ein experimenteller Vergleich des Empfängeres mit kommerziellen Funkuhren an verschiedenen Orten der Erde wäre sehr interresant.

Es folgen einige bewertende Bemerkungen:

Ein großer Nachteil des vorgestellten Empfängerkonzeptes ist die Bewegungsempfindlichkeit der Uhr. Ein Bewegen der Antenne kann zu einer Phasenverschiebung des empfangenen Signals führen. Solange dieser Phasenfehler nicht ausgeregelt ist, wird der Demodulator schlecht oder falsch arbeiten. Bei einer Phasenverschiebung um π liefert der Demodulator negative Samples anstelle von positiven. Der in Abschnitt 5.3 erläuterte Hysteresemechanismus kann durch Bewegungen der Uhr zunichte gemacht werden. Sind aber die zeitlichen Abstände der Bewegung wesentlich größer als die Zeit, die zum Ausregeln des dadurch entstehenden Phasenfehlers benötigt wird, so funktioniert die Uhr immer noch korrekt. An eine Implementierung als Armbanduhr ist aber in keinem Fall zu denken.

Die Uhr geht nur auf $\frac{1}{20}$ Sekunde genau. Mit verbesserten Methoden läßt sich diese Genauigkeit sicher erhöhen. Der wesentliche Punkt bei dem vorgestellten Empfänger ist aber, daß er die Uhrzeit überhaupt detektieren kann, auch unter möglichst schlechten Empfangsbedingungen.

Zwei weitere Nachteile des vorgestellten Entwurfes entstehen aus der Tatsache, daß das DCF77-Signal vor der Digitalisierung auf 4420Hz herabgemischt wird. Es wäre stattdessen viel günstiger, das Signal durch Unterabtastung zu digitalisieren. Leider ist Unterabtastung mit dem DSP56002EVM nicht möglich, da die Aliasfilter nicht umgangen werden können. Der erste Nachteil ist, daß die zur Mischung benötigte zusätzliche Analogelektronik ein Eigenrauschen hervorruft, welches die Qualität des digitalen Signals zusätzlich herabsetzt. Der andere Nachteil ist, daß sich die Drift des Quarzoszillators auf diese Weise viel stärker auswirkt. Die Mischfrequenz von 81920Hz wird ja durch Frequenzteilung um den Faktor 300 aus der Quarzfrequenz abgeleitet. Das bedeutet, daß die Drift der Mischfrequenz 300 mal so klein wie die Drift der Quarzfrequenz ist. Würde der Empfänger jedoch mit Unterabtastung arbeiten, so könnte dieser Faktor viel größer sein. In diesem Fall würde nur die Samplefrequenz vom Quarzoszillator abgeleitet werden. Eine günstige Samplefrequenz wäre zum Beispiel 12kHz. Diese könnte aus derselben Quarzfrequenz durch Teilung um den Faktor 2048 hervorgehen. Mit Unterabtastung wäre die Drift dann um den Faktor $\frac{2048}{300} \approx 6.8$ geringer. Die maximal erreichte Stufenanzahl der Filterkaskade wäre dementsprechend nicht sechs sondern sieben.

Es stellt sich weiterhin die Frage, wieso denn überhaupt die kohärente Demodulation angewendet wurde. Wenn die Schwachstelle des Empfängers tatsächlich das Fangen der PLL ist, warum demoduliert man das Signal nicht einfach per Quadraturmischung ? Dann könnte man auf den PLL-Regler und die Multiratenfilterkaskade ganz verzichten. Dazu ist zu sagen, daß alle inkohärenten Demodulationsverfahren mit einer Betragsbildung des empfangenen Signals einhergehen [KAM96]. Deswegen ist der im demodulierten Signal enthaltene Rauschanteil nicht näherungsweise symmetrisch um den Mittelwert null verteilt, und deswegen hebt sich das Rauschen bei der Korrelation auch nicht so schnell gegen sich selbst auf. Letztendlich hilft aber bei der Entscheidung dieser Frage wieder nur ein experimenteller Vergleich weiter.

Je größer die maximale Stufenanzahl der Filterkaskade ist, desto besser funktioniert der Hysteresemechanismus. Die maximale Stufenanzahl hängt aber vor allem von der Stabilität der Frequenz des Quarzoszillators ab. In Abschnitt 5.3.1 wurde die Driftgeschwindigkeit der Mischfrequenz $D = 0.268$Hz/h berechnet. Die Driftgeschwindigkeit der Quarzfrequenz beträgt das 300-fache, also 80Hz/h. Daß dieser Wert ist so hoch ist, liegt am Aufbau der Analogplatine, der noch in vielerlei Hinsicht optimierbar ist. Schon jeder leichte Luftzug kann die Quarzfrequenz beeinflussen.

Die im Empfänger implementierte Idee, eine Multiratenfilterkaskade und eine PLL-Regelung miteinander zu verbinden, könnte auch in anderen Anwendungen verwendet werden, nämlich überall dort, wo genaue Frequenzen benötigt werden.

Der in dieser Arbeit vorgestellte Empfänger unterscheidet sich in einem Punkt von den Funkuhren, die an der Technischen Hochschule in Darmstadt oder an der Universität in Stuttgart entwickelt wurden. Alle diese Empfänger entscheiden in jeder Sekunde, ob die empfangene Marke eine logische null oder eine logische eins ist. Wenn diese Entscheidung aber falsch war, dann geht viel Information verloren. Dies beeinträchtigt die Leistungsfähigkeit der Uhr auch dann, wenn mehrere solcher 0-1-Entscheidungen aus verschiedenen Minuten zur Redundanzausnutzung miteinander in Beziehung gesetzt werden. In dem in dieser Arbeit behandelten Empfänger werden solche 0-1-Entscheidungen im Sekundentakt gar nicht getroffen. Das Verfahren zur Redundanzausnutzung arbeitet nicht nur mit den beiden Werten 0 und 1 sondern mit einem Wertekontinuum.

Literaturverzeichnis

[AND86] G. Anderer in *Signalprozessoren 3*, Oldenbourg Verlag, 1986

[ARI85] M. Ben-Ari: *Grundlagen der Parallelprogrammierung*, Hanser,1985

[BAD87] Kurt Baderschneider: *Detektion deterministisch und stochastisch gestörter Zeitzeichen*, Dissertation, TH Stuttgart, 1987

[BEC72] G. Becker:*Aussendung und Empfang des Zeitmarken und Normalfrequenzsenders DCF77*, PTB-Mitteilungen Heft 4/1972, Vieweg

[BEC73] G. Becker, P. Hetzel:*Kodierte Zeitinformation über den Zeitmarken- und Normalfrequenzsender*, PTB-Mitteilungen Heft 3/1973, Vieweg

[BEC77] G. Becker, P. Hetzel:*Kodierung von Sommerzeit und anderen Informationen beim Sender DCF77*, PTB-Mitteilungen Heft 2/1977, Vieweg

[BEL89] Ulrich Bellmann: *Meßplatz zum Prüfen und Entwickeln von Funkuhrempfängern*, Dissertation, TH Darmstadt, 1989

[BER85] Rainer Bermbach: *Neue Funkuhrkonzepte durch enge Verflechtung von Empfänger und Microcomputer*, Dissertation, TH Darmstadt, 1985

[BES93] Roland Best: *Theorie und Anwendungen des phase locked loops*, AT-Verlag, 1993

[CRE88] Michael Cremer: *Regelungstechnik*, Springer Verlag, 1988

[FLI93] Fliege: *Multiraten-Signalverarbeitung*, Teubner, 1993

[GAR66] Floyd M. Gardner: *Phaselock Techniques*, New York: Wiley, 1966

[HAH89] Istvan Hahn: *Sonnentage - Mondjahre*, Uranina Verlag, 1989

[HET87] P. Hetzel:*Zeitübertragung auf Langwelle durch amplitudenmodulierte Zeitsignale und pseudozufällige Umtastung der Trägerphase*, Dissertation, Universität Stuttgart, 1987

[HIN91] K.Hinderer: *Grundbegriffe der Wahrscheinlichkeitstheorie*, Springer Verlag, 1972

[KAM96] K.D.Kammeyer: *Nachrichtenübertragung*, Teubner, 1996

[KER77] B.Kernighan,D.Ritchie: *The C Programming Language*, Prentince-Hall, 1977

[KLA92] Gerd Klawitter:*Zeitzeichensender = Time Signal Stations*, Siebel Verlag, 1992

[KNE57] Kneschke:*Differentialgleichungen und Randwertprobleme*, Volkseigener Betrieb Verlag Technik, 1957

[LAG87] Klaus Lagemann: *Rechnerstrukturen*, Springer-Verlag, 1987

[MOT95,1] *DSP56000 Digital Signal Processor Family Manual*, Motorola Inc., 1995

[MOT95,2] *DSP56002 Digital Signal Processor User's Manual*, Motorola Inc., 1995

[PRE81] Hans-Georg Preller: *Beiträge zur geschlossenen Behandlung des digitalen Phase-Locked Loop unter besonderer Berücksichtigung von Quantisierungseffekten*, Dissertation, Universität Erlangen-Nürnberg

[SHA69] C. Shannon, W. Weaver: *The Mathematical Theory of Communication*, Univ. of Illinois Pr., Urbana, 1996

[SIE95] Wolf Siebel:*Spezial-Frequenzliste, Ausgabe 1994/1995*, Siebel Verlag, 1991

[TIE93] U. Tietze, C. Schenk: *Halbleiter-Schaltungstechnik*, Springer Verlag, 1993

[WIE88] Joachim Wietzke: *Kriterien zur einheitlichen Beurteilung der prinzipiellen Leistungsfähigkeit verschiedener Zeitzeichenempfänger unter Einbeziehung neuer digitaler Varianten*, Dissertation, TH Darmstadt

Erklärung

Hiermit erkläre ich an Eides statt, die vorliegende Arbeit selbstständig durchgeführt zu haben, und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt zu haben.

Hamburg, den

Martin Wierich