On Synchronisation Issues in Wireless Mobile Digital Communications

Richard Clarke, B.E.(Hons), B.Sc

A thesis submitted in partial fulfilment
of the requirements for the degree of
Master of Engineering
in
Electrical and Electronic Engineering
at the
University of Canterbury,
Christchurch, New Zealand.

Communications Research Group
Department of Electrical and Electronic Engineering
University of Canterbury,
Christchurch, New Zealand
June 2002

ABSTRACT

Symbol timing recovery is an important function of any digital receiver. In the wireless mobile data field the task of establishing accurate symbol timing at the receiver is complicated by the time varying channel. This time varying channel also makes the use of coherent modulation schemes significantly more difficult. This is one of the major reasons that almost all existing mobile wireless digital standards utilise some form of differential modulation and detection.

This thesis takes a primarily practical approach to the investigation of timing and phase estimation problems. The main focus of the work is on the comparison of three existing all digital timing synchronisation algorithms, two of which were originally designed for the AWGN channel, and the third was designed from ML principles for the Rayleigh fading channel. In order to test these sub-systems in the wider context of receiver performance, a pilot symbol assisted (PSAM) receiver was simulated to compare the effects of the different synchronisers on receiver steady state performance.

Finally, because the real time implementation aspects of software radio are of considerable interest to the author, some attempt has been made to migrate the MATLAB synchronisation simulations to a real time DSP platform, specifically the TMS320C6701 Texas Instruments floating point device.

ACKNOWLEDGEMENTS

To begin, I would like to express my thanks to my supervisor Professor Des Taylor, without whom I may have blundered off on one too many tangents. His guidance, though in small and unobtrusive doses, without doubt helped greatly in bringing this project to a welcome but satisfactory conclusion.

My thanks also go to Tait Electronics and the New Zealand government in the form of the Foundation for Research, Science and Technology, whose financial assistance in the form of a Bright Futures Enterprise Scholarship helped to ease the pain of transitioning from full time work to full time study. I am also grateful to Tait Electronics for the opportunity to take long term leave from my job so that I could pursue this course of study.

A special thanks must go to my colleagues and friends in the Communications laboratory, who have, over the past two or so years, kept things interesting and fun, and provided many topics of interesting conversation, not all of them Engineering related! Many thanks also, to Mary Nash, an honorary Lab member, whose personality and great sense of humour has provided me with many laughs during the course of my work.

Finally, I wish to express gratitude to my friends and family who have provided support in one form or another during my somewhat lengthy University educational career. I am especially grateful to my parents for their support, guidance and encouragement throughout my educational career. Thanks Mum and Dad!

Richard Clarke

June 2002

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

GLOSSARY

AD	Amplitude Directed
API	Application Programming Interface
ASK	Amplitude Shift Keying
AWGN	Additive White Gaussian Noise
BER	Bit Error Rate
BPF	Band Pass Filter
BPSK	Binary Phase Shift Keyed
DD	Decision Directed
DDC	Digital Down Converter
DMA	Direct Memory Access
DQPSK	Differential Quadrature Phase Shift Keying
DSP	Digital Signal Processor
EVM	Evaluation Module
FCC	Federal Communications Commission
FSK	Frequency Shift Keying
HPI	Host Port Interface
I,Q	In-phase and Quadrature signal components
IF	Intermediate Frequency
ISI	Inter-symbol Interference
ML	Maximum Likelihood
M-PSK	Multi Phase Shift Keyed digital modulation
M-QAM	Multi level Quadrature Amplitude Modulation
NCO	Numerically Controlled Oscillator
NDA	Non-Data Aided
PAM	Pulse Amplitude Modulated
PAPR	Peak to Average Power Ratio
PCS	Personal Communications Services
PD	Phase Detector
PLL	Phase Locked Loop
PSAM	Pilot Symbol Assisted Modulation
PSP	Per Survivor Processing
QAM	Quadrature Amplitude Modulation
QPSK	Quadrature Phase Shift Keying (same as 4 QAM)
RF	Radio Frequency
SER	Symbol Error Rate
SNR	Signal to Noise Ratio
TED	Timing Error Detector
TF	Transfer Function
TTIB	Transparent Tone In Band
VCO	Voltage Controlled Oscillator
VLIW	Very Long Instruction Word

CHAPTER 1
INTRODUCTION

1.1 Mobile Digital

The adoption of wireless digital communications, as a replacement for the more traditional analog wireless communication systems, has accelerated in recent years, driven by market, technology and regulatory forces.

Digital wireless enjoys many advantages over analog systems, not least of which is the ability to squeeze more channels of information into the same amount of radio spectrum. This is possible due to the combination of efficient data compression techniques, and spectrally efficient multi-level modulation formats such as M-QAM, and M-PSK. Digital transmission provides additional advantages such as the ability to introduce new, value added services such as text messaging and mobile e-mail. Digital systems can potentially provide improved performance in a fading channel environment through the adoption of specialised receiver structures, signal processing and channel coding algorithms, which would not otherwise be possible with an analog system. Improved battery life is also an important factor, the value of which cannot be overestimated when it comes to portable, handheld devices. Extended battery life is usually achieved through a number of measures, including lower RF transmitter power, lower voltage digital electronics, and more efficient power down and standby states.

A particular case in point demonstrating the increasing pervasiveness of digital radio systems is the rise of so called Personal Communications Services or PCS systems. The FCC has allocated 3MHz of spectrum in the United States in the 900MHz band for narrowband PCS, and 140MHz of spectrum in the 2GHz band for broad-band PCS.

Quoting from the FCC web site (http://wireless.fcc.gov/pcs/),

> "Narrowband Personal Communications Services (PCS) is broadly defined by the Federal Communications Commission as a family of mobile or portable radio services that may be used to provide wireless telephony, data, advanced paging, and other services to individuals and businesses, and which may be integrated with a variety of competing networks. For example, narrowband PCS could be used for the development of advanced paging systems. For example, pagers may become equipped with a small keyboard allowing the subscriber to both retrieve and send complete messages through microwave signals (e.g. wireless E-mail)."

Broadband PCS is targeted at providing more data intensive services such as multimedia, and others yet to be conceived.

Narrowband PCS is distinguished from digital cellular by smaller cell sizes, lower radiated power (10-20mW, compared with average of 1W for cell phone handsets), [1], longer battery life, small cell sites that may be installed on power poles and suburban buildings, utilisation of simple high quality speech coding algorithms requiring less signal processing and hence less power use. The first commercially available PCS system became operational in Washington DC and Baltimore in the United States in November 1995.

PCS has been used as just a single, though important, example of the growing significance of mobile digital communications technology in the 21st century. An important topic when it comes to digital receivers is that of carrier and timing synchronisation. Only coherent receivers require that carrier phase synchronisation be performed, however every digital receiver must incorporate some form of timing recovery circuit or algorithm to allow the receiver to accurately determine the optimum time at which to sample the received data symbols, so that a decision can be made regarding the most likely symbol transmitted, with minimum probability of error.

In a mobile wireless environment this task becomes more challenging than for the more benign additive white Gaussian channel. This is due to the multi-path nature of radio propagation, combined with relative movement between transmitter and receiver, and of objects within the environment, creating a time varying channel affecting both the magnitude and phase of the signal of interest in a random time varying manner.

In this thesis three symbol synchronisation algorithms, suitable for all-digital implementation, will be simulated in software, and their performances in a slow, frequency flat, Rayleigh fading channel, will be compared. The effect of different fade rates and system pulse shape excess bandwidths on the relative performance of each algorithm will be investigated.

In addition to looking at the performance of each of the synchronisers as a stand-alone system, the effect of each of the synchronisers on the symbol error rate of a simulated pilot symbol assisted receiver, will be examined. This will help to place the synchronisers' performance in the context of a more realistic system, where factors external to the performance of the timing sub-system may have a larger impact on the overall receiver performance. This may help to define which of the synchronisers would be more suitable in a given situation, where in reality there is always a complexity versus performance trade-off.

1.2 Scope of Thesis

The scope of the work presented in this thesis is primarily of a practical and implementational nature rather than a highly theoretical treatise. The primary goal of this work is to investigate the problem of symbol synchronisation in all-digital receivers operating in a wireless mobile channel environment, primarily through the use of computer simulation. The secondary goal is to implement the simulated synchronisation algorithms in real time on a modern digital signal processor (DSP) platform, using modern DSP software development tools.

The primary goal was achieved through comparison of the performance of three symbol synchronisation algorithms operating under fading channel conditions. The performance of these algorithms was examined through the construction of appropriate computer based simulations. The scope of the simulations was expanded beyond the examination of the timing sub-system in isolation, by incorporating it into a specific receiver structure. The structure chosen for this work has been studied quite extensively in the literature [2-8]

and accommodates the fading channel environment through the insertion of known pilot symbols into the data stream. These are used at the receiver to estimate channel state, thereby providing information for the fading compensation of received data symbols. Using this structure it was possible to examine the impact of using each of the three synchronisers under investigation, on the simulated receiver's bit error rate performance.

1.3 Organisation of Thesis

A brief description of the organisation of the thesis is presented here.

The next chapter will present a brief introduction to some fundamental background wireless digital communication theory, chosen for its relevance to the material presented later in the thesis. This will include a system level description of a typical digital communications system, presentation of some fundamental digital modulation methods, and an explanation of the Rayleigh fading channel phenomenon from a mobile wireless perspective.

Because the topic of synchronisation is the primary focus of the work in this thesis, some background on this large and specialised topic is provided in Chapter 3. It includes an overview of some traditional methods of carrier and timing synchronisation, often based on analog electronic circuit realisations, and carries on to introduce timing synchronisation structures, sampling schemes, and timing error detectors more suited to contemporary implementations of all-digital receivers.

In Chapter 4 the three timing error detector (TED) algorithms investigated in this work are presented. The statistical characteristics of these detectors, under ideal and simulated fading channel conditions are examined and compared. These characteristics have an important effect on the performance of closed loop synchronisers that incorporate the TED's, and the performance of these synchronisers during timing acquisition and tracking is investigated and compared later in the chapter.

Of more interest perhaps is the actual difference each synchroniser makes to the overall bit error rate (BER) performance of the receiver. There is little point in incorporating the best performing but most complex of the synchronisers in a receiver design if other factors in the receiver design are the limiting factor in ultimate BER performance. Chapter 5 extends the computer simulations to include the simulation of a pilot symbol assisted receiver structure into which each of the synchronisers under investigation can be incorporated to provide a comparison of the effect of the synchroniser on the overall receiver BER performance.

Appendix A presents the more practical aspects of the work, including the 5 step design process used to proceed from MATLAB simulation to functioning synchronisation software on a modern DSP platform. The design and software structure of the simulation, and the real-time DSP implementation is presented, as is the hardware setup used to interface the I and Q baseband digital samples from a real digital receiver, to the DSP development board used in this work. Constellation plots are presented that demonstrate the functioning of the synchronisation software sub-system on a received QPSK signal.

The final chapter presents a summary of the work that has been reported, draws relevant conclusions and makes suggestions for future work.

CHAPTER 2
SOME BASIC DIGITAL COMMUNICATION CONCEPTS

This chapter presents some fundamental digital communications principles, that will prove to be useful in later chapters. To begin with a model of a typical digital communications system is presented and explained. The primary advantage of coherent over non-coherent communications is briefly introduced as motivation for the development of coherent and quasi-coherent fading channel communication system design, an example of which is examined in Chapter 4. Following this is a brief introduction to the mathematical concept of representing bandpass communication systems as equivalent complex baseband systems. This makes the computer simulation of such systems much more practical. An introduction to three of the basic digital modulation methods is then followed by a summary of the mobile radio fading channel concepts that underlie the channel model used in this work. Finally, a typical digital receiver structure, of the type used in this work, is introduced.

2.1 A Typical Digital Communication System

No communications thesis would be complete without introducing an overall block diagram model of a digital communication system model. The model for such a system can be broadly categorised into three fundamental parts. These are, the transmitter, the channel, and the receiver, illustrated in Figure 2.1. The particular diagram printed below presents the basic elements of a coded digital communications system [9, 10].

The purpose of a digital communications system is the same as that of an analog communications system, namely, the transfer of information between two points in space, or alternatively, two points in time. In each case this goal

is met in fundamentally different ways. The information source for an analog communication system is in the form of a continuous time varying electrical waveform, derived from sources such as audio, video etc. This is then used to modulate one or more of the three primary characteristics of an RF carrier, namely, frequency, phase and amplitude.

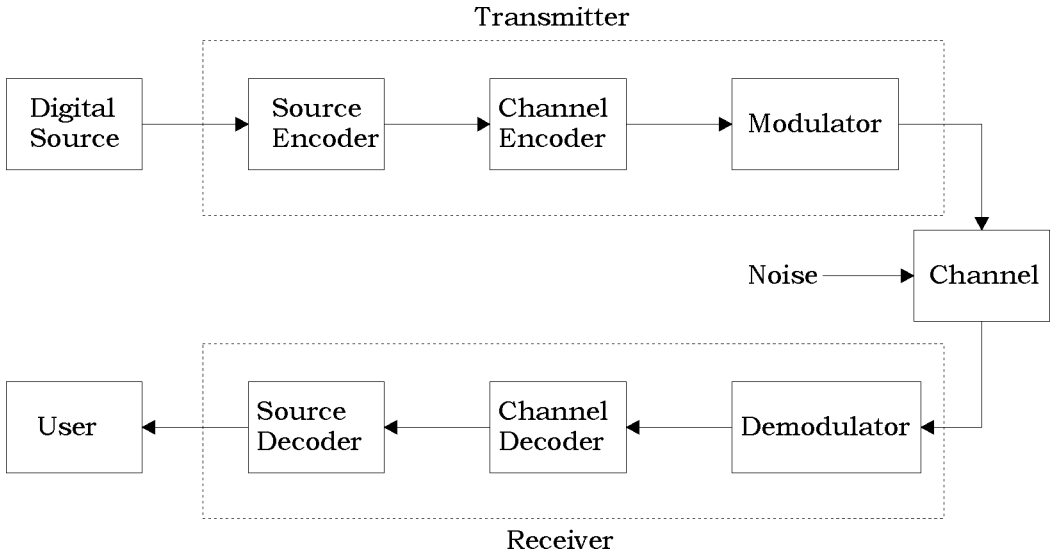

Figure 2.1 Basic components of a coded digital communications system.

In contrast the information source for a digital system is in the form of binary data. This may have been derived from audio, video, computer systems, or a wide variety of other sources. To begin with, the binary *source* data is *encoded* or equivalently, compressed, to eliminate as much redundant information as possible. The encoded binary data stream is then passed on to the *channel encoder*, which has two main functions. The first is to reintroduce some carefully controlled redundancy into the data stream for the purpose of protecting against channel induced errors. This is also known as error control coding. The second function is to segment the encoded data stream into groups of 1, or 2, up to 2^M bits. The particular size used in the design of a system will depend on what the system is to be used for, the type of channel environment it is to operate in, the type of data it is to carry, and many other factors, which are beyond the scope of this thesis. Each of these data segments is called a message or symbol. Finally, on the transmission side, it is the job of the *digital modulator* to map the digital symbols into analog

waveforms that are appropriate for transmission through the particular waveform channel in question.

The communications *channel* is the physical/electrical medium through which the signal passes in travelling from the transmitter to the receiver. In practical terms this may be a wire, a terrestrial radio channel, a fibre optic cable, or a satellite radio channel, to name just a few. Each of these channels has different characteristics and will affect a digitally modulated signal passing through it in different ways. Two of the major sources of signal corruption introduced by non ideal channels are noise and amplitude / phase distortion due to fading processes. The fading process is most usually associated with radio channels, and mobile radio channels in particular. Both of these channel types are discussed in more detail in the next section.

It is the role of the receiver to process the channel corrupted waveform and, given knowledge of the channel and source encoding methods used by the transmitter, to reverse the transmitters encoding and modulating procedures to extract an estimate of the originally transmitted data sequence. One measure of how well the receiver is doing at reconstructing the transmitted data is the error frequency in the reconstructed data sequence, or stated another way, the probability of an error occurring. The probability of error at the receiver depends on many different factors, including:

- signal to noise ratio
- modulation order and type
- channel type
- type of error control coding used

2.2 The Advantages of Coherent Communication

Any given digital receiver can be classified as either coherent or non-coherent in nature. These terms refer to whether the receiver's local phase reference is synchronised to the phase of the received signal. If this is the case then signal detection is termed 'coherent', and the receiver is called a coherent receiver. If

there is no phase synchronism between the receivers phase reference and the received signal then the type of detection is termed 'non-coherent' and the receiver is referred to as a non-coherent receiver. Some modulation types require coherent reception, while others can be demodulated in a non-coherent manner too. An example of a modulation type that can be demodulated either coherently or non-coherently is binary FSK. Coherent receivers for a given modulation type exhibit greater power efficiency (up to 3dB) over their non-coherent counterparts [9]. In this case power efficiency refers to the coherent receiver's ability to achieve the same data error rate as its non-coherent version, but at a lower received SNR. This means that for a given desired receiver error rate performance, a coherent communications system can utilise a lower transmitter power than the 'equivalent' non-coherent system. This is a primary motivation for developing coherent receivers for fading channel environments. To date, most wireless mobile digital schemes use non-coherent, differential detection because the problem of tracking phase in a fading channel is non trivial.

2.3 Complex Baseband Representation of BandPass Signals

All wireless digital communications systems are bandpass systems. This means that the baseband information signal has been modulated onto a high frequency electromagnetic carrier wave so that it may be transmitted through free space. Normally the bandwidth of the information signal is very small compared with the frequency, f_c, of the radio carrier wave. This presents a serious problem when it comes to simulating such systems by computer because in order to adequately represent the carrier component of the signal it would be necessary to use several samples per cycle of the carrier. In turn, this would mean that a huge number of samples per modulated symbol would be used in the simulation, making simulation times impractically long.

Fortunately it turns out that to usefully simulate all the relevant behaviours and characteristics of a bandpass digital communications system it is not actually necessary to explicitly simulate the carrier component of the signal. Such a system can be reduced to a complex low pass equivalent form without

any loss of generality. This dramatically reduces the amount of processing required to simulate any given bandpass system.

The concept of a low pass equivalent of a bandpass signal is introduced briefly below, and has been paraphrased from [11]. Further detail can be found in references [9, 11].

Any carrier modulated signal can be written in the form

$$s(t) = A(t) \cdot \cos(2\pi f_c t + \phi(t)) \tag{2.1}$$

This can be rewritten in the intermediate form shown below. Application of Euler's identity to the expression in the top line of Eq 2.2 produces the complex exponential form of the modulated bandpass signal, expressed on the bottom line of Eq 2.2.

$$\begin{aligned} s(t) &= \mathrm{Re}\left\{A(t) \cdot \left[\cos(2\pi f_c t + \phi(t)) + j \cdot \sin(2\pi f_c t + \phi(t))\right]\right\} \\ &= \mathrm{Re}\left\{A(t) \cdot e^{j\phi(t)} \cdot e^{j2\pi f_c t}\right\} \end{aligned} \tag{2.2}$$

In this expression *A(t)* is the amplitude modulation, $\phi(t)$ is the phase modulation of the signal and f_c is the carrier frequency. The signal represented by

$$A(t) \cdot e^{j\phi(t)} \tag{2.3}$$

contains all of the data modulation related information. This is known as the complex envelope of the signal and it is low pass in nature. Generally the complex envelope can also be written in the cartesian form

$$\begin{aligned} \tilde{s}(t) &= A(t) \cdot e^{j\phi(t)} \\ &= s_I(t) + j \cdot s_Q(t) \end{aligned} \tag{2.4}$$

where $s_I(t)$ and $s_Q(t)$ are referred to the as the 'in phase' and 'quadrature' components of the complex envelope. These are both real valued and low pass in nature.

Using Euler's identity again, and going almost full circle in the mathematical reasoning, the original bandpass signal s(t), can be expressed in canonical form as,

$$s(t) = s_I(t) \cdot \cos(2\pi f_c t) - s_Q(t) \cdot \sin(2\pi f_c t) \tag{2.5}$$

Complex baseband representation of bandpass communication systems is used by almost all commercial RF simulation software and the simulations constructed for the purpose of this thesis are no exception.

2.4 Fundamental Digital Modulation Methods

To transmit digital information from one place to another it is necessary to change it into a form that is compatible with the selected transmission medium. In digital communication systems the unit of information is binary data, typically referred to as 0's and 1's, or alternatively some M-ary encoded version of groupings of 0's and 1's. If the communications system is *baseband* in nature, for example optical data communications, or Ethernet, the information is transmitted as pulses of some form. In the case of *passband* modulation the information is impressed upon a sinusoidal radio carrier by systematically varying one or more of the three fundamental characteristics of the carrier in a way that is proportional to the original information signal. The three characteristics that may be varied are the amplitude, the frequency and the phase of the carrier. Through this process the original digital symbol is transformed into a sinusoidal waveform of duration T_s, equal to the symbol period.

Most forms of digital modulation are combinations of one or more of the three fundamental signalling schemes, *amplitude shift keying* (ASK), *phase shift keying* (PSK), and *frequency shift keying* (FSK), illustrated in Figure 2.2. In this example the data source is binary.

The remainder of this section introduces the modulation type commonly known as Quadrature Amplitude Modulation (QAM), a hybrid modulation that encodes digital information into both the amplitude and phase of the radio

carrier. This is the modulation method used in all subsequent work presented in this thesis.

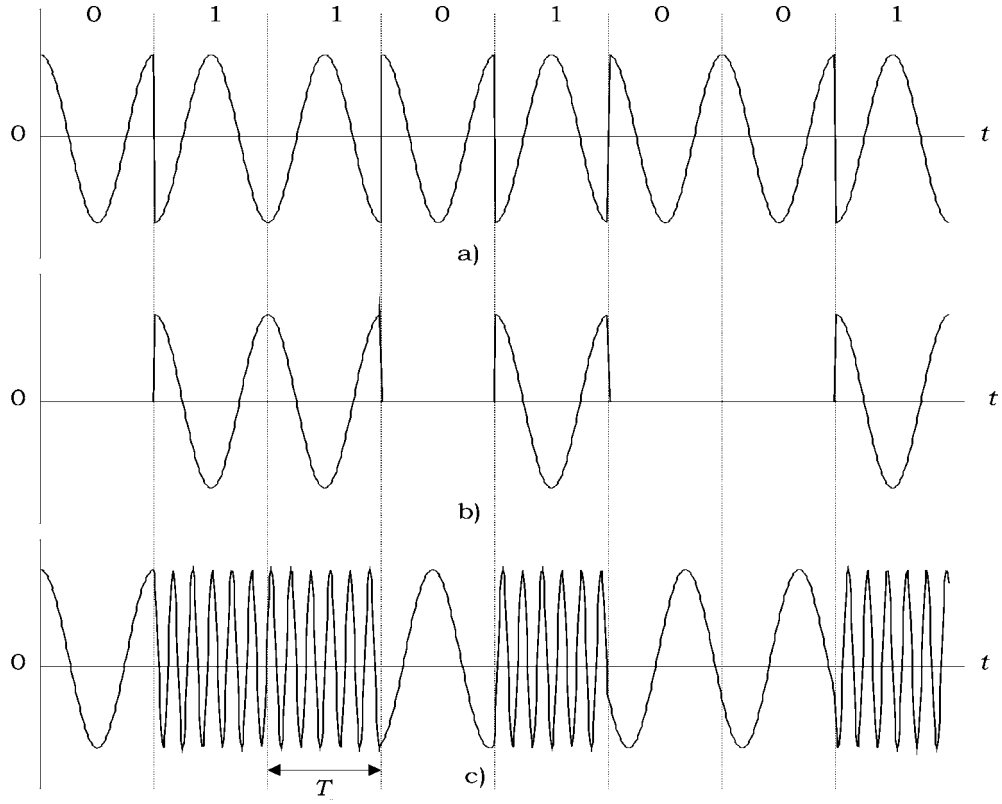

Figure 2.2 Modulated carrier for modulation types a) PSK, b) ASK, and c) FSK [9]

QAM is a popular digital modulation format for current and future proposed high speed digital communications services due to its spectral efficiency. In addition its independent quadrature I and Q components make for relatively simple implementation, and slicer implementation in the receiver is straightforward due to simple rectangular decision region boundaries [12]. A basic QAM modulator is illustrated in Figure 2.3. Note that the independent I and Q symbol sequences are applied to quadrature carrier components. These components are orthogonal to each other. This allows them to be independently modulated and then transmitted in the same spectral band. Due to this orthogonality the two symbol streams can be separated by coherent demodulation at the receiver.

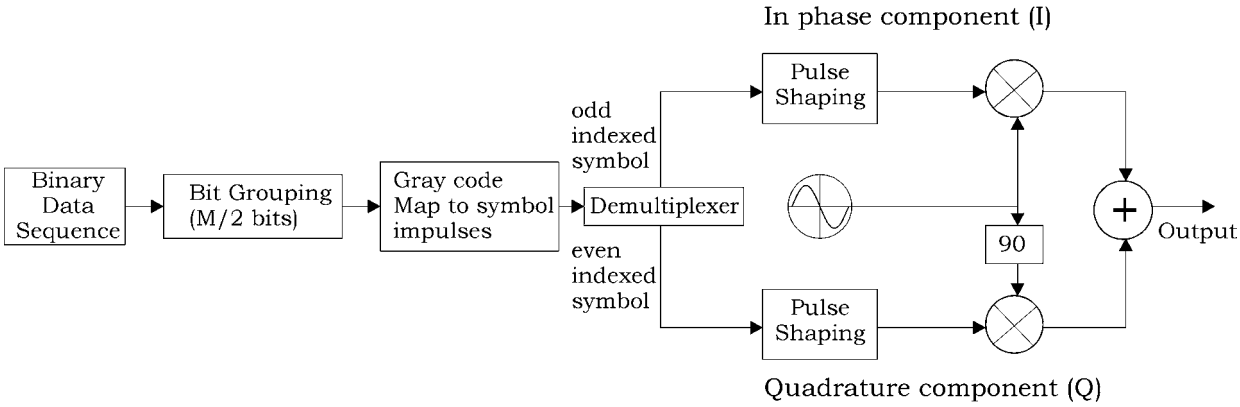

Figure 2.3 Standard bandpass QAM modulator

The name QAM can be applied to any modulation that encodes the source digital symbols into discrete radio carrier amplitude and phase states. There are many different forms of QAM, each with particular advantages and disadvantages that may make one more suitable to a particular application than another.

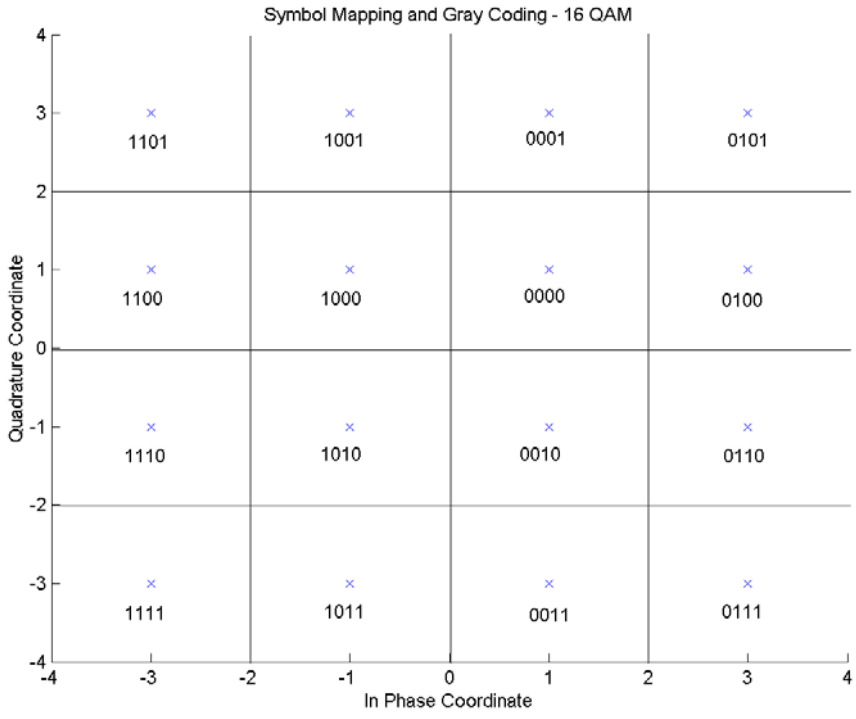

Figure 2.4 Constellation map for Gray coded 16 QAM.

If the magnitude of the independent I and Q baseband samples, obtained by sampling at the optimum sampling instant, were to be plotted onto a rectangular co-ordinate system graph, with the I 'channel' on the x axis and

the Q 'channel' on the y axis, the result would be a constellation diagram. Constellation diagrams are extremely useful tools for visualising various properties of the signal such as the minimum Euclidean distance between constellation points, and peak to average power ratio. The square constellation is one of the most popular QAM constellations, and is shown for 16-QAM in Figure 2.4.

2.5 The Mobile Fading Channel

Because one of the main themes of this thesis is the performance of different timing synchronisation algorithms in the presence of signal degradation due to a fading channel, it is appropriate at this point to provide a brief introduction to the topic of fading channels as they apply to terrestrial mobile digital communications. A great deal of attention has been devoted to this subject in the literature over the past several decades, however an excellent tutorial article can be found in references [13, 14], and a number of texts also provide good introductory treatments of this topic, a few of which have been listed in the references [10, 15, 16].

2.5.1 Large and Small Scale Fading

Fading can be separated into large scale and small scale phenomena and represents the most serious impediment to the performance of communications systems designed for use with the wireless atmospheric channel.

The shadowing, and attenuation of the average power of a signal at a receiver, caused by the movement of the receiver over significant distances (large compared to the wavelength of interest) within an environment composed of obstructing, and attenuating objects such as hills, forests, and buildings is termed large scale fading. Included within the mechanisms of large scale fading is the power reduction at the receiver due to its distance from the transmitter. This follows an inverse power law relationship, $r^{-\alpha}$, where α can take on values in the range 2 to 4. As the receiver gets further away, or closer

to the transmitter, it will experience a decrease or increase respectively, in the average signal power level. The signal attenuation attributable to the large scale fading phenomenon is illustrated by the dotted line in Figure 2.5a.

Superimposed upon the large scale fading is small scale fading, which is the fading mechanism of primary concern in this work. Small scale fading is the rapid fluctuations in signal attenuation that are observable over small changes in position by the mobile receiver. In this case the small positional changes mentioned are on the order of wavelengths of the received signal frequency. The small scale fading phenomenon is illustrated more clearly by separating it from the large scale fading, *m(t).* This is illustrated in Figure 2.5b.

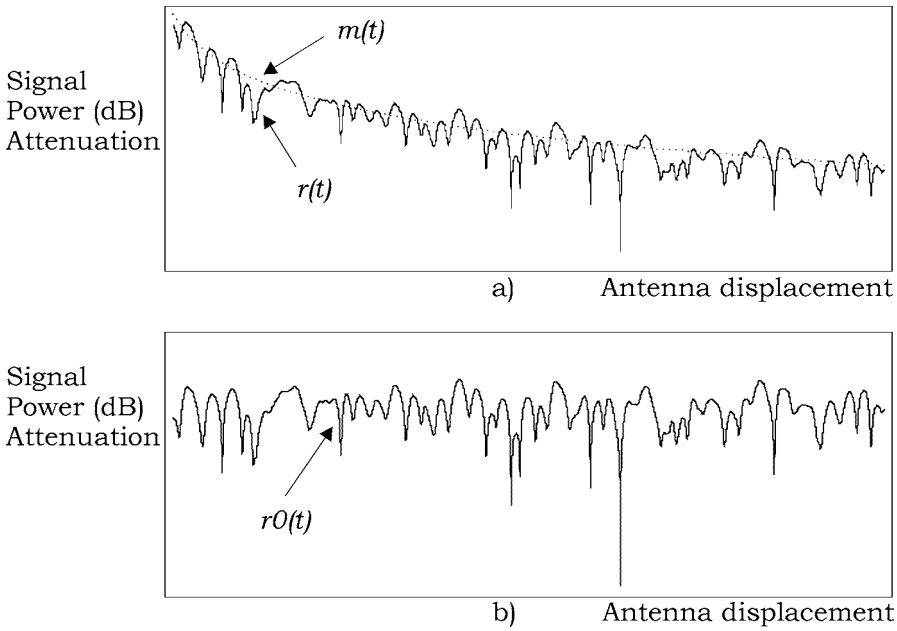

Figure 2.5 Illustrates the large and small scale composition of the signal fading experienced by a mobile receiver. a) shows the combined fading, and b) shows the small scale Rayleigh fading only[13].

As was mentioned earlier, small scale fading is the phenomenon of primary concern to the work contained in this thesis. For this reason the subject of large scale fading shall not be dealt with further here, but the next few subsections shall explain the mechanisms of small scale fading in a little more detail.

2.5.2 Small Scale Fading Mechanisms

Small scale fading is due to multipath radio wave propagation, and relative motion between the receiver and the transmitter. In an urban environment in particular, the received signal is often made up of multiple reflected rays, and perhaps even a line of sight component. The transmitted radio signal tends to encounter many objects within the environment, such as buildings, cars, tree foliage, and signs that reflect the original signal such that the receiver will typically see a signal that is the summation of many versions of the original signal, each with a different power, and each having been delayed by a different amount of time, τ. The real problem arises because each of these signals will arrive with a different carrier phase at the receiver. When summed together the resultant signals interfere constructively or destructively, causing significant variations in received signal power as a function of antenna position. This idea is illustrated in Figure 2.6. If all of the reflectors and scatters in the environment giving rise to each signal path are absolutely static then the power surface illustrated will also be static in nature. However in a realistic environment while some reflectors will be static, many will not. Examples include foliage rustling in the wind, and movement of vehicles. This will introduce some time variation of the power surface.

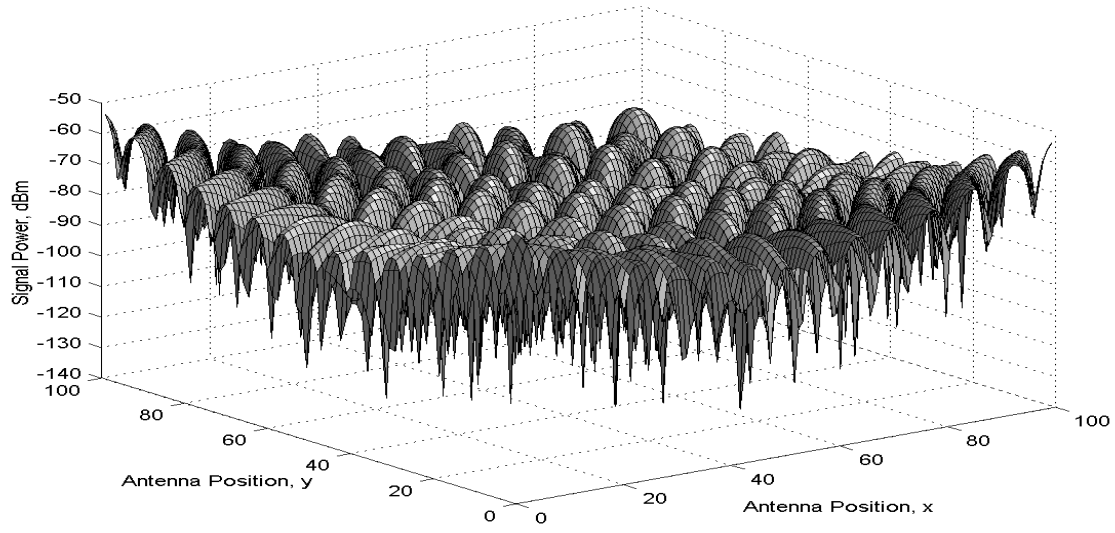

Figure 2.6 Geographically distributed power surface resulting from summation of 4 reflected signals. Deep power nulls occur at positions where all reflected rays add destructively.

When it comes to the relative delay between different reflected signal paths, two cases can be identified. The first is where the maximum time delay (the delay domain τ) between the arrival of the first path signal symbol and the arrival of the corresponding symbol on the last signal path carrying any appreciable power, exceeds or is on the order of a symbol period, T_s. This is known as the maximum excess delay, denoted T_m. The second case is where all delayed versions of the signal arrive within a space of time that is relatively short in comparison to the symbol period, that is $T_m << T_s$. These two situations give rise to the frequency selective fading channel, and the frequency non-selective or flat fading channel respectively, each imposing a different set of requirements on the receiver.

2.5.3 Statistical Characterisation of Small Scale Fading

As mentioned previously, the signal fading at the receiver is caused by the superposition of a large number of components with randomly fluctuating amplitudes and phases (due to doppler, and moving scatterers and reflectors). These are generally assumed to be independent of each other. Next, because the number of randomly fluctuating components is large, the central limit theorem is utilised to claim that the distribution of the resulting sum is a complex Gaussian random variable. This means that the real and imaginary components of the lowpass complex signal envelope are independent and have a mean of zero and a variance of σ^2, and the magnitude, r, of the signal envelope seen by the receiver has a Rayleigh probability distribution [10]. This can be expressed as

$$p(r) = \begin{cases} \dfrac{r}{\sigma^2} \cdot \exp\left[-\dfrac{r^2}{2\sigma^2} \right], & \text{for } r \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.6}$$

2.5.4 Manifestations of Small Scale Fading

As far as the receiver is concerned there are two main consequences to small scale fading. These are:

- Time spread (in τ, delay domain) of digital pulses in signal due to different time of flight of each reflected signal component.
- Time varying behaviour (in t) of the channel due to the receiver's relative motion with respect to the transmitter, and movement of scatterers and reflectors within the environment.

2.5.4.1 Time Spreading

In the first case, even without movement of the receiver, the received digital pulses may undergo distortion, or smearing in time due to the delayed arrival of multiple copies of each transmitted symbol. A term used to describe this property of a multipath channel is *multipath spread*. The concept is illustrated by Figure 2.7 for the case of a discrete multipath channel, where a very narrow pulse (wideband) is injected into the communications system and the signal monitored at the receiver. Each impulse shown corresponds to a separable multipath component, with a particular associated excess delay, τ, where in this context, the excess delay of the first multipath component to arrive is defined to be zero. This kind of diagram is called the *power delay spread*, or *the multipath intensity profile*.

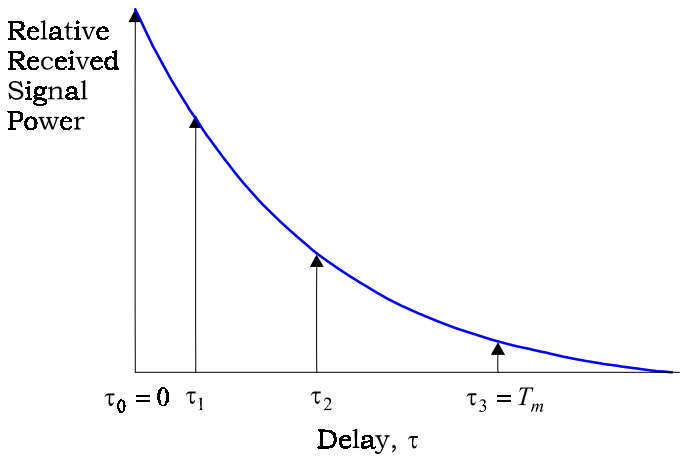

Figure 2.7 Power delay profile for a discrete multipath channel

A further delay spread measure is the RMS delay spread of the power delay profile. This is defined as the standard deviation of the delay of reflections (multipath components) weighted in proportion to the power in each component. In reality however, each distinguishable multipath component, τ_k, may actually be comprised of many unresolvable components whose excess delays all fall within $\pm d\tau$ of τ_k, where $d\tau$ is small compared with a symbol interval. This is illustrated with Figure 2.8 which shows two shaded scattering ellipses. The finite width of each ellipse represents all the multipath components that arrive within an unresolvably small band of excess delays.

These would only be observable as a single multipath component at the receiver.

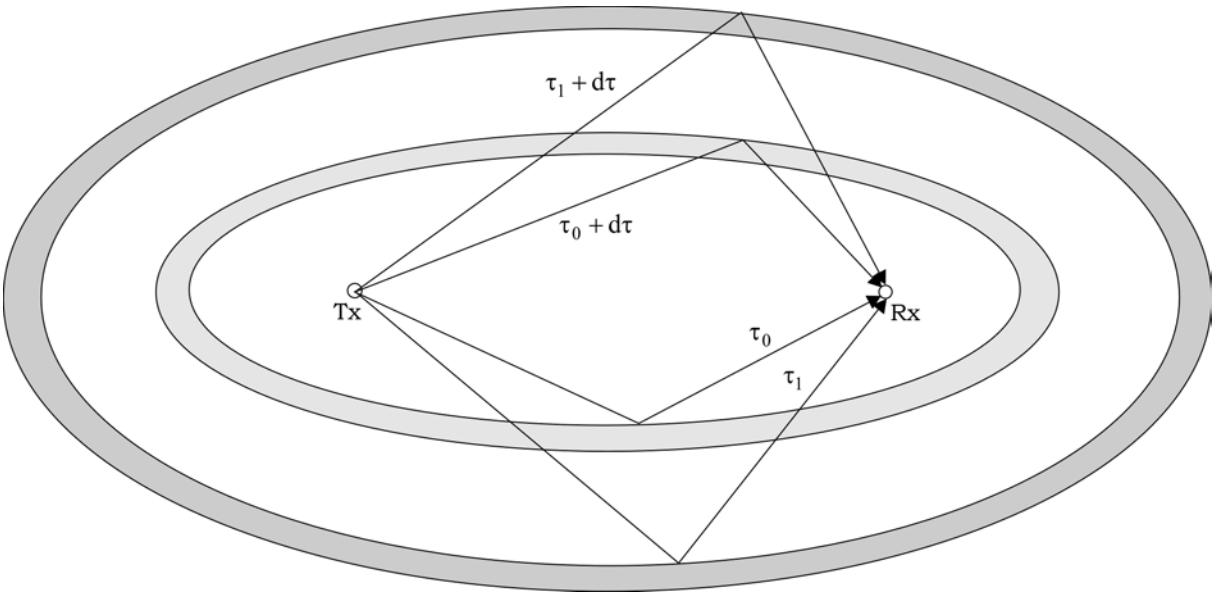

Figure 2.8 Scattering ellipses illustrate the multipath nature of the signal at the receiver and the fact that each resolvable multipath component is likely to be made up of many similarly delayed components. Adapted from [17]

If the delays between arrivals of each symbol copy is significant with respect to the symbol period then interference with subsequently transmitted symbols will result. This degradation is known as channel induced inter-symbol interference (ISI) and is a feature of the frequency selective fading channel. Receivers designed to operate in this kind of channel require an equaliser of some form to correct as much as possible for this ISI. On the other hand if the delay of the last symbol to arrive, carrying any significant power, is

significantly less than the symbol period, $T_m << T_s$, then the individual multipath components are not resolvable (flat fading). This means that there is essentially no distortion due to channel induced ISI, however the constructive and destructive interference of the components as described earlier, and as illustrated in Figure 2.6, acts to dramatically affect the SNR at the receiver, depending on its position.

2.5.4.2 Time Variation

In the second case, movement of the receiver through the power surface illustrated in Figure 2.6 introduces a time variant channel from the perspective of the receiver. This is in addition to any time variations of signal power at a particular geographic location due to the movement of scatterers and reflectors within the environment.

There is another effect associated with the movement of the receiver with respect to each of the multipath component sources. So far we have not taken account of the spectral spreading of signals due to the doppler effect. If an observer moves toward (positive) or away (negative) from a radio source of frequency f_c, at a constant velocity v, the measured frequency will change by an amount proportional to the component of velocity in the direction of the source, as shown in Eq 2.7.

$$\Delta f = f_d = \frac{v}{c} \cdot f_c \tag{2.7}$$

The change in frequency is called the doppler frequency, f_d, and c is the speed of light. This idea is illustrated by Figure 2.9.

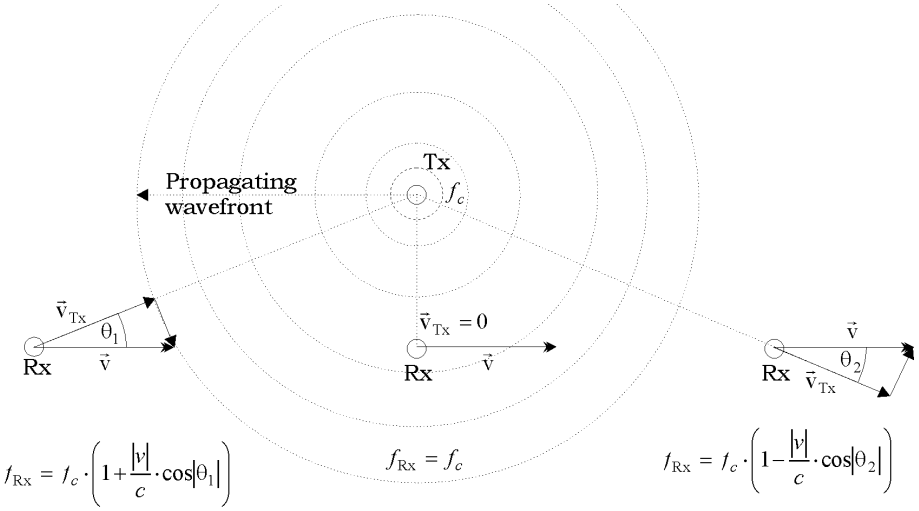

Figure 2.9 Doppler vector diagram illustrating how the received measured carrier frequency changes as the receiver position changes.

Now, consider a receiver moving at a constant velocity, v, and receiving many reflected signals from a single transmitter, which is generating a single unmodulated carrier frequency f_c. Each of the multipath signals at the receiver will have undergone a different doppler shift due to the different angles of incidence. The absolute maximum frequency that may be seen at the receiver is f_c+f_d and the absolute minimum frequency that may be observed at the receiver will be f_c-f_d. Assuming that there are a large number of arriving signal components, which are radially uniformly distributed, with equal magnitude reflection coefficients, and randomly occurring carrier phases, (known as the dense scatterer channel model [18, 19]) the receiver will see a continuum of frequencies between these two extremes. The resulting spectrum characterises the time variation of the channel and is commonly known as the Jake's power spectrum, and is illustrated by Figure 2.10.

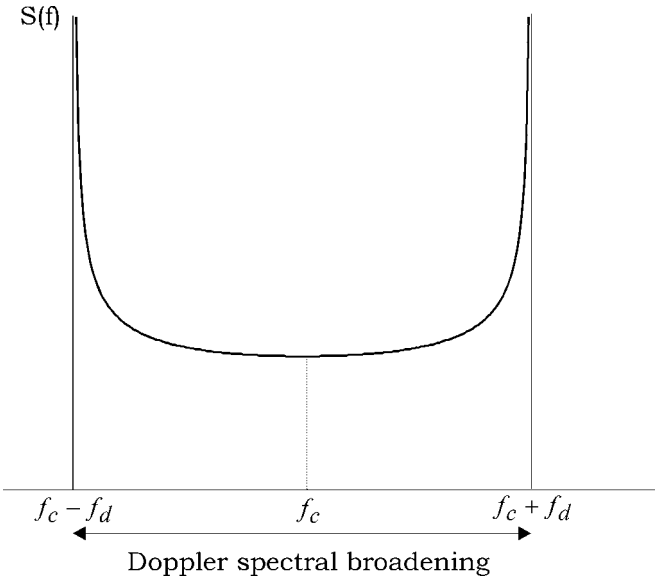

Figure 2.10 Jake's power spectrum at the receiver of a single frequency sinusoid transmitted through a dense scatterer doppler channel model.

This doppler spreading model implies that from a frequency domain point of view, the transmitted digital signal will undergo spectral spreading by an amount f_d that is dependent on the velocity of the receiver. The amount of spreading is also termed the *doppler spread*, and the *fading bandwidth*. This is the frequency domain dual of the time domain pulse spreading due to multipath delay spread.

A final classification can be made regarding time variant channels. The channel is usually regarded as *fast fading* if the *channel coherence time,* T_0, is less than a symbol interval. The channel coherence time is the time duration for which the channel remains correlated. This can be derived approximately as the time taken by a receiver moving at velocity v, to traverse two nulls in the power surface depicted in Figure 2.6, which is a distance of approximately half a wavelength of the carrier.

$$T_0 \approx \frac{\lambda/2}{v} = \frac{0.5}{f_d} \tag{2.8}$$

The channel is *slow fading* if $T_0 > T_s$. These criteria can be restated in terms of frequency domain quantities. A channel is fast fading if the signal bandwidth, B, is less than the fade rate, f_d, and slow fading if the signal bandwidth is greater than the fade rate.

2.5.4.3 The Slow, Flat Fading Rayleigh Channel

The previous few sections have introduced the fading channel and the mechanisms that produce it. They have also categorised the fading channel as frequency selective or flat, and as fast or slow fading. The category of channel that is considered exclusively by the work presented in this thesis in later chapters is the slow, flat Rayleigh fading channel. Generally speaking, in most of the current mobile digital communications systems, fading is almost always slow because doppler spreads tend to be of the order of a few hundred hertz at most whereas signaling rates are on the order of tens to hundreds of kilohertz. The frequency flat assumption is applicable to any environment where the RMS delay spread of the channel is less than approximately a tenth of a symbol period. This is applicable to many rural environment channel models, or in indoor or micro cellular channels. This assumption can also hold true for many dense urban macro cellular channels, particularly for the data rates common today, and projected for the near future [15].

2.6 Simple Mathematical and Simulation Channel Models

In the design and analysis of digital communications systems it is common practice to utilise simplified mathematical models that embody /mimic /duplicate the most important characteristics of the physical channel through which the information is to be transmitted. Obviously there is a trade off between the complexity of the model used and the accuracy with which it represents the behaviour of the real channel; however, it is often possible to create a reasonable approximation to the main channel characteristics with a model of modest complexity, and hence reasonable computational efficacy/efficiency. The two channels with which this thesis is primarily concerned are the additive white Gaussian noise (AWGN) and the frequency flat Rayleigh fading channel models. Each of these channel models is presented and explained further in the next two sub-sections.

2.6.1 Additive White Gaussian Noise

This is one of the simplest and most benign of the non ideal channels encountered in practice. Consequently it is also easy to model. In a digital communications system AWGN is typically attributed to sources such as thermal noise in front end receiver electronics. This is caused by the thermally induced random motion of charged particles in the receivers electronic components. These very small random currents combine in an additive fashion with the received signal induced currents to give a noise corrupted signal. Thermal noise in electronic components is modeled as a zero mean, stationary Gaussian random process with a power spectral density that is constant across an extremely wide range of frequencies [20]. This is the origin of the term white. It is an analogy to the case of white light which contains equal amounts of all light frequencies across the visible electromagnetic spectrum.

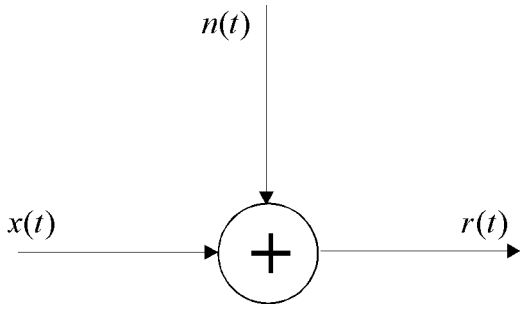

Figure 2.11 Mathematical model of AWGN channel.

In the model illustrated by Figure 2.11 the transmitted signal *x(t)* is corrupted by additive white Gaussian noise, *n(t)*, to produce the received signal *r(t)*, or in mathematical notation

$$r(t) = x(t) + n(t) \tag{2.9}$$

2.6.2 Frequency Flat Rayleigh Fading

The Rayleigh fading channel model is of particular relevance to mobile radio communications [21]. Mathematically the received complex lowpass signal can be represented as the product of the transmitted signal $\tilde{s}(t)$, and a complex

channel tap gain $\tilde{c}(t)$, summed with additive white Gaussian noise, $\tilde{n}(t)$. This is shown algebraically by Eq 2.10, and diagrammatically by Figure 2.12,

$$\tilde{r}(t) = \tilde{c}(t) \cdot \tilde{s}(t) + \tilde{n}(t) \tag{2.10}$$

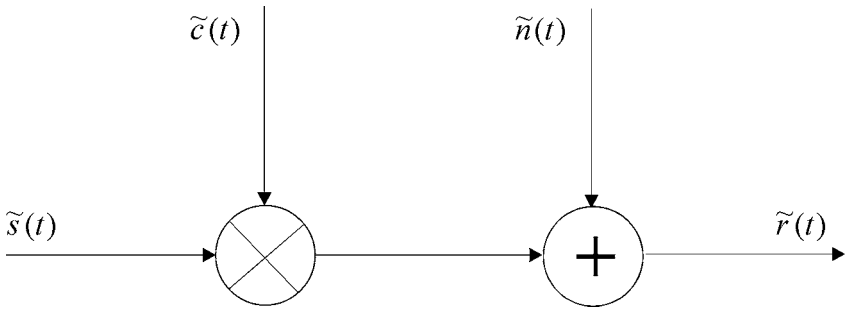

Figure 2.12 Mathematical model used to implement the slow, flat, Rayleigh fading channel.

The generation of the Rayleigh distributed complex fading channel in a computer simulation can be accomplished using one of several different methods. It has been widely recognised [22] that the particular shape of the doppler spectrum (U shaped in Figure 2.10) has little impact on system performance. Of much greater importance is the doppler spread parameter, f_d. For this reason, it is relatively common practice to simulate the fading distortion using the following method, illustrated in Figure 2.13. Two, independent Gaussian distributed random number streams are generated. These are then filtered using a lowpass filter with a sharp rolloff. In this case a 3rd order Butterworth filter with a –3dB bandwidth equal to the desired fade rate of the channel, f_d is used. The filtering acts to partially correlate the random processes, the filter cutoff frequency dictating the amount of correlation introduced. As expected, a channel experiencing a greater fade rate will have a lower correlation than a slower fading channel. The two filtered random Gaussian processes are then combined into a single complex random variable. This is the channel tap weight used to multiply each transmitted symbol sample.

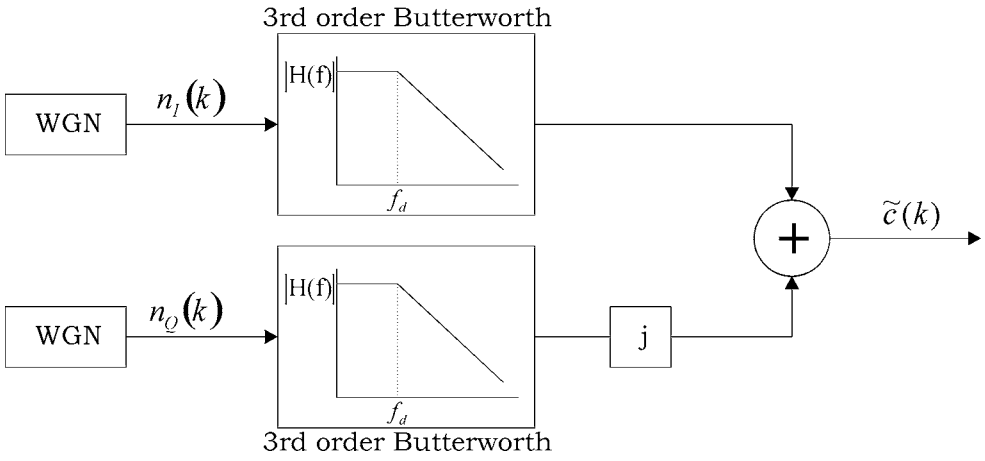

Figure 2.13 Simulation model used to generated fading channel tap weights.

This simulation model will often be normalised such that the average power of the fading envelope is equal to 1, or mathematically,

$$E\left[\left|\tilde{c}(k)\right|^2\right]=1 \tag{2.11}$$

This means that the average power of a signal transmitted into the channel is the same when it 'exits' the channel. In the simulation environment this makes it easier to set signal to noise ratios to desired values. This normalisation is achieved by appropriate scaling of the Butterworth filter coefficients.

2.7 A Typical All Digital Receiver Structure

A simplified block diagram level structure for an all digital receiver is illustrated by Figure 2.14. This is the kind of structure used in the simulations created for the purposes of the work presented in this thesis, and closely matches the structure of the actual digital radio receiver intended to be interfaced to the Texas Instruments TMS320C6701 EVM. The EVM is a PCI card based DSP development board intended to allow rapid prototyping of DSP applications. A photo of the EVM can be found in Figure A.11.

The block labeled "Phase and Time Synchronisation" is the functional component that is the primary focus of the work presented in this thesis.

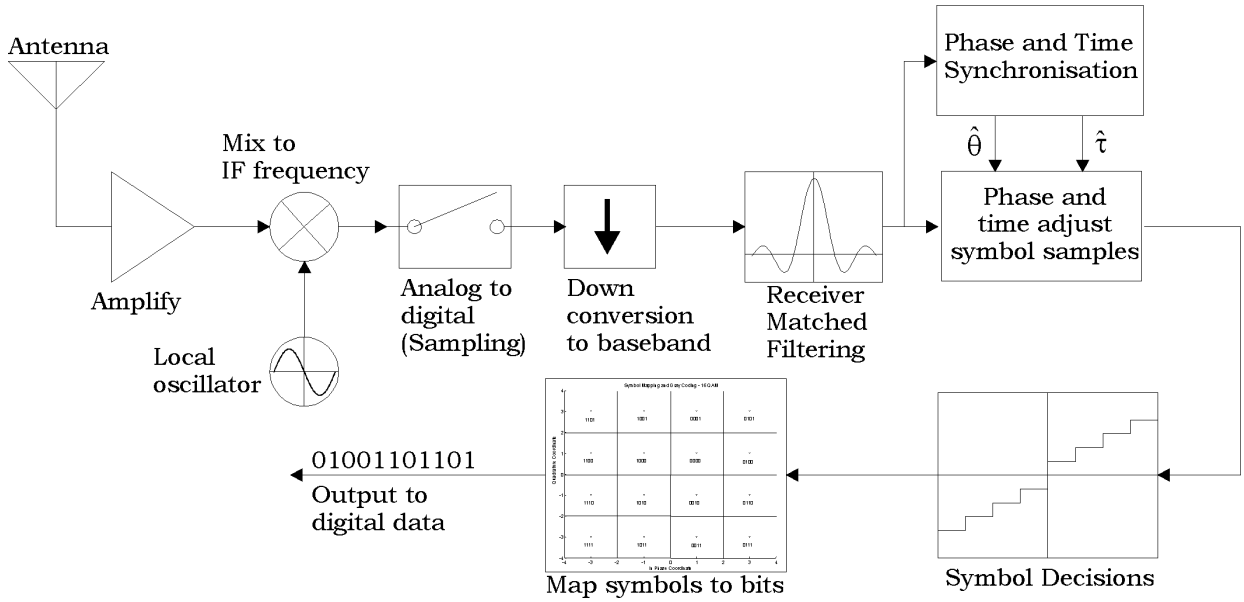

Figure 2.14 Generic high level architectural diagram for an all-digital radio receiver.

The received modulated RF carrier (for example 915MHz) is mixed down by analog methods to an intermediate frequency (IF), a typical value being 70MHz. The IF signal is then sampled. If an IF of 70MHz is being used then bandpass sampling is likely to be used. This allows the 70MHz IF to be sampled at a significantly lower frequency, for example 40MHz. As long as certain restrictions regarding IF and sampling frequency ratios are observed then this is perfectly feasible and does not violate the Nyquist sampling theorem [23]. Next, the sampled IF signal is digitally down converted to baseband and is normally represented in quadrature (I and Q) form. The complex baseband signal can now be matched filtered, timing and phase synchronised, and the resulting symbol samples are used to make decisions on the identity of the most likely symbol to have been transmitted. The final stage is to map the symbols back to a binary bit stream.

CHAPTER 3
AN INTRODUCTION TO SYNCHRONISATION

3.1 Introduction

To synchronise events means to make them happen at the same time. This is a very important requirement in the context of digital communications systems, where several levels of synchronisation must be established before data recovery can take place successfully. Three of these synchronisation levels are *carrier synchronisation, symbol synchronisation,* and *frame synchronisation.*

In the analog implementation of a digital wireless (RF) receiver, carrier synchronisation is the process whereby a local carrier signal is generated that has a phase and frequency very closely matching that of the received carrier, upon which the transmitted information signal is modulated. Synchronisation refers to both the initial acquisition and the continued tracking of the received carrier frequency and phase. In a fully digital receiver implementation it is not required that a local version of the carrier be explicitly generated, however the receiver must still be able to estimate the phase and frequency of the received signal at the sampling instants. Phase synchronisation is only required by receivers using coherent detection, which in general provides better bit error rate versus signal to noise ratio performance than non-coherent receivers [9, 23]. This performance benefit does come at the price of the additional receiver complexity required to perform phase recovery.

Digital communications relies on the ability to represent the original information source as a series of discrete states, or transitions between states. The transmitter further encodes these digital states, or state transitions into analog waveforms that are suitable for transmission through the waveform channel relevant to the particular system in question. To be able to recover this information, the receiver must know when the received signal is going to change state. Because a digital receiver only 'looks' at the received signal at

discrete points in time (sampling) it must know the optimum times at which to sample the signal. The usual definition of 'optimum' in this context are those timing instants that will minimise the receiver's average probability of making a symbol decision error, given that all other factors influencing the receiver's probability of error are held constant. The act of sampling the symbol at the optimum instants in time usually means that the effects of intersymbol interference have been minimised, and the symbol signal to noise ratio has been maximised. The process of determining when these 'optimum' timing instants are, often from the received data waveform only, is called timing or symbol synchronisation.

The third level of synchronisation required by most digital communications systems is frame synchronisation. Usually the unit of information in a system is not a symbol, but rather a grouping of symbols, or a packet of data. The start and end of these packets are usually marked with special data patterns. Acquiring and maintaining the knowledge of where these data packets begin and end is termed frame synchronisation.

This thesis is primarily concerned with the problem of timing synchronisation, especially in the slow, frequency flat, Rayleigh fading channel. Phase recovery for the frequency flat fading channel is briefly investigated only through the study and implementation of a previously proposed method called pilot symbol assisted modulation [2]. This work allowed the author to simulate a QAM receiver incorporating two levels of synchronisation.

The topic of framing synchronisation is outside the scope of the work presented in this thesis.

3.2 Carrier Synchronisation

The performance advantages of coherent receivers over non-coherent receivers makes signal phase estimation in digital receivers an important problem. Much effort has been devoted to the development of new ways of estimating signal phase, and also to the improvement of existing methods. The problem of signal phase estimation is also variously known as *carrier recovery*, *phase*

recovery, and achieving *phase lock*, to name a few. In the AWGN channel the problem is not particularly difficult and is commonly achieved using either an analog or digital phase locked loop approach. In this type of channel, the problem of carrier recovery is almost the same as for timing recovery, and consequently the solutions can be very similar.

In the case of a time varying channel such as those commonly encountered in mobile radio applications, the problem of phase recovery becomes more complicated. This is because the channel fading affects both the amplitude and the phase of the received signal in a random and possibly rapid manner. Phase estimation of the received signal becomes a problem of estimating how the channel has affected the phase of the transmitted signal, or equivalently, it becomes a problem of channel estimation.

This section begins by briefly introducing a couple of the methods commonly used to achieve carrier recovery in the AWGN channel. It then carries on to introduce some of the methods that have been developed to counter the effects of fading on phase estimation of the received signal.

3.2.1 Traditional Carrier Synchronisation Methods for AWGN Channels

Traditionally, phase recovery has often been implemented using one or both of two primary components, the band pass filter (BPF) and the phase locked loop. Carrier recovery from signals containing an unmodulated sinusoidal component at the carrier frequency is relatively straightforward and simply involves tracking the frequency and phase of the unmodulated component, typically using a PLL. The bandwidth of the PLL will be made narrow enough that the spectral components of the message will not significantly affect the operation of the PLL. The transmission of an unmodulated component merely for the purposes of phase synchronisation at the receiver could be considered an inefficient use of transmitter power. This power carries no information.

A preferable solution is to employ carrier synchronisers that are capable of extracting phase information from the more power efficient suppressed carrier modulation formats such as M-PSK, or M-QAM. This process is somewhat

more complex but is the preferred approach in practice, for the reason of power efficiency stated earlier.

In principle most existing carrier recovery schemes for suppressed carrier modulations regenerate a carrier related component by passing the signal through some form of non-linearity and then filtering out the carrier related component using a narrow band filter. This conceptual approach is illustrated by Figure 3.1. The particular non-linearity used will depend on the type of modulation, however, often it takes the form of an M^{th} power law device.

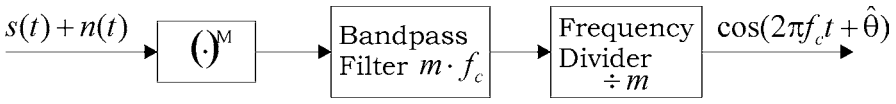

Figure 3.1 Carrier recovery process conceptually involves regenerating a carrier coherent component using a non-linearity, and narrow band filtering to extract the carrier reference.

The effect of using the M^{th} power non-linearity is best explained with an example. Consider the simple case of a modulated signal of the form

$$s(t) = A(t) \cdot \cos(2\pi f_c t + \theta) \tag{3.1}$$

If *A(t)* has the form shown in Eq 3.2, and a_k takes on real values of ±1 then *s(t)* is a BPSK signal, and is a type of double sideband suppressed carrier modulation. Because the modulating signal has no dc component (assuming +1 and –1 are equally likely to occur) the spectrum of *s(t)* will not contain any component at the carrier frequency.

$$A(t) = \sum_k a_k \cdot p(t - kT_b - \tau) \tag{3.2}$$

If we apply a square law non-linearity to the signal then a phase coherent component will be generated at a frequency of $2f_c$, as shown below.

$$\begin{aligned} s^2(t) &= A^2(t) \cdot \cos^2(2\pi f_c t + \theta) \\ &= \frac{A^2(t)}{2} + \frac{A^2(t)}{2} \cdot \cos(4\pi f_c t + 2\theta) \end{aligned} \tag{3.3}$$

This component can in principle be isolated using a narrow band filter tuned to a centre frequency of $2f_c$, and then a frequency divider can be used to

generate a phase coherent component at the carrier frequency. This can then be used as a local oscillator to coherently demodulate the received signal *s(t)*.

A similar principle applies to higher order modulations such as M-PSK and M-QAM, except that in these two cases the power law non-linearity used has to be M^{th} power and 4^{th} power respectively. The order of the power law non-linearity used is directly related to the N-fold phase symmetry of the modulation in question. Square constellation QAM is 90° symmetric, meaning that the constellation can be rotated by a multiple of 90° and look exactly the same. Further details on this may be found in references [10, 24].

In practice using a fixed centre frequency bandpass filter to extract the phase coherent component is not very satisfactory because the generated component will contain some level of distortion, may be contaminated by phase noise, and may not be at the precise centre frequency of the bandpass filter.

The usual solution is to use a PLL in place of the bandpass filter. The PLL acts as a very narrow bandpass filter with a centre frequency that tracks the frequency of the generated coherent component. The output of the PLL will be a sinusoidal signal that is phase and frequency locked to the regenerated carrier harmonic. The output of the PLL can be frequency divided to produce the final phase coherent local oscillator signal. A structure based on this idea, suitable for use with M-PSK type signals is shown in Figure 3.2.

The Costas loop [25, 26] is another practical structure that has been extensively used for suppressed carrier recovery purposes. A bock diagram of the Costas loop is illustrated in Figure 3.3. The noise performance of the Costas loop and the Mth power loop (when M=2) are mathematically equivalent. There are also modified versions of the Costas loop that are suitable for higher order modulations.

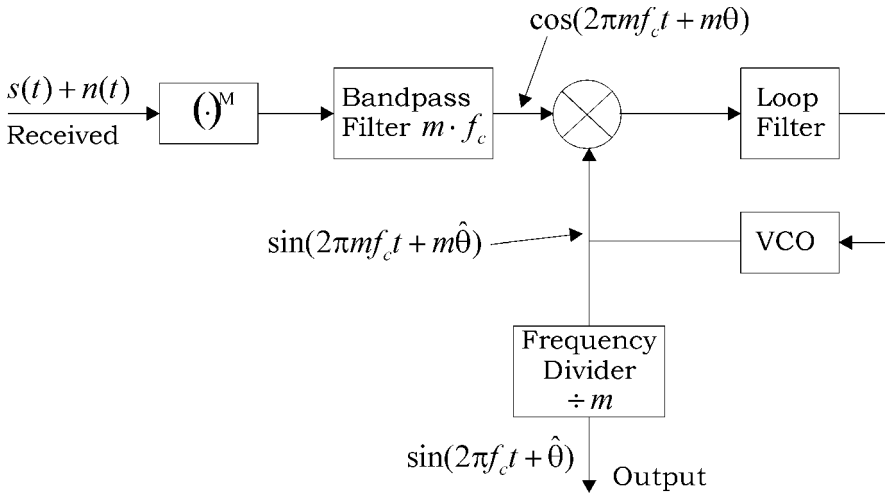

Figure 3.2 M[th] power law carrier recovery using a wider bandwidth bandpass filter and a PLL [10].

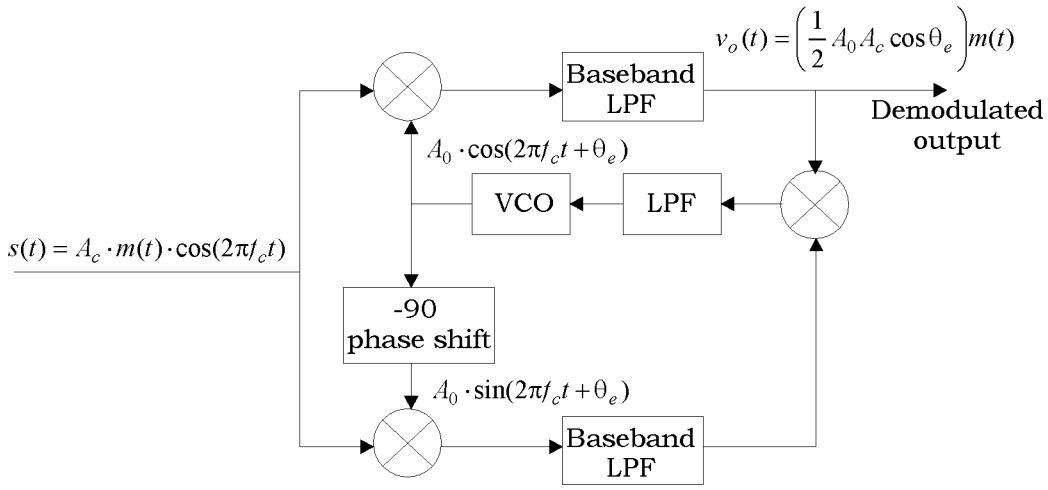

Figure 3.3 Costas carrier recovery loop for suppressed carrier signals.

3.2.2 Contemporary Phase Synchronisation Methods for Fading Channels

More recently, the estimation of signal phase in a fading channel environment has become relatively more important as the incentive to implement more bandwidth or power efficient modulation schemes in wireless mobile digital systems increases. In a fading channel, the phase estimation problem becomes equivalent to estimating the state of the channel at each received signal point. Standard phase recovery based on the methods introduced in the last section are not generally useful in most fading channels where the occurrence of deep fades will usually result in loss of lock of any PLL structure.

This section very briefly outlines some modern methods that have been proposed to help make coherent communications in fading channels possible. They are *pilot tone assisted, pilot symbol assisted,* and channel gain estimation using *per-survivor-processing* (PSP) techniques.

The most successful of the pilot tone assisted methods is known as *transparent tone-in-band* (TTIB), where it is discussed in the 1988 paper [27] with regard to the transmission of 16 QAM data to and from mobile data terminals. In its simplest form the TTIB technique involves splitting the original modulated signal spectrum into two halves, and then separating them to provide a spectral gap in which a pilot tone can be added. The assumption is that the pilot tone, being located in the middle of the data signal spectrum, will undergo the same channel induced degradations, thus making it possible to use this information to compensate for the fading. Drawbacks to this method include bandwidth expansion, and an increase in peak to average power ratio. Further detail on the TTIB technique can be found in [28], and a comparison between TTIB and pilot symbol assisted modulation can be found in Cavers and Liao's paper [3].

The next method will only be mentioned here briefly for completeness as a much fuller description is contained in Chapter 5. Pilot symbol assisted modulation (PSAM) has received considerable attention in the literature, see [2, 4-8, 29] for example. PSAM involves inserting special symbols known to both the transmitter and receiver in order to sample the channel. The receiver is then able to reconstruct the behaviour of the channel at each data symbol location using interpolation, provided that the pilot symbols were spaced with due consideration to the expected maximum fade rate. This channel knowledge can be used to compensate the received fading corrupted symbols prior to decisions. This method is generally considered to be superior to the TTIB method [3].

Per Survivor Processing is effectively a decision directed, sequence detection method based on a trellis structure and path searching. It constitutes a general framework for approximating the Maximum Likelihood Sequence

Estimation algorithms whenever unknown quantities (such as phase, timing or channel state) prevent the use of the classical Viterbi sequence detection algorithm. It avoids the classical approach of using segregated sub-systems to estimate unknown parameters (for example, carrier phase and timing synchronisation, frequency offset tracking and so on), and data detection and instead realises that decision directed estimation of these unknown parameters should be an inherent part of the trellis path search algorithm itself. Thus it effectively combines the two main functions of a receiver (unknown parameter estimation and data sequence detection) into a single self-coupled process. Additional information on the PSP method and it's application to digital mobile communications can be found in [30-35].

3.3 Timing Synchronisation

In timing recovery, the receiver must be able to accurately estimate both the clock frequency (1/T), and the position within each symbol interval, at which to sample the output of the matched filter. Timing synchronisation typically involves two main processes. The first is the *estimation of the timing phase error*, and the second is *correction* of the sampling phase based upon the current error estimate. The task of achieving timing synchronisation is made even more difficult under fading channel conditions. This will also be investigated in more detail in the coming chapters.

3.3.1 Traditional Timing Synchronisation Methods for AWGN Channels

A very simple method for achieving timing synchronisation is for the transmitter to simultaneously transmit the clock signal, or some multiple of the clock signal, with the information signal itself. The receiver then only needs to employ a narrowband filter to extract the timing signal for use in the sampling circuit. Although this is very simple to implement, there are several disadvantages to this particular method. The two primary disadvantages relate to power and bandwidth. The transmitter has to allocate some of its transmitted power to the clock signal, reducing the amount available to the information signal. Secondly, a small fraction of the system bandwidth has to

be used in the transmission of the clock signal. Despite these drawbacks, this method is used in some applications, particularly where the overhead involved in transmitting the clock signal can be spread over multiple users, such as in telephone transmission systems, where wide bandwidths are used to carry the data from many simultaneous phone conversations. The transmitted clock signal can be used in the timing synchronisation of many users' signals at once.

Perhaps of more general application are systems where the clock signal is derived from the data signal itself. These are self synchronising systems, and this is the type of symbol synchronisation considered exclusively in this thesis.

Traditionally, self synchronising receiver timing sub-systems have been based around one of two main functional circuit blocks, the band pass filter (BPF) or the phase locked loop (PLL). In many instances the solutions that have been used are very similar to those used to solve the carrier synchronisation problem.

A classic approach to the symbol synchronisation problem is the early-late gate symbol synchroniser [10]. This is briefly explained here and illustrated in Figure 3.5.

Consider a rectangular baseband pulse shape, defined by (refer to Figure 3.4a)

$$g(t) = \begin{cases} a & 0 \le t \le T \\ 0 & elsewhere \end{cases} \tag{3.4}$$

The output of a receive filter matched to g(t) is illustrated by Figure 3.4b. The pulse exhibits a magnitude maximum at the optimal sampling time, t=T. If however, the output of the matched filter is sampled *early* at t=T-δ_oT and *late* at T+δ_oT, on average the magnitude of the samples at those times will be equal, and smaller than the maximum value at t=T. Therefore, the difference between them will be zero and the optimum sample will be exactly half way between them. This is the timing error detector mechanism used to drive a PLL structure to adjust the sample timing so as to minimise this error. A

structure for achieving this is illustrated in Figure 3.5. In this diagram, correlators have been used in place of the matched filter. It is equivalent.

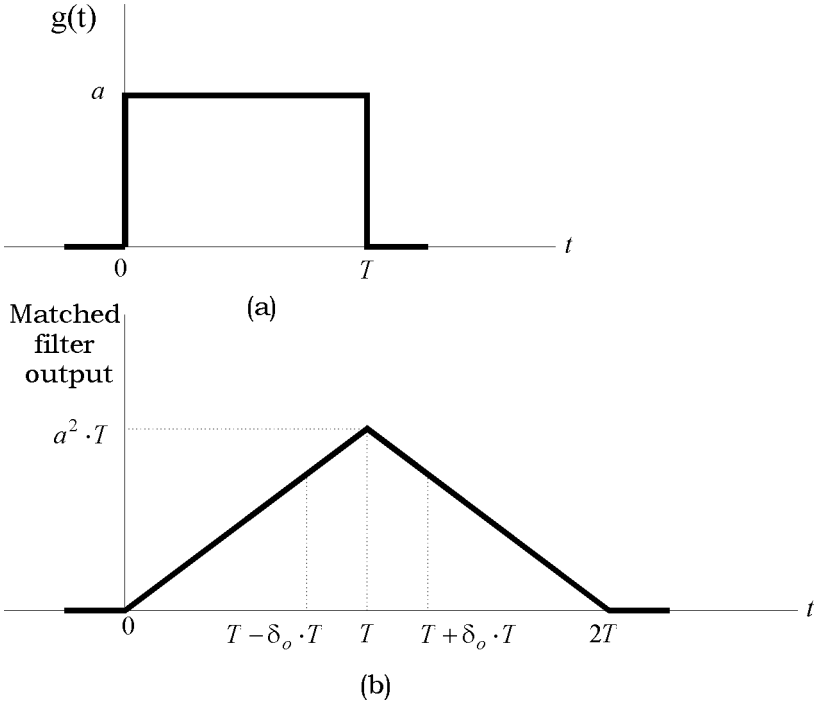

Figure 3.4 (a) Baseband rectangular pulse, and (b) the corresponding matched filter output [9].

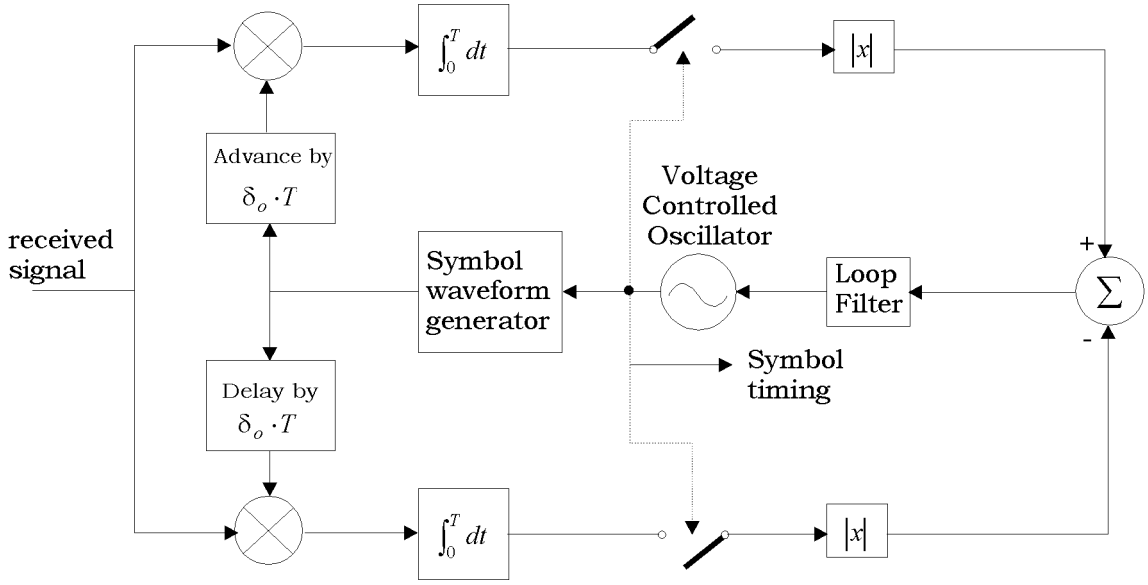

Figure 3.5 Block diagram of one implementation of the early-late type of symbol synchroniser [10].

3.3.2 The Timing Synchronisation Benefits of an All-Digital Receiver Implementation

With the past decade or two of rapid advances in semiconductor device technology, the implementation of all-digital receivers has become a reality. A fully digital receiver implementation can be defined as one in which the received radio signal is sampled and digitised as close to the antenna as possible, with all subsequent processing of the signal done entirely in the digital domain, either in hardware or by software running on a general purpose programmable processor, typically a digital signal processor (DSP). Such fully digital approaches have made new methods of timing synchronisation possible. The methods are no longer constrained by the behaviours and characteristics of physical electronic components and can be performed purely in the mathematical domain. An all-digital receiver also makes possible the concept of non-synchronous sampling, where timing adjustment is performed purely in the digital domain This is explained in more detail in the next section.

3.3.3 Synchronous and Non-synchronous Sampling

Synchronous sampling was commonly used in previous generations of digital receiver. In this case the sampling clock of the analog to digital converter is directly controlled by the output of some form of timing correction circuit, the output of which is dependent on the estimated timing error, as derived from the received signal. An example of this sampling scheme is illustrated in Figure 3.6.

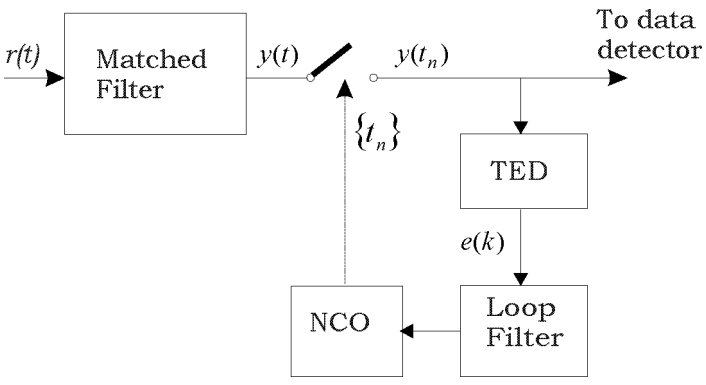

Figure 3.6 Block diagram of synchronous sampling and clock recovery scheme [36].

A fully digital receiver can utilise a non-synchronous sampling scheme. In this case the sampling of the incoming analog signal is not locked to the incoming pulse rate. Sampling instead is governed by a fixed frequency free running sampling clock, and all timing adjustments are done in the digital or mathematical domain, whichever way you like to think of it. This will typically involve some form of interpolation to calculate what the signal sample value would have been at the current estimate of the optimum sampling time. This sampling method is illustrated by the block diagram shown in Figure 3.7.

In all of the work that follows in this thesis, the sampling scheme used in simulations, and in the digital receiver hardware intended to provide actual Rayleigh channel faded signal data to the timing synchronisation algorithms implemented on the TMS320C6701 EVM platform, is of a non-synchronous nature. All sample timing adjustment is performed mathematically without any direct adjustment of the sampling frequency or phase of the analog to digital converter.

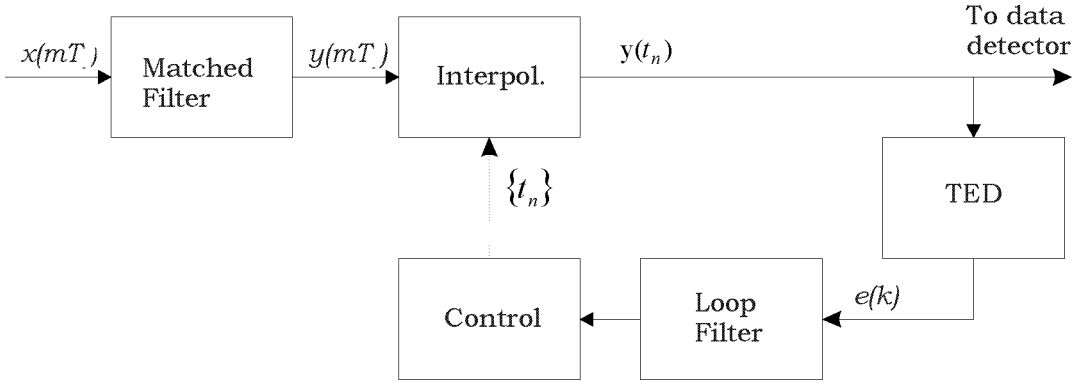

Figure 3.7 Block diagram illustrating non-synchronous sampling and clock recovery [36].

3.3.4 Feed-forward and Feedback Estimators

Timing estimators can be constructed using open or closed loop structures. In both cases a timing error detector block is an integral part of the estimator. The timing error detector (see section 3.3.5) produces an output signal that is in some way related to the magnitude of the discrepancy between the current sampling phase, and the optimum sampling phase. This error signal is used to drive some form of sampling phase correction process. The non-

synchronously sampled (see section 3.3.3) all-digital version of this structure is illustrated by Figure 3.8. In this diagram AAF is an anti-alias filter, whose purpose is to remove any signal components above the Nyquist frequency prior to sampling.

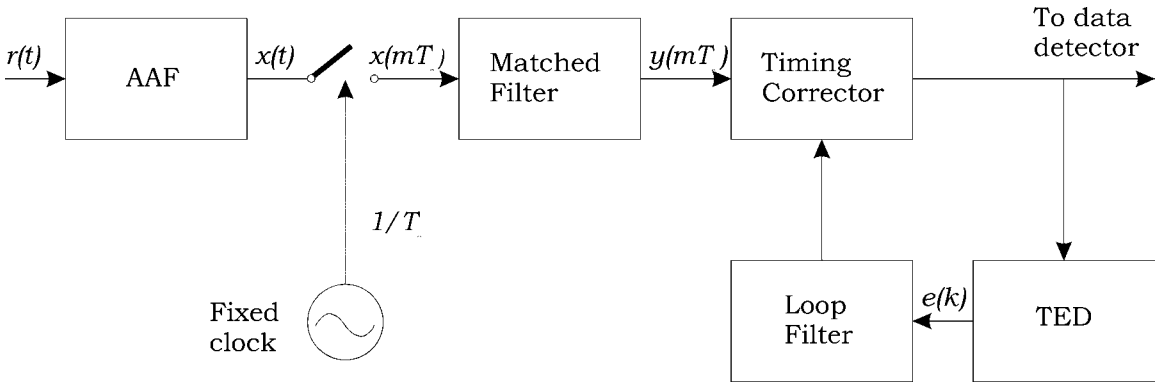

Figure 3.8 Feedback synchronisation structure [37]

In the case of a feed-forward structure, a block of symbols is buffered and then processed to derive a timing correction estimate. This timing correction is then applied to the same buffer of symbols prior to passing them to the data detector. The non-synchronously sampled version of this structure is illustrated by Figure 3.9

Feed-forward schemes may be preferable to feedback ones when short acquisition times are required, such as for burst communications (e.g time division multiplexed systems). They are not prone to the hang up (see section 3.3.5) phenomenon that can slow the acquisition performance of feedback based synchronisers.

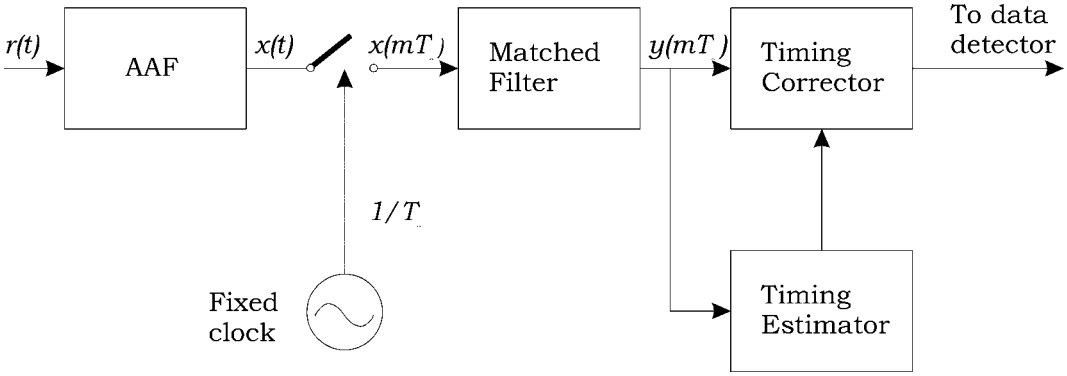

Figure 3.9 Feed-forward synchronisation structure [37]

The work described by this thesis is at all times concerned only with the feedback type of timing synchroniser. Feed-forward synchronisers will not be discussed any further in this work. As may be noted from Figure 3.8 and Figure 3.9, non-synchronous sampling can be used with both feed-forward and feedback timing estimator schemes, whereas it only makes sense to use synchronous sampling with feedback sampling schemes.

3.3.5 Timing Error Detectors

One of the critical 'components' in all timing recovery schemes is the timing error detector (TED), also known as a timing estimator. The task of this block is to calculate, or otherwise produce, a signal that is in some way dependent on the difference between the current sampling phase and the optimum sampling phase. Its role is exactly analogous to that of the phase detector in a standard electronic analog or digital phase locked loop. In a fully digital receiver implementation, which is the kind that this thesis is concerned with, the TED will be some kind of mathematical algorithm implemented either in digital hardware, or in software on a DSP. It will calculate an error signal based on information from the received signal only, or it may additionally make use of knowledge of the transmitted data symbols (as in a training sequence) or alternatively the receiver's own symbol decisions. A TED and hence its associated timing recovery scheme can be classified according to these three situations as Non Data Aided (NDA), Data Aided (DA), or Decision Directed (DD). To summarise, each of the TED types can be classified in the following manner.

- Non Data Aided – TED uses only sample values from the corrupted received signal to calculate a timing error signal.
- Data Aided - TED makes additional use of known transmitted symbols, derived for example from a known preamble sequence.
- Decision Directed - TED instead makes additional use of receiver symbol decisions.

In the literature there are two important tools that are often used in the investigation and comparison of the performance of different timing error detectors. The first is the S-curve of the TED, and the second is the variance curve of the TED output. These curves provide information on the statistical behaviour of the TED output error signal in response to the input of a large number of random data symbol samples having a fixed and known sample timing error. The S-curve is actually the expected, or average, timing 'phase detector' characteristic and is constructed by determining the expected value of the TED output *e(k),* for different values of the sample timing error, ranging from minus half a symbol period to plus half a symbol period each side of the optimum sampling time. The characteristics of the TED S-curve contribute significantly to the overall timing recovery loop's behaviour. It normally takes the shape of an 'S' tipped on its side, hence the name. Stated mathematically,

$$S(\tau-\hat{\tau}) \stackrel{\Delta}{=} E\left\{e(k)\middle|(\tau-\hat{\tau})\right\} \tag{3.5}$$

where $\tau-\hat{\tau}$ is the normalised (to the symbol period) difference between the current sampling phase and the optimum sampling phase. A scheme used in simulation for measuring the S-curve and variance curve of a particular TED is illustrated by Figure 3.10.

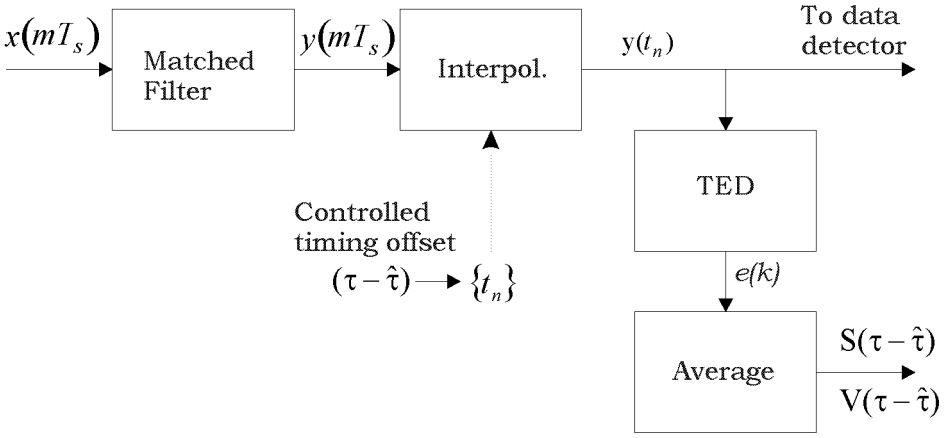

Figure 3.10 Scheme for measuring the S-curve of a digital TED.

In this figure, $x(mT_s)$ is the received signal sampled with a period of T_s, $y(mT_s)$ is the same signal after matched filtering, $y(t_n)$ is the timing adjusted and sub-sampled signal, where the number of samples per symbol has now been adjusted to suit the particular TED being tested. The output of the TED is an

error sample, typically at a rate of one sample per symbol, hence the change in discrete time subscript to *k*.

S-curve features that are normally considered desirable include:

- Effect of signal bandwidth on amplitude is negligible
- 'Large' mean at the edges, i.e doesn't return to zero
- Passes through zero at optimum sampling phase

The second curve typically used in TED comparisons traces out the variance of the TED error signal output as the sampling phase error changes over the range mentioned previously. Characteristics of the TED variance curve that are considered desirable from the point of view of constructing a closed loop synchroniser include:

- zero self noise variance at optimum sample phase
- large variance at extremes of sampling phase offset

In terms of synchroniser performance, zero variance of the TED output at the optimum sampling phase will contribute to good tracking performance while the synchroniser loop is in lock. On the other hand, a large variance of the timing error signal at the extreme ranges of sampling phase offset can help to 'kick' a synchroniser away from a region of the S-curve where the average output of the TED may be relatively small. For TED's having a small or zero mean output value at sampling phase errors of ±T/2 the TED is providing relatively little 'steering force' to the loop when the timing error is in these regions. This can result in excessive loop acquisition times. The increased variance of the TED output can help to mitigate the synchronisers dwell time in this region of the S-curve, thereby speeding up acquisition. This is a common problem for closed loop synchronisers and is termed *hangup*. The phenomenon of hangup has been investigated in depth in the literature, some examples of which include [38-41]. Hangup can be avoided if the TED's S-curve has discontinuities at the extremes of the range of sampling phase

error. This means that the 'steering force' provided to the loop will remain large, on average, resulting in rapid movement toward the S-curves stable null, at the optimum sampling phase.

3.3.6 Interpolation as a Timing Adjustment Mechanism

With non-synchronous fully digital receiver implementations, timing adjustment must be carried out through the use of interpolation techniques. This is because the analog to digital converter's sampling clock is not adjustable. Instead it is free running and almost certainly not synchronous with the symbol rate. This means that none of the actual signal samples are likely to have been taken at the precise instant corresponding to the optimal sampling phase. Thus, it becomes necessary for the receiver to calculate what the sample would have been had it been able to physically sample it at the currently optimal timing estimate. This is achieved through a process of interpolation.

There are many interpolation methods in use, ranging from simple linear interpolation through to n^{th} order polynomial, Gaussian, and sinc based interpolation. For the purposes of timing adjustment in this thesis a cubic polynomial interpolation scheme has been utilised. There is a particular implementation structure that is popular in the implementation of this interpolation method. This structure can be implemented relatively efficiently as an FIR filter and is known as the Farrow interpolator [42]. This structure has an advantage over the switched coefficient interpolator, also commonly used in interpolator implementation, in that it is continuously variable. It provides the ability to adjust sample timing on a continuous rather than a discrete basis. The fractional sample interval, μ, at which the sample interpolant is to be generated, can take on any value between 0 and 1, where a value of 1 would be the next actual sample point. The Farrow structure for the special case of 3^{rd} order polynomial (cubic) interpolation is illustrated by Figure 3.11.

This structure can alternatively be written in matrix algebra form,

$$y(\mathbf{x},\mu) = \boldsymbol{\mu} \cdot \mathbf{A} \cdot \mathbf{x}_n \tag{3.6}$$

where the coefficient matrix **A** is just the FIR filter coefficients used in the FIR filter structure illustrated in Figure 3.11, **x** is a vector of four input samples (four because this is a cubic polynomial interpolation), selected by the basepoint index *n*, for which an interpolant is to be calculated, and μ is a cubic polynomial coefficient vector in μ. These are shown explicitly below for clarity.

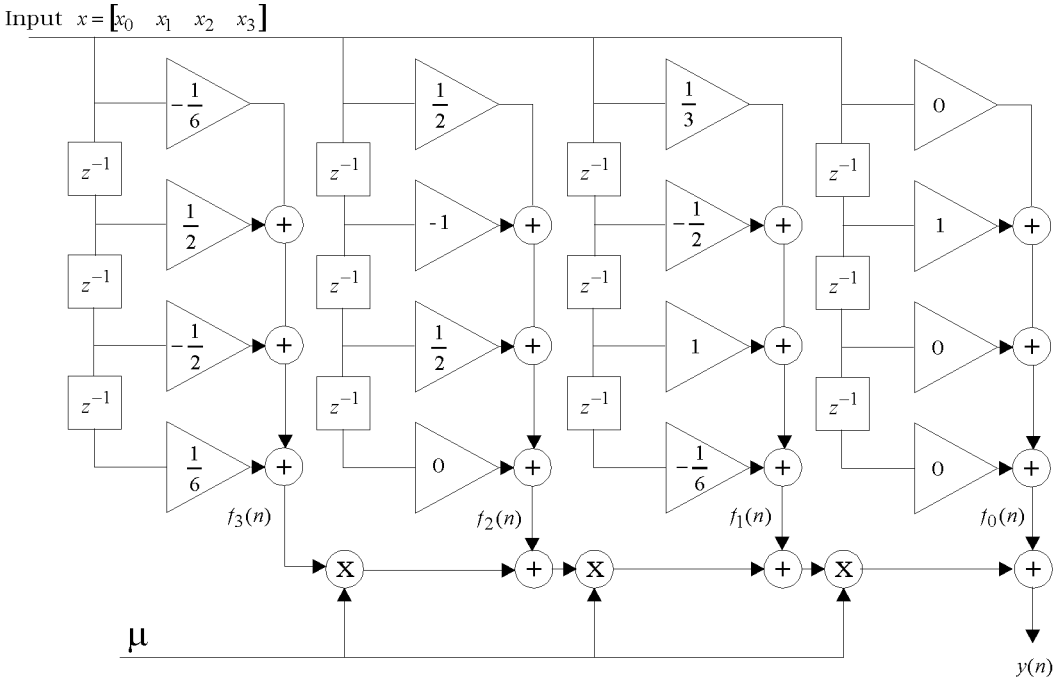

Figure 3.11 Farrow FIR structure for implementing cubic polynomial interpolation.

$$\mathbf{A} = \begin{bmatrix} \frac{1}{6} & -\frac{1}{2} & \frac{1}{2} & -\frac{1}{6} \\ 0 & \frac{1}{2} & -1 & \frac{1}{2} \\ -\frac{1}{6} & 1 & -\frac{1}{2} & -\frac{1}{3} \\ 0 & 0 & 1 & 0 \end{bmatrix} \qquad \mathbf{u} = \begin{bmatrix} \mu^3 & \mu^2 & \mu & 1 \end{bmatrix} \qquad \mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

The interpolation inputs are illustrated in Figure 3.12 to further clarify their meaning. The interpolant to be calculated will lie somewhere between input samples x_1 and x_3. The exact location will depend on the sign and magnitude

of the fractional sample μ. If μ=0 then the calculated interpolant will correspond exactly with the actual sample x_2.

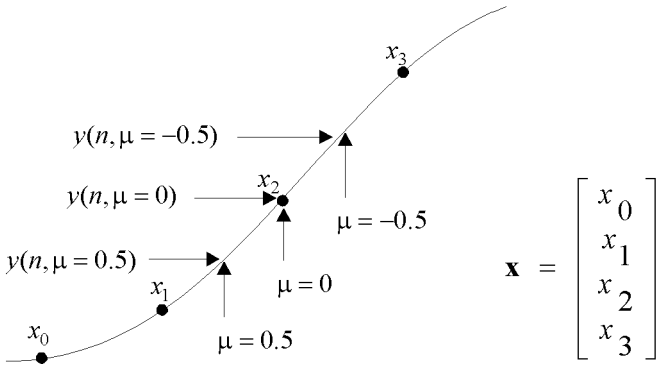

Figure 3.12 Interpolant generation using the cubic Farrow interpolator, given the fractional sample interval μ.

3.4 Phase Locked Loops as Synchronisation Structures

This brief introduction to PLL operational theory is presented because the all digital timing synchronisation algorithms presented later in this thesis are based on the basic PLL structure.

The phase locked loop (PLL) is one of the cornerstones of communications technology. Its roots stretch as far back as 1919 [43], however the first widespread use of phase lock techniques did not occur until the development of television [44]. Here, it was used in the synchronisation of the horizontal and vertical scans to marker pulses embedded in the video signal. Today it is used in many guises and forms to achieve significant improvements in performance of both transmitter and receiver devices, over the conventional methods used prior to the invention of the PLL. Much of the current advanced communications technologies in existence would not be possible without the PLL.

In simple terms the PLL is a feedback control structure designed to precisely control the phase of a local oscillator to match that of some external reference signal. Although the conceptual explanations presented in the initial sections of this chapter tend to talk about the PLL in terms of analog components and concepts, each of the basic PLL building block functions can be duplicated in software or digital hardware. This will be illustrated later in this chapter when

a software implementation of a PLL, used in simulation work, will be demonstrated.

Much greater detail on the operation and behaviour of PLL's can be found in a variety of texts and journal papers, including [26, 45, 46]. In addition, the author has produced an internal report [47] that brings together many of the concepts of analog and digital phase locked loops, and includes an example of software implementation and simulated loop responses to various input signal stimuli. The following sections only aim to present a very brief summary of important PLL concepts and results.

3.4.1 Basic Theory of Operation

Conceptually, a PLL consists of three fundamental functional blocks (Figure 3.13):

- phase difference detector
- loop filter
- controllable oscillator (whose phase is to be controlled)

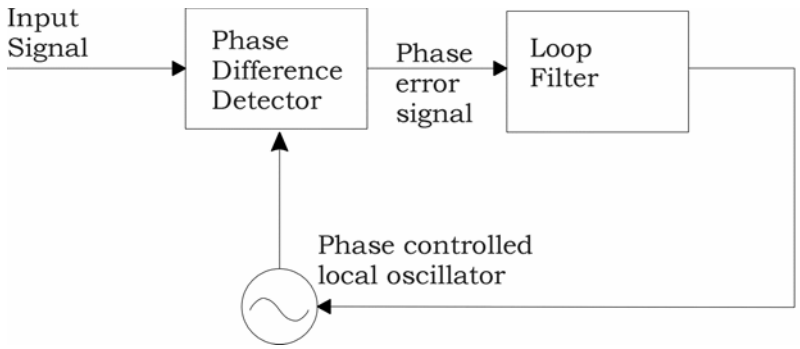

Figure 3.13 The three basic functional components of a PLL.

The basic idea of operation is very simple. The phase difference detector makes a comparison between the phase of the reference signal and the locally controlled signal. The output is a signal that is in some way proportional (not necessarily linearly) to the phase difference. The error signal may be smoothed by a loop filter prior to being presented as the control signal to the voltage controlled oscillator. Due to the negative feedback arrangement, this signal

changes the frequency and hence phase of the local oscillator in a direction that reduces the phase difference between the two signals. When the local oscillator is matched in both frequency and phase to the reference signal, the control signal will have reached a value such that it no longer acts to change the frequency and phase of the local oscillator. This is termed *lock.*

3.4.2 Linearised Baseband Model of a PLL

Analysing a PLL in its locked state means that we can make the assumption that the phase error is very small. Thus for a PLL with a sinusoidal type phase detector (PD) it is possible to make the approximation that,

$$\sin(\theta_e) \approx \theta_e \tag{3.7}$$

This linearises the PD, meaning that in the locked / tracking state, the device may be mathematically modelled as a linear system with little loss in accuracy. Furthermore, this means that we may employ Laplace Transforms to aid in the analysis of the system. A block diagram with the appropriate Laplace notation is shown in Figure 3.14.

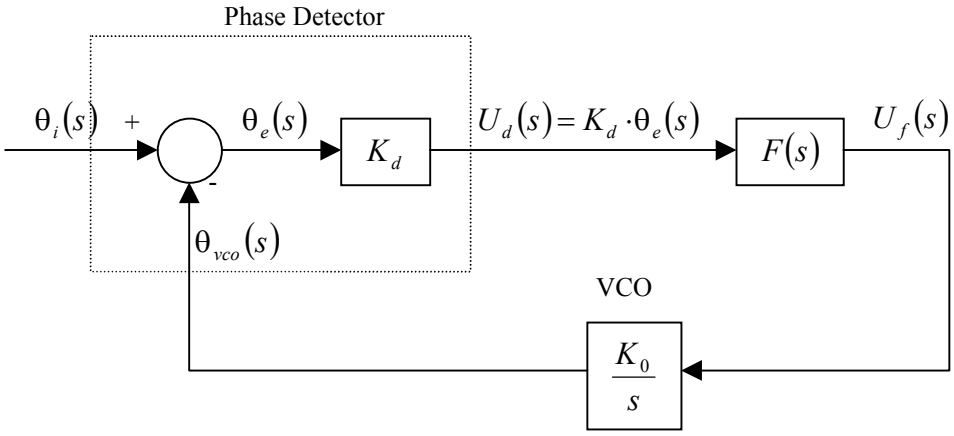

Figure 3.14 Linearised baseband model of PLL.

The notation of Figure 3.14 is explained below and the system parameters are defined as follows:

K_d Phase detector gain – it relates the magnitude of the input phase difference to the resulting output voltage. For a PD with a given characteristic

(e.g sine or saw tooth), the gain is determined by the gradient of the characteristic, and more specifically, in the linear analysis presented here where phase errors are assumed to be small, K_d is the gradient at $\theta_e=0$.

K_o VCO Sensitivity. This relates the input control signal magnitude to the resulting change in frequency of the VCO. In an analog VCO this would have the units of Hz per volt. Changing the frequency of the VCO also changes its phase.

$F(s)$ Loop Filter transfer function (TF). This is a constant in the case of a first order PLL, a 1st order TF for a second order PLL, and a second order TF for a third order PLL.

$\theta_i(s)$ Input signal phase. For example this may be the radio signal being received, or in baseband synchronisation applications may be the received waveform shaped data signal.

$\theta_{vco}(s)$ Local reference VCO signal phase. It is this signal that the PLL control structure is trying to make the same as the input signal phase, i.e track it.

$\theta_e(s)$ Phase error signal.

It is a relatively straightforward procedure to apply Laplace analysis to the model of Figure 3.14 to derive the phase and error transfer functions of this linearised PLL system. The details will not be reproduced here, however the analysis of such a system can be found in many communications and control system text books, for example [26]. In terms of the notation of Figure 3.14, the phase and error transfer functions are given, respectively, by,

$$H(s)=\frac{\theta_{vco}(s)}{\theta_i(s)}=\frac{K_0\cdot K_d\cdot F(s)}{s+K_0\cdot K_d\cdot F(s)} \tag{3.8}$$

$$H_e(s)=\frac{\theta_e(s)}{\theta_i(s)}=\frac{s}{s+K_0\cdot K_d\cdot F(s)} \tag{3.9}$$

3.4.3 First Order PLL

A first order PLL is one in which the loop filter is omitted entirely, or consists merely of a gain block. For the purposes of this analysis we shall consider $F(s) = G_{loop}$. Thus Eq.3.8 becomes,

$$H(s) = \frac{K_0 \cdot K_d \cdot G_{loop}}{s + K_0 \cdot K_d \cdot G_{loop}} = \frac{G}{s+G} \tag{3.10}$$

where, for the sake of simplicity the 3 constants have been absorbed into a single overall open loop gain G. It is simple to see that the –3dB loop bandwidth is just,

$$\omega_{-3dB} = G \tag{3.11}$$

With a first order PLL, loop gain is the only parameter available to the designer for adjustment of the devices performance. Often there are conflicting requirements that cannot satisfactorily be resolved with only one degree of freedom. For example, to achieve a small steady state phase error it is necessary to have a large loop gain. However a large loop gain implies a large loop bandwidth and hence poorer tolerance to noise. A second order loop has an additional degree of design freedom allowing some optimisation of the PLL to better suit both requirements.

3.4.4 Second Order PLL

A second order PLL is implemented by using a first order loop filter. This provides some ‘smoothing’ of the error signal prior to presenting it as the control signal to the VCO. Second order PLL’s have a number of performance advantages over first order PLL’s and are more commonly used in practical applications. Two kinds of first order loop filter are possible. The first involves an imperfect integrator, in which the filters pole does not lie at the Lapace s-plane origin. The second is a perfect integrator with a pole at s=0. The

illustration below shows how each of these may arise in an analog electronics context.

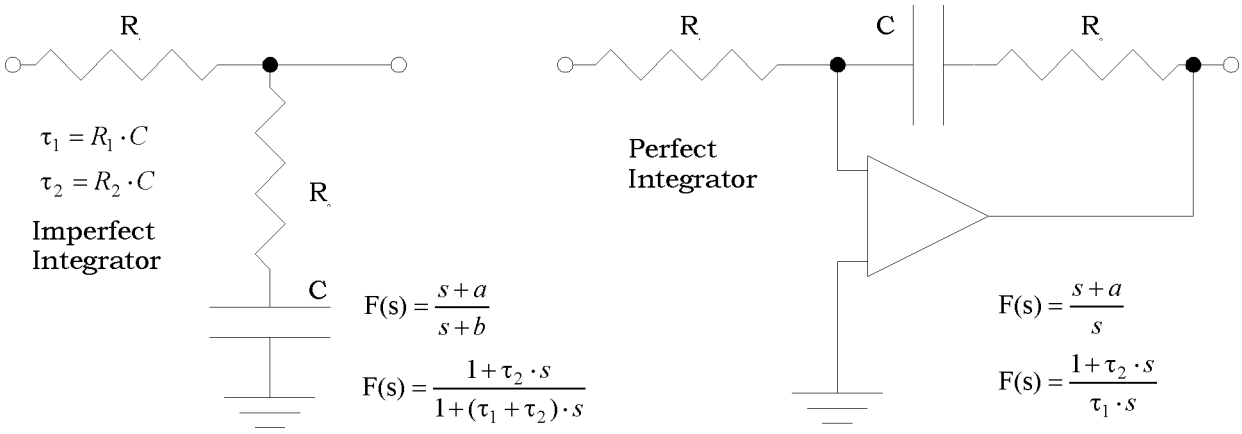

Figure 3.15 Analog electronic implementations of first order loop filters. The passive circuit implements an imperfect integrator, while the active circuit performs the function of a perfect integrator.

For simplicity of illustration, and greater generality, consider the first order loop filter described by the transfer function,

$$F(s) = \frac{s+a}{s+\lambda \cdot a} \tag{3.12}$$

This can represent either the perfect or the imperfect integrator illustrated in Figure 3.15 simply by setting $\lambda = 0$ or $\lambda \neq 0$. In the case where $\lambda \neq 0$, λ will set the fractional offset of the imperfect integrators pole from the origin, relative to the integrator's zero displacement from the origin. In the special case where $\lambda = 1$, the PLL will degenerate to a first order system since F(s) = 1.

Substituting this general loop filter transfer function into Eq.3.8 produces,

$$H(s) = \frac{G \cdot (s+a)}{s^2 + (G + \lambda \cdot a) \cdot s + G \cdot a} \tag{3.13}$$

where in this case the substitution $G = K_o K_d$ has been used. This is a second order system and it is common to write the TF in terms of the damping coefficient, δ^* and the natural frequency, ω^*_n. The standard second order system 'template' denominator is written as,

$$s^2 + 2 \cdot \delta^* \cdot \omega_n^* \cdot s + \omega_n^{*2} \tag{3.14}$$

Setting the denominator of Eq.3.13 equal to Eq.3.14 and then performing some algebraic substitution, manipulation and simplification reveals,

$$\begin{aligned} \delta^* &= \sqrt{\frac{G}{4 \cdot a}} + \lambda \cdot \sqrt{\frac{a}{4 \cdot G}} \\ &= \delta + \lambda \cdot \left(\frac{1}{4 \cdot \delta} \right) \end{aligned} \tag{3.15}$$

and,

$$\omega_n^* = \omega_n = \sqrt{G \cdot a} \tag{3.16}$$

where δ and ω_n are the damping and natural frequency for a perfect second order system. It is more useful to be able to specify the natural frequency and damping factor required for the desired second order PLL and, given the desired loop filter pole relative offset λ, have the loop filter zero location, *a*, and the open loop gain, G, calculated.

3.4.5 Software Implementation of a PLL

There are two approaches to implementing a PLL in software. The first is to come up with an expression for the closed loop transfer function of the PLL and then implement the appropriate difference equation in software. The second approach is to treat the PLL as a system of discrete building blocks, such as the phase detector, the loop filter, and the integrator (or VCO). Each of these can be implemented separately and then strung together as sequential software tasks. This is the approach that will be demonstrated briefly here.

For the purposes of this example a phase detector with a sine characteristic will be used. The simulation model that we are implementing is shown in Figure 3.16.

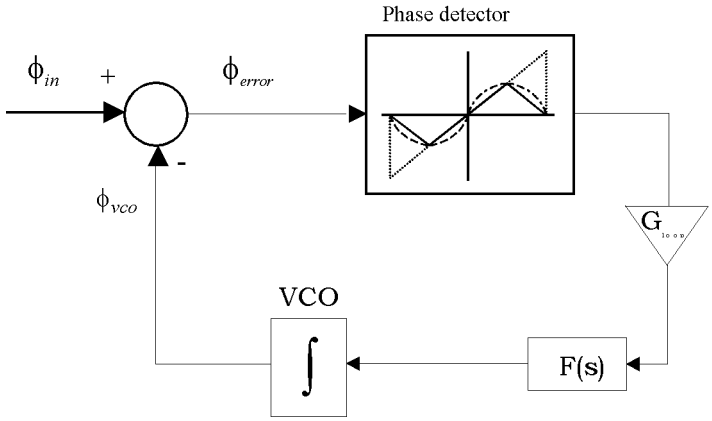

Figure 3.16 Baseband simulation model of phase locked loop system.

3.4.5.1 Integration Block

Straightforward application of the bilinear z transform to the Laplace equivalent of time domain integration, namely 1/s, yields a difference equation that can be used to implement numerical trapezoidal integration.

The Bilinear transform is given by,

$$F(z) \Leftrightarrow F(s), s = \frac{2}{T} \cdot \left[\frac{1 - z^{-1}}{1 + z^{-1}} \right] \tag{3.17}$$

Therefore the z domain transfer function representing integration is simply the reciprocal of Eq.3.17,

$$\frac{Y(s)}{X(s)} = \frac{1}{s} \Rightarrow \frac{Y(z)}{X(z)} = \frac{T}{2} \cdot \left[\frac{1 + z^{-1}}{1 - z^{-1}} \right] \tag{3.18}$$

This may be converted to the discrete time domain by applying the inverse Z-transform, yielding the following difference equations.

$$\begin{aligned} y(n) &= \frac{1}{2 \cdot f_s} \cdot \left[w(n) + w(n-1) \right] \\ w(n) &= x(n) + w(n-1) \end{aligned} \tag{3.19}$$

Additional information on the transformation can be found in [47]. This algorithm can be represented as a discrete signal flow diagram, as shown below.

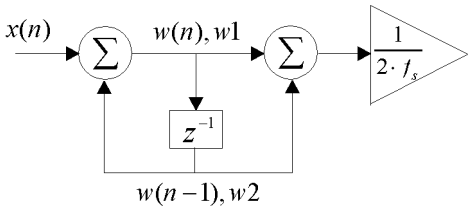

Figure 3.17 Discrete time implementation of trapezoidal integrator.

The corresponding MATLAB code that implements the trapezoidal integrator block would be,

```
w2=0;                       %Initialise delay buffer variable

for i=1:npts                %Run loop for as many signal points as we've got. This loop is
                            %running at the sample rate.
    w1=x(i)+w2;             %w2 is the delayed version of w1, calculated last iteration.
    y(i+1)=(w1+w2)/(2*fs);
    w2=w1;                  %Performs the delay operation

end
```

3.4.5.2 Loop Filter

Applying the bilinear transform to Eq.3.12 gives,

$$\frac{Y(z)}{X(z)}\cdot = \frac{(a-2\cdot f_s)\cdot z^{-1}+(a+2\cdot f_s)}{(\lambda\cdot a-2\cdot f_s)\cdot z^{-1}+(\lambda\cdot a+2\cdot f_s)} \tag{3.20}$$

This can be written as discrete time difference equations in the form,

$$y(n)=(a-2\cdot f_s)\cdot w(n-1)+(a+2\cdot f_s)\cdot w(n) \tag{3.21}$$
$$w(n)=\frac{x(n)-(\lambda\cdot a-2\cdot f_s)\cdot w(n-1)}{(\lambda\cdot a+2\cdot f_s)}$$

Once again, this may be represented by the following discrete time signal flow block diagram.

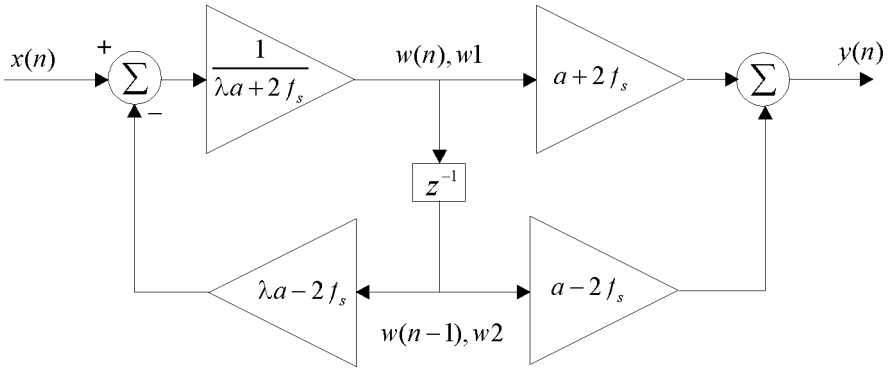

Figure 3.18 Discrete time implementation of first order loop filter.

If we make the following substitutions for the sake of clarity,

$$\begin{aligned} g_m &= a - 2 \cdot f \\ g_p &= a + 2 \cdot f_s \\ h_m &= \lambda \cdot a - 2 \cdot f \\ h_p &= \lambda \cdot a + 2 \cdot f_s \end{aligned} \tag{3.22}$$

then the MATLAB code that implements this filter can be written as listed below.

```
w2=0;
for i=1:npts                    %npts is the number of samples in the signal waveform,
                                %thus, this loop is run at the sample rate.
   w1=(1/hp)*(x(i)-hm*w2);
   y(i+1)=gm*w2+gp*w1;
   w2=w1;                       %Implements the delay function.
end
```

3.4.5.3 Conclusion

As these examples have shown, implementing a software based phase locked loop is relatively straightforward and affords a high degree of flexibility. In additions, software based PLLs can often be designed to perform in ways that would be very difficult or impossible to achieve with purely hardware based solutions, although normally with a compromise in achievable operating frequencies. Phase locked loops are therefore extremely useful building blocks in the design of timing synchroniser sub-systems.

CHAPTER 4
A COMPARISON OF THREE SYNCHRONISATION ALGORITHMS

4.1 Introduction

The purpose of this chapter is to present the three timing error detectors used throughout the author's research work and to compare the performance of their associated closed loop synchronisers in the AWGN and frequency flat Rayleigh fading channels.

The comparison will prove to be interesting because two of the synchronisers are Non Data Aided (NDA) and have been designed with the AWGN channel in mind. The third synchroniser could be classified as decision directed (DD) when used without the benefit of a known start up pre-amble sequence, and additionally, was designed for the flat Rayleigh fading channel using maximum likelihood principles.

In comparing synchronisation sub-systems both transient and steady state behaviour is of interest. Transient behaviour includes such things as acquisition time, lock in range and pull out range. Steady state behaviour is more relevant when it comes to the effect of the operation of the embedded synchronisation sub-system on the overall performance (for example BER) of the receiver system. The two major steady state synchroniser performance indicators of interest are the bias and the jitter, or variance in the estimate. A synchroniser's parameter estimate is *unbiased* if the long term time average of the estimate equals the actual parameter value being estimated. The jitter is a measure of how widely the estimate varies around the long term average

4.2 Three Timing Error Detectors

The basics of timing error detector's (TED's) have been previously introduced and discussed in section 3.3. The purpose of this section is to present the

three specific timing error detectors utilised in the construction of the symbol synchronisers used throughout the rest of this thesis.

4.2.1 Gardner TED

The Gardner TED presented in this section was first proposed by Gardner [48] as a simple non-data aided algorithm for the detection of timing error in a synchronously sampled BPSK or QPSK digital receiver. It was developed using physical reasoning from data waveforms rather than through the application of any particular mathematical optimisation principles, such as maximum likelihood. The algorithm requires two samples per symbol and was stipulated originally to be best suited to signals with an excess bandwidth in the range 40-100%. The Gardner TED does not depend on receiver decisions. There is also no requirement that phase lock be achieved prior to timing recovery because the timing error detector characteristics are completely independent of carrier phase.

Mathematically the algorithm takes the form [48]

$$e(k) = y_I(k-1/2)\cdot\left[y_I(k) - y_I(k-1)\right] + y_Q(k-1/2)\cdot\left[y_Q(k) - y_Q(k-1)\right] \qquad (4.1)$$

where,

$y_I(.)$ is the in-phase "I" symbol sample
$y_Q(.)$ is the quadrature "Q" symbol sample
$y(k-1/2)$ is the sample between symbol $y(k-1)$ and $y(k)$

The notation of this equation relates to a sampled receiver where the output of the receiver matched filter is a sequence of sample pairs, $y_I(.)$and $y_Q(.)$. Figure 4.1 should help to clarify the algorithm and its notation. The TED algorithm extracts the timing information from this stream of half symbol period spaced samples, generating one error signal sample per symbol. The index (k-1/2) refers to the signal sample that is halfway between the symbol spaced k^{th} and

(k-1)th signal samples. A pictorial representation of a structure that implements this algorithm is shown in Figure 4.2.

An intuitive explanation of the operation of this TED can be made relatively simply [48]. Closer examination of Eq.4.1 shows that the error signal is composed of the product of a midpoint symbol 'transition' sample and a value that contains some slope information. In the case of optimal timing the midpoint sample will be zero on average, assuming an ideal noiseless channel.

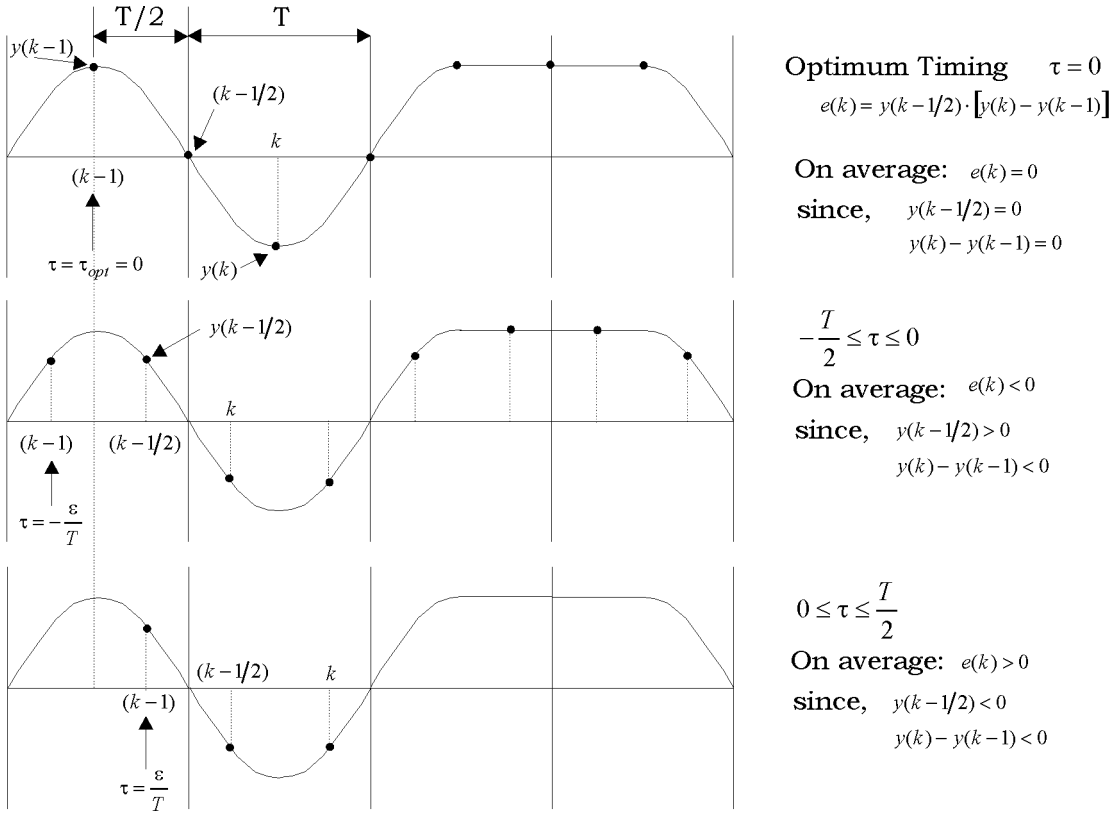

Figure 4.1 Waveform diagram illustrating the basis for the Gardner timing error detection algorithm.

If there is a timing error then the midpoint sample, will be non zero. The magnitude will depend on the size of the timing error. The slope term, calculated by the algorithm from an examination of the two sample values either side of the midpoint sample, provides information, indicating the direction in which the sampling phase must adjust to reduce the error. There is no timing information available if there is no transition. In such a situation, assuming no timing error, the two samples each side of the midpoint sample will be the same, thus the slope term will be zero. This has the benefit of

preventing the non-zero midpoint sample value from contributing to an inappropriate non-zero error value. A lack of regular data transitions can cause a problem for timing recovery schemes such as this. Data scramblers are one practical solution that are used to prevent long strings of zeros or ones appearing in the data stream.

Following similar physical reasoning, it is straightforward to deduce that this TED will suffer from self noise at the optimum sampling phase if the received signals raised cosine pulse shape roll off is anything less than 100%. For pulse shapes with less than 100% roll-off, the zero crossings of symbol transitions no longer necessarily lie exactly halfway between the two nearest strobe points (optimal sample points). The average zero crossing point is midway, but there is now a scatter of zero crossing locations around the midway point. This means the output of the TED at the optimum sampling phase will now only be zero on average, but will exhibit self noise on an instantaneous basis.

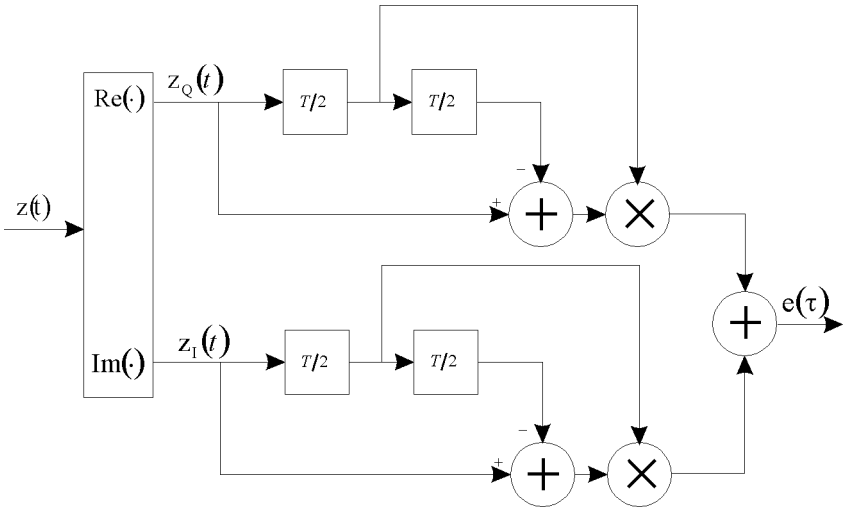

Figure 4.2 Block diagram of the Gardner TED structure.

4.2.2 Amplitude Directed TED

The Amplitude Directed TED was first proposed in a paper [49] by Verdin and Tozer. Its construction arose from their realisation that the extraction of timing information from a complex pulse amplitude modulated (PAM) data waveform should be considered as a complex operation rather than a real

operation performed separately on the in-phase and quadrature components of the signal.

Mathematically the TED algorithm takes the form [49]

$$e(k) = \mathrm{Re}\left\{\left[\mathrm{sign}\left(y(k-1)\right) - \mathrm{sign}\left(y(k)\right)\right] \cdot y^{*}\left(k-1/2\right)\right\} \tag{4.2}$$

where the notation is the same as that defined for the Gardner TED algorithm in Eq.4.1 and sign(x) is the complex signum function, defined by Eq.4.3. Taking the sign(.) of a complex value normalises that complex quantity, so that it has a magnitude of 1, and the same phase as the original. In this context it can be thought of as a quantiser, whose function lays somewhere between a data-slicer (giving a decision directed TED) and no non-linearity (giving a non data aided TED). The resulting scheme was termed 'Amplitude Directed' by Verdin and Tozer. It has the same advantage as the Gardner scheme, in that it is invariant to carrier phase rotation, and so can be used before carrier phase lock has been achieved.

This scheme, as originally proposed, was applied to M-PSK modulated signals and was shown to have performance somewhere between completely decision directed and purely non-data aided TED's. In this thesis the Amplitude Directed TED has been applied instead to M-ary QAM signals rather than M-PSK modulated signals and has still been found to perform quite well. Although this TED was not designed specifically for Rayleigh fading channels, work to be presented later in this chapter has found that it handles Rayleigh channels with slow to moderate fade rates reasonably well. The improved performance of the AD TED over the Gardner TED is at the expense of a small increase in complexity. It makes use of the complex signum function and therefore requires some additional computation when compared with the simpler Gardner TED. Further information on this complexity trade off can be found in Appendix A which deals with some of the practical aspects of implementing these algorithms on a real time DSP platform. Some of the

advantages of this TED claimed by the authors when compared with standard NDA TED's, such as the Gardner, include:

- Lower self noise
- Immunity to hangup

The structure of the AD TED is illustrated below. A comparison between this and the equivalent structure for the Gardner TED, illustrated in Figure 4.2, reveals significant similarity. The most obvious difference is in the AD's treatment of the signal samples as complex entities rather than extracting the in-phase and quadrature parts and processing them independently.

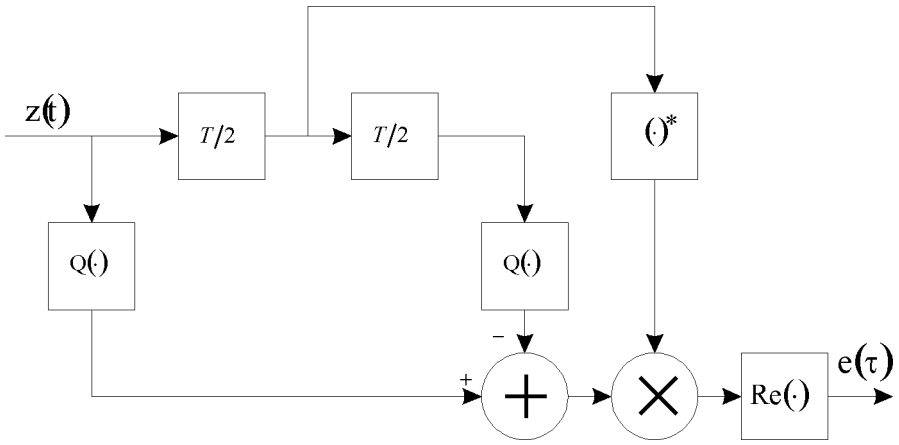

Figure 4.3 Structure of the two samples per symbol Amplitude Directed timing error detector.

For the readers reference, the complex signum function, denoted by Q(.) in Figure 4.3, is defined mathematically to be,

$$Q(x) = \begin{cases} \dfrac{x}{|x|} & x \neq 0 \\ 0 & x = 0 \end{cases} \tag{4.3}$$

where $|x|$ for a complex quantity is the magnitude of that complex value.

4.2.3 Watkins FFML1 TED

The final TED investigated by this thesis was designed mathematically from consideration of maximum likelihood parameter estimation principles for a

received QPSK, or DQPSK signal experiencing flat, Rayleigh fading. The TED presented here is the simplest of a family of such TED's designed by Gabriel Watkins during his Master's research [50] at the Department of Electrical Engineering, University of Canterbury.

This TED differs from the previous two in that the optimal form of the detector requires that the receiver have knowledge of the transmitted symbols. This condition can be satisfied if a pre-amble sequence is used for initial acquisition, however, in normal operation this requirement cannot be met. Therefore, in tracking mode it is necessary to operate the detector in a sub-optimal decision directed mode.

Mathematically the TED takes the form [50]

$$e(k) = \left[\frac{y^*(k-1)\cdot y(k-1)}{a(k-1)\cdot F_0 \cdot a^*(k-1) + \sigma_n{}^2} - 1 \right] \cdot \left[\frac{F_0}{a(k-1)\cdot F_0 \cdot a^*(k-1) + \sigma_n{}^2} \right] \times \left[a^*(k-1)\cdot [a(k) - a(k-2)] + a(k-1)\cdot [a(k) - a(k-2)]^* \right] \tag{4.4}$$

In the context of this equation, the notation is defined in the following manner.

$y(.)$ symbol spaced complex signal samples
$a(.)$ actual symbol transmitted
F_0 covariance of the fading process
$\sigma_n{}^2$ variance of the additive gaussian noise
$(.)^*$ complex conjugate

When the detector is to be operated in tracking mode then the actual transmitted symbols will be replaced with the result of a preliminary hard symbol decision from a quantiser unit. The levels of the quantiser are set to match the levels in the transmitted data symbols. Considering that this TED is intended for use in a fading channel environment, it is not immediately obvious that this approach would have any merit, because the preliminary hard decisions are being derived from fading corrupted signal samples. Intuitively it may be expected that these decisions would provide an extremely

poor approximation to the actual symbols transmitted, thereby rendering the output of the TED devoid of any useful timing information. However, as explained by Watkins [50], despite their inherent phase ambiguity due to the fading, the sequence of symbol estimates derived in this manner is often adequate for providing useful timing information.

Because of the need to use preliminary symbol estimates derived from faded signal samples, sub-optimal use of this algorithm, as required during tracking, is not very successful with multilevel QAM type modulations. This is because the preliminary hard decisions on a faded multilevel QAM signal bear a very poor resemblance to the actual transmitted symbols. All future direct comparisons between the three timing error detectors, and their associated closed loop synchronisers will utilise 4 QAM modulated signals. This of course is equivalent to QPSK.

4.2.4 TED Performance Metrics - Computer Simulation

The statistical characteristics of the three timing error detectors previously introduced were calculated using computer simulation techniques. Initially the simulations were developed in MATLAB, an advanced mathematical package with its own high level interpreted language. Later, for reasons of computational speed, and to partially satisfy other project goals, these simulations were rewritten in the 'C' high level language. The simulations saw a performance increase of between 10 and 20 times as a result of the re-coding. MATLAB was used to post process the results from the 'C' based simulations and to generate graphical displays of the results.

In all cases, except where otherwise noted, the modulation format used in the simulations is QPSK, or equivalently, 4 QAM. Root raised cosine filtering, with a pulse shape excess bandwidth of α where ($0 \leq \alpha \leq 1$), was used at both the transmitter and receiver. The transmitted data was generated randomly using a uniform discrete random number generator based on the Mitchell-Moore algorithm from Knuth, volume 2, [59]. Refer to A.3.1 for more detail.

4.3 Statistical Characteristics of Three Timing Error Detectors

The figures in the following few pages plot the mean and variance of each of the timing error detector outputs as a function of the actual sampling phase error. The effect of three communication system parameters on the statistical characteristics of the TED's are examined. These include the signal to noise ratio E_b/N_0, the excess bandwidth α, and the fade rate. The fade rate of the channel is characterised by f_DT, which is the product of the maximum doppler spread f_D and the symbol period T. [To limit the number of plots presented, in most instances only typical and worst case parameter values have been selected for display]. In the simulations used to generate the plots in this section, the transmitter used a square root raised cosine pulse shaping filter with a roll-off of α. The receiver utilised the square root raised cosine matched filter. Neglecting the effect of the channel, this results in a full roll-off raised cosine pulse shape. In most cases, results for two values of alpha have been shown, α=1 represents the least stringent value that could be used in a receiver, and α=0.35 represents a more realistic value.

The plots shown in Figure 4.4 provide a set of reference mean and variance curves for the Gardner, Amplitude Directed and both the ideal FFML1 and the sub-optimal decision directed FFML1 timing error detectors, in the case of ideal channel conditions. In this case *ideal* means noise free, and no fading. In each case, unless otherwise noted, the plots have not been normalised. Normalisation of the S curves refers to the process of scaling the data so that each mean curve has a slope of –1 at the zero timing error point. This would be achieved by dividing each mean curve by G_{TED}, which is the slope of the curve at ε = 0, and by dividing the associated variance curve by G_{TED}^2. This convention is followed in [50].

One obvious feature that distinguishes the four TED S curves is the point at which they begin to fall back towards zero. The Gardner S curve is very 'sine' like and starts to decrease toward zero for normalised timing error offsets of > ±0.25. The Amplitude Directed S curve has a peak maximum magnitude at approximately ±0.35, however it maintains significant magnitude until much

nearer to the extreme sampling phase offsets, falling quickly toward zero for normalised timing error offsets > ± 0.45 in the plot presented in Figure 4.4. In actual fact this is an artifact of the number of points used to plot the curve. If more points were used then the curve would show that the S curve maintains significant magnitude right out to the extremes of ±0.5. This is a very desirable characteristic as it helps to provide the associated closed loop synchroniser with some immunity to hangup, described previously in section 3.3.5. As will be noted later, this desirable characteristic is degraded with the introduction of noise, fading, and reduced pulse shape excess bandwidth. The sub-optimal decision directed and the optimal data aided implementation of the FFML1 TED show similar behaviour, although the magnitude and slope at the origin are less than for the Gardner and AD TED S curves. All TED's exhibit an output variance characteristic with a zero minimum at the optimum sampling phase. At this point none of the TED's suffer from self noise.

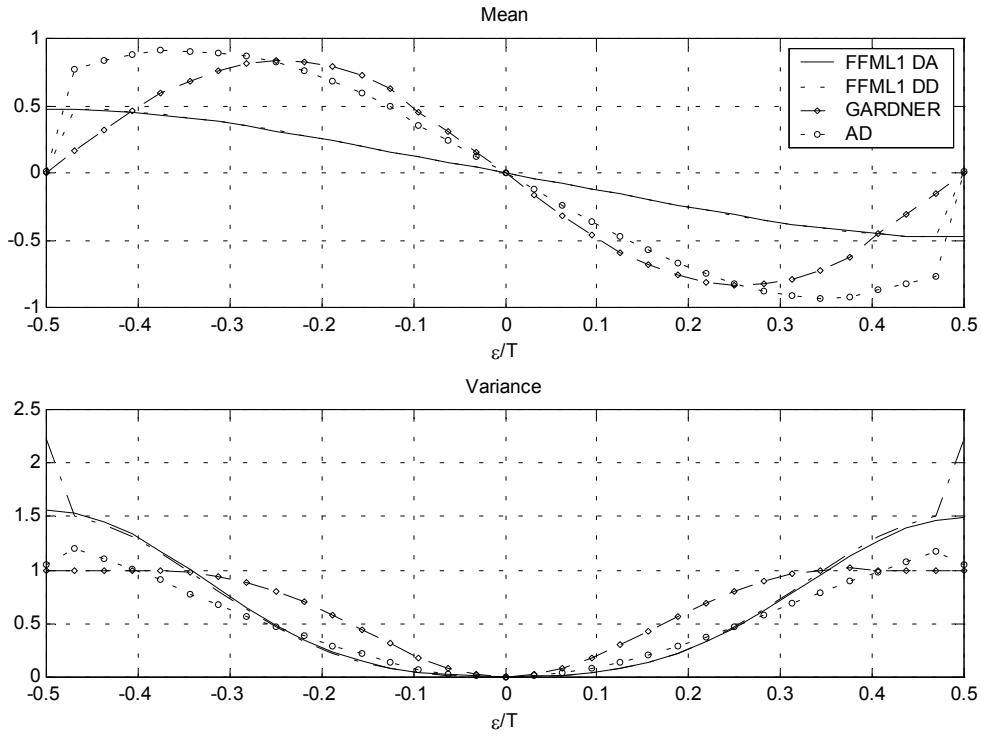

Figure 4.4 Reference mean and variance curves of each of the timing functions under investigation. In each case the modulation is QPSK, SNR = ∞ dB, α=1, and there is no fading.

The following few figures illustrate the effect of the pulse shapes excess bandwidth, α, on the statistical characteristics of each TED. In all cases the variance of the timing error produced by each TED decreases to zero at the optimum sampling time, but only for a full roll off raised cosine pulse shape, i.e $\alpha = 1$. As the pulse shapes excess bandwidth is reduced, the variance of the TED output, at the optimum sampling phase, becomes non zero. The exception to this is the FFML1 TED which maintains zero variance output for both the optimum (DA) and the sub-optimum (DD) versions.

The source of this self noise was discussed briefly in section 4.2.1. As may be noted from Figure 4.5, the Gardner TED mean curve magnitude is affected quite significantly by the reduction in the excess bandwidth.

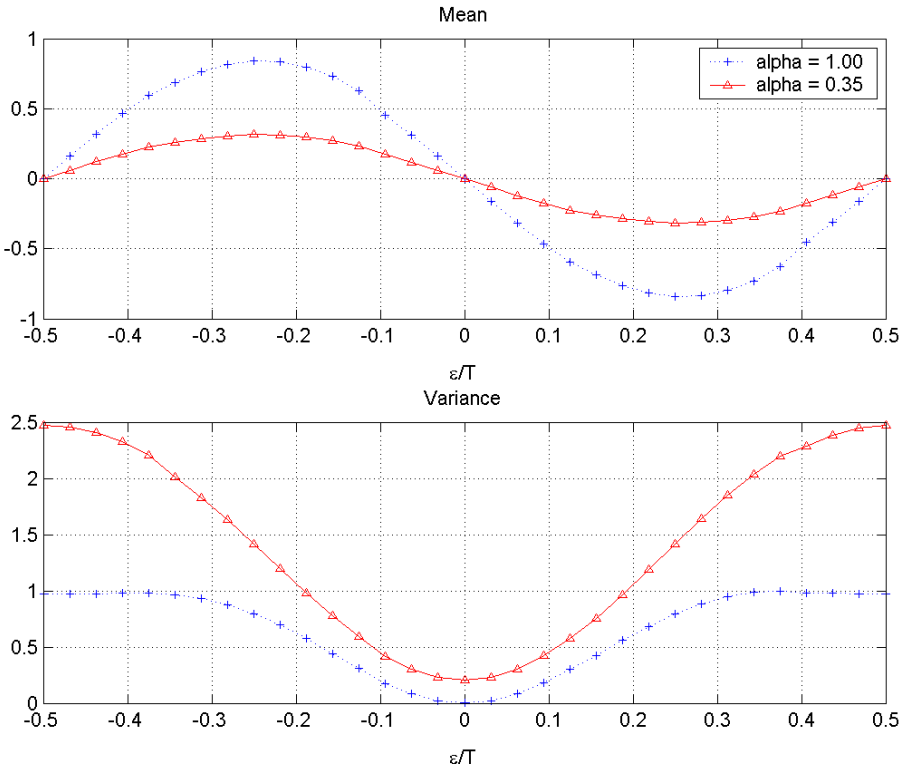

Figure 4.5 The effect of excess bandwidth on the mean and variance characteristics of the Gardner TED. Modulation is QPSK, SNR = ∞ dB, and there is no fading.

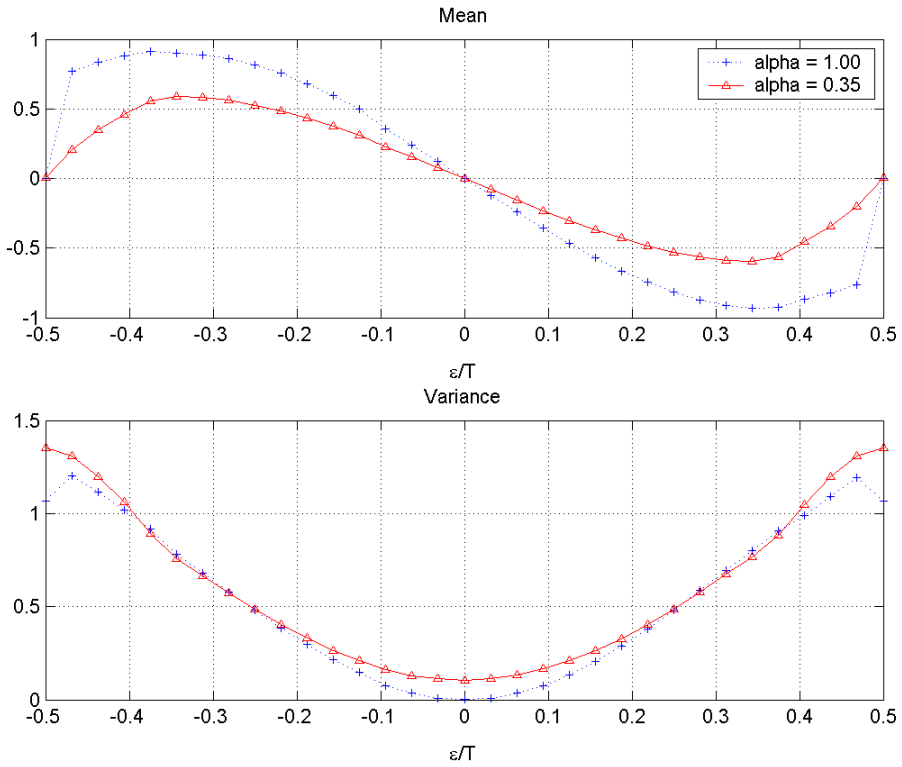

Figure 4.6 The effect of excess bandwidth on the mean and variance characteristics of the AD TED. Modulation is QPSK, SNR = ∞ dB, and there is no fading.

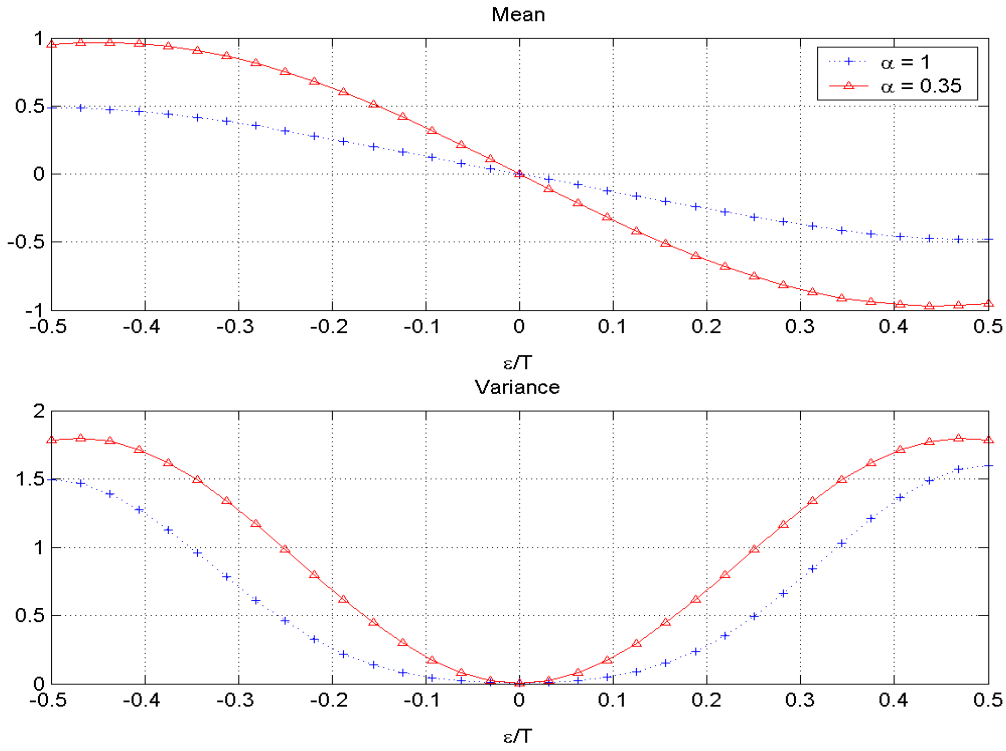

Figure 4.7**:** The effect of excess bandwidth on the mean and variance characteristics of the FFML1 DA TED. Modulation is QPSK, SNR = ∞ dB, and there is no fading.

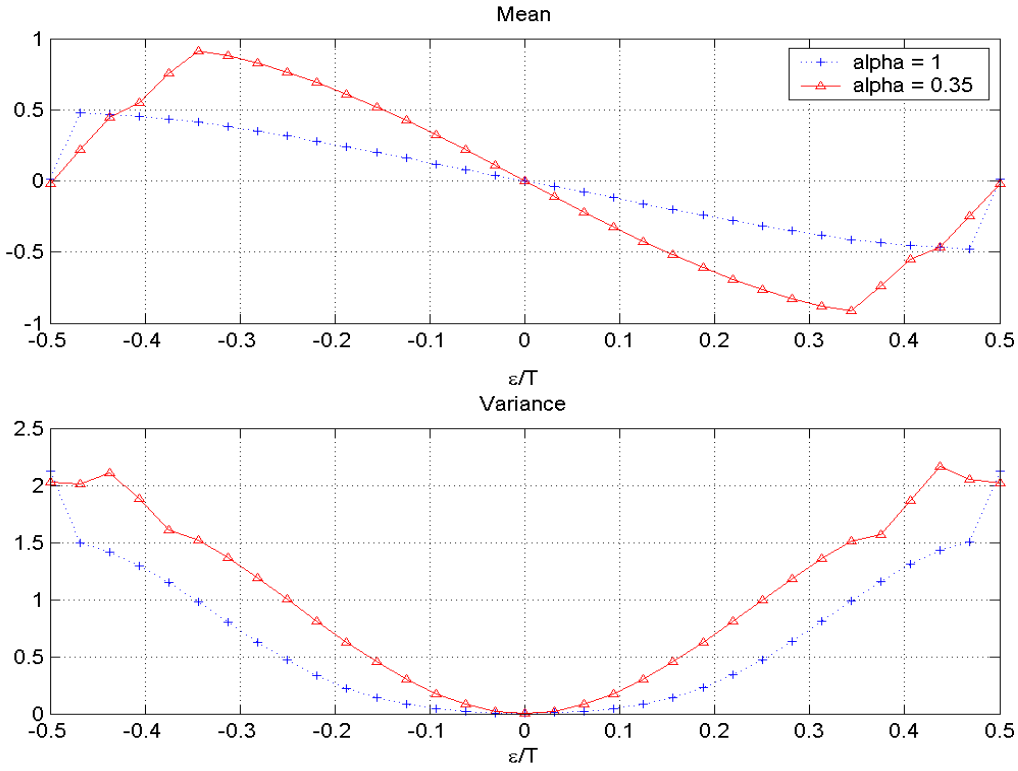

Figure 4.8 The effect of excess bandwidth on the mean and variance characteristics of the FFML1 DD TED. Modulation is QPSK, SNR = ∞ dB, and there is no fading.

A similar effect is seen in the case of the Amplitude Directed TED, although the proportional reduction in amplitude is not as great. The opposite effect is observable with the FFML1 TED's. The magnitude of the mean curve increases with decreasing excess bandwidth. This is an interesting characteristic and indicates that the FFML1 TED may be better suited to systems incorporating more severely band limited pulse shapes, as will be the case in most real systems. Another point of difference to note between the three TED mean curves is the timing error at which they begin to fall back to zero. In the case of the Gardner TED in Figure 4.5, this becomes apparent after approximately $\pm 0.25T$.

The AD TED mean curve maintains a significant magnitude until much closer to the sampling phase extremes of $\pm 0.5T$. This characteristic is helpful in speeding up acquisition of the associated synchroniser when the initial sampling phase is significantly offset from the ideal. This property is somewhat compromised by the reduction in excess bandwidth, although it still shows an improvement over the equivalent Gardner curve. In comparison, due to the data aided nature of the optimal FFML1 algorithm, illustrated in Figure 4.7, the TED does not exhibit a declining timing error mean as the sampling phase approaches $\pm 0.5T$. It is this property that would ensure the associated synchroniser is not affected by the hangup phenomenon that has been mentioned earlier in this work.

The output of the decision directed, sub-optimal version of the FFML1 TED, illustrated in Figure 4.8, like the Gardner and AD TED's, shows a decline in the mean error magnitude after approximately $\pm 0.35T$, in the worst case. This happens because, as the sample phase offset approaches $\pm 0.5T$, the number of symbol errors made by the receiver increases, eventually approaching 0.5. No useful timing information is being extracted from the decisions and the output of the timing function approaches zero.

4.3.1 Simulation Parameter Values

The parameter values that have been chosen for many of the simulations, have been done so with a typical target communications system in mind. Some of the system parameters are defined below.

The United States Federal Communications Commission (FCC) has allocated spectrum in the 2GHz band for wireless personal communication services (PCS). This is the primary reason behind the choice of 2GHz as a typical carrier frequency, used in the calculation of doppler spreads below.

Doppler spreads have been calculated for data rates of 80K baud and 500K baud, and mobile receiver velocities of 50km/h (typical) and 120km/hr. One should note, at data rates as high as 500K baud, in some environments where the RMS delay spread of the channel (see section 2.5.4.1) is significant, the channel may become frequency selective. This scenario is ignored here. The following table shows the normalised fade rates that are applicable in these cases.

Doppler Spread, f_D	80K baud	500K baud
93Hz (50km/h)	~0.001	~0.0002
222Hz (120km/h)	~0.003	~0.0004

Table 4.1 Typical normalised fade rates applicable to the defined communications system.

In some simulation results, higher fade rates are used to illustrate particular fundamental behaviours, or to explore the limits of the synchroniser performance where it may be useful to do so. The lower fade rates applicable to the 500K baud data rates are not treated in the simulations due to the extensive simulation times required and because the results add little value to those demonstrated by the somewhat higher fade rates more applicable to the 80K baud rate.

Two values of the pulse shape roll-off parameter, α, have been utilised. The first, $\alpha=1$, is used because it is a limiting value, and represents a 100% excess bandwidth. For two of the synchronisers presented it represents a 'best

performance' case. The second value is α=0.35. This has been chosen relatively arbitrarily, but is a figure that is more typical of a real world system, where bandwidth efficiency requirements make it unlikely that a value of 1.0 would be used. The FFML1 synchroniser on the other hand shows improving performance with decreasing excess bandwidths, as will be shown by the results presented in the next few pages.

Signal to noise ratio, or more accurately, the bit energy to noise density ratio, E_b/N_o, has been chosen to be 10dB for all simulations. This represents a symbol to noise density ratio, E_s/N_o, for QPSK, of approximately 13dB. In fading channel conditions instantaneous E_b/N_o and thus probability of error are constantly changing. All E_b/N_o and error rates in this case are average values.

4.3.2 Channel Fading and the TED Characteristics

The TED characteristic curves shown so far have all been in an ideal channel without noise and without fading. These conditions are useful for examining the underlying statistical characteristics of each of the TED's, however they are hardly representative of real world channel conditions. The effects of channel fading and noise on the characteristics are examined in the next few diagrams.

Figure 4.9 and Figure 4.10 demonstrate the effect of the Rayleigh fading channel on the Gardner TED statistical characteristics. Both were generated for the case of a signal to noise ratio (energy per bit to noise density ratio, E_b/N_o) of 10dB, with a roll off value of 1.0 for the former and a more realistic value of 0.35 for the latter. The non zero variance at the optimum sampling phase in Figure 4.9 is due entirely to AWGN. If the SNR is made very large then the variance at this sampling phase reduces to zero, for all fade rates, provided that the pulse shape roll off is 1.0. The effect of pulse shape roll-off on TED self noise has been mentioned earlier in this chapter. At sampling phases other than the optimum, the time variation of the channel produces greater variance in the TED output. The faster the channel variation the

greater the additional variance of the TED error output. As is to be expected, the mean is not affected significantly because the effect of fading is averaged out. Note also, that for the Gardner TED under fading conditions, the TED output variance is once again minimised at the optimal sampling phase when the pulse shape roll-off value, $\alpha = 1.0$. In fact this holds true for all sampling phase offsets from optimum. This characteristic was also evident in the noiseless, fadeless case illustrated in Figure 4.5, where the TED exhibited self noise for any value of α less than 1.0. The contribution to the TED variance due to fading reduces to zero in the full roll-off case, as is evidenced by the intersection of the variance curves for non-zero fade rates with that of the no-fading case, at the optimum sampling phase.

The equivalent curves for the AD TED, shown in Figure 4.11 and Figure 4.12, show that the AD TED is less affected by the fading channel. The TED output variance suffers only a minor increase as the channel changes from a non fading AWGN situation to a fade rate of 0.003. This increase is even less pronounced for the narrower bandwidth case of α=0.35. Unlike the Gardner TED case, the AD TED shows an almost zero contribution to the TED variance output from fading at the optimal sampling phase, for both values of α, and the output variance shows only a minimal departure from that for the no-fading case, across the entire sampling phase range. These results indicate that the AD TED will be more suited to operation in a fading channel environment than the Gardner TED. This will be confirmed by the synchroniser results presented in section 4.4.1.2.

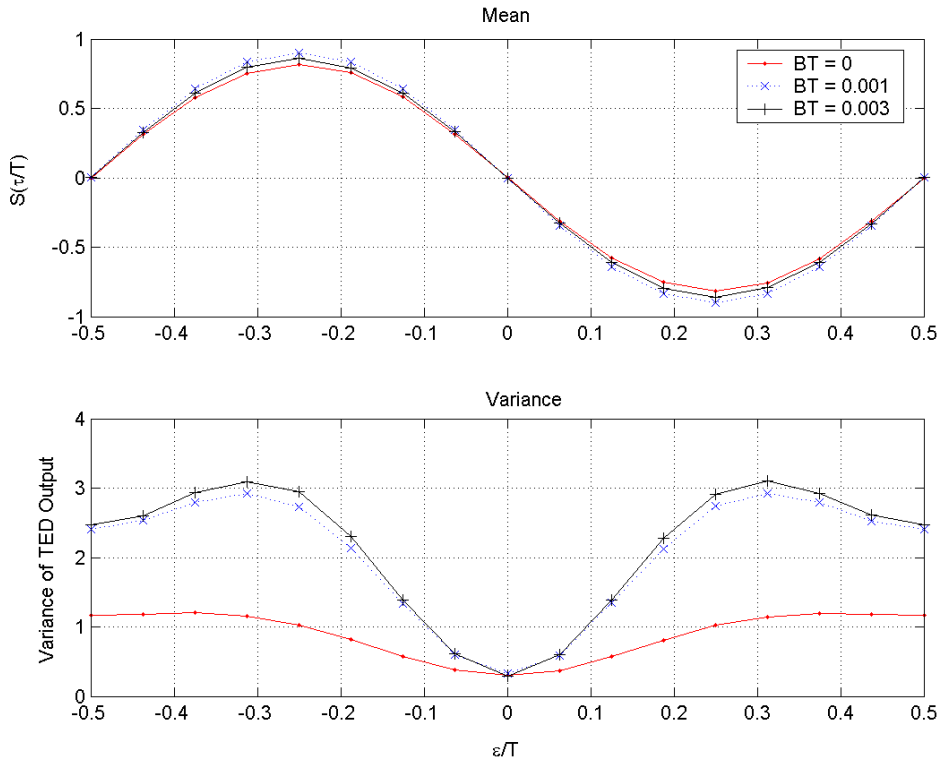

Figure 4.9 The effect of Rayleigh channel fading on the mean and variance characteristics of the Gardner TED. Modulation is QPSK, E_b/N_o = 10 dB, and α=1.0.

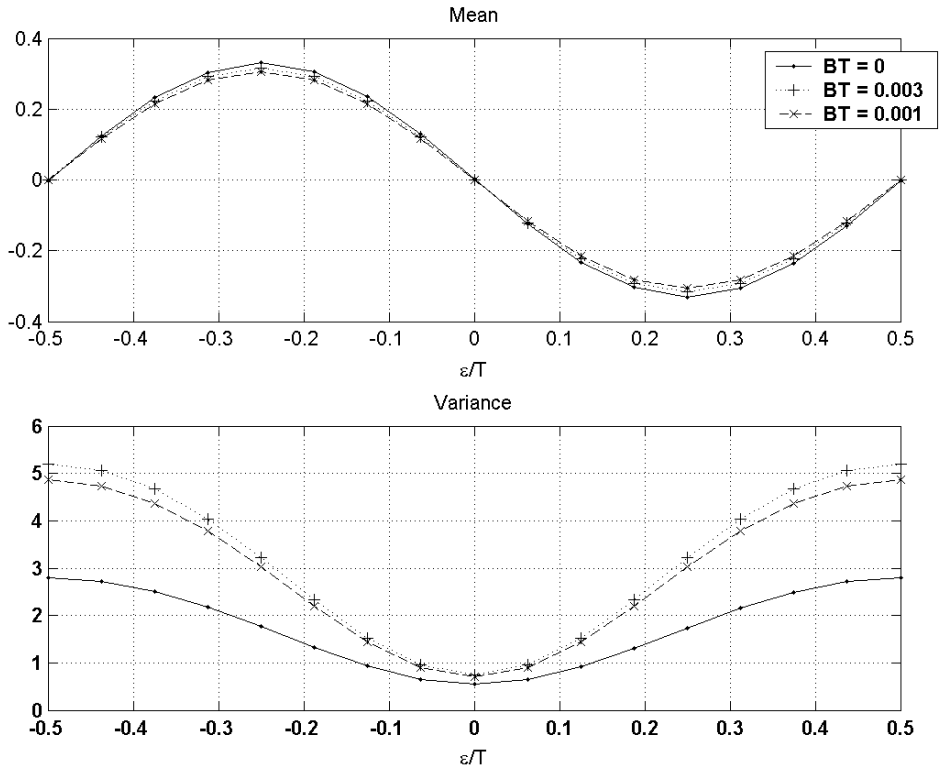

Figure 4.10 The effect of Rayleigh channel fading on the mean and variance characteristics of the Gardner TED. Modulation is QPSK, E_b/N_o = 10 dB, and α=0.35.

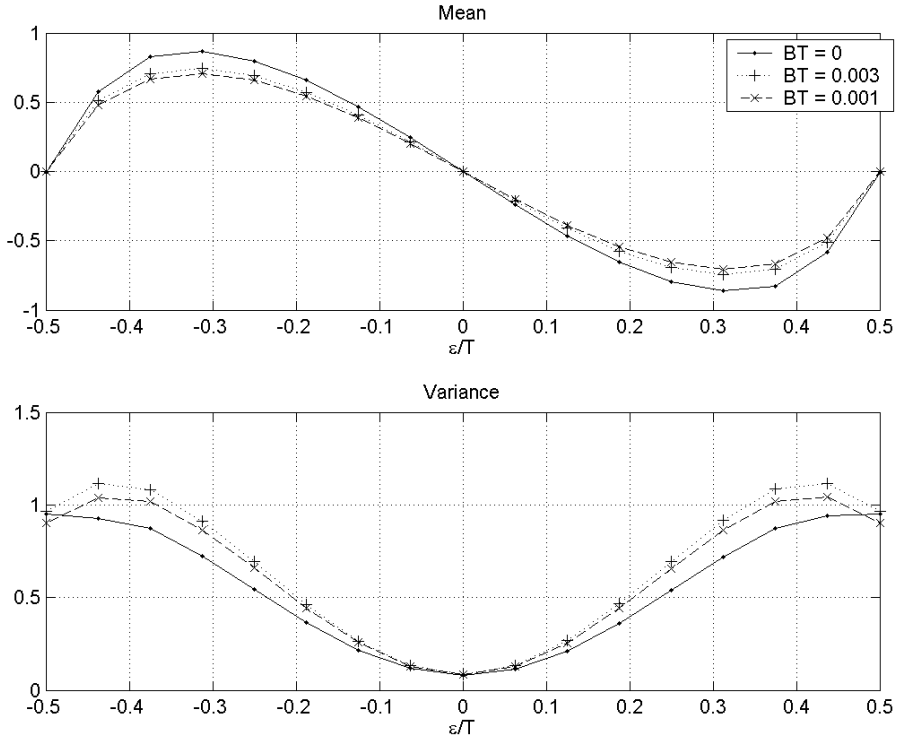

Figure 4.11 The effect of Rayleigh channel fading on the mean and variance characteristics of the AD TED. Modulation is QPSK, E_b/N_o = 10 dB, and α=1.0.

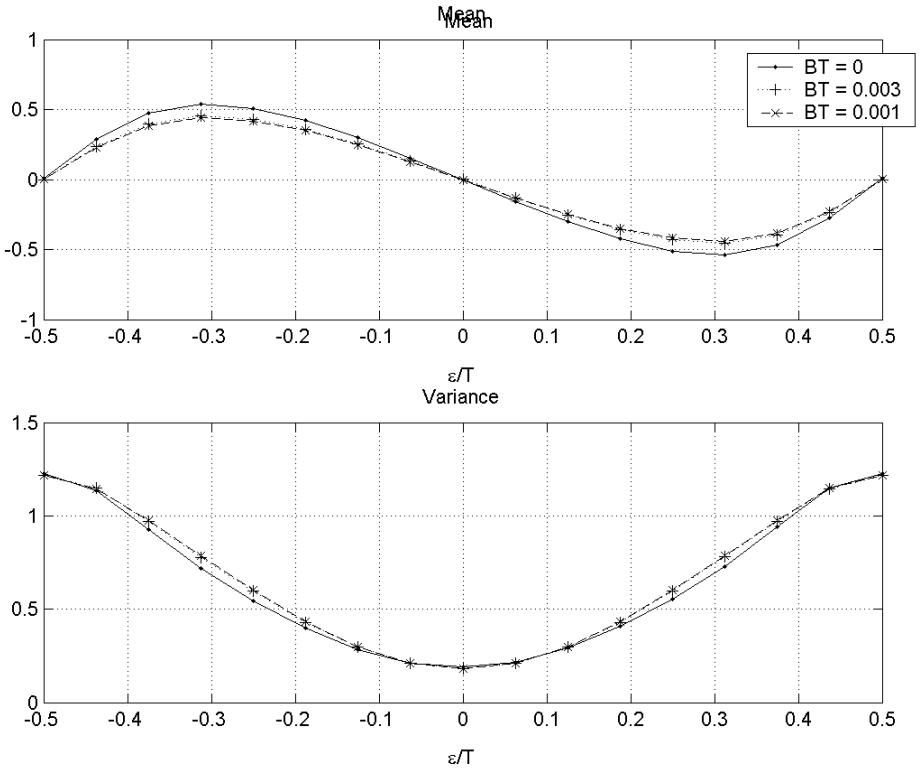

Figure 4.12 The effect of Rayleigh channel fading on the mean and variance characteristics of the AD TED. Modulation is QPSK, E_b/N_o = 10 dB, and α=0.35.

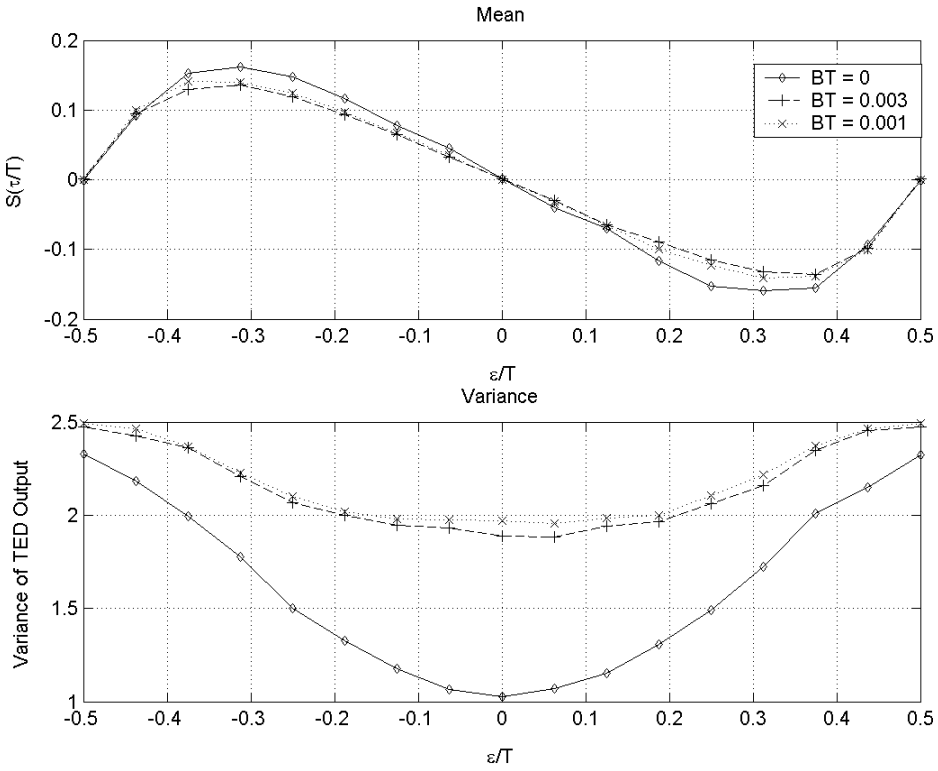

Figure 4.13 The effect of Rayleigh channel fading on the mean and variance characteristics of the FFML1 DD TED. Modulation is QPSK, E_b/N_o = 10 dB, and α=1.

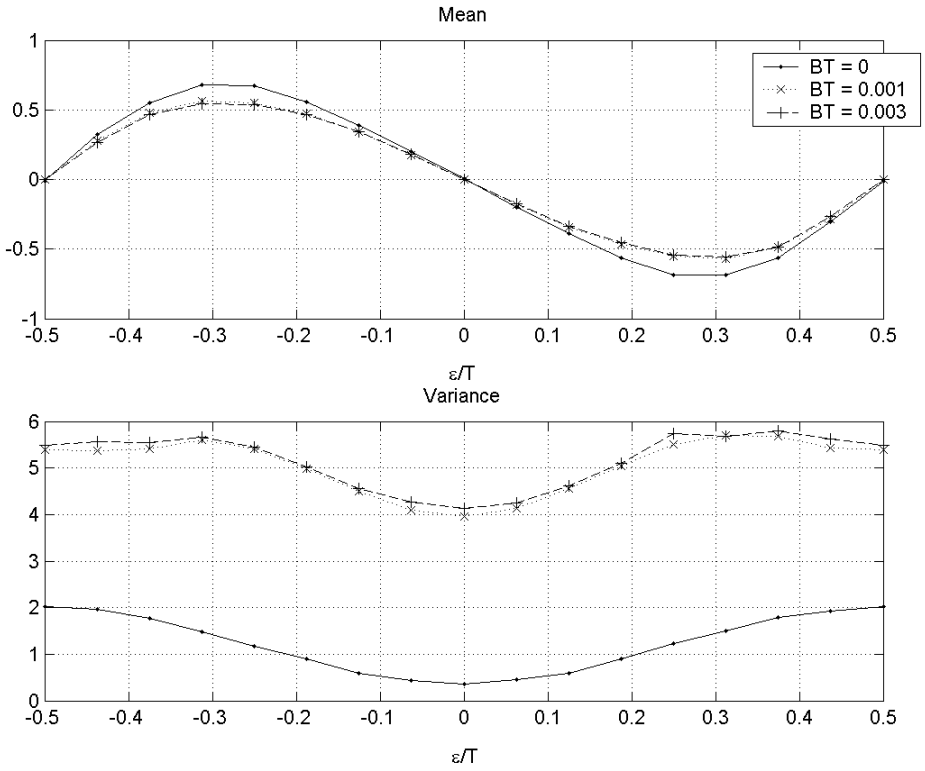

Figure 4.14 The effect of Rayleigh channel fading on the mean and variance characteristics of the FFML1 DD TED. Modulation is QPSK, E_b/N_o = 10 dB, and α=0.35.

In decision directed mode, the results of which are illustrated in Figure 4.13 and Figure 4.14, the Watkins FFML1 TED demonstrates a poorer variance performance than either the Gardner or the Amplitude Directed timing error detectors. As the sampling phase error approaches zero, the variance of the detector output does not reduce by as much as that seen in either of the other two TED's. This relatively poor performance is not entirely unexpected, because this sub-optimal version of the FFML1 TED relies on preliminary symbol decisions in making its error assessment. In the event that these decisions are unreliable, as in the case of fading corrupted symbols, the performance of a decision directed TED will suffer accordingly. Generally speaking, non data aided detectors should, in principle, do better than decision directed detectors when decisions are not available or are unreliable [36].

4.4 Performance Comparisons of Three Closed Loop Synchronisers

In the following few sub-sections, the performance of synchronisers based on the three timing error detectors under discussion will be examined. The general form of the synchroniser structure is illustrated by Figure 4.15. This diagram shows the sample rates at various points within the algorithm, labelled as x s/s (samples per symbol). The integrator block performs the role of the VCO in the phase locked loop structures introduced in section 3.4, summing the filtered error terms from the timing error detector (TED) block to produce timing control signal terms { τ, μ } which adjust the timing of the output symbol samples, produced at 2 samples per symbol, each half a symbol period apart.

Simulation results showing acquisition trajectories and tracking variance behaviour will be presented primarily for first order synchronisers, however results will be presented for a second order synchroniser based on the FFML1 TED as an example of the benefits that may be gained by using a second order implementation. A brief introduction to phase locked loop concepts has been presented earlier in this thesis (Chapter 3) and will not be repeated here. For all of the simulation results presented here, parameter values have been

chosen according to the reasons explained in section 4.3.1, and the modulation is QPSK in all cases.

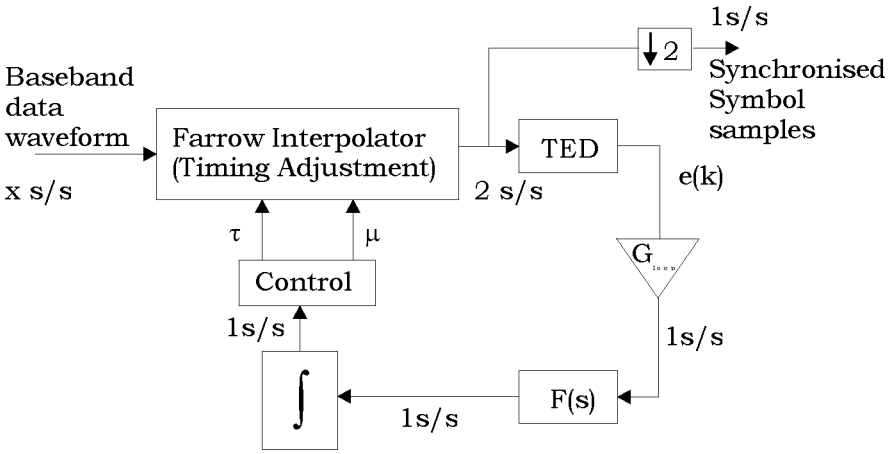

Figure 4.15 General structure of the closed loop synchroniser used in the simulations.

4.4.1 First Order Loop

As previously shown in section 3.4.3, the first order loop is defined by a constant gain loop filter, F(s) = G, or in terms of the defined structure of Figure 4.15, where a separate constant gain factor of G_{loop} has been defined external to the loop filter, F(s) = 1.

4.4.1.1 Acquisition – Ideal Conditions

If we assume that the initial timing error is relatively large (the maximum being half a symbol period), then it is during the acquisition phase that most PLL structures are operating in the non-linear portion of their timing error detectors characteristic S curve. Some representative plots of the acquisition trajectories of each of the three timing synchronisers, have been generated by computer simulation and are presented in the following pages.

The first two acquisition trajectory plots, illustrated in Figure 4.16 and Figure 4.17 show the fundamental acquisition performance differences, under ideal conditions, (no fading and no noise) between the first order synchronisers employing each of the three TED algorithms.

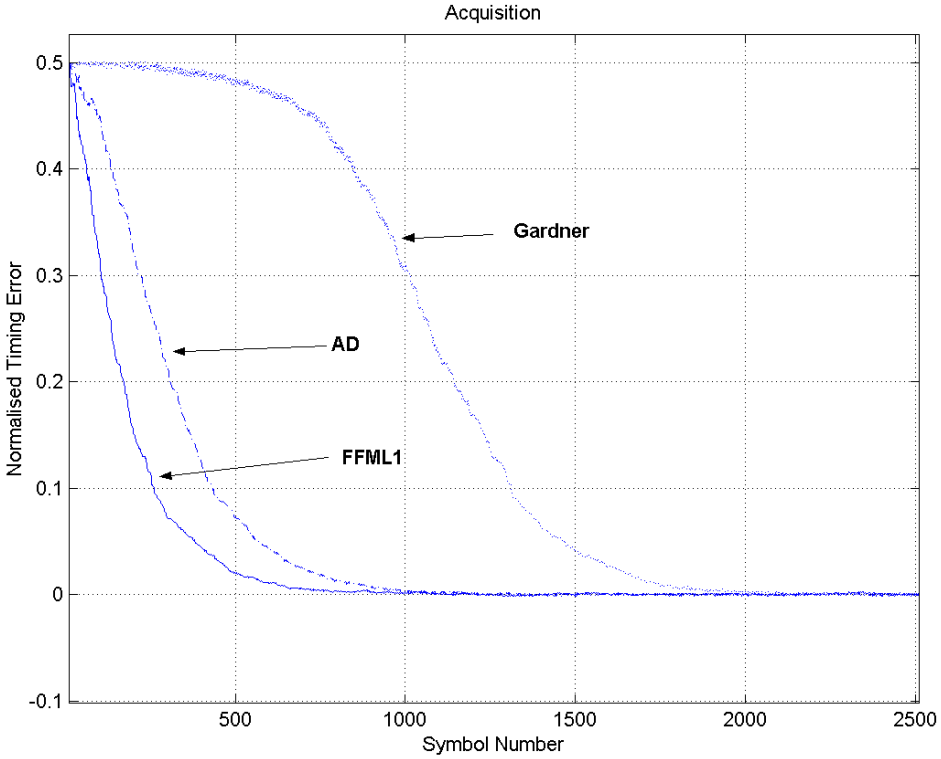

Figure 4.16 Noiseless acquisition trajectories for the Gardner, AD, and FFML1 first order synchronisers. In each case the open loop gain, G, is 0.01, α = 0.35, and there is no fading.

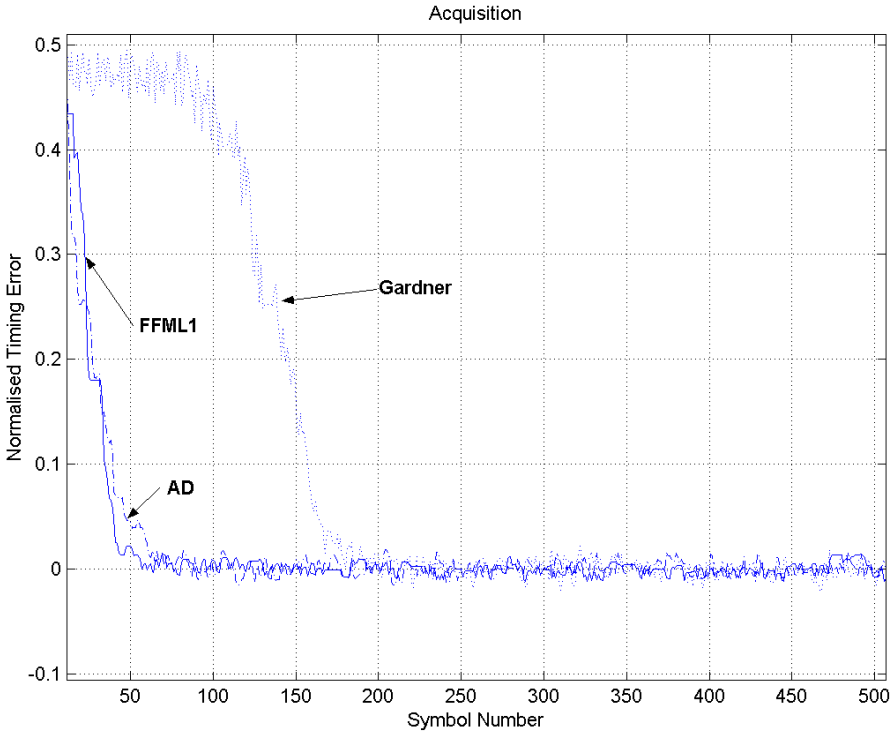

Figure 4.17 Noiseless acquisition trajectories for the Gardner, AD, and FFML1 first order synchronisers. In each case the open loop gain, G, is 0.1, α = 0.35, and there is no fading.

Acquisition has been shown for two different open loop gain cases, note the differing time axis scales between the two plots. As expected from standard phase lock loop theory, the greater the open loop gain (and hence bandwidth, in the case of a first order loop) the faster the acquisition, though this will be at the expense of poorer tracking variance performance. This is easily visible in Figure 4.17 where the timing estimate in 'lock' mode is significantly more noisy than for the lower open loop gain case illustrated in Figure 4.16. Because these plots illustrate the ideal case where $E_b/N_o = \infty$ dB, the noise apparent in the figures is 'self-noise' due to the random pattern nature of the data symbols.

The term 'lock' will be defined for the purposes of this and all subsequent discussion, as the state where the normalised timing error has reduced to less than or equal to 0.05.

	Open Loop Gain G = 0.01		**Open Loop Gain** **G = 0.1**	
Synchroniser	α = **0.35**	α = **1.0**	α = **0.35**	α = **1.0**
Gardner	1460	665	163	64
AD	560	295	46	33
FFML1	380	760	40	85

Table 4.2 First order synchroniser lock times in symbol periods for the noiseless, and zero fading case, for two open loop gains, and two values of symbol pulse shape roll-off, α.

Additional acquisition trajectories were generated for the case where α, the pulse shape excess bandwidth factor, has a value of 1.0. As has been noted in section 4.3, the properties of the Gardner and Amplitude Directed TED's relevant to synchroniser acquisition and tracking performance, improve with increasing excess pulse bandwidth, while those of the FFML1 TED tend to worsen. These properties include the slope of the S-curve at the origin, the normalised timing error at which the S-curve begins to fall back towards zero, and the variance of the TED error signal at the optimum timing estimate. These trajectory plots have not been presented here due to space considerations , however Table 4.2 summarises and compares the lock times,

in symbol periods, for each of the synchronisers, for the cases where α = 0.35 and 1.0, and open loop gain, G, equals 0.01, and 0.1. From the table it is apparent that the Gardner and the AD synchronisers both experience a reduction in lock time as α increases from 0.35 to 1.0, while on the other hand, the FFML1 synchroniser shows an increase in lock time. This behaviour is consistent with the effect of α on the properties of their respective TED's, readily apparent from Figure 4.5, Figure 4.6 and Figure 4.8. In the case of the Gardner and AD TED's, increasing α increases the slope of the S-curve at the origin, while also increasing its magnitude at larger normalised timing offsets. The increased slope results directly in an increased open loop gain of the corresponding synchroniser, for small timing offsets. This helps to speed timing acquisition. The larger S-curve magnitude at normalised timing offsets nearer to $\pm T/2$ tends to reduce dwell times of the associated synchroniser. This is directly related to the *hangup* phenomenon that many closed loop synchronisers tend to suffer from, and is briefly described in section 3.3.5.

4.4.1.2 Acquisition – Non-Ideal Conditions

The acquisition characteristics of the three synchronisers under ideal conditions, as presented in the previous section, are of fundamental theoretical interest. However the primary focus of this thesis is their performance under conditions that more closely approximate real world conditions. This section presents simulation results for the acquisition behaviour of each of the synchronisers under fading channel and additive noise conditions.

The effect of the pulse shape excess bandwidth, α, on the acquisition behaviour of each of the synchronisers, under ideal conditions, has been established in the preceding section. Because simulations have shown that this effect continues to hold true in the fading channel case, the following trajectory plots have all been produced using only a single value of α.

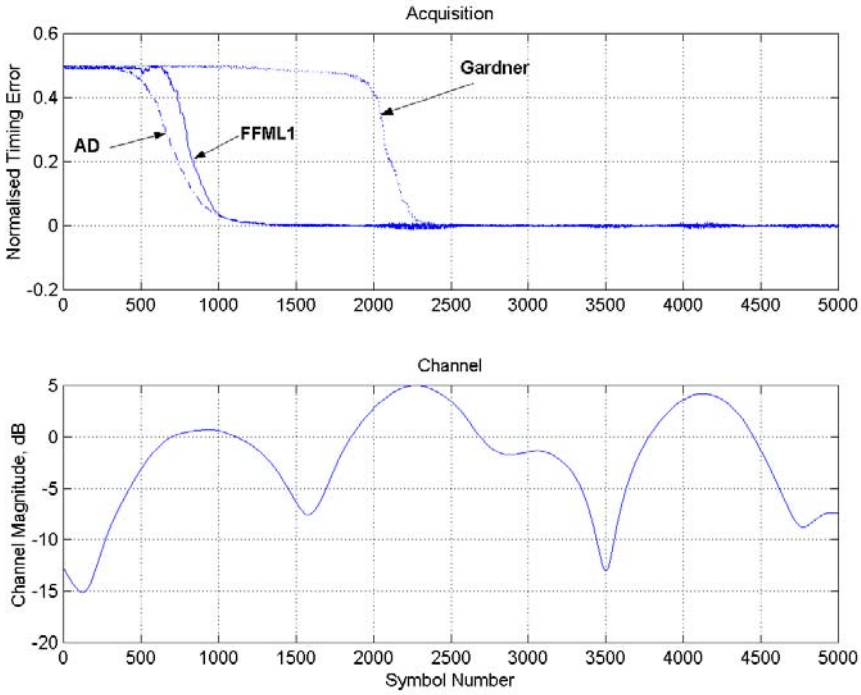

Figure 4.18 Noiseless acquisition trajectories for the Gardner, AD, and FFML1 first order synchronisers. In each case the open loop gain, G, is 0.01, $\alpha = 0.35$, and the fade rate is $f_DT = 0.001$. The bottom plot illustrates the magnitude of the fading channel during the same time interval.

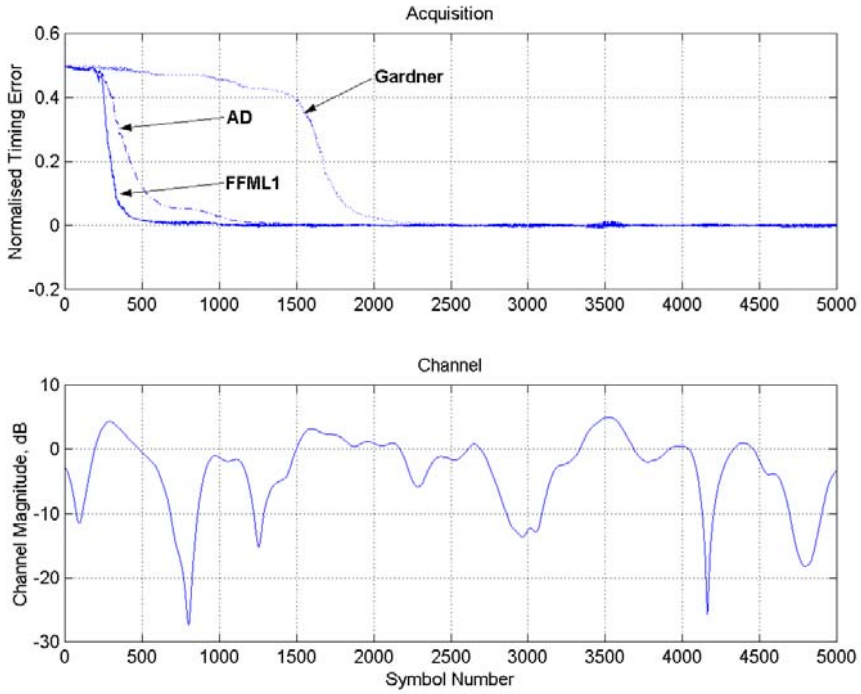

Figure 4.19 Noiseless acquisition trajectories for the Gardner, AD, and FFML1 first order synchronisers. In each case the open loop gain, G, is 0.01, $\alpha = 0.35$, and the fade rate is $f_DT = 0.003$.

The effect of the multiplicative, slowly fading channel on the acquisition trajectories of each of the synchronisers, in the noiseless case, is illustrated by Figure 4.18 and Figure 4.19, for normalised fade rates of 0.001 and 0.003. In the presence of slow fading each of the synchronisers is still quite capable of acquiring and tracking symbol timing. The first thing to note is that the time to lock for each synchroniser in the more slowly fading channel ($f_DT = 0.001$) is generally longer than for the faster fading channel ($f_DT = 0.003$). One should be careful not to read too much into the result, as it is specific to this particular simulation, and the channel state that prevailed in this case. In the $f_DT = 0.001$ instance, the channel magnitude started off at a relatively low value and remained small for a significant period of time (below 0dB for approximately 650 symbol periods). This reduces the acquisition bandwidth of the synchronisers during this interval, acting to slow lock. The greatest contributor to the increased lock time is the lengthened dwell time at large timing error offsets. On the other hand, in the case of the faster fading channel, the magnitude does not remain at low levels for so long (below 0dB for approximately 200 symbol periods), and actually increases to greater than 0dB for the next 300 symbol periods. During this time the bandwidth (or open loop gain) of the synchronisers are increased thereby speeding the lock process. If this simulation comparison was to be repeated a large number of times, each with a different channel 'snap shot' , the lock times of the synchronisers experiencing the faster fading channel conditions will, on average, be less than those experiencing the slower fading channel. The reason is relatively intuitive given the preceding explanation, however to clarify, simulation has shown that synchroniser dwell times are generally reduced for faster fading conditions. This is primarily due to the probability of the channel magnitude being in a fade (below 0dB) for a time that is significant compared with the natural lock time of the synchroniser (in ideal conditions), being smaller for faster fading channels than for slower fading channels. This point is highlighted further by the simulation results shown in Figure 4.20 for the relatively fast fade rate of $f_DT = 0.01$. This has been presented merely to illustrate a point, not because this fade rate is in any way realistic for the type of system under consideration here (see section 4.3.1).

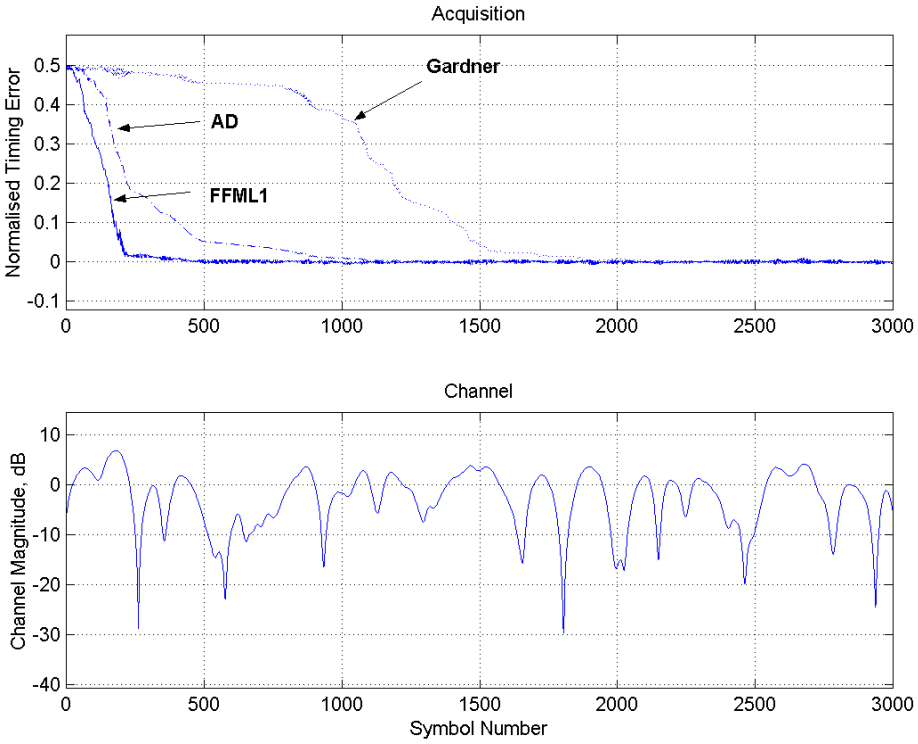

Figure 4.20 Noiseless acquisition trajectories for the Gardner, AD, and FFML1 first order synchronisers. In each case the open loop gain, G, is 0.01, $\alpha = 0.35$, and the fade rate is $f_D T = 0.01$.

Note the change in x axis scale for Figure 4.20 compared with the two previous figures. It is also more obvious in this illustration that the synchronisers make the greatest progress (steeper slope of trajectory) towards the optimum timing estimate during periods of larger channel magnitude and 'slow' down during fades, when the open loop gain of the synchronisers is suppressed.

Introduction of additive noise into the synchroniser simulations does not change the results dramatically. An E_b/N_o value of 10dB has been used in each case presented below. In the case of the Gardner synchroniser the addition of the noise actually helps to reduce the lock time slightly for each of the fade rates presented. The reason for this may be due to the fact that the Gardner synchroniser suffers from the hangup phenomenon to a greater extent than either the AD or FFML1 synchronisers. This is entirely due to the characteristics of its S-curve, particularly at the extreme sampling phase offset errors of $\pm T/2$.

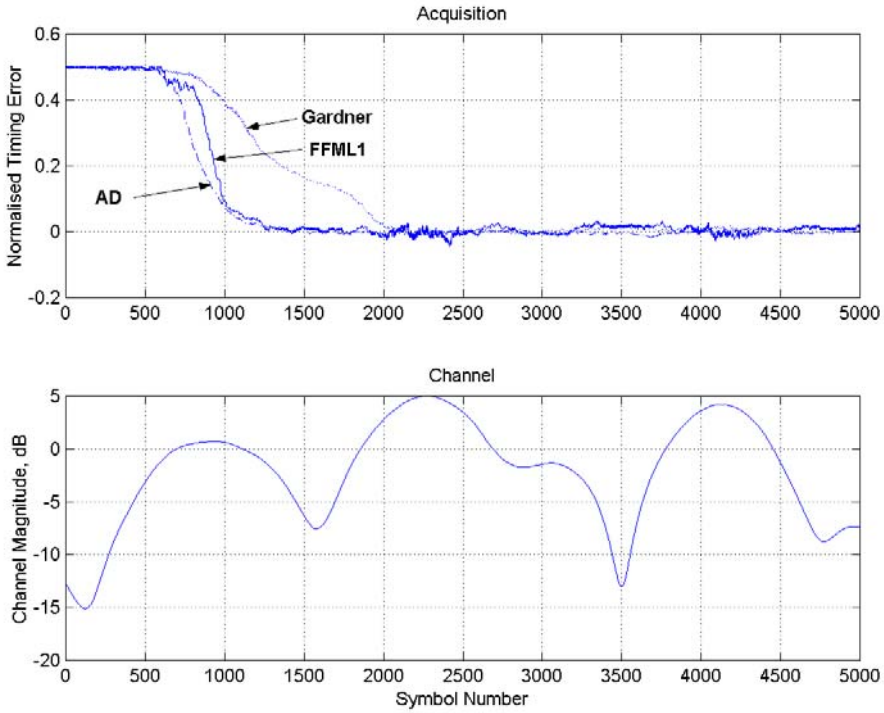

Figure 4.21 Acquisition trajectories for the Gardner, AD, and FFML1 first order synchronisers, in fading, f_DT = 0.001, and noise, E_b/N_o = 10dB. In each case the open loop gain, G, is 0.01, and α = 0.35.

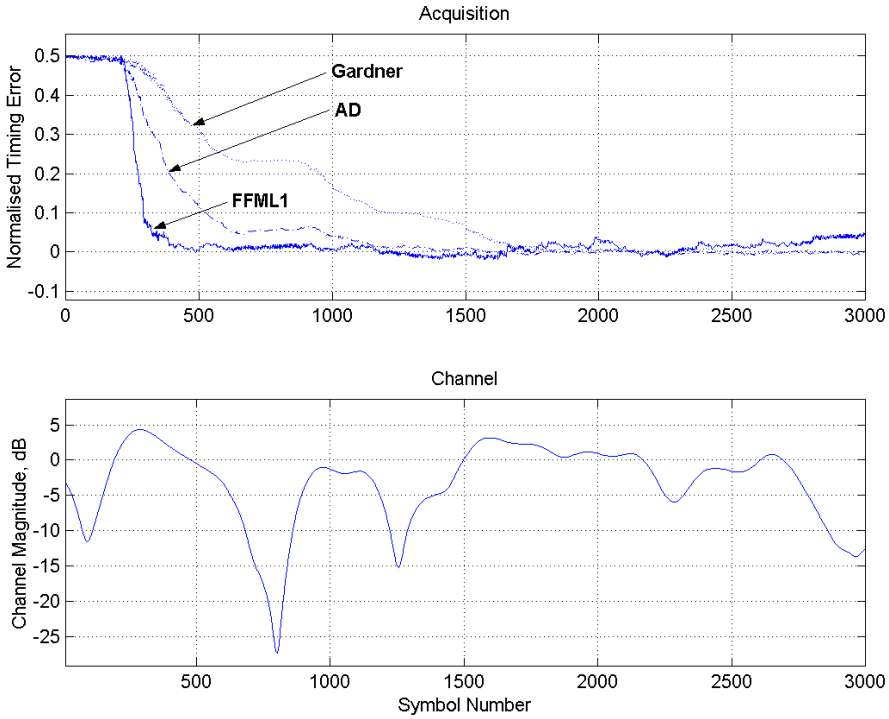

Figure 4.22 Acquisition trajectories for the Gardner, AD, and FFML1 first order synchronisers, in fading, f_DT = 0.003, and noise, E_b/N_o = 10dB. In each case the open loop gain, G, is 0.01, and α = 0.35.

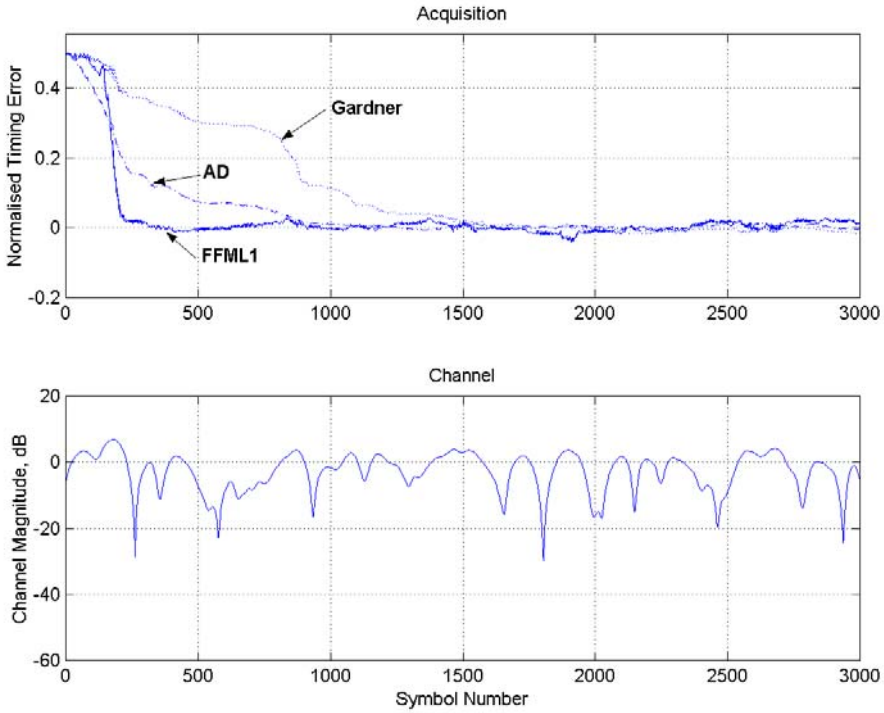

Figure 4.23 Acquisition trajectories for the Gardner, AD, and FFML1 first order synchronisers, in relatively fast fading, $f_DT = 0.01$, and noise, $E_b/N_o = 10$dB. In each case the open loop gain, G, is 0.01, and $\alpha = 0.35$.

The Gardner S-curve (refer to Figure 4.10) has a smaller magnitude toward the extremes than either the AD or FFML1 TED's. This acts to reduce the 'force' driving the loop toward lock, thereby extending the dwell time of the synchroniser at large sampling phase errors. This is quite evident in all of the trajectories presented thus far. The effect of noise on the lock-in process of each of the synchronisers under fading conditions is summarised for select cases by Table 4.3.

Because this dwell time is the major contributor to the extended lock times experienced by the Gardner synchroniser, it seems possible that the introduction of noise may act to reduce the lock time by increasing the probability that a noise event, or events will possess sufficient energy to 'kick' the synchroniser away from the extreme sampling phase error, thereby moving it further 'up' the S-curve to larger mean values of the TED error output, and hence to faster acquisition.

f_DT	0.001		0.003		0.01	
E_b/N_o dB	∞	10	∞	10	∞	10
Gardner	2228	1925	1865	1540	1480	1200
AD	940	1050	835	1000	550	775
FFML1	975	1120	380	320	200	200

Table 4.3 Comparison of the lock times, in symbol periods, for the three synchronisers, under fading conditions, with and without noise.

4.4.1.3 Gear Shifting

One method that can potentially be used to overcome the conflicting first order loop requirements of larger open loop gain for fast acquisition, smaller open loop gain for good tracking performance is to use 'gear shifting'. This is a method whereby the open loop gain of the loop is reduced once lock has been established, or after a predefined number of symbols have been received. An example of the potential benefit to this method is shown in the acquisition plots of each of the synchronisers, shown in Figure 4.24.

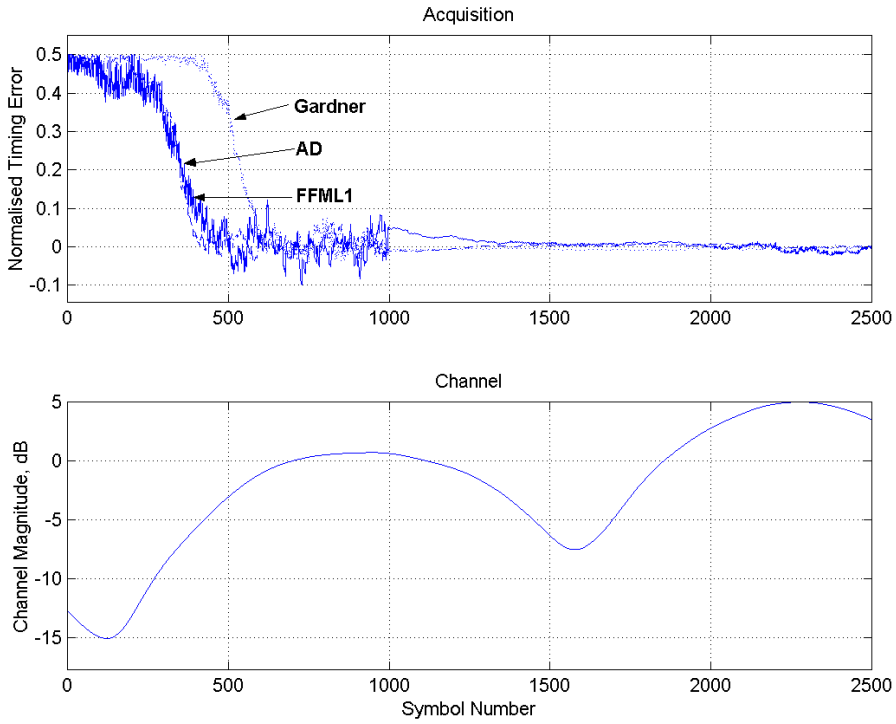

Figure 4.24 Acquisition trajectories for the Gardner, AD, and FFML1 first order synchronisers, in fading, $f_DT = 0.001$, and noise, $E_b/N_o = 10$dB, with $\alpha = 0.35$. The open loop gain, G, is reduced from 0.1 to 0.005 after 1000 symbols.

Comparing Figure 4.21 and Figure 4.24, the reduction in lock time is significant. A word of warning is appropriate at this point. The designer must be careful not to select the acquisition open loop gain to be too large otherwise the loop can become unstable. This is particularly true if the channel magnitude happens to be greater than 1 during the period of acquisition, because this will contribute to increase the open loop gain of the loop still further.

4.4.1.4 Tracking

In addition to the acquisition performance of the timing synchronisers, the tracking performance is of interest. This characteristic is of greater relevance to the digital receiver sub-system than the acquisition performance, when in normal, steady-state operation. In a first order loop implementation however, choices made for open loop gain to influence lock time, also impact the tracking performance of the synchroniser and vice versa, thus compromises must be made. This is one reason why second order loops can be very useful, as they provide an additional degree of freedom to the designer. Synchroniser implementations using second order PLL structures are examined briefly in section 4.4.2.

Note from Figure 4.26 how the FFML1 and Gardner synchronisers begin to exhibit a variance floor. The AD synchroniser only begins to show a variance floor at E_b/N_o values greater than 35 dB. This improved performance can be directly attributed to the better variance performance of the AD timing error detector, which was presented in section 4.3.2.

As expected, based on the statistical performance characteristics of the TED presented in section 4.3, the synchroniser using the sub-optimally operated FFML1 TED exhibits the poorest tracking variance performance due to the decision directed mode in which it is operated. Also, a comparison of Figure 4.25 and Figure 4.26 shows that the increase in the channel fade rate from 0.001 to 0.003 has the greatest impact on the Gardner synchroniser's tracking variance performance. The Amplitude Directed and the FFML1 based

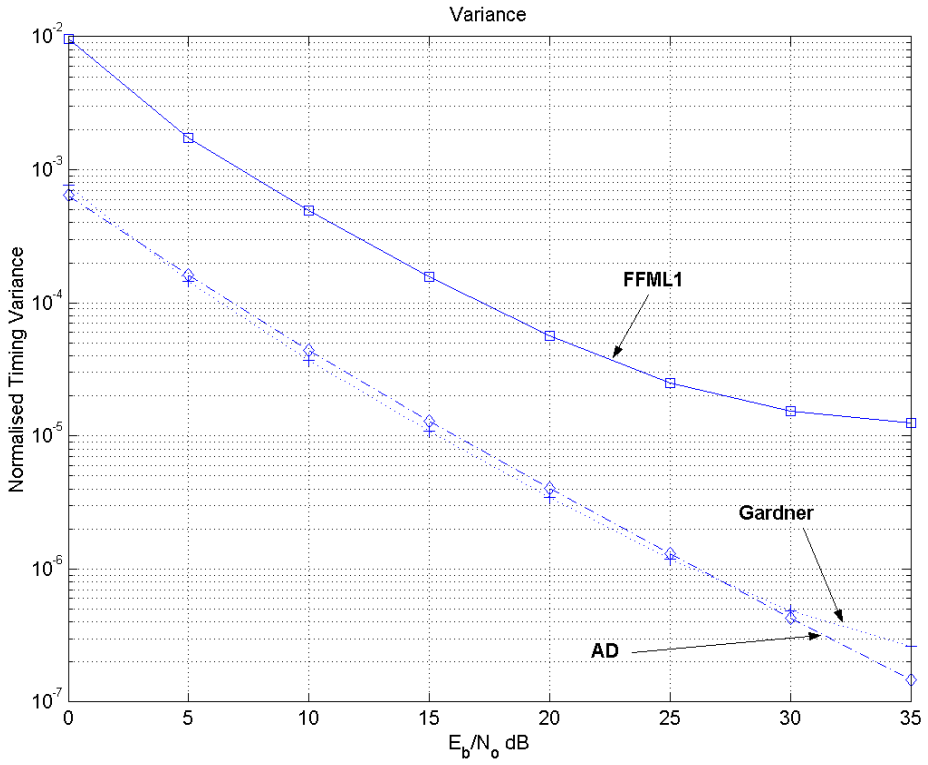

Figure 4.25 The timing estimate variance performance of each of the three synchronisers. In this simulation f_DT is 0.001, α is 1.0 and synchroniser open loop gain, G=0.01.

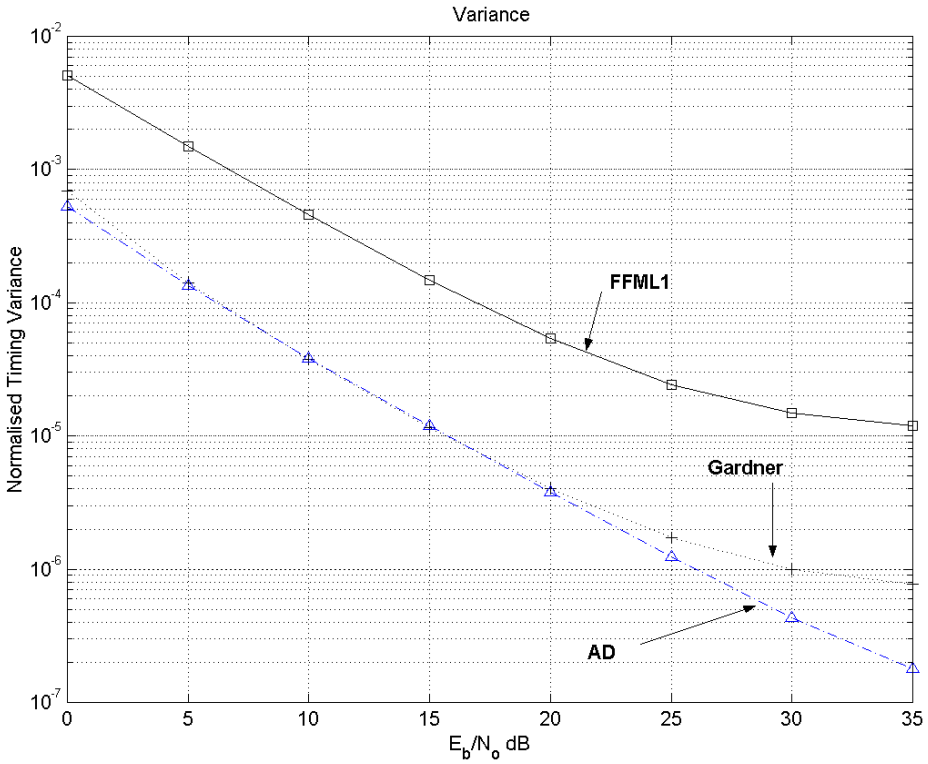

Figure 4.26 The timing estimate variance performance of each of the three synchronisers. In this simulation f_DT is 0.003, α is 1.0 and synchroniser open loop gain, G=0.01.

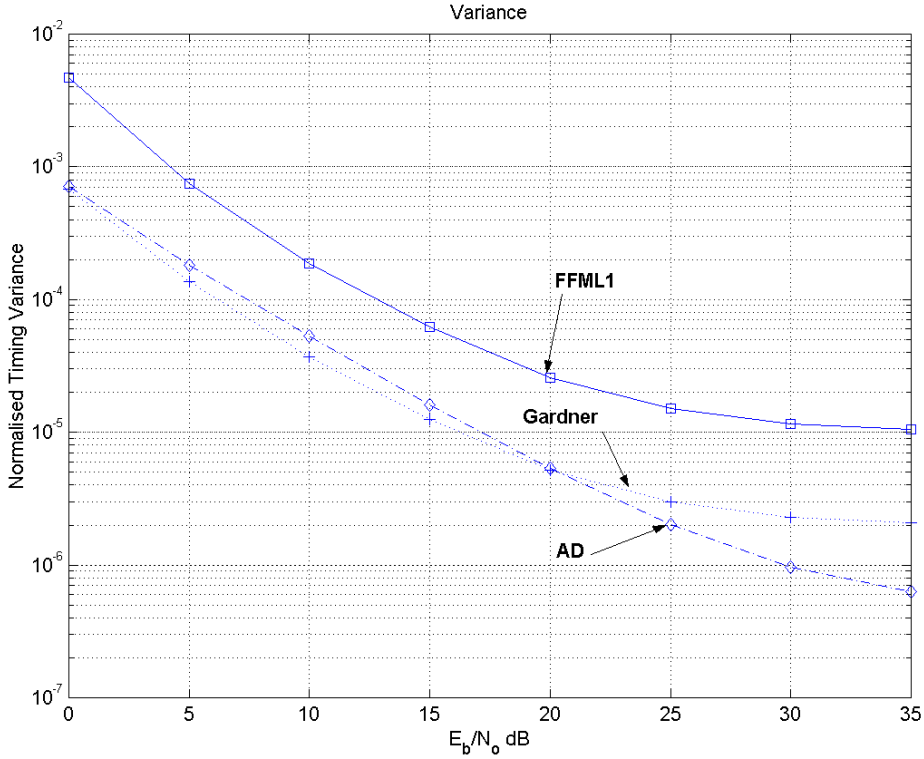

Figure 4.27 The timing estimate variance performance of each of the three synchronisers. In this simulation f_DT is 0.001, α is 0.35 and synchroniser open loop gain, G=0.01.

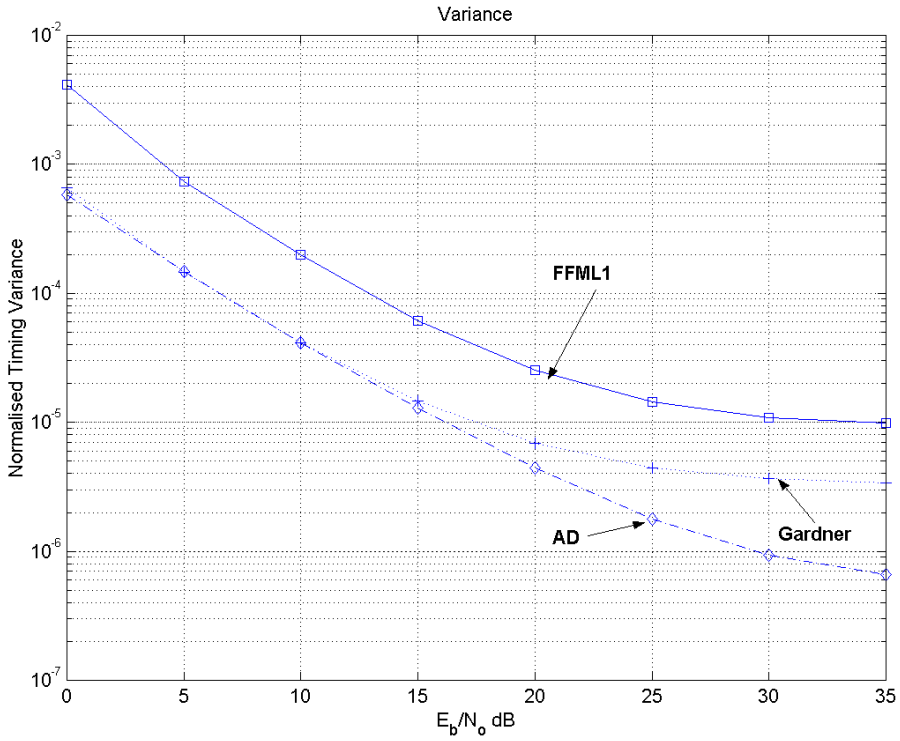

Figure 4.28 The timing estimate variance performance of each of the three synchronisers. In this simulation f_DT is 0.003, α is 0.35 and synchroniser open loop gain, G=0.01

synchronisers are almost unaffected. The poorer tracking variance performance of the FFML1 synchroniser is also quite visible in the acquisition plots presented earlier in this chapter.

The tracking variance performance of each synchroniser is affected by the value of the pulse shape excess bandwidth, α. This can be observed by comparing the variance curves of each of the synchronisers in Figure 4.25 and Figure 4.27, for a fade rate of $f_DT = 0.001$, and Figure 4.26 and Figure 4.28 for a fade rate of $f_DT = 0.003$.

This behaviour is as expected, since the output error variance of each associated TED has been shown earlier (see section 4.3.2) to either be improved or degraded by reducing α from 1.0 to 0.35. Changing the fade rate from 0.001 to 0.003 has a negligible impact on the tracking performance of any of the synchronisers when $\alpha = 0.35$.

4.4.2 Second Order Loop

In terms of the synchroniser structure illustrated in Figure 4.15, the only difference between the first order and second order synchroniser is the loop filter F(s), and the actual value used for G_{loop}. The loop filter has the form,

$$F(s) = \frac{s + a}{s + \lambda \cdot a} \tag{4.5}$$

The first order structure has F(s) = 1 for all frequencies. The second order structure has $F(s)=(1/\lambda)$ at DC, and rolling off at higher frequencies. This improves the smoothing of the error signal through the increased attenuation of higher frequency noise, reducing the tracking variance of the synchroniser. The discrete time implementation of this first order filter is briefly described in section 3.4.5.2.

To demonstrate the performance advantages of the second order synchroniser implementation over the first order synchroniser, simulation results for a second order synchroniser based on just one of the TED's is presented in this section.

The advantage to employing a second order synchroniser is not limited to its improved tracking variance performance. The second major advantage stems from a second order structure's ability to track the timing phase with very little steady state error when there is a constant frequency difference between the transmitter and receiver clocks. This ability is derived from the integrating effect of the extra pole provided by the loop filter.

In the acquisition and tracking variance performance comparisons of the first and second order synchroniser, the loop bandwidth has been made approximately the same. The actual PLL parameters used in the simulation that produced these results, with reference to the PLL model illustrated in Figure 4.15, and the discrete loop filter implementation described in section 3.4.5.2, were:

First Order: $G_{loop} = 0.01$

Second Order: $G_{loop} = 0.008$, $\lambda = 0.08$, $a = 0.35$.

4.4.2.1 Acquisition

The FFML1 TED algorithm has been used as the basis for the second order synchroniser whose acquisition and tracking performance results are presented in this and the next sub-section.

Figure 4.29 compares the acquisition trajectories of a first order and a second order implementation of the synchroniser employing the FFML1 algorithm as the timing error detector. The loop bandwidth of the synchronisers, under non-fading conditions, is the same. Because of this, and because the channel fading experienced by the synchronisers, illustrated by the bottom plot, is the same, the acquisition times are very similar. In the example shown here the loop filter of the second order synchroniser has been given a pole position of z=0.97. It is quite apparent that the normalised timing error output from the second order loop is 'smoother' than the corresponding output from the first order loop.

The second order loop has suppressed the higher frequency noise components in the estimate, and as a consequence the output will have lower variance. This improvement is gained at a very minor increase in complexity.

A second point of interest is the increase in the normalised timing error for both the first and second order loops during periods of large channel magnitude. This is particularly noticeable around symbol periods 1550-1650 and 3350-3450 in Figure 4.29. Although this may seem counterintuitive, in actual fact an increase in the channel magnitude can equivalently be viewed as an increase in the open loop gain of the timing PLL, if you consider the channel and the PLL as one system. This has the effect of temporarily increasing the bandwidth of the loop, and hence a reducing the noise immunity of the PLL.

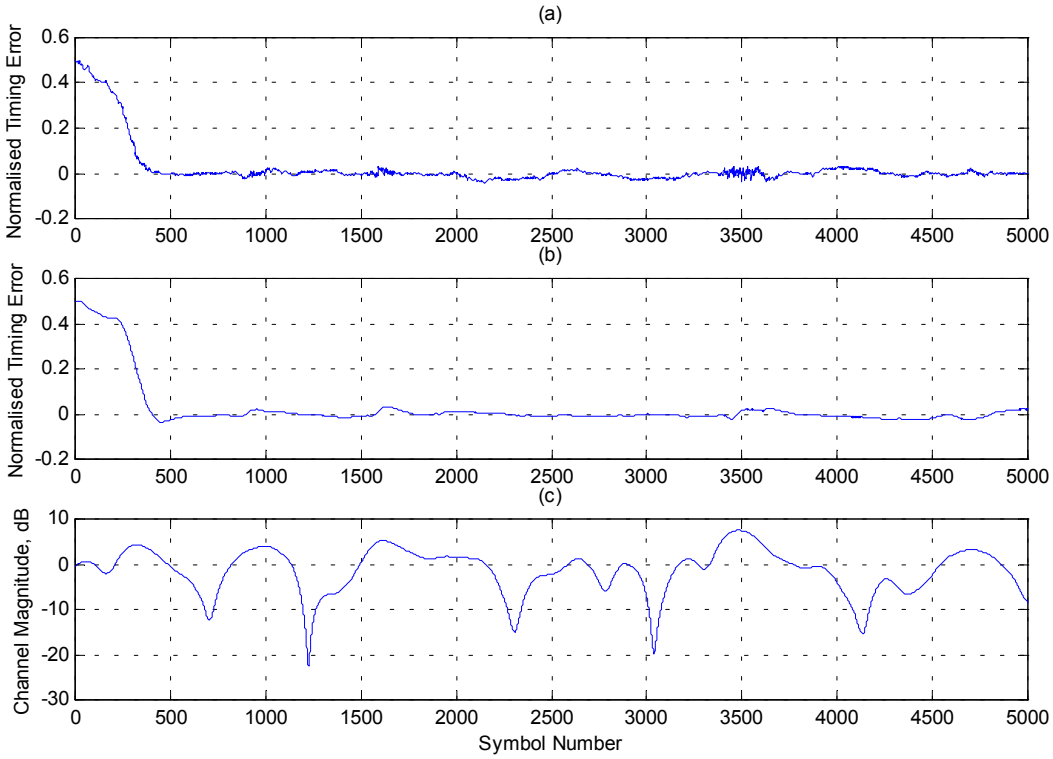

Figure 4.29 Example acquisition trajectories for (a) first and (b) second order synchronisers based on the FFML1 TED. Both loops have the same bandwidth. For the first order loop, G=0.01. The fade rate f_DT is 0.003, α is 0.35, and E_b/N_o = 10dB

4.4.2.2 Tracking

As expected from the comparison of the acquisition trajectories, Figure 4.30 illustrates an improvement in the second order synchronisers timing estimate tracking variance, particularly at higher values of E_b/N_o, where the contribution to the noise in the timing estimate is now dominated by the random time varying nature of the channel and its affect on the received signal. As for the acquisition case presented in the last section, this comparison is for two synchronisers having the same loop bandwidths, with $\alpha = 0.35$, and the modulation is QPSK, as in all previous cases.

At low E_b/N_o values the noise in the timing estimate is dominated by additive white noise, and because the loop bandwidths of the synchronisers are similar, they exhibit similar timing estimate variances. As the contribution to the noise in the synchroniser output from the fading of the signal becomes proportionately greater than that due to white noise, the second order loop shows an increasing performance advantage over that of the first order loop.

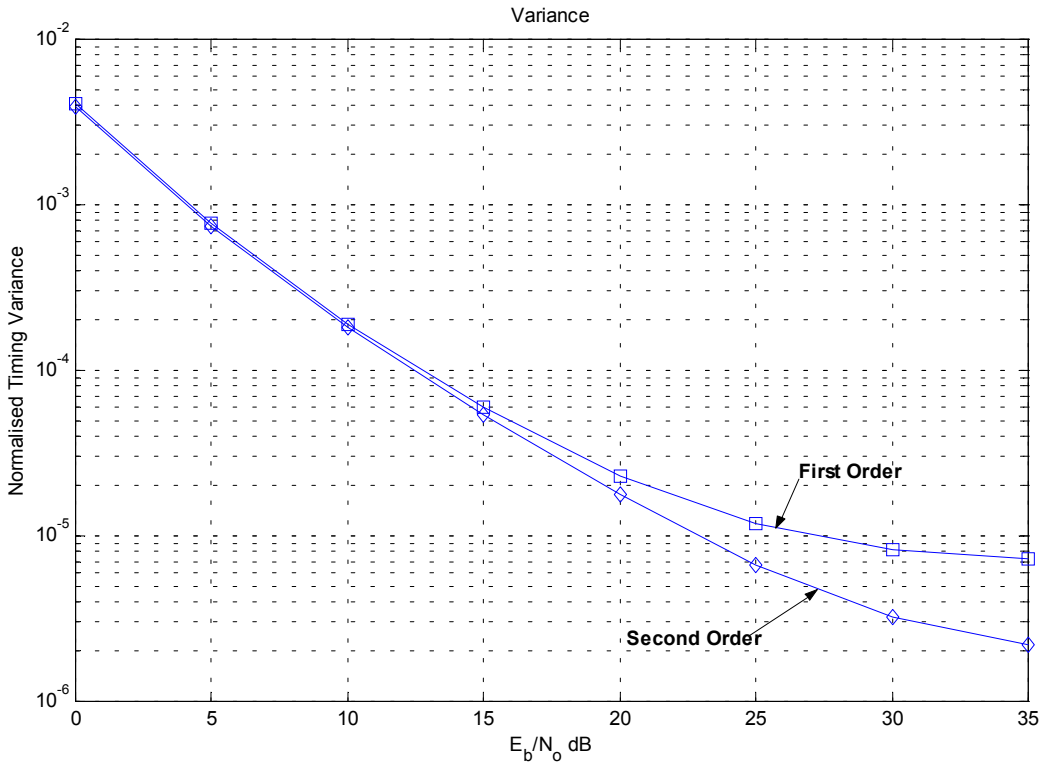

Figure 4.30 Comparison of the tracking variance performance of the first and second order synchronisers. Fading is $f_DT = 0.003$.

4.5 Summary and Conclusion

From the TED and associated synchroniser simulation results presented in this chapter, it is apparent that the Gardner synchroniser is the least suited to operation in a Rayleigh fading channel environment. It suffers from hangup to a greater degree than either of the other synchronisers, and hence has an acquisition time that is invariably longer, usually by a significant margin, than either the AD or the FFML1 TED based synchronisers. Its tracking timing estimate variance is also the most severely degraded by increasing channel fade rate. The AD and FFML1 TED's and associated synchronisers' tracking timing estimate variance is relatively insensitive to the fade rate. On the other hand, the performance of both the Gardner and AD TED are quite sensitive to the pulse shape excess bandwidth, α, and exhibit degraded performance, both in acquisition and tracking, as the bandwidth is restricted to values less than 100% roll-off, as would be the case in most real systems. The FFML1 TED and synchroniser exhibits the opposite behaviour, showing improved performance with reduced pulse shape excess bandwidth. The FFML1 based synchroniser actually showed the best acquisition performance, in terms of shortest times till lock, of the three synchronisers.

The FFML1's tracking error performance, though the poorest of the three, is still entirely adequate for inclusion in a QPSK receiver operating in a fading channel environment, with negligible impact on symbol error rate. If complexity of implementation is not an issue then a synchroniser based on the FFML1 TED would be a good choice for a QPSK mobile digital receiver. However, complexity usually is an issue in real world systems, and given the reasonable performance of the AD TED based synchroniser, and the fact that it can operate on multilevel modulations such as M-QAM, as well as QPSK, it would have to be recommended as being the most suitable TED on which to base a timing synchroniser for operation in a Rayleigh fading channel environment.

CHAPTER 5
A FADING CHANNEL QUASI COHERENT RECEIVER USING PSAM

5.1 Introduction

This chapter introduces a technique that has received considerable attention in the literature over the past decade or so [2, 4, 5, 51], as a means for mitigating the effects of a slow fading channel on the received signal. The technique is called pilot symbol assisted modulation (PSAM) and it is conceptually straightforward to understand. It has been reported [2] that the technique is adequate for channel fade rates up to approximately $f_DT=0.05$.

The basic concept revolves around the idea of sounding the time varying channel through the periodic insertion of known symbols into the data stream.

5.2 General PSAM Description

The transmitter periodically inserts symbols from a known sequence into the data symbol stream. The receiver has knowledge of these symbols and their location in the fading channel corrupted received symbol sequence. The receiver splits the received symbol stream into a data stream and a reference stream. The reference stream is composed of the received pilot symbols. These are used to form an estimate of the channel state (amplitude and phase) at each pilot symbol location. Estimates of the channel state at all data symbol locations are then made using interpolation. The use of interpolation to reconstruct the behaviour of the channel at all data symbol positions other than the pilot symbols pre-supposes that the channel has been 'sampled' at a frequency appropriate to the rate at which it is changing (That is, the Nyquist sampling criterion has been met and the underlaying continuous random fading process/signal can be perfectly reconstructed from the sampled signal). In effect this means that the spacing of pilot symbols must be chosen with

regard to the expected normalised doppler fade rate, f_DT, of the channel. Pilot symbols must be spaced more closely together for a channel experiencing a higher normalised fade rate than for one with a slower fade rate.

The accuracy with which the fading estimation is made is dependent on the type and order of interpolator used [4, 52, 53], and also on the signal to noise ratio of the received pilot symbols. Because the received pilot symbols will have been corrupted with AWGN as well as the complex multiplicative distortion of the Rayleigh fading channel, calculation of the channel sample will necessarily result in a noisy estimate of the channel state.

Figure 5.1 shows a block diagram of a typical PSAM system, and its associated data and pilot frame structure.

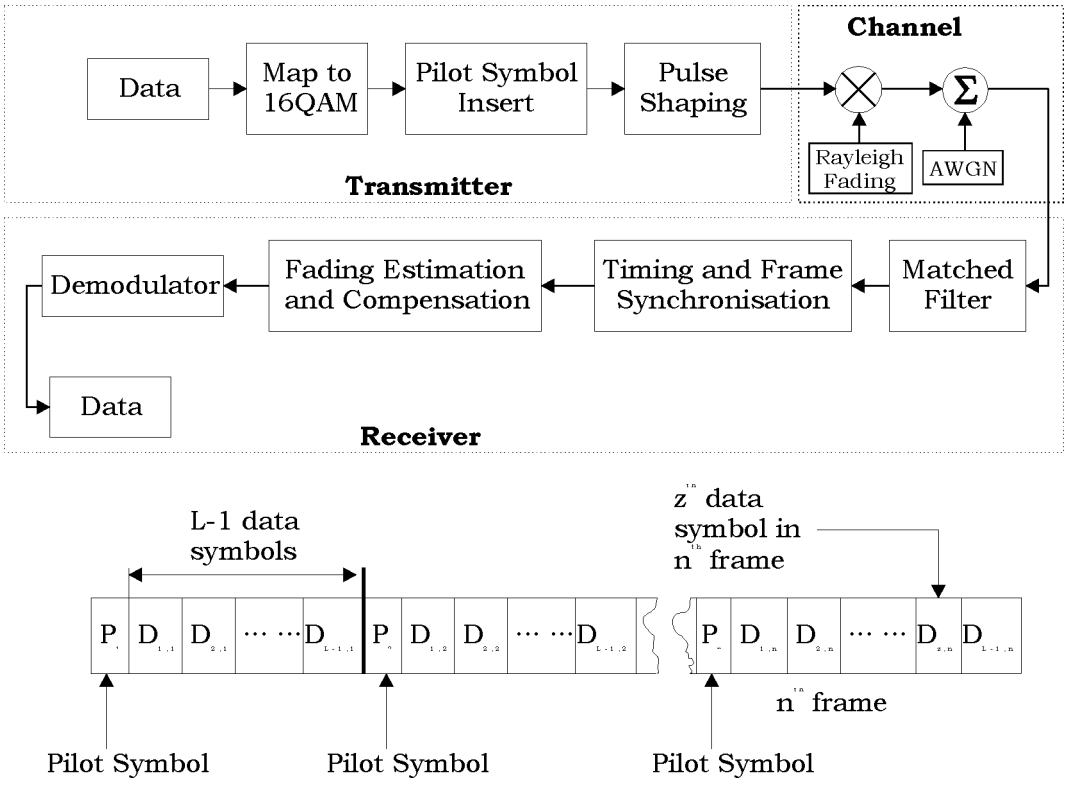

Figure 5.1 A generalised PSAM system and the underlaying symbol frame structure [4].

The receiver, now armed with estimates of the channel state at each data symbol location, is able to scale and rotate (phase adjust) the received symbol so as to compensate for the effect of fading on the symbol during its transmission through the channel.

5.3 Interpolation

One of the key design choices in a PSAM system is the type of interpolator to use in the reconstruction of channel state samples at the non pilot symbol locations. The decision is basically one of complexity versus performance. The performance of several different interpolators has been examined by various authors [2, 4, 8, 53]. The interpolation schemes studied have included linear, 2nd and higher order Gaussian, optimal Wiener filter, and sinc function interpolators. The authors of [4] conclude in their paper that although Gaussian interpolation is very simple to implement and works well when pilot symbols are inserted at a rate significantly above the Nyquist rate for the channel, a sinc function interpolator can provide performance that is almost as good as the optimal Wiener filter estimator used in [2], and has the advantage that it does not require any *a priori* information regarding the autocorrelation function of the channel gain, the Doppler frequency, or the average SNR.

To illustrate the channel estimation process, consider complex baseband symbol rate samples at the output of the receiver matched filter, assuming ideal symbol timing. These can be written as

$$r_k = s_k \cdot c_k + n_k \tag{5.1}$$

In this equation r_k is the fade and noise corrupted received complex symbol sample, s_k is the corresponding originally transmitted symbol, c_k is the complex channel gain, taken from a sequence of zero mean complex Gaussian random variables, representing the sampled fading channel process at time k. Finally, n_k is taken from an additive white Gaussian noise sequence with a variance of $\sigma_n^2 = N_0/2$.

Channel state estimates are formed by dividing the received symbols (in the known pilot symbol time slots) by the known pilot symbols, labelled P_n.

The channel sample estimate at pilot symbol P_n , in the n^{th} frame can then be calculated as

$$\hat{c}_n = \frac{r_n}{P_n} = c_n + \frac{n_n}{P_n} \tag{5.2}$$

where c_n is the actual fading present at the pilot symbol in the n^{th} frame. Note the presence of the second term indicates that the value calculated is a noisy estimate of the actual value.

The channel state at any particular data symbol location within a frame is calculated as the weighted sum of the closest K channel samples, derived from the nearest K pilot symbols. The channel samples are taken from the [(K-1)/2] previous frames, the current frame, and the next [K/2] frames [4].

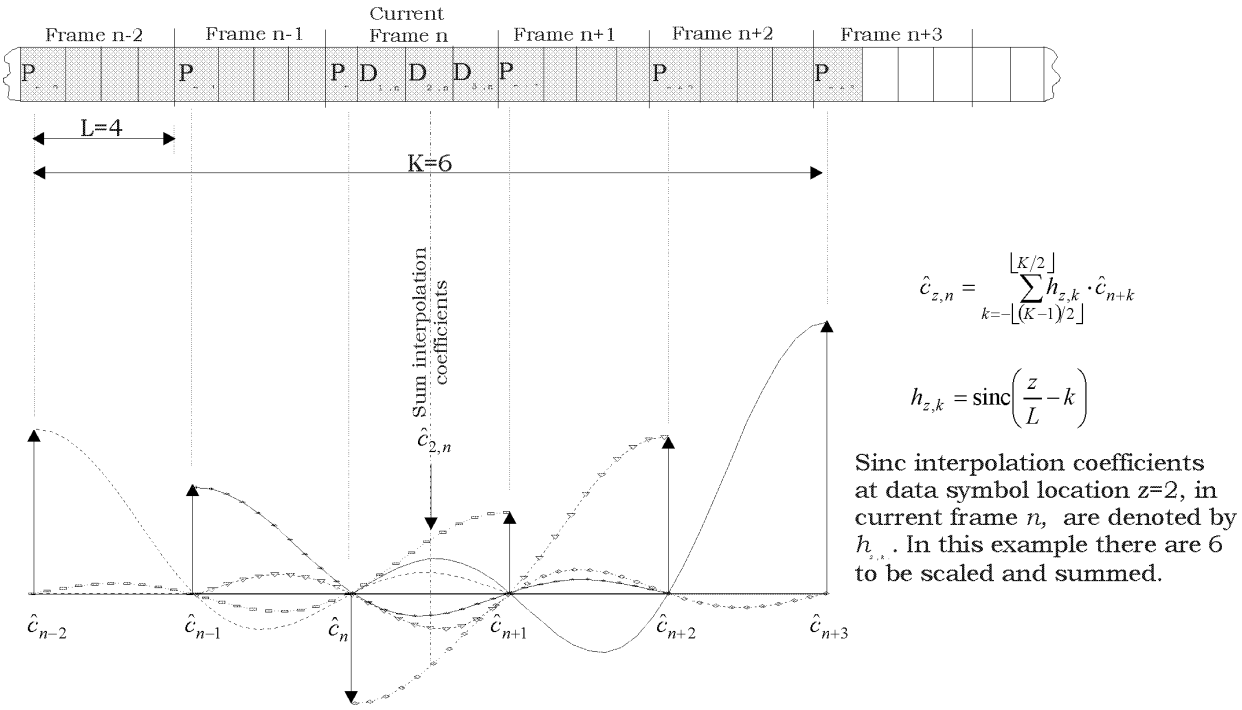

Figure 5.2 An example illustrating the process of sinc interpolation to estimate channel state at data symbol locations within a frame.

Figure 5.2 illustrates the channel estimation process for the specific case where the interpolator used is a sinc function, and values of the interpolation order *K=6*, and frame size *L=4* have been used for clarity.

In mathematical terms the fading estimate at data location *z*, in the current frame *n*, is given by

$$\hat{c}_{z,n} = \sum_{k=-\lfloor (K-1)/2 \rfloor}^{\lfloor K/2 \rfloor} h_{z,k} \cdot \hat{c}_{n+k} \tag{5.3}$$

The floor function, $\lfloor x \rfloor$, denotes the largest integer not greater than x, z=1,…,L-1 is the index of the data symbol within each frame, k is the frame index within the frame cluster used to calculate the channel interpolants within the current frame. The number of neighbouring frames used in the interpolation is defined to be K. This is the interpolation order. The larger K is the more accurate any given interpolant will be. Due to the 1/x decay of the sinc interpolator coefficients there will be an interpolation order that it makes little sense to exceed, as the improvement in accuracy becomes less and less significant.

The interpolation coefficients, $h_{z,k}$, are real numbers and in the case of this thesis are derived from the sinc function. The length of $h_{z,k}$ will be equal to KL. The magnitude of the sinc coefficient at any given data symbol position z, within the current frame, contributed by the pilot symbol channel sample estimate from frame k, is given by

$$h_{z,k} = \mathrm{sinc}\left(\frac{z}{L} - k \right) \tag{5.4}$$

The sinc coefficients for the example introduced earlier where K=6, and the frame size L=4 are shown in Figure 5.3. Note, in practice the sinc interpolator coefficients may be windowed using a spectral windowing function such as the Hamming or Hanning to avoid the spectral problems associated with abrupt truncation of the sinc function coefficients.

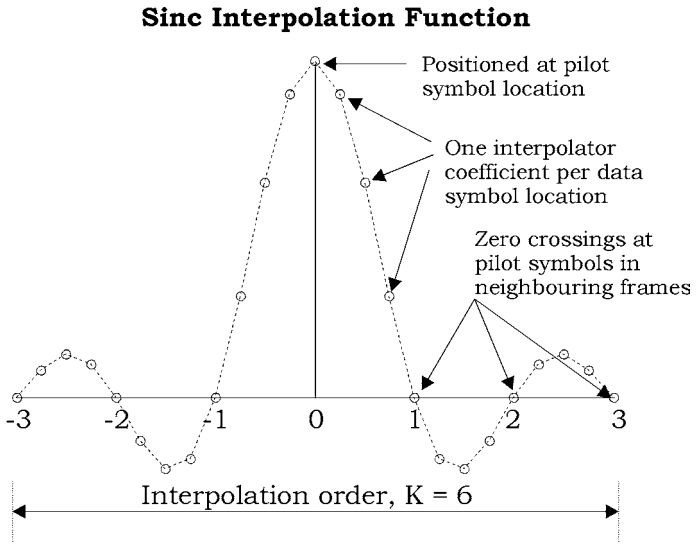

Figure 5.3 Sinc interpolator coefficients for the example where K=6, L=4

5.3.1 Effect of Interpolation Order

The interpolation order, *K,* has several effects on the operation of a PSAM system. Increasing the interpolation order means that more pilot symbols are utilised in the calculation of an interpolant. This increases the complexity of the computation but perhaps more significantly, increases the inherent processing delay of the receiver. This is because before a channel estimate can be calculated for a given data symbol, it is necessary to buffer *K* frames of *L* symbols first. Interpolation order also has an effect on the symbol error rate (SER) performance of the receiver. If the number of neighbouring pilot symbols used to make the current channel estimate is too small then the estimate will be a poor approximation of the actual channel state. This will result in inaccurate symbol compensation leading to a greater probability of the receiver making an incorrect symbol decision.

It has been reported in the literature [4] that in practice, interpolation orders greater than 10 at normalised doppler fade rates, f_DT, of 0.01 and 0.05, see very little additional improvement in SER performance. They recommend therefore that setting *K*=10 is sufficient for implementing a real time system. This interpolation order has been used in all PSAM simulation results presented in this thesis.

5.4 PSAM Receiver Performance in Rayleigh Fading

Although it is interesting to examine the performance of the synchronisation sub-systems in isolation, in the real world it is their impact on the performance of the entire receiver system, that is of greater relevance. One of the major steady state performance measures for a receiver in a communications system, is the average bit or symbol error rate as a function of the signal to noise ratio, E_b/N_o or average signal to noise ratio (E_b/N_o) in the case of a fading channel. Comparison with the same modulation scheme utilising perfect timing estimation can be informative. This can highlight the loss in receiver performance attributable to the non-ideal timing synchronisation sub-systems used.

To be able to present receiver system comparisons using the different timing synchronisation schemes presented in the previous chapter, it is necessary to adopt a receiver structure that is capable of operating in a Rayleigh channel. For the purposes of this thesis, the PSAM receiver, introduced in the previous few sections, has been implemented in simulation for the dual purpose of investigating its performance as a quasi-coherent receiver structure for slowly fading, frequency flat Rayleigh channels, and for examining the impact of non-ideal timing synchronisation on its performance.

To keep the simulation task manageable, certain assumptions are made and built implicitly into the simulation. These include ideal operation of various receiver blocks and functions not being explicitly simulated, such as the RF modules. The signal entering the digital domain is assumed to be within the dynamic range of the digital to analog converter. This would be achieved by some form of slow acting automatic gain control, which would maintain some set point for the average signal power. so that the signal is scaled correctly with respect to the receivers decision threshold grid.

In the PSAM receiver simulation presented here, the focus is on the effect of the timing synchronisation sub-system on the symbol error rate performance of the receiver. It will also be interesting to compare the performance results obtained for the quasi-coherent PSAM receiver with those that have been

previously published for a receiver using a non-coherent modulation format such as differential QPSK, and with the unattainable ideal coherent receiver which operates with perfect knowledge of the channel state.

Within the complex digital baseband processor, a full implementation of a PSAM receiver would have to include some kind of pilot symbol synchronisation capability. This would allow the receiver to initially determine, and then track, where the pilot symbols are located in the received symbol stream. This aspect of synchronisation is outside the scope of the work presented and therefore is one of the parameters that is assumed to be 'known' perfectly by the receiver for the purposes of this work.

The simulated system is uncoded. This means that no channel coding has been applied to the data to protect it from the damaging effects of the fading channel distortion. In a real system, channel coding is a powerful tool in helping to ensure that a digital communications system will operate successfully in a fading channel. Without the protection provided by channel coding, the bit error rate at the receiver, at any reasonable value of SNR, is too high for many applications.

5.5 PSAM Receiver Structure

In the first part of this chapter a general PSAM receiver structure was presented. In this section a more detailed view of the specific structure used in the receiver simulations presented in this thesis will be given. For reasons of computational efficiency all receiver simulations were performed using a block based, rather than a streaming symbol by symbol based approach.

The PSAM communications system simulation was initially implemented using the MATLAB scripting language. This environment provides many high level mathematical constructs such as native support for vector, matrix, and complex data types, and advanced graphing capabilities. This made initial development and testing of the simulation simpler than it would have been if it had been developed exclusively in a high level language such as C.

The results of the PSAM simulation were compared against results that have been published for similar systems in the literature. For example, various aspects of M-QAM PSAM receiver performance were examined by Cavers [2] in which he provides a solid mathematical analysis of PSAM. The paper by Young-Su Kim et al [4] proposed the use of a sinc function interpolator for estimating channel samples and compared the performance of the resulting receiver with receivers using other types of interpolator.

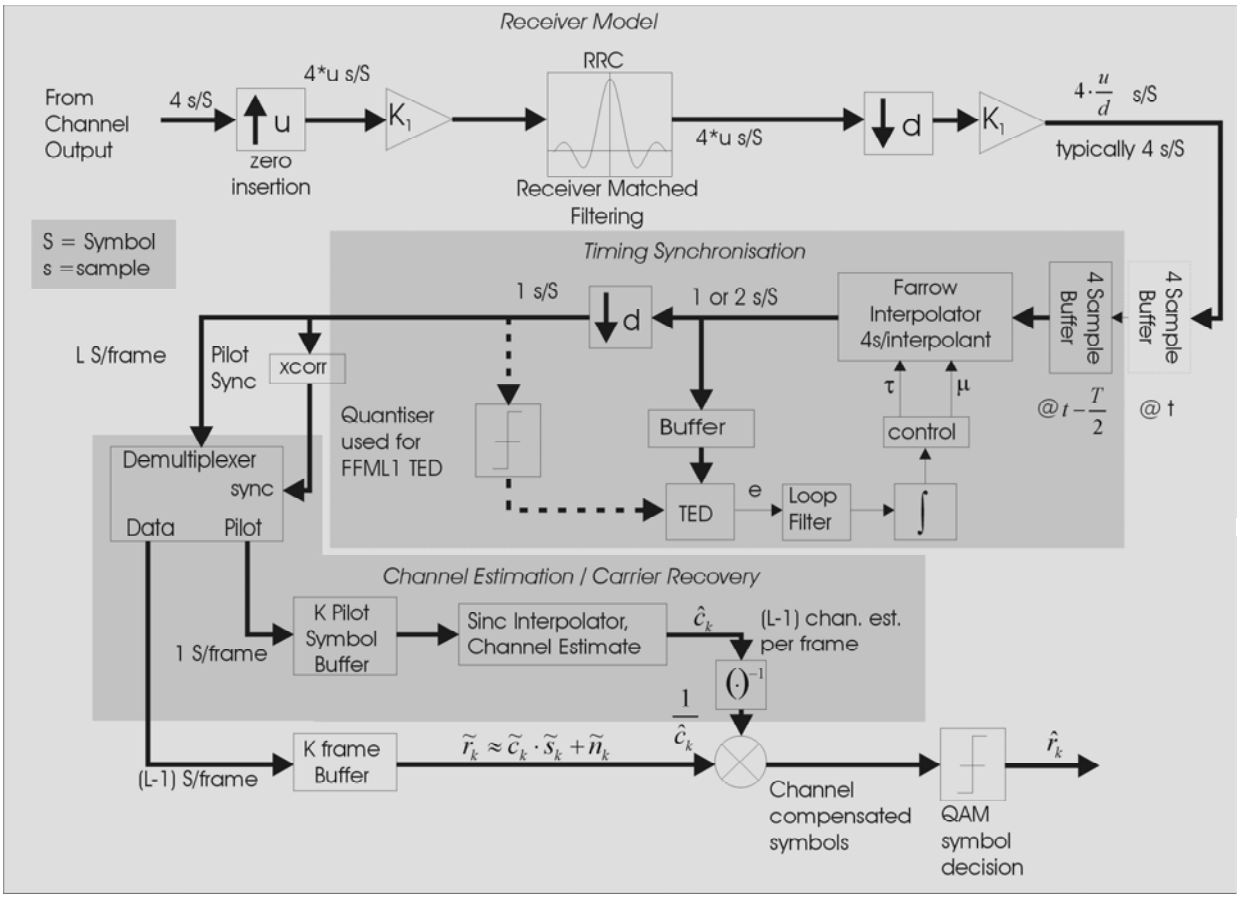

Figure 5.4 PSAM receiver simulation structure.

The simulated receiver presented here also makes use of the sinc function interpolator. Sampei and Sunaga [52] examine the performance of 16, 64 and 256QAM PSAM in a simulation environment and compare the performance of an actual 16QAM modem implementation with their simulated results.

Once the MATLAB implementation was deemed to be producing results that were comparable with those presented in the literature mentioned above, it was re-written in C. This provided a significant increase in execution speed,

typically running between 10 and 20 times as fast as the MATLAB version. The next few sub-sections will explain in more detail the various components of the receiver simulation structure illustrated in Figure 5.4.

5.5.1 Digital Down Converter

A digital down converter (DDC) typically consists of a synthesiser, quadrature mixer, and low pass filter. It performs down conversion, narrowband low pass filtering and decimation to produce a complex (I and Q) baseband signal. In an actual all-digital software radio implementation, the DDC and the preceding IF digital to analog converter act as the interface between the analog RF signal domain and the digital complex baseband signal domain. All of the 'intelligent' operations required in a digital radio, such as demodulation, synchronisation, equalisation, and decoding take place entirely in the digital baseband domain.

In the receiver simulation it is not necessary to simulate the DDC itself. In fact the simulation really only begins at the output of the DDC. It is sufficient to provide complex baseband samples for the rest of the simulation system to process.

5.5.2 Upsampling and Matched Filtering

If desired it is possible to increase the sample rate of the signal by a factor of u by inserting $u-1$ zero samples between consecutive complex samples prior to matched filtering. The matched filter is a low pass root raised cosine filter with a rolloff of α which matches that of the transmit filter. The filter is designed for the upsampled signals higher sample rate. This filter also acts as an interpolation filter, by reconstructing the signal at the input zero sample instants. In the frequency domain this can be thought of as simply a consequence of the low pass filters removal of the image frequencies that were generated by the sample rate expansion.

Most simulations were performed using a value of 16 samples per symbol (integer number of samples per symbol used in most cases) at the output of

the DDC. After the matched filter this sample rate was normally downsampled by 4 for the remainder of the receiver processing.

5.5.3 Timing Synchronisation

This is the sub-system whose stand alone performance was examined in the last chapter. It is a phase locked loop type structure whose timing error detector can be chosen to be any of the three detectors introduced in the last chapter. The input to this block is the non-synchronously sampled complex baseband signal, defined at Nu samples per symbol, where N is deliberately held to be an integer for simplicity. This requirement was relaxed in a later version of the software that was developed with real time DSP implementation in mind. The output of this block is one complex sample per symbol, generated by the interpolator at the current optimum timing estimate.

5.5.4 Channel Estimation

In this block the timing corrected channel corrupted symbols are demultiplexed into a data stream and a pilot symbol stream. It is at this point that knowledge regarding the pilot symbol locations is exploited. This information is determined in the simulation by performing a cross correlation between the transmitted complex symbol sequence and the timing synchronised but channel corrupted received symbol sequence. This needs to be re-calculated for each SNR at which the simulation is run. This is required because during the acquisition process it is possible for the timing synchroniser to converge to a timing estimate that differs by one symbol period depending on how the faded signal evolves with time. This does not affect the validity of the timing estimate, just the alignment between the transmitted symbol block and the corresponding received and synchronised symbol block. It is an artefact of the block-based approach of the simulation. The correct alignment has to be known at the receiver so that the pilot symbol locations will also be known, and so that the transmitted and received symbol blocks can be compared for error counting purposes.

Pilot symbols are buffered so that there are always a total of K available. These pilot symbols come from the $[(K\text{-}1)/2]$ previous frames, the current frame, and the next $[K/2]$ frames. The interpolation technique described in at the beginning of this chapter is used to generate channel samples for each data symbol location within the current frame.

5.5.5 Symbol Channel Compensation

Each data symbol in the current data frame buffer is then multiplied by the reciprocal of its corresponding complex channel sample estimate. This reduces the effects of the fading channel distortion on the symbol. Note that it is not possible to eliminate it because the channel sample is derived from a pilot symbol suffering from additive noise.

5.5.6 Simulation Stopping Criteria

To achieve reasonable statistical accuracy in the measurement of receiver error rates in the presence of a fading channel, two stopping criteria were used in the simulation. The first was based on error counting. The simulation would continue *at least* until N errors had been observed. This was a necessary but not sufficient condition for stopping. The other condition was based on the requirement to accurately model the statistics of the Rayleigh fading channel. This criteria was satisfied by ensuring that the total number of symbols transmitted during the simulation was sufficient to encompass a particular number of fading events. A trial and error method was used to determine what a suitable number of fading events was. Suitability of the number was judged from the 'smoothness' and consistency of the resulting symbol error rate curve. A value of at least 1000 fading events was settled upon as sufficient for statistical accuracy[1].

[1] The channel statistics were assumed ergodic and stationary. This implies that simulation over an ensemble of channel instances can be replaced by simulating a single channel instance over a suitable length of time. The length of simulation was determined empirically through stepped increases in length until multiple simulations (utilising different random number channel seed's) yielded insignificant differences in the SER performance curve. This avoids the need to be concerned with properties such as the channel decorrelation distance.

5.6 Pilot Symbols

In choosing the pilot symbols it is desirable that they not increase the peak to average power ratio (PAPR) of the transmitted signal [51]. It is well recognised that signal envelopes with larger PAPR's place greater demands on the design of the transmitter power amplifier. The amplifier must have good linearity over a wider operating range than one used with a signal possessing a lower PAPR. This usually means operating the amplifier on the most linear section of it's input to output power transfer characteristic. Unfortunately this region of the amplifiers characteristic is usually not its most efficient operating point.

A PSAM receiver does a better job at estimating and compensating for the gain/phase distortion, and the frequency offset as the power of the pilot symbols increases. This provides better immunity to noise. The pilot symbols were not chosen from the standard data symbol alphabet, or constellation. Instead they were chosen to lie entirely on either the I or the Q axis of the constellation, and to have a power equal to that of the outermost symbol in the constellation. The pilot symbol to be transmitted in any given frame was selected randomly from the 2 symbol alphabet. This avoids introducing any periodic component in the transmitted baseband spectrum. Introduction of such a component into the signal spectrum, while not affecting the SER performance of the receiver, is undesirable from a practical standpoint due to the resulting increase in adjacent channel interference.

5.7 Average Symbol Error Rate Performance

Symbol error rate performance results are presented for the PSAM receiver in the following few subsections. The presentation begins with the benchmark performance of an ideal coherent QAM receiver. This is useful in judging the relative merits of the non ideal quasi coherent PSAM receiver implementations. Following this are some fundamental performance results for PSAM assuming perfect timing estimation. These results will show how various system parameters such as the frame length, and interpolation order affect the behaviour of the receiver for fixed channel parameters values such as fade rate and excess bandwidth, α.

5.7.1 Ideal Coherent Receiver Performance in Rayleigh Fading

As a basis for comparison it is useful to know what the theoretical best average SER performance of an M-QAM receiver operating in a flat Rayleigh fading channel actually is. To achieve this ideal it is necessary for the receiver to have perfect knowledge of the channel state at each ideal symbol sampling instant. Initially it may seem that a receiver possessing this ability should be able to achieve the performance of a receiver in an AWGN channel, since with perfect channel knowledge it would appear sensible that the effects of the channel can be perfectly compensated for. This would indeed be the case in a zero additive noise system, however the symbols suffer from additive noise superimposed on the already multiplicatively distorted symbols (refer to Figure 2.12). When the noisy, faded symbols are compensated by multiplying them by the reciprocal of the relevant fading channel state, the additive noise component of the symbol can actually be enhanced. This results in an SER performance that is significantly worse than for the AWGN channel. Performance for 4, and 16 QAM ideal coherent receivers in fading is shown in Figure 5.5.

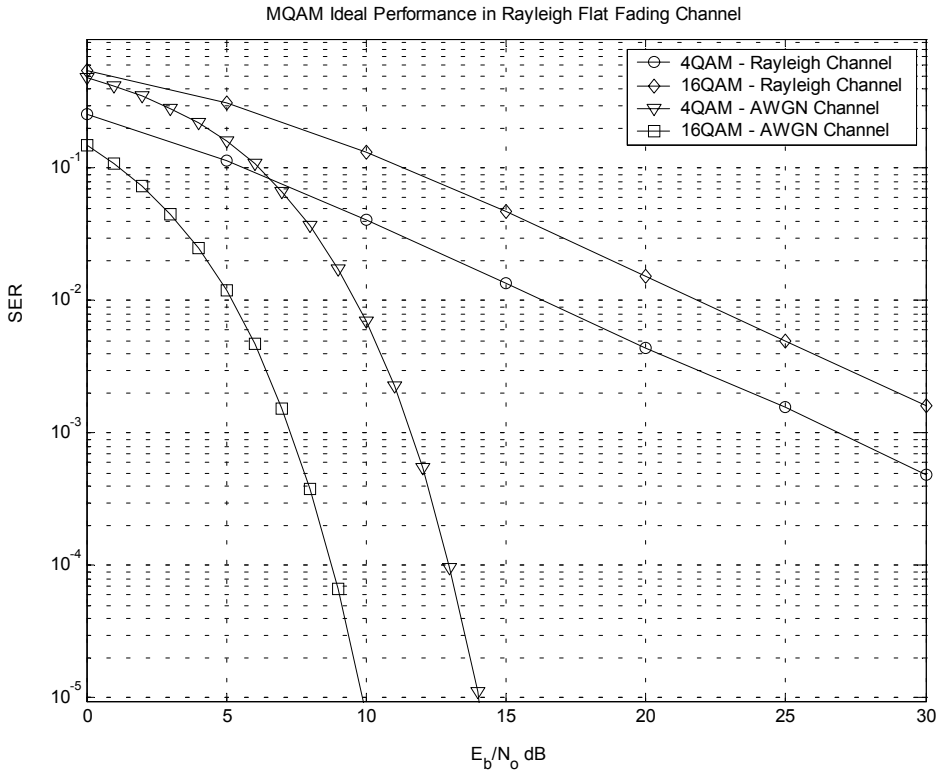

Figure 5.5 Performance of ideal coherent M-QAM receivers in a flat Rayleigh fading channel. Performance in an AWGN channel is shown for comparison.

As the reader will note from Figure 5.5, there is a significant performance penalty to be paid for operating in a fading channel environment. Even with perfect coherent signal detection (perfect channel knowledge), a receiver will generally be designed to utilise error control coding techniques, and perhaps even signal diversity techniques to shift down, or steepen, the SER curves shown in Figure 5.5, thereby achieving more useful symbol error rates.

5.7.2 Fundamental PSAM Receiver Performance Parameters

The frequency with which pilot symbols are inserted into the data stream is defined by the frame size. This is an important parameter affecting the performance of a PSAM communications system. The frame size also directly affects the latency of the system. The larger the frames the more processing delay there will be in the system as enough pilot symbols are gathered for the purposes of channel estimation. Another performance-critical parameter is the interpolation order used in calculating the channel sample at each data symbol location. The greater the interpolation order the more neighbouring

pilot symbols (and thus channel samples) are used in producing each data symbol channel sample estimation, and the greater the accuracy of the estimation. This also comes at the cost of larger processing delay, due to the increased buffering requirements.

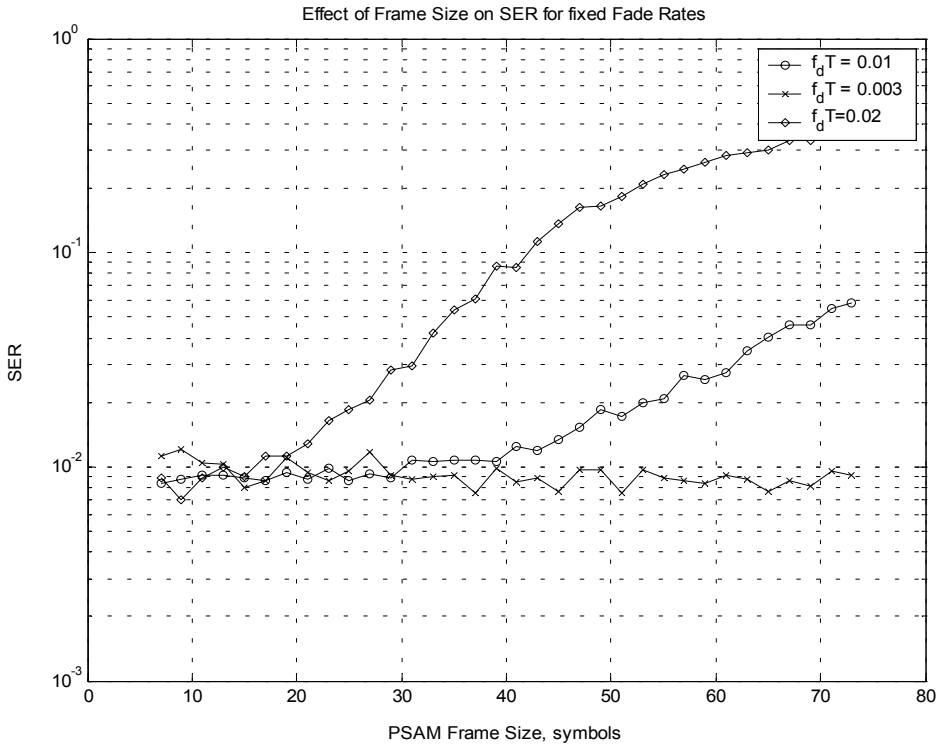

Figure 5.6 The effect of the spacing between pilot symbols on the symbol error rate for three different normalised fade rates. Interpolation order is 10, α = 1, timing is perfect, and E_b/N_o=20dB.

As expected from basic sampling theory, the pilot symbol spacing at which the symbol error rate begins to degrade significantly, reduces as the normalised channel fade rate increases. Three fade rates have been chosen for display here. The slowest rate of 0.003 represents a worst case scenario for the kinds of systems envisioned in this thesis (see Table 4.1). The higher fade rates have been used so that this behaviour can be illustrated with a reasonable frame size. It has been shown in other simulations that for a fade rate of 0.003, the pilot symbols can be spaced as far apart as approximately 110 symbols before the symbol error rate begins to show significant degradation. This means that the SNR penalty due to the insertion of pilot symbols can be very small for slow fading. Assuming the pilot symbols have a power equal to the maximum data symbol power (in QPSK all data symbols have the same power, equal to 2

in these simulations), then in this case the use of pilot symbols carries an SNR penalty of,

$$\Delta E_b / N_o = 10 \cdot \log_{10}(R) \tag{5.5}$$

where

$$\begin{aligned} R &= \frac{(N_s - 1) \cdot \overline{P}_{data} + P_{pilot}}{(N_s - 1) \cdot \overline{P}_{data}} \\ &= \frac{(110 - 1) \cdot 2 + 2}{(110 - 1) \cdot 2} \\ &= 1.009 \end{aligned} \tag{5.6}$$

Thus, to achieve the same symbol error rate as a hypothetical system that does not require pilot symbols, this system would require an increase in transmitter power of just 0.04dB. If the pilot symbol spacing is reduced to 15, the overhead climbs to 0.3dB, in the case of QPSK.

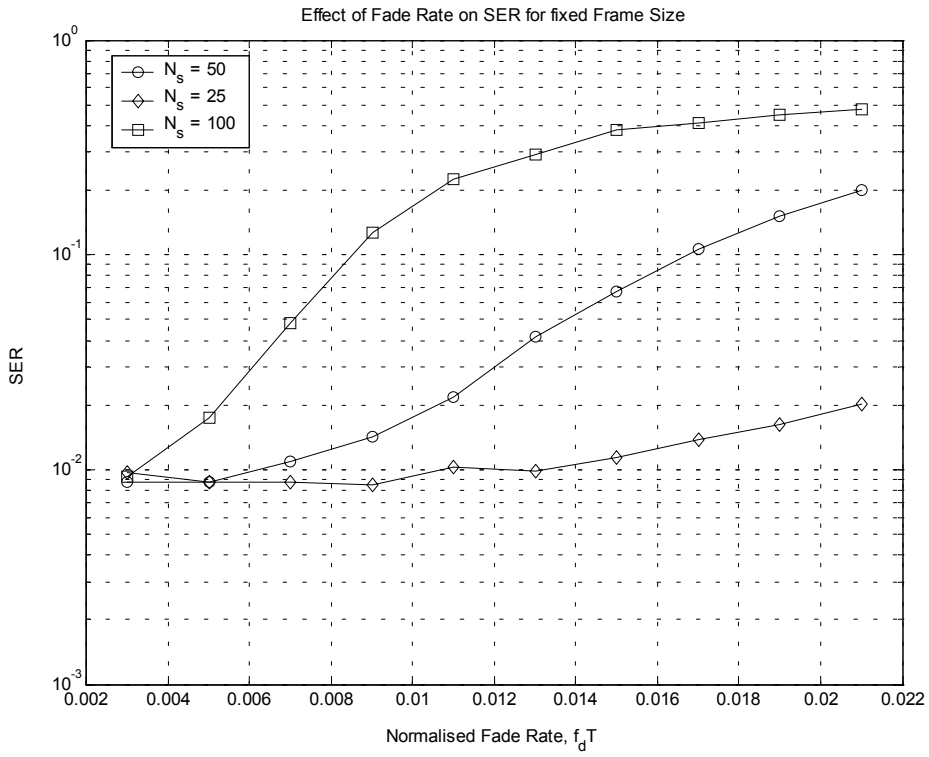

Figure 5.7 Three fixed frame sizes and the effect of increasing channel fade rate, f_dT, on the receivers symbol error rate performance.

This behaviour can also be illustrated in a slightly different way, by showing how the receivers symbol error rate is affected as the normalised fade rate

increases. From Figure 5.7 it is apparent that a pilot symbol spacing of 25 provides adequate channel sampling, and hence channel estimation, up to a fade rate of approximately 0.009. For pilot symbol spacings of 50 and 100, the corresponding fade rates are 0.005, and 0.003 respectively. In reality, typically encountered channel fade rates (see Table 4.1) at today's medium (a few tens of Kbaud) and the medium term future's higher data rates (a few hundred Kbaud), in an urban environment, will be less than 0.001. This means that pilot symbols can be spaced widely thereby minimally impacting power efficiency of the system.

Simulations have shown that the symbol error rate is relatively insensitive to the interpolation order. In fact the results indicate that increasing the interpolation order to more than 6 does not actually result in much further improvement in channel estimation accuracy at the slow fade rates this thesis is concerned with. For all subsequent receiver simulations however, an interpolation order of 10 is used. This is the value adopted by [4].

5.7.3 Simulated PSAM Receiver with Non Ideal Timing Synchronisation

Figure 5.8 illustrates the symbol error rate performance that is achieved by a quasi-coherent receiver using pilot symbol assisted modulation. Four different timing synchronisation scenarios are shown. The first is the receiver's symbol error rate with perfect timing synchronisation. The rest of the curves illustrate the receivers performance when each of the three timing synchronisers being examined, is used. As the reader will note, in the steady state, there is a negligible performance impact on the error rate in using the non-ideal synchronisers, even for the FFML1 synchroniser with its significantly poorer timing estimate variance. The symbol error rate for DQPSK is provided for comparison.

When the frame size chosen is too large for the channel fade rate experienced by the receiver, the symbol error rate exhibits an irreducible error floor due to the random fluctuations of the channel. As the fade rate, f_dT, increases (for example through increased vehicle velocity), the error floor will increase, even

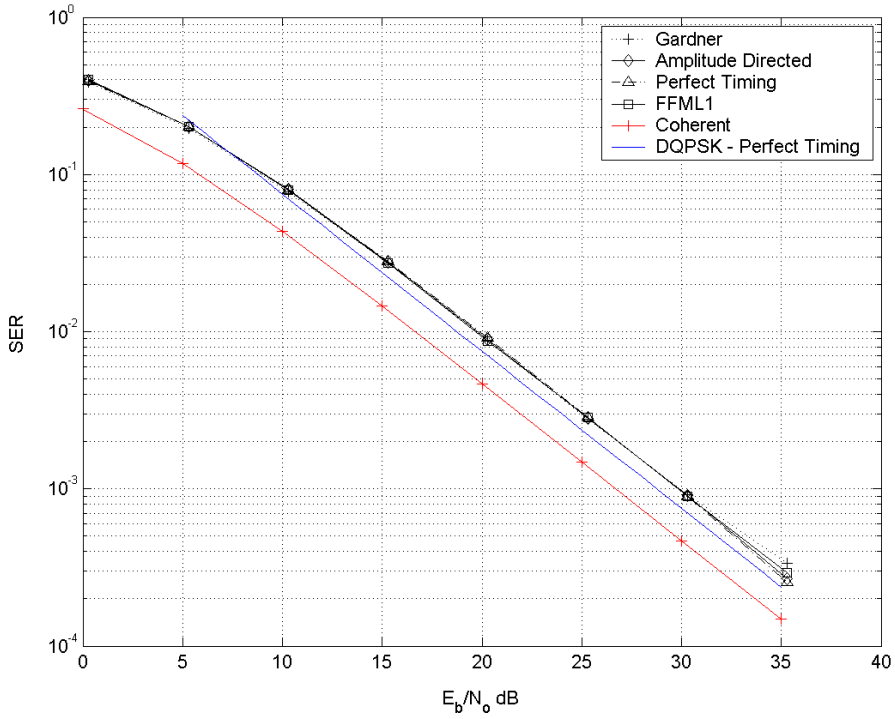

Figure 5.8 The symbol error rate performance of a PSAM receiver with perfect timing and with each of the three synchronisers. In this simulation f_dT is 0.003, α is 1.0 and synchroniser loop gain, G=0.01. Frame size is 15.

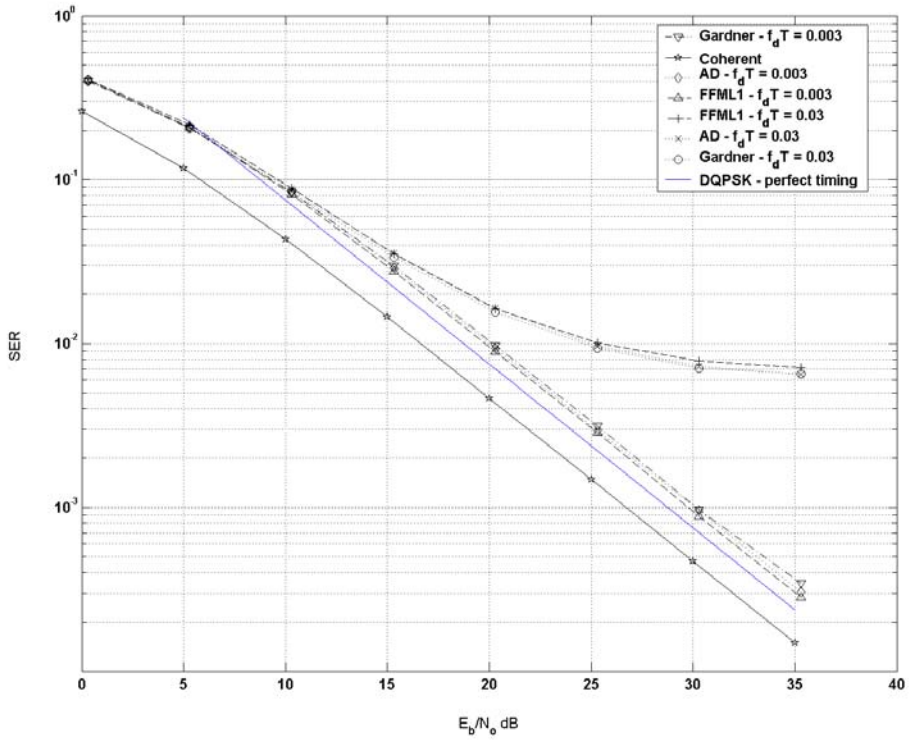

Figure 5.9 PSAM receiver SER performance with each of the timing synchronisers, for two fade rates. The frame size is 15 which is adequate for sampling the more slowly fading channel but not the faster channel. α=0.35.

if E_b/N_o is increased. Once a certain value of E_b/N_o is reached, the dominant factor contributing to receiver errors is the poor channel estimation, rather than noise. The second set of curves in Figure 5.9 illustrates this error floor phenomenon. The pilot spacing of 15 is insufficient to provide accurate channel estimate for a fade rate as high as 0.03.

5.8 Conclusions

For all practical purposes, the steady state performance of any of these synchronisers is sufficient to allow the PSAM receiver to very nearly achieve the level of symbol error rate performance that would be expected from a PSAM receiver with perfect timing information.

The PSAM receiver simulated here shows a performance degradation of between 3 and 3.5dB when compared with the ideal coherent receiver.

CHAPTER 6
CONCLUSIONS

6.1 Summary

Three existing all-digital timing error detector (TED) algorithms, and their associated first and second order timing estimation synchronisers, have been simulated and their performance compared in the additive white Gaussian noise (AWGN), and more importantly, in the flat, slow fading Rayleigh channel. The Rayleigh fading channel is of primary importance to mobile digital receivers.

Two of the synchronisers have only been designed with the AWGN channel in mind, while the third (FFML1) TED was designed by a former student of the Electrical and Electronic Engineering department [50] from maximum likelihood principles to provide an optimum error estimate in a Rayleigh fading channel. This detector is optimum only if the actual transmitted symbols are known to the receiver. Unless a training sequence is initially transmitted this is generally not the case, and during normal operation it is certainly not the case. This estimator was used in a sub-optimum decision directed mode for the purposes of the comparisons in this thesis. Even so, the FFML1 based synchroniser's (first and second order) showed good acquisition performance in fading, generally having the shortest lock time of the three synchronisers. Unfortunately it's steady state tracking error variance was relatively poor, in comparison to the Gardner and AD synchronisers, however, as demonstrated by the PSAM receiver simulation results in Chapter 5, this made very little difference to the overall receiver symbol error rate (SER) performance. External factors, primarily the limitations of the channel estimation algorithm, based, as it is, on noisy pilot symbols, had a far greater influence on the SER, swamping any effects due to differences in the performances of the timing synchronisation sub-systems.

It was found therefore that, the tracking performance of all of these synchronisers is acceptable when incorporated into a pilot symbol assisted receiver. Of more relevance, perhaps, in selecting a particular synchroniser, is the acquisition time in fading conditions, and the complexity of the implementation. A simple measure of the complexity of each TED algorithm was presented in Appendix A, where the execution time of each synchroniser was measured when running on a real DSP platform. This measure, given in cycle counts, gives approximate relative complexities (where the Gardner TED is 1) of 10 for the AD and 18 for the FFML1 TED. Further optimisation of the AD and FFML1 implementation could no doubt achieve some increase in efficiency, however the relative complexity ranking's are unlikely to change.

Being closed loop feedback type structures, none of the synchronisers exhibited exceptionally fast lock times in conditions of fading and additive noise. Lock times were normally on the order of at least several hundred symbol periods. The actual lock time in any specific case will depend on the state of the channel magnitude at the beginning of the timing acquisition process, and also on the initial open loop gain value, G. This would likely make them unsuitable in a time division multiplexed communication system if the receiver is required to re-aquire lock at every assigned timeslot. In such a case, some form of open loop feedforward method may be a more suitable candidate since feedforward techniques are not prone to the same hangup phenomenon that feedback structures are.

Acquisition times of the feedback synchronisers can be significantly reduced if used in combination with 'gear shifting', which involves starting the synchroniser with a larger open loop gain (wider bandwidth) than it's ultimate tracking open loop gain. The open loop gain is reduced in one or more steps after a predefined number of symbols have been received, or after lock has been detected.

6.2 Suggestions for Future Work

An obvious extension to the work presented in this thesis is to complete the comparisons of the DSP synchroniser implementations with the computer simulations. This will involve extending the DSP application to provide data transfer capabilities between the EVM and host PC, most likely utilising shared memory and the host port interface. This would be similar to the technique used in the hybrid PC/DSP simulation presented in section A.4. The corresponding PC application will also need to be designed and implemented. Its primary function would be the downloading of the DSP code into the EVM, the configuration of the DSP synchronisation software, gathering of data from the DSP, analysis and display of desired quantities. The DSP software should be 'instrumented' so that data from various key points within the synchronisation algorithm can be transferred to the PC for analysis and display if relevant. This data communication process should be implemented in such a way that it does not impact on the DSP's ability to process received samples from the receiver's digital down converter in real-time.

Performance measurement of the synchronisation algorithms running in real-time on the DSP platform should be performed in both an AWGN and a Rayleigh fading channel environment. The Rayleigh fading channel environment can be created using the Hewlett Packard RF channel simulator, HP11759B.

Further practical extension to this work could include the integration of the synchronisation sub-system simulated and implemented here, with the Motorola DSP receiver software currently running on Peter Green's [54] digital receiver platform, which currently lacks any form of timing synchronisation. Because the target processor is different (Motorola 56303), the synchronisation software will need to be ported to the new environment. There are two approaches possible for this task. The first would be to restructure the current Motorola DSP receiver software into a C implemented framework, with each of the critical receiver tasks being a C callable assembler function. The

only task that would not be written in assembler would be synchronisation, since this has already been created as a C only implementation.

An alternative approach would be to completely rewrite the synchronisation algorithm in Motorola assembler and integrate that into the existing Motorola DSP receiver software structure.

Some additional work is also required on the Motorola DSP receiver software to complete the receiver functionality to the point where binary data is output from the radio.

This work has only examined closed loop synchronisers. As has been noted several times in the body of the thesis, these are prone to the phenomenon known as hangup. This acts to potentially lengthen the dwell time of the synchroniser at extreme sampling phase offset errors, with a consequent increase in acquisition time. It would be interesting to examine a few feedforward synchronisation algorithms so that their performance in fading channel conditions can be evaluated and compared with the feedback structures presented here.

The channel model used exclusively in this work was that of flat, slow, Rayleigh fading. This model has wide relevance to current and near term digital wireless systems, however, as data rates increase, the duration of a data symbol will more commonly be on the order of, or less than, the RMS delay spread of the types of channels affecting most users, particularly urban, high rise environments. This means that a more relevant channel model for future systems may be the Rayleigh, slow fading, frequency selective channel. Examination of the affect of frequency selectivity on the performance of these synchronisers, and how well these synchronisers are able to perform in combination with channel equaliser structures would be useful.

REFERENCES

[1] K. Pahlavan and A. H. Levesque, *Wireless Information Networks*. New York: John Wiley & Sons, Inc, 1995.

[2] J. K. Cavers, "An Analysis of Pilot Symbol Assisted Modulation for Rayleigh Fading Channels," *IEEE Transactions on Vehicular Technology*, vol. 40, pp. 686 - 693, 1991.

[3] J. K. Cavers and M. Liao, "A comparison of pilot tone and pilot symbol techniques for digital mobile communication," presented at IEEE Global Telecommunications Conference, 1992. , GLOBECOM '92. Communication for Global Users.,, Orlando, FL, USA, 1992.

[4] Y.-S. Kim, C.-J. Kim, G.-Y. Jeong, Y.-J. Bang, H.-K. Park, and S. S. Choi, "New Rayleigh fading channel estimator based on PSAM channel sounding technique," presented at 1997 IEEE International Conference on Communications, 1997. ICC '97 Montreal, Towards the Knowledge Millennium, Montreal, Que., Canada, 1997.

[5] H. Lo, D. Lee, and J. A. Gansman, "A study of non-uniform pilot spacing for PSAM," presented at 2000 IEEE International Conference on Communications, 2000. ICC 2000., New Orleans, LA, USA, 2000.

[6] M. H. C. Ng, S.W., "Bandwidth-efficient pilot symbol aided technique," *Electronics Letters*, vol. vol. 34, pp. 1548 -1550, 1998.

[7] E. Okamoto, H.-B. Li, and T. Ikegami, "A pilot symbol assisted compensation scheme of fading and frequency offset for 16QAM," presented at IEEE 1998 International Conference on Universal Personal Communications, 1998. ICUPC '98., Florence, Italy, 1998.

[8] J. M. Torrance and L. Hanzo, "Comparative study of pilot symbol assisted modem schemes," presented at Sixth International Conference on Radio Receivers and Associated Systems, 1995., Bath, UK, 1995.

[9] S. Haykin, *Communication Systems*, 3rd ed: John Wiley & Sons, Inc, 1994.

[10] J. G. Proakis, *Digital communications*, 2nd ed. New York: McGraw-Hill, 1989.

[11] M. C. Jeruchim, P. Balaban, and K. S. Shanmugan, *Simulation of communication systems*. New York: Plenum, 1992.

[12] R. Howald, "Modulation Basics for Digital Communications, Part 2," in *Communication System Design*, March 2001.

[13] B. Sklar, "Rayleigh Fading Channels in Mobile Digital Communication Systems Part 1: Characterization," in *IEEE Communications Magazine*, vol. 35, 1997, pp. 136-146.

[14] B. Sklar, "Rayleigh Fading Channels in Mobile Digital Communication Systems Part 2: Mitigation," in *IEEE Communications Magazine*, vol. 35, 1997, pp. 148-155.

[15] T. S. Rappaport, *Wireless Communications - Principles and Practice*, 1st ed. New Jersey: Prentice-Hall Inc, 1996.

[16] M. C. Jeruchim, P. Balaban, and K. S. Shanmugan, *Simulation of communication systems : modeling, methodology, and techniques*, 2nd ed. New York: Kluwer Academic/Plenum Publishers, 2000.

[17] A. F. Molisch, *Wideband Wireless Digital Communications - Editor Andreas F. Molisch*. New Jersey: Prentice Hall, 2001.

[18] W. C. Jakes, *Microwave mobile communications : Edited by William C. Jakes*. N.Y.: Wiley, 1974.

[19] R. H. Clarke, "A Statistical Theory of Mobile Radio Reception," *Bell Syst. Tech. J.*, vol. 47, pp. 957-1000, 1968.

[20] M. C. Jeruchim, P. Balaban, and K. S. Shanmugan, "Models of White and Band-Limited White Noise," in *Simulation of Communication Systems*, R. W. Lucky, Ed. New York: Plenum, 1992, pp. 255.

[21] J. G. Proakis, "Digital Signaling Over Fading Multipath Channels," in *Digital Communications*, *McGraw-Hill series in electrical engineering. Communications and signal processing*, S. W. Director, Ed., 2nd ed. New York, 1989, pp. 706.

[22] F. Gini, M. Luise, and R. Reggiannini, "Cramer-Rao bounds in the parametric estimation of fading radiotransmission channels," *IEEE Transactions on Communications*, vol. 46, pp. 1390 - 1398, 1998.

[23] *The Communications Handbook*: CRC Press and IEEE Press, 1996.

[24] L. E. Franks, "Carrier and Bit Synchronization in Data Communications - A Tutorial Review," *IEEE Transactions on Communications*, vol. Vol. Com-28, pp. 1107-1120, 1980.

[25] L. W. Couch II, *Digital and Analog Communication Systems*, 6th ed. New Jersey: Prentice-Hall, Inc, 2000.

[26] F. M. Gardner, *Phaselock techniques*, 2nd ed. New York: Wiley, 1979.

[27] P. M. Martin, A. Bateman, J. P. McGeehan, and J. D. Marvill, "The implementation of a 16-QAM mobile data system using TTIB-based fading correction techniques," presented at IEEE 38th Vehicular Technology Conference, 1988, Philadelphia, PA, USA, 1988.

[28] A. Bateman, "Feedforward transparent tone-in-band: its implementations and applications," *IEEE Transactions on Vehicular Technology*, vol. 39, pp. 235-243, 1990.

[29] J. P. Seymour and M. P. Fitz, "Two Stage Carrier Synchronization Techniques for Nonselective Fading," *IEEE Transactions Vehicular Technology*, vol. vol. 44, pp. 103-110, 1995.

[30] C.-K. Tzou, R. Raheli, and A. Polydoros, "Applications of per-survivor processing to mobile digital communications," presented at IEEE Global Telecommunications Conference, 1993. GLOBECOM '93, Houston, TX, USA, 1993.

[31] R. Raheli, A. Polydoros, and C.-K. Tzou, "Per-Survivor Processing: a general approach to MLSE in uncertain environments," *IEEE Transactions on Communications*, vol. 43, pp. 354-364, 1995.

[32] R. Raheli, A. Polydoros, and C.-K. Tzou, "The Principle of Per-Survivor Processing: A General Approach to Approximate and Adaptive MLSE," presented at IEEE Global Telecommunications Conference, 1991. GLOBECOM '91, Phoenix, AZ, USA, 1991.

[33] G. M. Vitetta and D. P. Taylor, "Maximum likelihood decoding of uncoded and coded PSK signal sequences transmitted over Rayleigh flat-fading channels," *IEEE Transactions on Communications*, vol. 43, pp. 2750-2758, 1995.

[34] G. M. Vitetta and D. P. Taylor, "Viterbi decoding of differentially encoded PSK signals transmitted over Rayleigh frequency-flat fading channels," *IEEE Transactions on Communications*, vol. 43, pp. 1256-1259, 1995.

[35] G. M. Vitetta and D. P. Taylor, "Double-filtering receivers for PSK signals transmitted over Rayleigh frequency-flat fading channels," *IEEE Transactions on Communications*, vol. 44, pp. 686-695, 1996.

[36] U. Mengali and A. N. D'Andrea, *Synchronization techniques for digital receivers*. New York: Plenum Press, 1997.

[37] U. Mengali and A. N. D'Andrea, "Timing Recovery in Baseband Transmission," in *Synchronization Techniques for Digital Receivers*, *Applications of Communications Theory*. New York: Plenum Press, 1997, pp. 353-409.

[38] F. M. Gardner, "Hangup in Phase-Lock Loops," *IEEE Transactions on Communications*, vol. vol. COM-25, pp. 1210-1214, 1977.

[39] F. M. Gardner, "Equivocation as a Cause of PLL Hangup," *IEEE Transactions on Communications*, vol. vol. COM-30, pp. 2228-2241, 1982.

[40] H. Meyr and L. Popken, "Phase Acquisition Statistics for Phase-Locked Loops," *IEEE Transactions on Communications*, vol. vol. COM-28, pp. 1365-1372, 1980.

[41] D. Verdin and T. C. Tozer, "Hangup in asynchronous timing recovery loops," *Electronics Letters*, vol. vol. 29, pp. 1914 -1915, 1993.

[42] C. W. Farrow, "A Continuously Variable Digital Delay Element," *IEEE International Symposium on Circuits and Systems*, vol. 2915, pp. 2641-2645, 1988.

[43] J. H. Vincent, "On Some Experiments in Which Two Neighbouring Maintained Oscillatory Circuits Affect a Resonating Circuit," *Proceedings of the Royal Society*, vol. 32, part 2, pp. 84-91, 1919-1920.

[44] K. R. Wendt and G. L. Fredendall, "Automatic Frequency and Phase Control of Synchronization in Television Receivers," *Proceedings of the IRE*, vol. 31, pp. 7-15, 1943.

[45] R. E. Best, *Phase-locked loops : theory, design, and applications*. New York: McGraw-Hill, 1984.

[46] Y. R. Shayan and T. Le-Ngoc, "All Digital Phase-Locked Loop: Concepts, Design and Applications," *IEE Proceedings-Radar, Sonar and Navigation*, vol. vol. 136 Pt. F, pp. 53-56, 1989.

[47] R. C. Clarke, "Analog and Digital Phase Locked Loop Concepts," University of Canterbury, Christchurch, Internal Report November 2000.

[48] F. M. Gardner, "A BPSK/QPSK Timing-Error Detector for Sampled Receivers," *IEEE Transactions on Communications*, vol. vol. COM-34, pp. 423-429, 1986.

[49] D. Verdin and T. C. Tozer, "Symbol-Timing Recovery for M-PSK Modulation Schemes Using the Signum Function," presented at IEE Colloquium on New Synchronisation Techniques for Radio Systems, London, UK, 1995.

[50] G. Watkins, "Maximum likelihood symbol timing recovery for digital wireless communication," in *Electrical and Electronic Engineering*. Christchurch: University of Canterbury, 1997, pp. vii, 105 leaves.

[51] Y. Murakami, M. Orihashi, T. Matsuoka, and M. Sagawa, "A new design of pilot symbol in 16 QAM channels," presented at 2000 IEEE 51st Vehicular Technology Conference Proceedings, 2000. VTC 2000-Spring Tokyo, Tokyo, Japan, 2000.

[52] S. Sampei and T. Sunaga, "Rayleigh fading compensation for QAM in land mobile radio communications," *IEEE Transactions on Vehicular Technology*, vol. vol. 42, pp. 137 -147, 1993.

[53] E. Okamoto, H.-B. Li, and T. Ikegami, "Rayleigh Fading Compensation for QAM by Using FFT," presented at PIMRC'96, Seventh IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, 1996., Taipei, Taiwan, 1996.

[54] P. J. Green, "Smart Antenna Software Radio Test System User Manual," University of Canterbury, Christchurch August 8 2001.

[55] L. E. Miller and J. S. Lee, "BER Expressions for Differentially Detected pi/4 DQPSK Modulation," *IEEE Transactions on Communications*, vol. 46, pp. 71-81, 1998.

[56] *TMS320C62x/C67x CPU and Instruction Set - spru189f*. Dallas: Texas Instruments, 2000.

[57] *TMS320C6201/6701 Evaluation Module Users Guide - spru269d*. Dallas: Texas Instruments, 1998.

[58] *TMS320C6201/6701 Evaluation Module Technical Reference - spru305*. Dallas: Texas Instruments, 1998.

[59] D. E. Knuth, *Seminumerical Algorithms*, vol. 2, 3rd ed. Reading, Massachusetts: Addison-Wesley, 1997.

[60] A. Williams, "A Better Random Number Generator", Web site, A Better Random Number Generator, 1999, http://www.planet-source-code.com/vb/scripts/ShowCode.asp?txtCodeId=239&lngWId=3

[61] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C: The Art of Scientific Computing*, 2nd ed. Cambridge: Cambridge University Press, 1997.

[62] F. M. Gardner, "Interpolation in digital modems. I. Fundamentals," *IEEE Transactions on Communications*, vol. vol. 41, pp. 501 -507, 1993.

[63] Texas Instruments: B. G. Carlson, "Application Report - TMS320C6000 EVM Daughterboard Interface", PDF file, spra478.pdf, 1998, www.dspvillage.ti.com

[64] *TMS320C6201/6701 Peripherals Reference Guide - SPRU190D*. Dallas: Texas Instruments, 2001.

APPENDIX A
DSP IMPLEMENTATION OF SELECTED RECEIVER FUNCTIONS

A.1. Introduction

One of the original goals of this thesis project was to gain experience in implementing embedded C real time signal processing software on a modern DSP platform. The intended development target was to be the Texas Instruments TMS320C6701 floating point DSP. This DSP was available in the form of an Evaluation Module (EVM) that could be installed in a PCI slot of a PC (host). It was envisioned that the EVM would be interfaced to the digital receiver hardware available to the author [54]. This would provide baseband I and Q samples, direct from the digital down converter, for processing by the digital receiver algorithms implemented on the C6701. This was achieved to a limited extent, however, it could not be completed in full primarily due to multiple development board failures. This chapter will detail the development stages implemented by the author in progressing from the original MATLAB simulations to the real time streaming DSP software framework.

A.1.1. Scope of Implementation

It was realised early in the project that to try and implement a full PSAM receiver on the EVM would be an overly ambitious task. To have done so would have required the implementation of techniques and processes that were outside the scope of the thesis, and furthermore would have also required modification of the digital transmitter software, again outside the scope of the project. Thus the real time DSP implementation goals were modified to reflect a more realistic approach. A more reasonable goal was the real time streaming implementation of the timing synchronisation sub-system.

A.1.2. The DSP Development Platform

A brief introduction to the DSP development platform will be provided here. Additional detail can be found in references [56-58].

As mentioned earlier, the development platform used was the Texas Instruments (TI) TMS320C6701 EVM. This particular DSP is a 32 bit floating point device that can run at up to 167MHz externally, although the EVM limited this to a maximum of 133MHz. Internally, it is based on an advanced Very Long Instruction Word (VLIW) architecture that provides the DSP with the capability of executing up to eight 32bit instructions every cycle. This provides peak performances of [56]:

- Max 1064 MIPS at 133 MHz
- Max 796 MFLOPS at 133 MHz for single precision operations
- Max 200 MFLOPS at 133 MHz for double precision operations
- Max 548 MFLOPS at 133 MHz for multiply and accumulate operations

The EVM provides a wealth of hardware resources for the DSP developer, however the most important from the point of view of this project included, complete access to the DSP's external peripheral interface, and a communications channel between the host PC and the DSP via the host port interface (HPI), accessible via the PCI bus.

Device 'emulation' was via an embedded JTAG controller and this was also accessed by the PC through the PCI bus. This is totally transparent to the developer but provides a software development and debugging environment in which it is possible to download code to the target device, run it, single step, and perform a myriad of other debugging and profiling tasks. JTAG is an industry standard (IEEE 1149.1) communication interface used for boundary scan testing of complex integrated circuits (IC) and, in this case, non-invasive communication with the DSP's internal resources such as CPU registers, and the entire memory map.

A.1.3. The DSP Software Development Tools

Texas Instruments could arguably claim to have the most sophisticated embedded DSP software development tool set currently available. Code Composer Studio is a fully integrated environment that brings together all host and target tools in a single uniform environment. Traditional tools for editing, building, debugging, application profiling and project management are included along with more advanced features such as data graphing, file I/O and a scripting language for enhanced user flexibility and customisation. The author saw the opportunity to familiarise himself with this very useful tool as an important motivation for implementing the timing synchronisation sub system in streaming, real time mode on a modern Texas Instruments DSP.

The TI EVM development system extended beyond the DSP board and the Code Composer Studio environment. In addition a significant number of application programming interface (API) software libraries and macros were provided for performing some of the more mundane tasks such as peripheral configuration at a much higher level of abstraction, potentially allowing significant time to be saved in getting an initial software framework up and running.

A.1.4. The PC Software Development Tools

All host-side software was developed as Win32 command console applications (DOS box) using the Microsoft Visual C++ Studio v6.00 development environment. In all software development, the C language was used exclusively, for reasons of portability (between host and DSP), ease of development, and because the author already possessed experience with this language. A host side API library, and low level Windows™ drivers were also provided with the DSP development kit. The low level driver, evm6x.vxd, provides the Win 9x operating system with low level access to the EVM hardware. The user API library is contained in the Win32 DLL, evm6x.dll. This provides all the functions required by user developed host software for control and communication with the EVM hardware, primarily by way of the host port

interface. These functions, and the HPI effectively allow the PC to access any part of the DSP memory map.

A.2. Development Stages

All initial timing and PSAM receiver simulations were implemented in MATLAB. This specialised matrix based mathematical package is ideally suited to rapid prototyping of signal processing algorithms, and subsequent analysis and visualisation of the signal during all stages of processing by the algorithm. At this stage it is generally most efficient to process the signal in large blocks rather than on a sample by sample basis. The primary reason for this, particularly as it pertains to MATLAB, is to keep looping structure overhead to a minimum. MATLAB is very efficient at calculating with matrices and vectors but relatively inefficient when it comes to loops.

Once the MATLAB simulations were judged to be producing reasonable results, assessed through comparison with results from other works on similar systems, work began on rewriting the simulation in C as a Win32 command console application, labeled 'Phase 2' in Figure A.1.

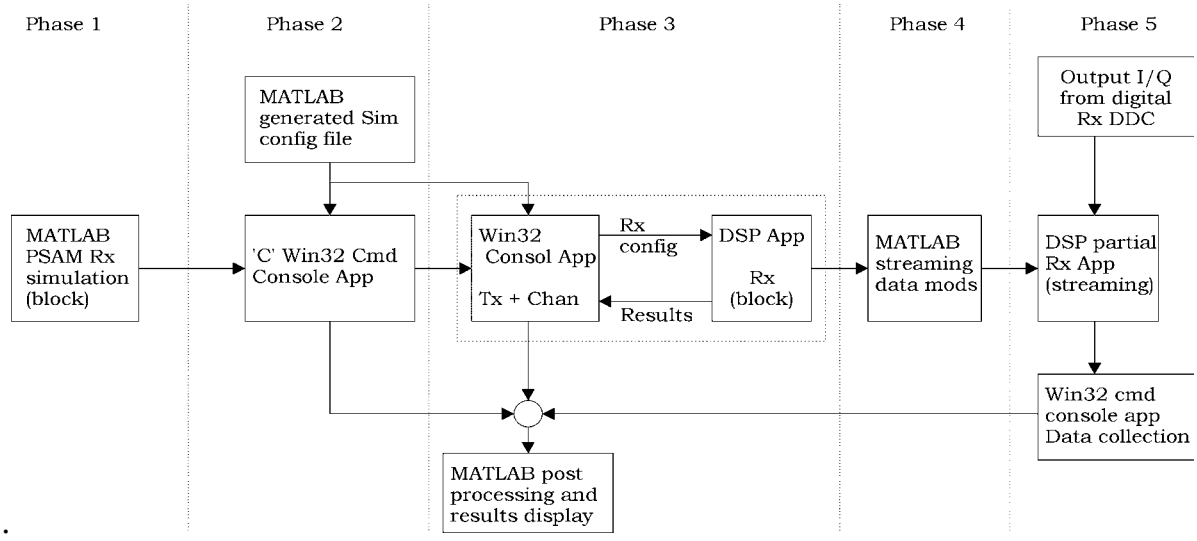

Figure A.1 Diagrammatic depiction of the various development stages involved in progressing from the initial MATLAB simulation software to a real time streaming application on the DSP

There were several benefits to performing such a rewrite. To begin with there is an immediate and dramatic increase in execution speed, up to a factor of between 10 and 20 times. Next, because C is a high level language that is

supported by both the PC and DSP development environments, functions written for the PC can potentially be ported to the DSP environment with little initial modification. The word 'initial' is used here because the normal process of developing real time code for a DSP usually involves multiple stages of code profiling, refinement and optimisation of the software until real time objectives are met. However the first stage typically involves using non-optimised, easy to understand (easily readable) routines that are known to be functionally correct. An obvious place to source these routines is the 'C' PC simulation model that has been verified against its equivalent MATLAB model.

The next stage, labeled 'Phase 3' in , would not ordinarily be part of the standard development phase in an industry project. In this case however, such a stage provided several benefits including familiarisation with the DSP development environment, and gaining experience with host/DSP HPI communications, while maintaining the overall software structure as similar to the PC based simulation as possible. The goal of this phase is to split the PC PSAM communication simulation into two parts. The first part includes the transmitter and the fading channel processes, while the second part includes all the PSAM receiver functionality and executes on the DSP. Communication of the large quantities of data involved takes place via the HPI.

Following this, a return to the MATLAB environment is warranted for the development of the necessary modifications to the simulation to account for real world, streaming, implementation requirements. This will be explained in more detail in section A.5.

The final stage involves interfacing the EVM to the digital receiver hardware and implementing the timing synchronisation algorithm.

A.3. PC ANSI C Block Based Simulation

The C simulation performs the signal processing tasks shown in the simulation signal flow diagram illustrated in Figure A.2. The input to the simulation is a binary (not text) parameter configuration file containing

various user selected simulation parameters. This file is the output of a specially created MATLAB m-file.

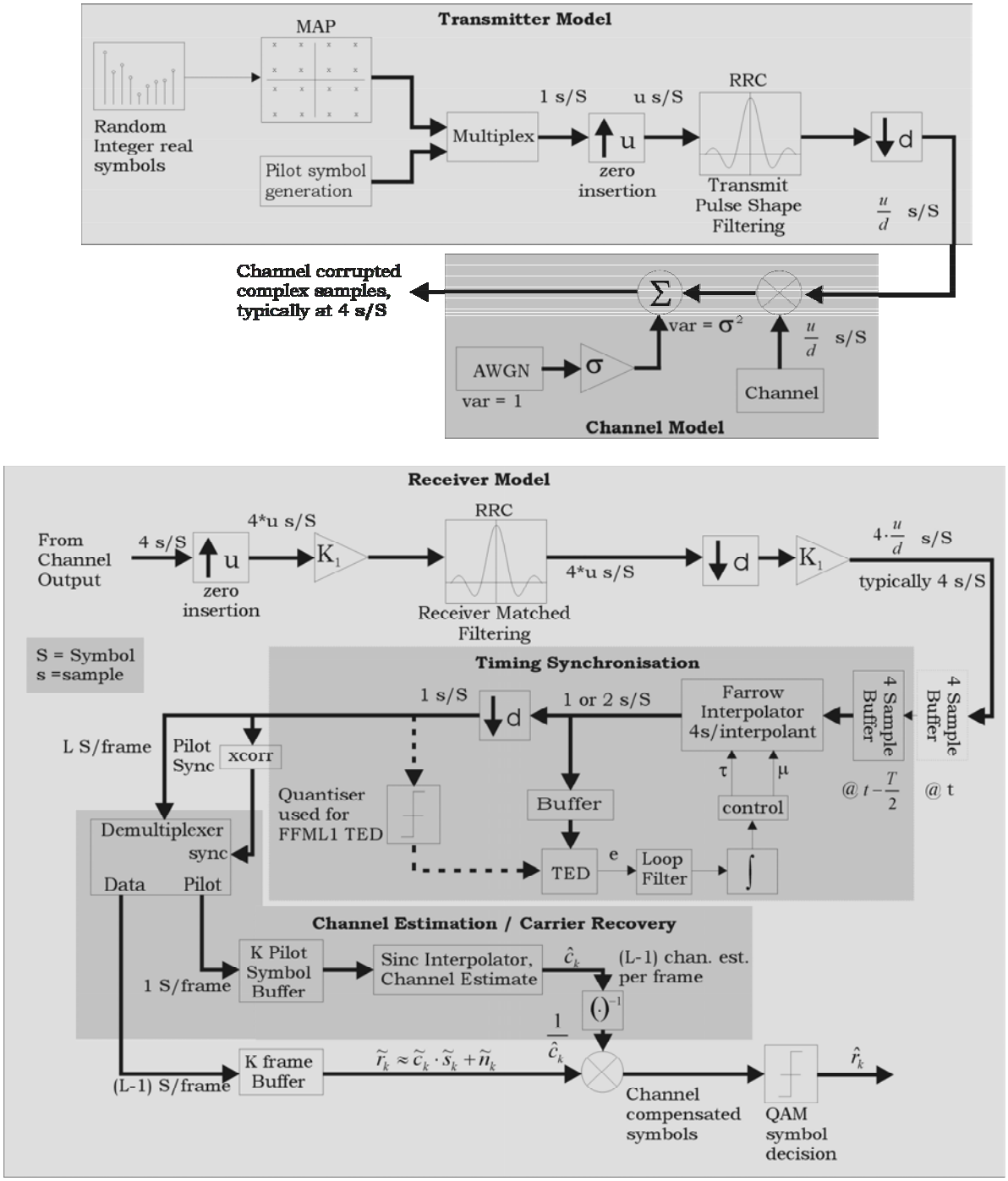

Figure A.2 PSAM communication simulation signal flow diagram.

The thick signal flow lines in Figure A.2 above represent signal samples with a complex number data type, representing the baseband I and Q samples. Thin signal lines represent real numbers.

A.3.1. Transmitter Model

The transmitter simulation model is relatively straightforward. It involves the generation of a block of random integers taking on values between 0 and M-1, where M is the constellation size, for example M=16 for 16 QAM. The discrete integer uniform random number generator used is based on the Mitchell-Moore algorithm as detailed in [59]. The source code was obtained from www.planet-source-code.com [60]. A block is typically composed of N_f frames of symbols, each consisting of N_s-1 data symbols. These parameter values are all modifiable by the user. Typical values for many of the simulations that were run included N_f=800 frames of (N_s-1)=14 data symbols each.

The transmitter generates N_f random pilot symbols, using the same discrete uniform random number generator mentioned above. The data symbols, at this point are still real integers with values between 0 and M-1. They are mapped to their complex baseband I and Q form, with the help of a constellation look up table. At the same time the pilot symbols are multiplexed into the data symbol block so that there is one pilot symbol per frame, making the total frame length N_s=15 in the typical case mentioned earlier.

At this point each symbol is represented by only one complex sample. A waveform level simulation such as this requires more samples per symbol for reasonable accuracy. It is generally recommended that a simulation's sampling rate should be between 4 and 16 times the simulation bandwidth [11]. The exact multiple used will depend on the nature of the system being simulated, and the presence or absence of any system non-linearities. This over sampling rate helps to avoid signal aliasing problems, and to provide accurate modelling of filters (reduces frequency warping effects).

The signal sampling rate is increased by a factor of *u*, through the insertion of *(u-1)* zeros between every sample, and subsequent filtering with the root raised cosine pulse shaping transmit filter. This filter is designed for a sample rate of *u* samples per symbol. This process of zero insertion followed by low pass filtering is a form of interpolation.

The root raised cosine transmit filter is implemented as an FIR filter. The coefficients are generated by calculating appropriately spaced samples of the time domain impulse response of the desired frequency domain filter function. The resulting sampled impulse response is windowed with a spectral shaping function (Hamming in this case) to reduce spectral leakage problems which result from truncation of the infinite duration impulse response.

A down-sampling function is provided so that the user is able to independently control the number of samples per symbol presented to the receiver model. This also provides a way to trade off simulation time with accuracy. It has been found that presenting the signal at a sample rate of 4 samples per symbol to the receiver algorithm, and subsequent processing within the receiver at that rate, does not result in any significant degradation in symbol error rate performance, and provides a significant boost in execution speed. It does however have some negative impact on the timing error estimate's variance as a function of signal to noise ratio, E_b/N_o.

A.3.2. Channel Model

The channel model relies heavily on the generation of continuous normally distributed random numbers. Random numbers with arbitrary probability distributions are generated by first generating a random number having a uniform distribution, and then applying the appropriate functional transformation. To achieve statistically reliable receiver error counts it is very important to use a uniform random number generator with good statistical properties. As is pointed out in [61], the implementation of random number generators (rand()), in many ANSI C libraries is flawed, so rather than take the risk of using a flawed system random number generator the author has utilised a generator that is known to have passed all statistical tests (for randomness) that it has been subjected to so far. The generator is based on the Park and Miller Minimal Standard generator with a Bays-Durham shuffle of the output, and additional safe guards, to remove low order serial correlations [61].

The gaussian generator is used to produce two statistically independent blocks (I and Q) of random double precision floating point values with zero mean and unit variance. These blocks have the same length as the current transmitted signal block, which, to use our earlier notation, is $N_f N_s f_s$. Rayleigh fading channel taps are now produced according to the model introduced in Chapter 2. The two independent gaussian random sequences are scaled and filtered with a 3rd order butterworth IIR filter. The coefficients for this filter are pre-calculated by MATLAB and are contained in the configuration file loaded by the simulation on start up. The 3dB bandwidth of this filter is determined by the desired normalised fade rate, $f_D T$, of the channel. The output of the filtering process is an array of complex Rayleigh faded channel taps.

A complex valued additive white Gaussian noise (AWGN) vector is produced using the same generator as mentioned above. It is then scaled appropriately and added to the faded signal array. In this case the scaling of the complex noise determines the signal to noise ratio (E_b/N_o) of the signal. It is important to be able to accurately set this in a simulation because it is usually desired that various receiver performance characteristics be plotted as a function of E_b/N_o.

A high level software flow chart illustrating the structure of the C simulation is shown in Figure A.3. Note that there are three main loops involved. The innermost loop increments through each E_b/N_0 signal array column. For computational efficiency, the transmitter only generates one sequence of random data symbols per block. The fading is applied to this block and is stored in column 1 of the signal array. For each E_b/N_0 value specified in the simulation configuration file, an appropriately weighted copy of the complex noise vector is added to a copy of the 'reference' signal in column 1. This is stored in the next available column of the array. The receiver function only processes one signal column at a time. Once the receiver has processed all E_b/N_0 signal columns in the current symbol block, the next higher loop is executed, causing a new block of symbols to be generated by the transmitter.

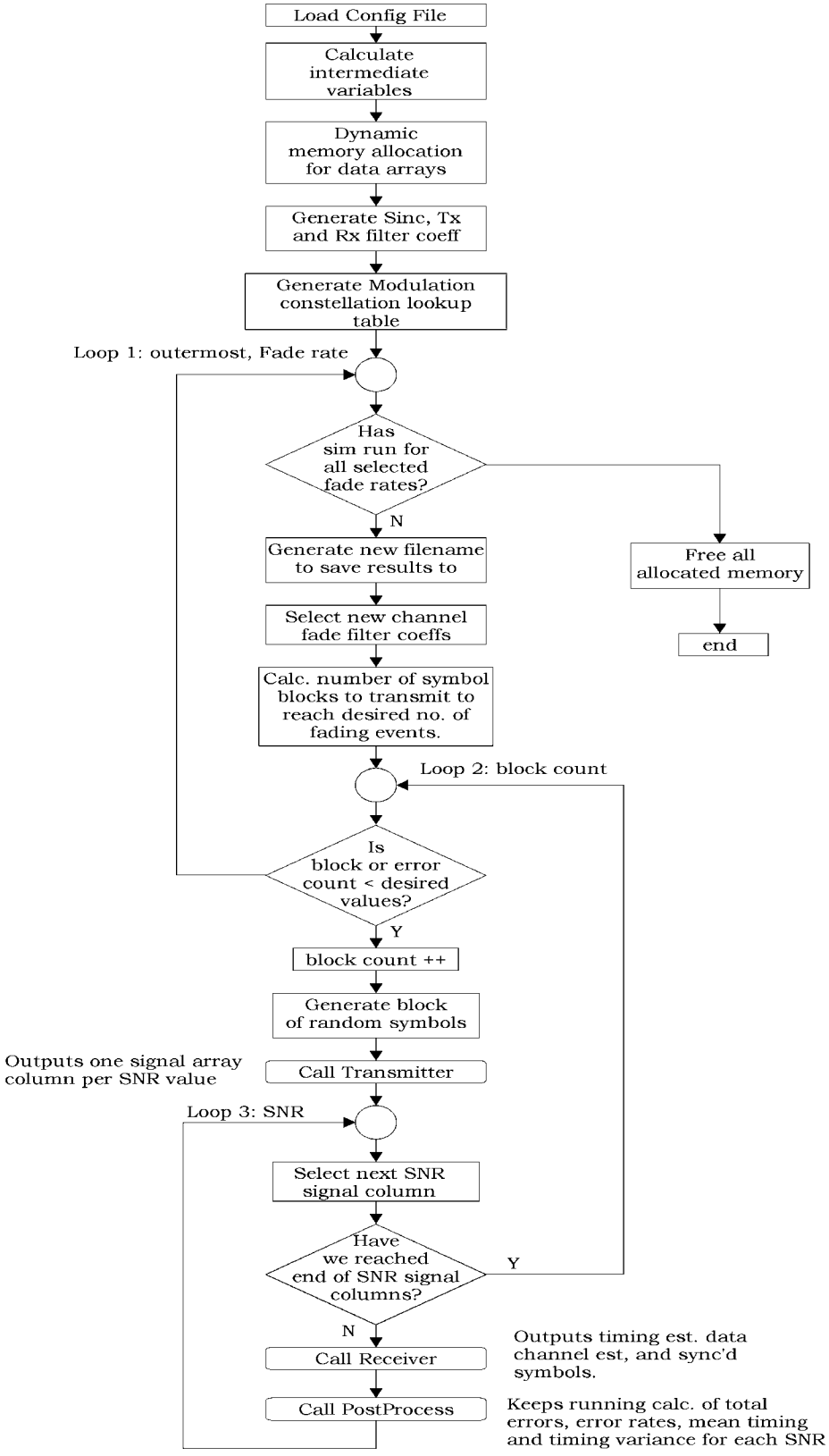

Figure A.3 High level view of the PSAM simulation software structure.

This loop will be executed until the simulation termination conditions have been met. Two conditions must be satisfied before the simulation, at the

current channel fade rate setting, will terminate. The first is that the specified error count has been reached, and the second is that the specified number of channel fade events has been met or exceeded. The final, and outermost, loop will repeat the entire simulation for each value of the channel fade rate specified in the configuration file. The simulation results for each unique value of the fade rate, $f_D T_s$, are saved in a separate file.

A.4. Hybrid PC/DSP C Block Based Simulation

This development stage provided an excellent introduction to the DSP development board and software development tools. The aim was to maintain the software structure as close as possible to that used in the PC simulation described above, but to off-load the PSAM receiver aspect of the simulation to the EVM. This was only possible due to the large amount of external memory available on the EVM, a total of 2 banks of 1M X 32bit, or 8 Mbytes of 100MHz SDRAM. No attempt was made to optimise the efficiency of the receiver simulation code that ran on the DSP, apart from turning on simple compiler optimisations. Efficiency was not one of the requirements of this stage of development. As a result this hybrid simulation ran significantly more slowly than the pure PC version. Some comparative execution times illustrating this, and reasons for the poor performance are presented later in this section.

A number of predefined (by the author) DSP memory locations, known to both the DSP and the host, allow the DSP to communicate the addresses of relevant dynamically allocated buffers to the host. As the DSP dynamically allocated memory for each of its required data buffers, it would save a copy of the buffer address in one of these 'known' locations. Because of the way things were designed, the host is able to implicitly associate a particular DSP memory location as holding the address of a specific buffer in the DSP receiver. Once the host received the signal from the DSP to indicate that it had

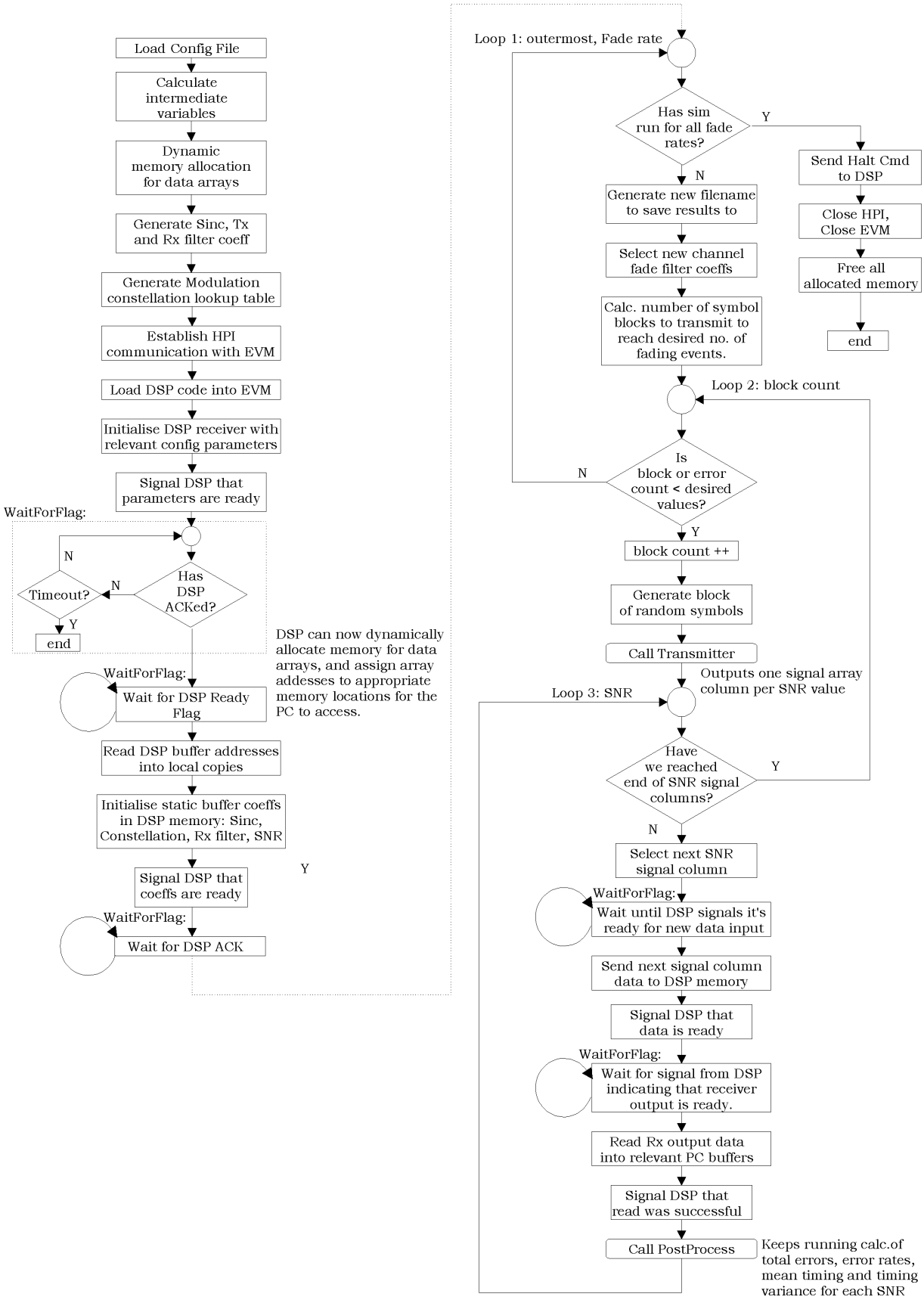

Figure A.4 Host side of the hybrid PC/DSP PSAM communications simulation.

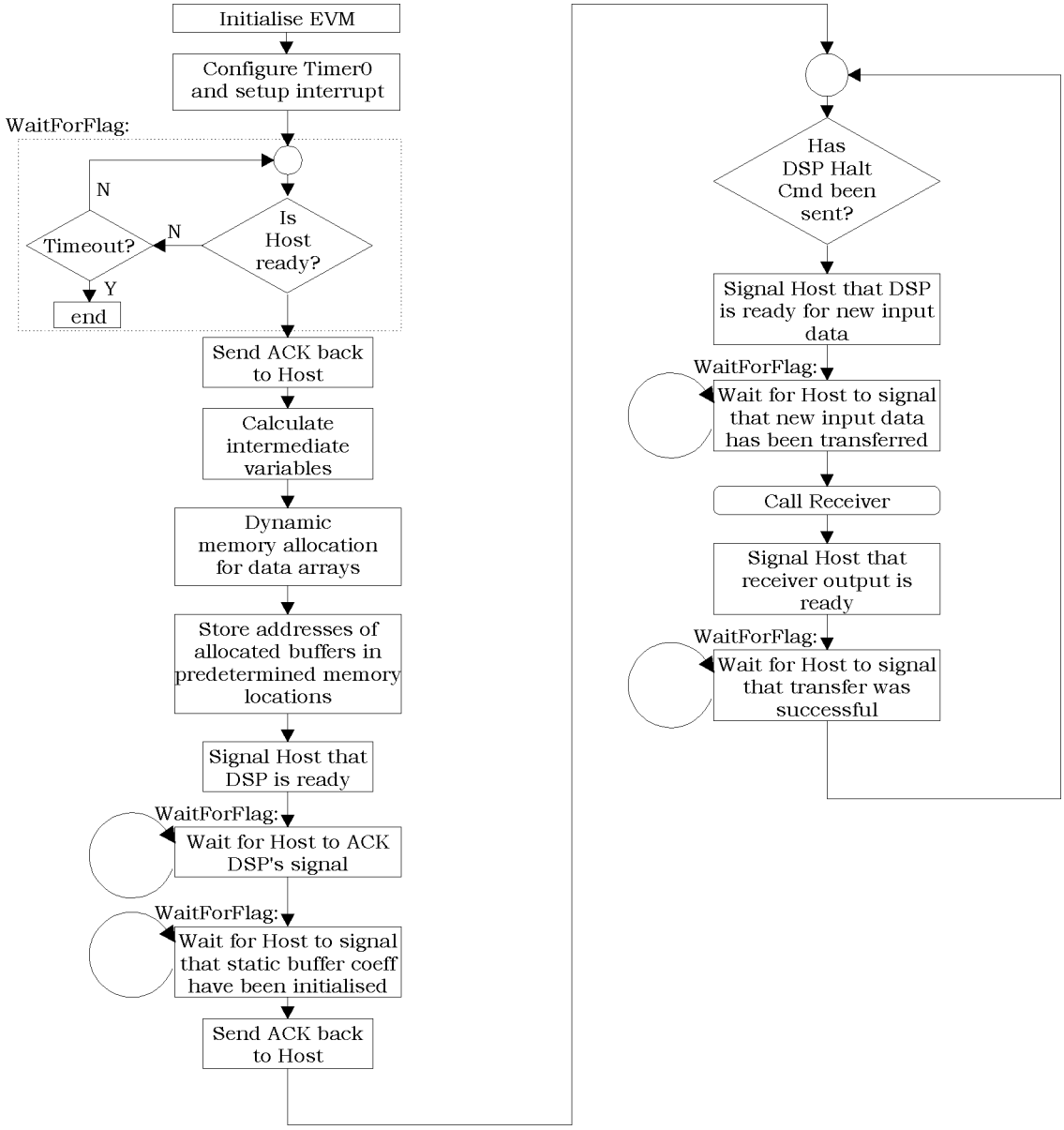

Figure A.5 DSP receiver software structure for the hybrid PC/DSP PSAM communications simulation.

completed allocating memory for data buffers, it would read all the addresses from DSP memory and store them in local variables for later use when transferring data to and from the DSP.

Processing synchronism between the DSP and the Host PC is maintained through a simple handshaking protocol which utilised the polling of shared memory flags as the method of communication. The host and DSP software structure is illustrated in the flow charts of Figure A.4 and Figure A.5. These diagrams should help clarify the following brief explanation of the workings of the hybrid simulation.

Several flags are used, each with a predefined function, known to both the host and the DSP. One flag '*uDsp_ReadyforDataInputFlag*' is used to manage the transfer of transmitter signal data to the DSP. Following the generation of a new block of channel corrupted transmitter signal data, the host polls this flag to determine when the DSP is ready for new data input. Once the data had been transferred successfully, the host resets the flag. This is a signal to the DSP that the new data is now ready for processing.

Once the DSP has finished processing a block of input data, it signals this to the host by asserting the '*uDsp_ReadyforDataOutputFlag*'. The host, having been sitting in a polling loop waiting for this signal, immediately transfers all appropriate results buffers from DSP memory to buffers in host memory. It indicates a successful transfer to the DSP by clearing the same flag. The host now executes the post processing routine which performs an ongoing analysis of the receiver results, the most important functions of which include, counting total symbol errors to date, calculating symbol error rate, mean timing estimate and the timing estimate variance.

A.4.1. Hybrid Simulation Performance

As an example of the performance of the hybrid simulation, the following figures were recorded for a simulation consisting of 12 blocks of symbols, 800 frames per block, 15 symbols per frame, and an SNR range from 0 to 30dB in increments of 5. The times recorded for the PC only simulation (PIII, 500MHz), the hybrid simulation with no DSP compiler optimisation, and the hybrid simulation with compiler optimisation (level –o3), were 1m 12s, 10m 8s, and 7m 46s, respectively. As the reader will note, the hybrid simulation suffers a significant performance penalty for utilising the EVM in the role of the simulated receiver. Reasons for this poor performance include:

- No attempt at efficiency in writing of the C code. Direct port from the PC platform.
- No use of optimised DSP math libraries, just standard ANSI C libraries.

- DSP only running at 100MHz, and external SDRAM is only clocked at half DSP clock rate. It is possible to increase this to 133MHz.
- No attempt was made to utilise the Direct Memory Access (DMA) controller. In this application it could have been used for the transfer of large chunks of data from slow external to fast internal data memory for the CPU to process.

A.5. MATLAB Simulation Modification for Streaming Data

To adapt the MATLAB simulation for real world streaming data conditions, it is necessary to recognise the two main limitations of the block based simulation software architecture. Firstly, the integer part of the timing estimates produced by this implementation were in the form of symbol block index offset values. This index offset value, denoted τ, would normally converge to an integer value between $-N/2$ and $+N/2$, where N is the number of samples per symbol, normally assumed to be an integer value. The original block based implementation simply steps through the very large input signal sample array by a fixed number of samples, N, each iteration. This is a problem in real world streaming situation. Consider the case where there is actually a non integer number of samples per symbol. This is in fact the most likely scenario in a real system.

A.5.1. Non Integer Samples per Symbol

To have an exactly integer number of samples per symbol, even assuming perfectly stable oscillator frequencies at both the transmitter and the receiver, requires that the receivers free running but fixed frequency signal sample rate be an exact multiple of the signal baud rate. In the case of this application the DDC had a programmable decimation factor which set the baseband I/Q sample rate relative to the *nominal* input IF sample rate of 40 MHz. The decimation factor can only take on discrete values between 16 and 32768. As an example, consider a signal baud rate of 4800. If it was desired to have 8 samples per symbol at the receiver then the DDC output sample rate would

have to be set to exactly 38400 Hz. This gives rise to a decimation requirement given by Eq A.1.

$$HDF = \frac{40\times10^6}{4\times f_s} = 260.4167 \tag{A.1}$$

This cannot be represented exactly by the 15 bit decimation integer. The closest value is 260. This will give a sample rate of 38461.5 Hz. The true number of samples per symbol is therefore,

$$N = \frac{38461.5}{4800} = 8.013 \tag{A.2}$$

This simple example shows that combinations of baud rate and receiver sampling frequency exist which will result in non integer number of samples per symbol. This is even before we account for oscillator frequency inaccuracies (small offsets from nominal), and instabilities.

The effect of this scenario on the symbol synchronisation algorithm implementation in the block based simulations is a continuously ramping value of the timing index offset value, τ. This occurs because the algorithm is stepping through the signal array by a constant N (8 in this example) samples each loop. This is actually very slightly less than one symbol period which results in a constant timing error which the synchronisation loop integrates to form the continuously ramping timing estimate. This is entirely analogous to the response of a standard first order phase locked loop to a constant frequency offset between its local oscillator and the input reference signal.

This approach is not suitable for the streaming case where we are working on a sample by sample basis rather than on a huge array of signal samples. The modification detailed in the next section accommodates this difference.

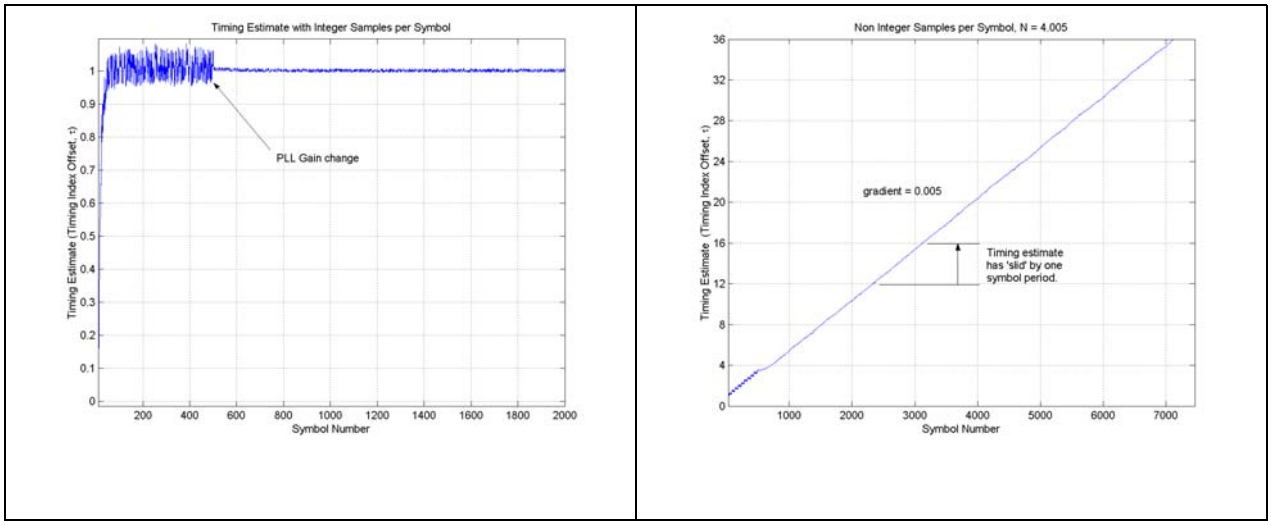

Figure A.6 Timing estimate output of the block based timing synchronisation algorithm with exactly 4, and with 4.005 samples per symbol.

A.5.2. The Sliding Input Window and Numerically Controlled Oscillator

To accommodate the fact that timing must be established and maintained on a sample by sample basis, the estimate cannot be allowed to change by several physical symbol locations as it can in the block-based simulations used to this point. The timing estimate must become window based. This means that a window of several samples (the number of which is dependent on the nominal number of samples per symbol) must be maintained, within which the optimal timing instant exists.

New samples are continually shifted into this window, or buffer. A mechanism is required for determining when the synchronised output symbol interpolant should be calculated. As before, this mechanism is provided by the familiar PLL feedback structure, although in this implementation the function of the pure integrator is replaced by a numerically controlled oscillator, the operation of which will be discussed in more detail shortly.

In the case of an integer number of samples per symbol, the number of samples shifted into the window before it is time to calculate a new interpolant (symbol output) will always be the same. On the other hand, when there are a non-integer number of samples per symbol, the number of samples shifted into the window before it is decided that a new symbol (and its optimum sampling instant) is contained within the window will not always

be the same. Once every few hundred, or few thousand symbols (depending on the magnitude of the offset from integer samples per symbol) one more, or one less sample may be shifted into the window before it is time to calculate a new output symbol sample. In this way the window will always 'keep up' with the symbols, ensuring that the optimal symbol timing estimate will always lie within the window. This process is illustrated in Figure A.7.

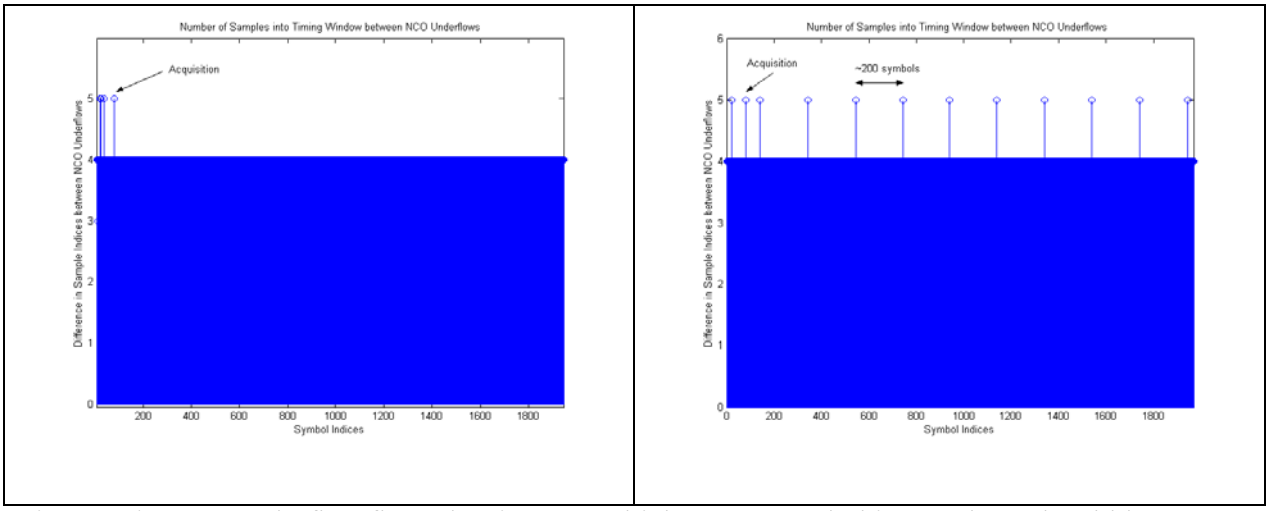

Figure A.7 NCO underflow flags the times at which a new symbol interpolant should be calculated. In this case the time is in terms of the number of samples that have been clocked into the timing window. The cases shown are for exactly 4 samples per symbol and for 4.005.

Note, that in the case of an exact integer number of samples per symbol (4 in this example), the NCO always underflows after 4 samples have been shifted into the timing window, as expected. In the next case, there are 4.005 samples per symbol, thus to keep up with the optimal symbol timing instant the NCO periodically underflows only after 5 samples have been shifted into the window. This occurs approximately every 200 symbols.

Numerically Controlled Oscillator

The numerically controlled oscillator (NCO) becomes the replacement for the pure discrete integration block used in the block based simulations. The NCO is simply a counter which is operated with an average period that's equal to the symbol rate. Underflow of the counter indicates that a new interpolant is to be calculated, using signal samples currently stored in the timing window buffer. Thus, the base-point index of the samples to be presented to the Farrow interpolator, are identified by flagging the right set of signal samples

rather than explicitly computing the index, as was done in the block based simulations. Further detail on the use of an NCO in this application can be found in [62].

The fractional interval, μ_k, is calculated from the contents of the NCO's counter upon underflow. This is illustrated in Figure A.8. The NCO can be defined by a difference equation.

$$\eta(m) = \left[\eta(m-1) - W(m-1)\right] \bmod N \tag{A.3}$$

where the NCO counter contents computed at the m[th] clock tick is designated as η(m) and the NCO control word as *W(m)*. The control word is adjusted by the timing recovery loop so that the output of the interpolator is the symbol sampled at near optimal timing.

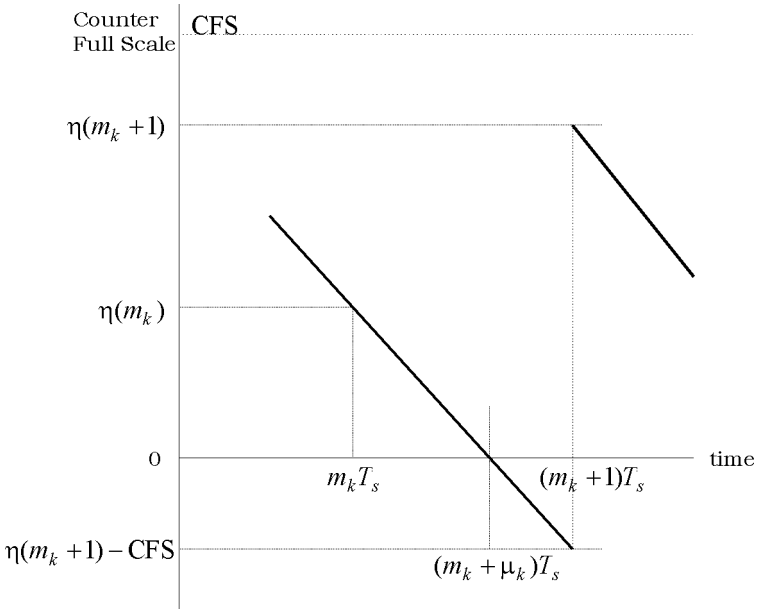

Figure A.8 NCO relationships

Using simple geometry it is straightforward to show that μ_k is given by,

$$\begin{aligned}\mu_k &= \frac{\eta(m_k)}{\mathrm{CFS} - \eta(m_k+1) + \eta(m_k)} \\ &= \frac{\eta(m_k)}{W(m_k)}\end{aligned} \tag{A.4}$$

In practice the NCO control word is simply calculated as the sum of a fixed component, and a variable component, where the variable component is just

the product of the timing error detector output and a gain factor. The single line of MATLAB code that implements the main function of the NCO is shown below.

```
eta=mod((eta-G*TEDerror-1),CounterFullScale);
```

A.6. Digital Receiver / DSP Streaming Data

This marked the final stage of the development process. The goal of this phase is to implement the timing synchronisation sub-system on the EVM. It would operate in real time on streaming I and Q samples from the digital down converter in the digital radio receiver. It was also intended that the performance of each of the three synchronisers would be examined under various fading channel conditions and compared with results obtained earlier from the simulations.

The goals of this phase were only partially met due, as mentioned earlier, to time constraints caused by multiple development board failures. The most significant achievement of this phase was the design and implementation of a software architecture suitable for real time processing of streaming symbol sample data, and a functioning symbol synchronisation sub-system, in which each of the three TED's could be enabled. This was developed with low baud rate data (4.8Kbaud) however significantly higher data rates could be accommodated with additional software optimisation. Detail regarding the implementation is contained in this section.

A.6.1. The Radio Test System

The digital radio system used during this stage in the development process is illustrated in the block diagram of Figure A.9. It consists of the digital transmitter, the channel, and the digital receiver, which in this case comprises the RF and digital down converter hardware, connected by high speed serial bus to the Texas Instruments EVM and host PC. The channel used initially was a direct atmospheric RF link between the transmitter and receiver, later to be replaced by a coaxial cable connection to eliminate

received signal strength and phase fluctuations due to the movement of people within the laboratory environment. It was intended to replace this direct RF link with a Hewlett Packard RF Channel Simulator (HP 11756B) at a later stage. This would allow testing of the synchronisation performance of the DSP software under different fading channel conditions.

The transmitter was set to transmit a QPSK pseudo random data sequence. The data rate was set to 4800 baud and no transmit pulse shaping filtering was performed. These were default settings in the transmitter software and the author saw no reason to modify them at this point in the receiver synchronisation software development process. Full details on the digital transmitter and receiver test system, designed by Peter Green during the course of his Masters and Phd work in the Department of Electrical Engineering at the University of Canterbury, can be found in [54].

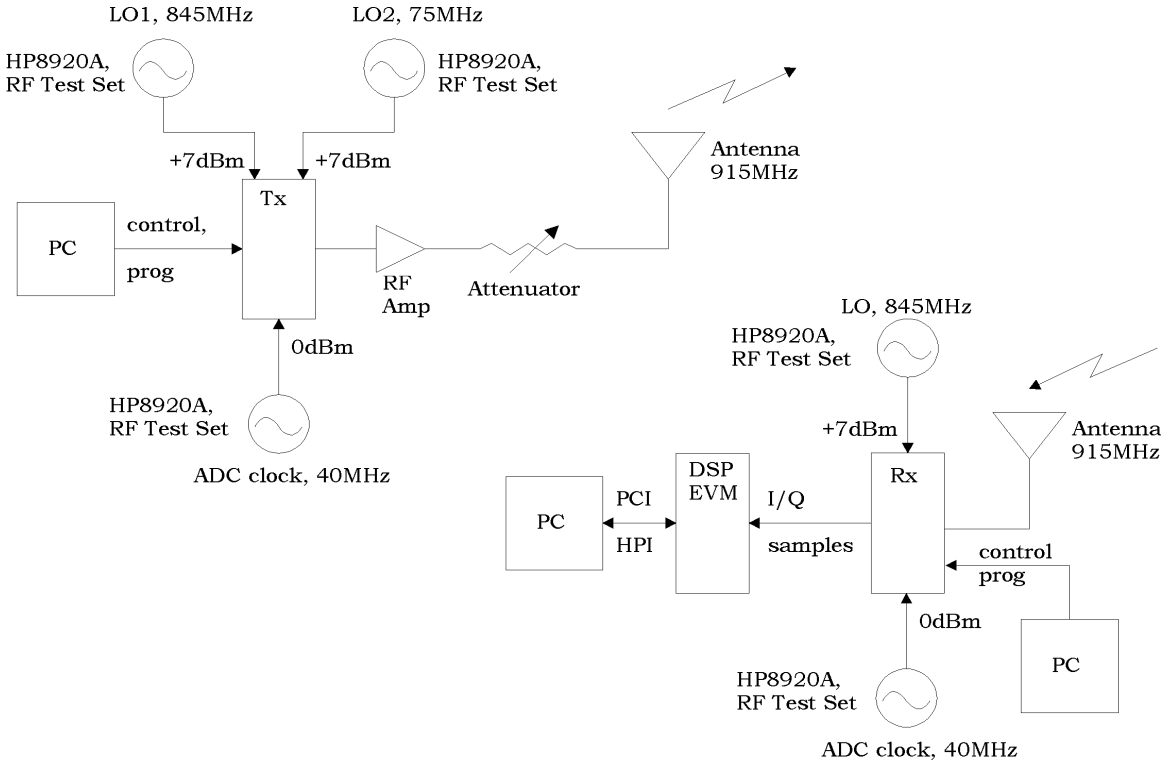

Figure A.9 Hardware configuration of the digital radio test system used during development of the real time software on the EVM.

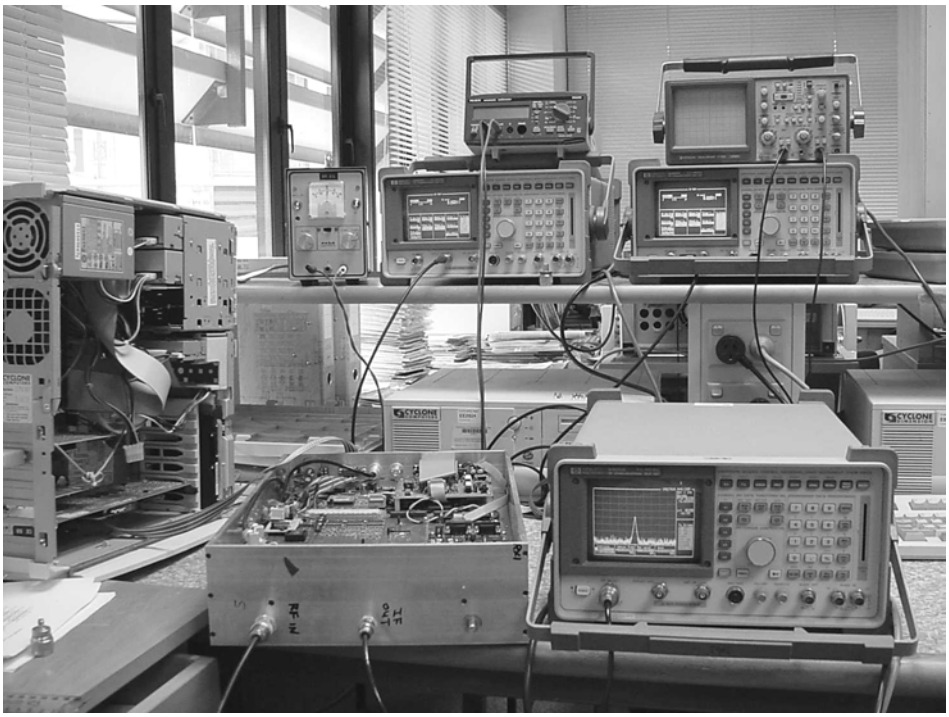

Figure A.10 Equipment configuration for the receiver side of the system.

A.6.2. Digital Down Converter / EVM Interface

The interface between the digital receiver and the EVM was a high speed serial communications bus, labelled the special communications bus (SCB) in [54]. The SCB is basically a direct connection to the DDC and consists of 4 main signal lines, including I, Q, IQSTB, and IQCLK. A simple EVM daughterboard was designed to interface the digital radio's SCB and the DSP's two high speed buffered synchronous serial ports. This is illustrated in Figure A.11. Additional detail concerning EVM daughterboard design requirements can be found in [63].

The DDC (Intersil HSP50016) is user configurable. I and Q samples from the DDC can be programmed for output on the I and Q signal lines simultaneously, or alternatively as I followed by Q, on the I signal line only. IQCLK is the I/Q sample data bit clock, and IQSTB is the frame synchronisation pulse which marks the start of each sample word. The output format of the sample words can be set to 16. 24, 32, or 38 bit fixed point 2's complement, signed magnitude or offset binary. It is also possible to select single precision floating point as the numerical format.

EVM		SCB	
Pin #	Signal Name	Pin #	Signal Name
29	XFSR0	9	IQSTB
41	XFSR1	9	IQSTB
27	XCLKR0	10	IQCLK
39	XCLKR1	10	IQCLK
30	XDR0	11	I
42	XDR1	12	Q

Table A.1 EVM peripheral connector and digital receiver SCB pin connectivity.

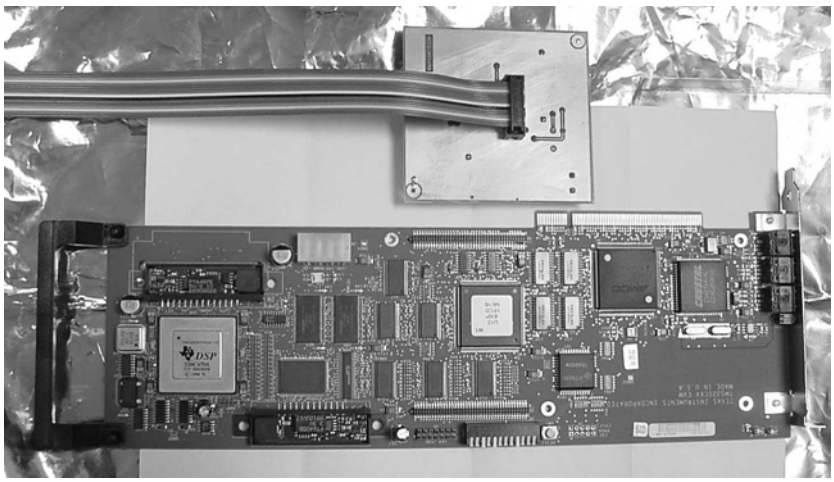

Figure A.11 The 'C6701 EVM and the DDC/EVM interface PCB.

Another programmable feature is the decimation ratio. Given a particular IF signal input sample rate, this ratio will set the output baseband signal sample rate. Depending on the baud rate of the signal involved, it is quite possible that due to the restriction of decimation ratios to discrete values, a non integer number of samples per symbol will result. The digital receiver algorithms prior to the output of the timing synchronisation system must be implemented such that they are able to handle this.

In this application, several of these settings were tried, however the final configuration utilised the I followed by Q, single precision floating point option. This choice meant that the DSP was not required to perform the fixed

to floating point number conversion. In addition only a single serial port needed to be utilised, and consequently only one DMA channel was required for handling the incoming DDC data samples.

A.6.3. DSP Software Development

The software structure was interrupt driven, with the primary interrupt being due to the DMA controller.

DMA and Ping Pong Double Buffering

Direct Memory Access (DMA) controllers are included as a standard peripheral device on many DSPs these days. A DMA controller is an extraordinarily useful way of removing the burdensome task of data movement from the CPU. The controller is capable of handling many kinds of data movement around the DSP memory map. Once it has been configured, it can operate in the background, independently of the CPU, transferring data while the CPU performs its number crunching tasks. The 'C6701 DSP incorporates a 4+1 channel DMA controller. The additional 1 channel is called the auxiliary channel and allows the DMA controller to handle requests from the HPI. Full details concerning the capabilities and configuration of the DMA controller can be found in [64].

In this application only one DMA channel was used. This was configured to automatically transfer I/Q samples from the serial port (McBSP0) and store them in one of two buffers that were allocated for the purpose. A ping pong double buffering scheme was adopted for both the input and output data. Although this scheme uses more memory than a single circular input buffering scheme, it offers certain advantages. The main one is that the CPU is able to access and process one of the input buffers at the same time the DMA controller is filling the other. This relies on the designer splitting the allocation of the two buffers across two memory blocks.

In the 'C6701 DSP the 64Kbytes of internal data RAM is organised into two blocks of 32Kbytes. Each of these blocks is organised into eight interleaved 2K

banks of 16 bit halfwords. This is illustrated in Figure A.12 This memory is single ported, meaning that there can only be one access to each bank per cycle. As long as the DMA controller and CPU are accessing different banks there is no conflict. To take maximum advantage of the DMA controller it is important that the designer minimise the possibility of memory conflicts between it and the CPU. Conflicts cause a delay of at least one cycle to the CPU or the DMA controller, depending on which one has been given the higher priority.

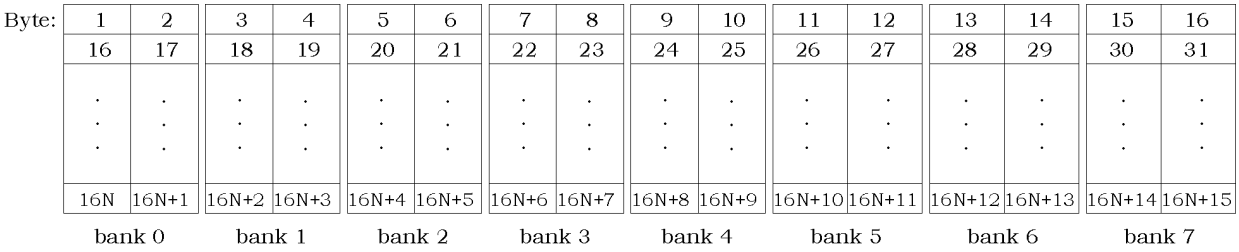

Figure A.12 The interleaved, byte addressable, memory bank structure of the 'C6701 internal data RAM. Source: Code Composer Studio v2 online help.

As mentioned above, with the dual buffering scheme, memory conflicts can be eliminated by locating one of the buffer pairs in block 0 and the other in block 1. These blocks do not share any memory banks.

In this application, DMA channel 0 was configured to operate in autoinitialisation mode. This means that once a block of data has been transferred, the DMA channel will automatically re-initialise itself without any intervention from the CPU. Because separated dual buffers were being used to store the input data from the serial port, it was necessary to use the advanced address generation capability of the DMA controller to calculate the start address of the second buffer (in block 1). Once the second buffer has been filled the DMA controller re-initialises the destination address for DMA Channel 0 to be the start of the first input buffer. Thus, the destination for the blocks of data from the serial port, will bounce between the two defined input buffers. This ping pong buffer structure is illustrated by Figure A.13.

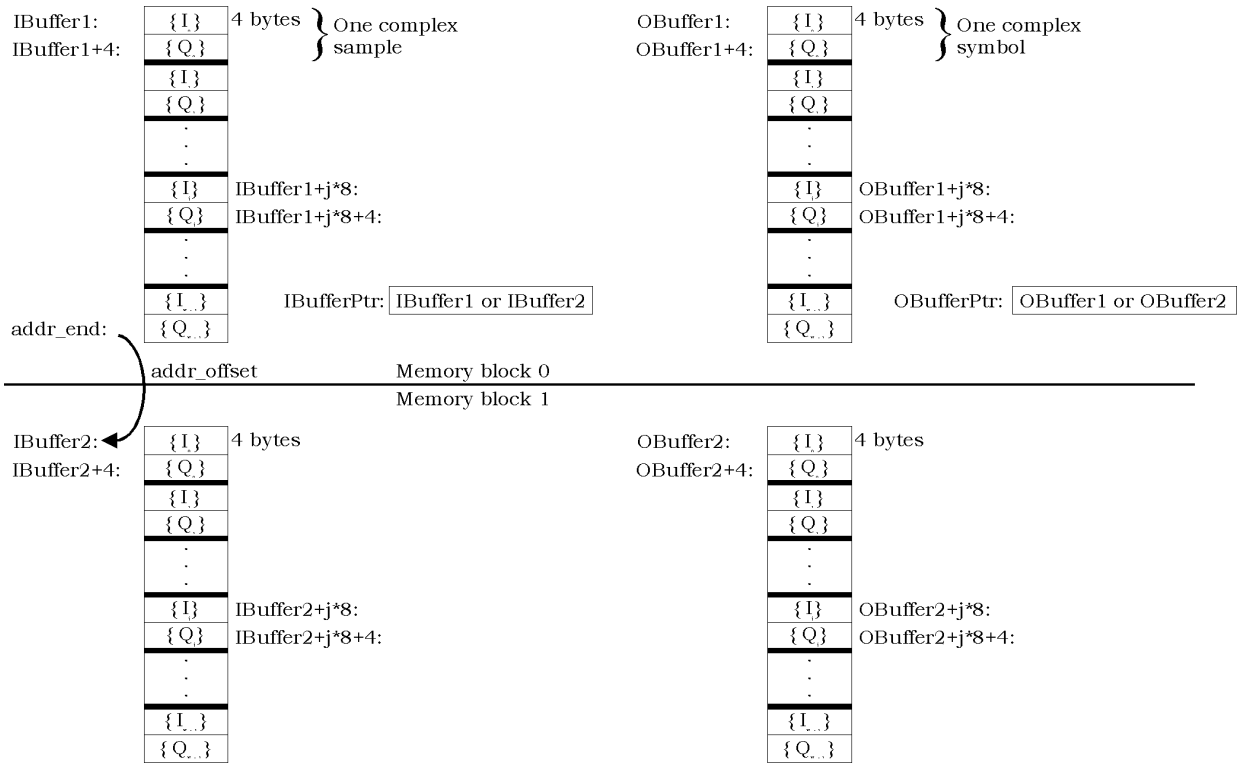

Figure A.13 Input and output ping pong double buffering scheme. The DMA controller handles the filling of the two input buffers.

The CPU must keep track of which of the input and output buffers it should be using at any given time. Two pointers are defined for this purpose, *IBufferPtr* and *OBufferPtr*. The input buffer pointer is updated whenever one of the input buffers is filled. The filling of a buffer generates a DMA Channel 0 interrupt to the CPU. While the CPU executes the interrupt service routine, which includes the signal processing and synchronisation routines, the DMA controller is busy filling the other input buffer. The overall software structure for this application is shown in flow chart form in Figure A.14.

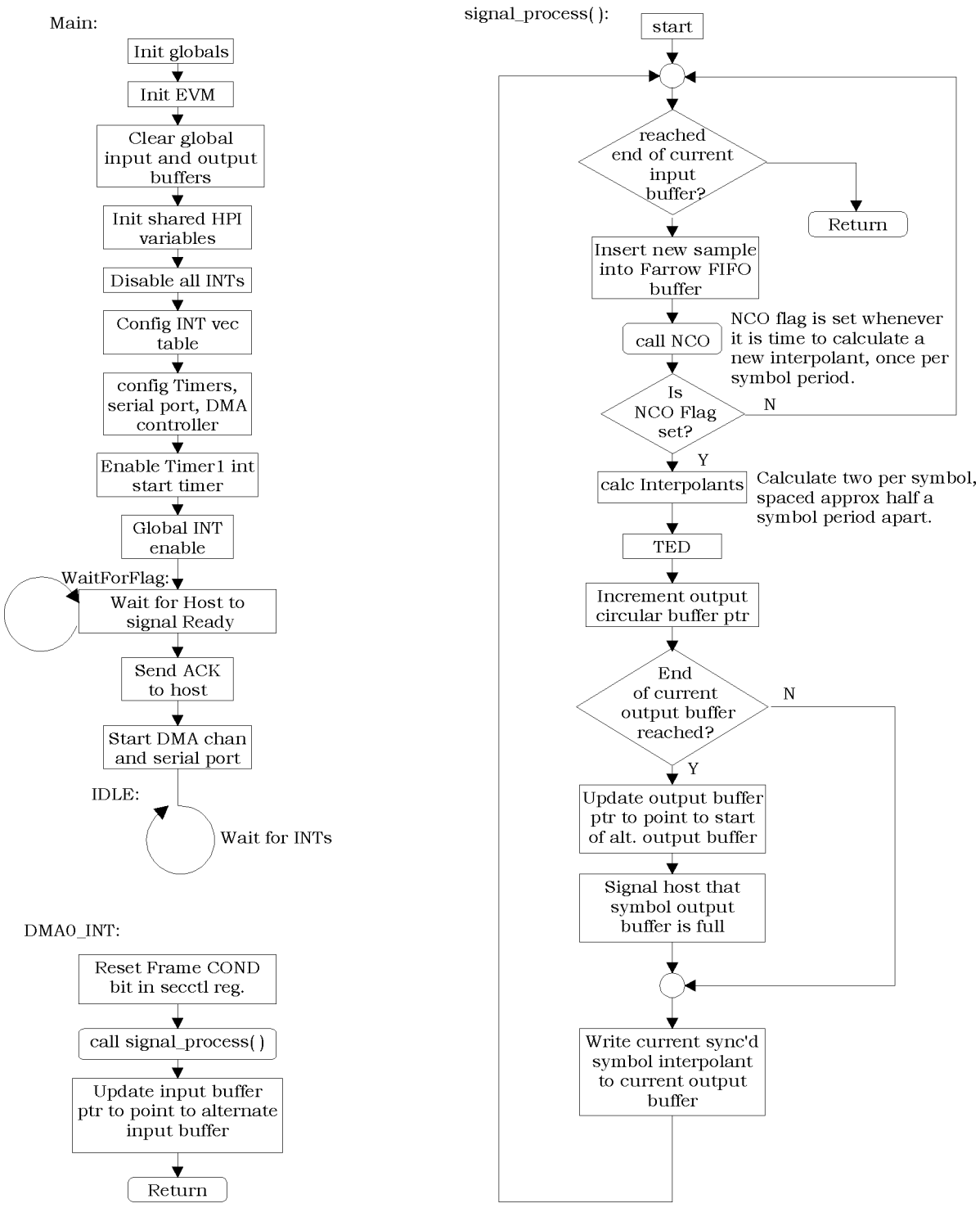

Figure A.14 Simplified flow diagram illustrating the structure of the real time DSP streaming synchronisation software framework.

A.6.4. Real Time Performance and Code Profiling

One of the steps in the DSP software design cycle is the profiling of important functions within the application. This gives the designer a good idea about which functions are contributing the greatest execution times to the overall

application, and thus where most advantage may be obtained from optimisation efforts. If real time objectives are not met initially there are a couple of levels of optimisation available to the designer. These include refining the C code further, and if that is not enough, rewriting some of the most critical functions in linear assembly code.

The profiling results for the most important of the signal processing functions, all executed from within the function *signal_process()*, which is itself executed from within the DMA Channel 0 interrupt service routine, are shown in Table A.3. The function execution lengths are given in terms of CPU cycles which is a more general measure than time, since it is independent of CPU clock frequency. Execution times can be easily calculated from these figures for any given clock frequency.

In all profile cases shown below, the standard C6000 run-time support (RTS) library has been replaced with a new run-time support library called 'FastRTS' from Texas Instruments. The 'C67x FastRTS library is a hand coded assembly optimised floating-point math function library for the 'C67x family of DSP's. It contains all the floating-point functions found in the standard C6000 RTS library. By using the routines found in this library instead of those in the existing standard RTS library, it is possible to achieve significant increases in execution speed without re-writing existing code.

As an example, consider the execution times of the *AmplitudeDirectedTED()* function using the standard RTS library, and with the FastRTS library. No compiler optimisation was used.

	Standard RTS Library		FastRTS Library	
	Average (cycles)	Max (cycles)	Average (cycles)	Max (cycles)
AmplitudeDirectedTED()	1630	1630	368	369

Table A.2 Profile statistics revealing the benefit of using the assembly optimised FastRTS library.

	No Compiler Optimisation		Compiler Optimisation –o3	
Software Functions	**Average (cycles)**	**Maximum (cycles)**	**Average (cycles)**	**Maximum (cycles)**
DMA Chan 0 INT	~1010000	Not Available	311700 357445 398880	312584 358670 404358
AmplitudeDirected()	368	369	200	200
GardnerTED()	34	34	20	20
GabeTED()	638	647	360	369
QAMDecideSingle()	453	463	178	186
CubicFarrow()	811	812	292	295
NCO()	86	138	80	220
mvmult() (assembly)	103	103	103	103

Table A.3 Software profiling results illustrating the effect of compiler optimisation on execution times.

A.6.5. Timing Error Detector Complexity Comparison

Three execution cycle counts are shown for the DMA Chan 0 interrupt service routine in Table A.3 because its execution time depends on which of the timing error detectors is being used in the symbol synchroniser. The complexity of the timing error detectors, as measured by computational effort, increase in the following order, Gardner, Amplitude Directed, and Watkins FFML1 (otherwise known as the Gabe TED).

In terms of their current software implementation, the Amplitude Directed TED is ten times as computationally intensive as the Gardner TED. The main reason for this is the required use of the complex signum function. The Watkins FFML1 TED is approximately eighteen times as computationally intensive as the Gardner TED, in its current implementation.

A.6.6. Operational DSP Software

The real-time DSP software structure described to this point was successful in establishing symbol synchronisation in an AWGN channel environment. The opportunity to extensively test the performance of the synchronisation algorithms, particularly in a fading channel environment did not present itself due to the reasons stated earlier. The following few figures show I-Q plots of received QPSK DDC signal samples, stored in the DSP's input buffer (1024 complex samples) and the associated synchronised symbol samples, stored in the output buffer.

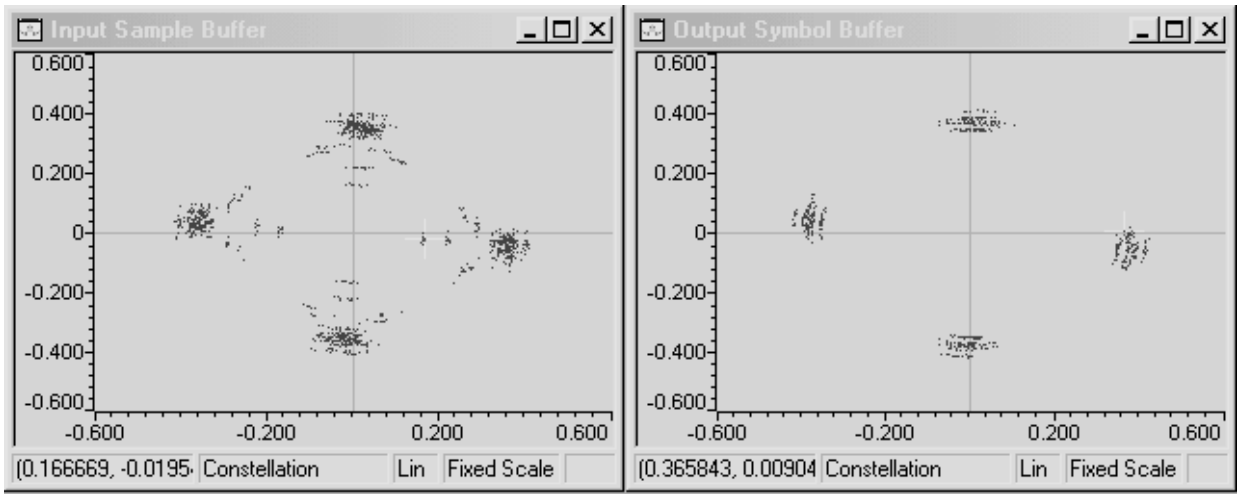

Figure A.15 Screen capture plot from Code Composer Studio showing the I-Q plots of the received QPSK DDC samples from the digital receiver, and on the right is the symbol synchronised output samples from the synchroniser algorithm.

The left hand plot in Figure A.15 illustrates the received unsynchronised DDC samples. The complex samples are arriving at a rate of 38400 samples per second, and the nominal baud rate is 4800, giving approximately 8 samples per symbol. The synchronisers output symbol synchronised samples are placed in the output buffer. The output buffer was defined to be 1024 complex samples, which means that 8 input buffers worth of samples are processed to fill up one output symbol buffer. The size of these buffers is easily modifiable at compile time.

The first thing to note from these plots is that there is significant noise present on the input signal. If things were ideal then the QPSK constellation points would be 4 small points rather than the 'clouds' seen in Figure A.15

The source of this noise was never identified. The next thing to note is that the constellation is rotated from its ideal position. This indicates that there is a constant phase offset between the transmitter and the receiver. In the test setup the transmitter and receiver were frequency and phase locked to each other through the 10MHz reference input/outputs on the HP8920A RF test sets. This was done because the DSP software did not have any phase synchronisation function and determining whether timing synchronisation had been achieved through visual examination of the constellation plot, would have been impossible if the constellation was continually rotating due to frequency offset between the transmitter and receiver.

The right hand plot of Figure A.15 indicates that lock has been achieved. This is apparent from the lack of samples showing on trajectories between the constellation points. If synchronisation had not been correctly established then some of the output symbol samples will not lie on valid constellation points, and will show up somewhere on a trajectory linking the constellation points. The synchronisation sub-system established timing lock for all three of the TED's under investigation.