EUROPEAN SPACE AGENCY

FINAL REPORT
ESTEC Contract No. 8437/89/NL/RE

The work described in this report was under ESA contract. Responsibility for the contents resides in the authors who prepared it.

DIGITAL DEMODULATOR SYNCHRONIZATION

PERFORMANCE ANALYSIS

Thierry Jesupret
Marc Moeneclaey
Gerd Ascheid

R.A. Harris (TST),
ESTEC Technical Manager

June, 1991

SAIT Electronics S.A.
Chaussée de Ruisbroek, 66
B-1190 Brussels
(BELGIUM)

Rijksuniversiteit Gent
Sint-Pietersnieuwstraat, 41
B-9000 Gent
(BELGIUM)

CADIS GmbH
Jülicher Str. 320
D-5100 Aachen
(GERMANY)

ABSTRACT

This study deals with carrier and symbol synchronization for narrowband (excess bandwith not exceeding 100%) uncoded BPSK, QPSK and OQPSK signals in a satellite communication environment.

The emphasis is on synchronizer algorithms suitable for digital implementation. In a digital synchronizer carrier phase errors are corrected digitally in a phase rotator and timing errors are corrected by digital interpolation.

The first part gives a systematic classification of various synchronizer algorithms (many of them being related to the maximum-likelihood criterion). A general overview of analytical methods and software models used for the analysis of feedback and feedforward synchronizer performance is presented.

Part II deals with the tracking performance of various carrier and symbol synchronizers, operating on a noisy modulated signal. These synchronizers are analyzed under the assumption of "ideal" conditions : additive white Gaussian noise channel, zero carrier frequency error, Nyquist-I pulse at the matched filter output, and ideal receiver operations (no quantization,...).

In Part III, we investigate the influence of tracking errors on the bit error rate (BER). The BER degradation due to tracking errors is defined as the increase of E_b/N_O, necessary to maintain the same BER as without tracking errors. As in Part II, "ideal" conditions are assumed.

The synchronizer must be capable of tracking a frequency error due to oscillators frequency drift, Doppler shift and phase noise. This puts a lower bound on the feedforward filter or loop noise bandwith. Part IV considers phase noise and frequency error influence on tracking performance. These topics are strongly related with the postfilter structure of the feedforward synchronizer.

In Part V, we compare qualitatively the carrier and clock synchronizers acquisition performance. In feedback synchronizers, the memory involved in the acquisition process (transition from some initial state to steady state) is introduced by the loop filter and VCO. In feedforward synchronizers memory is introduced by the postfilter.

Part VI concerns the cycle slipping properties of carrier and symbol synchronizers. Synchronizers should be operated in such a way that cyle slipping is a very rare event. Consequently, studying cycle slips by means of computer simulations is not adequate. We restrict our attention mainly to feedback synchronizers, using Fokker-Planck theory applied to discrete time systems.

Influence of distortions which may be present on the channel (including receiver front-end) are taken into account in Part VII. Distortions may be split between linear and nonlinear. In this study, we consider only linear distortion with independent timing transfer functions and nonlinear distortions resulting from high power amplifiers (HPA).

In the last part of the report, we study the influence of demodulator implementation aspects on synchronizer performance. We investigate quantization on matched filter, carrier and symbol synchronization algorithms, loop filters, phase rotator and interpolator. The optimimum signal level with a given ADC has been studied in more detail.

DIGITAL DEMODULATOR SYNCHRONIZATION PERFORMANCE EVALUATION

TABLE OF CONTENTS

GLOSSARY G–1

PART I: SYNCHRONIZATION ALGORITHMS AND ANALYSIS METHODS

I.1 Preliminaries I–1
- I.1.1 Signal representation I–2
- I.1.2 DD and NDA log–likelihood functions I–4
- I.1.3 Feedback and feedforward synchronization; coupling between algorithms I–7
- I.1.4 Digital implementation I–10

I.2 Classification of synchronization algorithms I–16
- I.2.1 Carrier synchronization I–17
- I.2.2 Symbol synchronization I–25
- I.2.3 Joint carrier and symbol synchronization I–39

I.3 Theoretical analysis methods I–43
- I.3.1 Feedback synchronization I–43
- I.3.2 Feedforward synchronization I–49

I.4 Simulation tools and models I–52
- I.4.1 Introduction I–52
- I.4.2 General models I–54
- I.4.3 Carrier synchronizer models I–60
- I.4.4 Symbol synchronizer models I–67

Figures I–74

Appendix I.A I–134

PART II: TRACKING PERFORMANCE UNDER IDEAL CONDITIONS

II.1	Preliminaries		II–1
	II.1.1	Introduction	II–1
	II.1.2	Theoretical analysis of tracking performance	II–2
	II.1.3	Simulation analysis of tracking performance	II–10
II.2	Tracking performance results for carrier synchronization		II–13
	II.2.1	Introduction	II–13
	II.2.2	DDML, Decision directed maximum likelihood carrier synchronization. Timing assumed to be known	II–14
	II.2.3	NDAML, Non data–aided maximum likelihood carrier synchronization. Timing assumed to be known	II–17
	II.2.4	V&V, Non data–aided Viterbi and Viterbi carrier synchronizer. Timing assumed to be known	II–19
	II.2.5	DDML$\bar{\text{T}}$, Decision directed maximum likelihood carrier synchronization independent of symbol timing	II–21
	II.2.6	NDAML$\bar{\text{T}}$, Non data–aided maximum likelihood carrier recovery independent of symbol timing	II–24
	II.2.7	NDASQFF$\bar{\text{T}}$O, MSK type carrier recovery	II–27
	II.2.8	Self–noise performance	II–28
II.3	Tracking performance results for symbol synchronization		II–30
	II.3.1	Introduction	II–30
	II.3.2	DD symbol synchronization	II–33
	II.3.3	NDA symbol synchronization	II–37
	II.3.4	Self–noise performance	II–41
II.4	Tracking performance results for joint carrier and symbol synchronization		II–45

II.5 Conclusions II–46

Figures II–49

PART III: THE INFLUENCE OF TRACKING ERRORS ON THE BIT ERROR RATE

III.1 Derivation of the expression for the BER degradation III–1
- III.1.1 Carrier phase errors only III–1
- III.1.2 Timing errors only III–6
- III.1.3 Both carrier phase errors and timing errors III–8

III.2 Validity of the Gaussian approximation III–9

III.3 Discussion of the expression for the BER degradation III–13

III.4 Numerical results III–18

III.5 Conclusions III–25

III.A Appendix III–26

Figures III–28

PART IV: PHASE NOISE AND FREQUENCY ERRORS

IV.1 Introduction IV–1

IV.2 Postfiltering IV–4
- IV.2.1 A (linearized) optimality approach IV–4
- IV.2.2 Planar filtering IV–15
- IV.2.3 Frequency estimation IV–16

IV.3 Influence of phase noise on tracking performance IV–17

IV.4 Influence of frequency errors on tracking performance IV–22
IV.4.1 Introduction IV–22
IV.4.2 Carrier synchronization IV–22
IV.4.3 Symbol synchronization IV–27
IV.4.4 Burst transmission IV–28

IV.A.1 Appendix: Kalman filter approach IV–29

Figures IV–32

PART V: ACQUISITION

V.1 Introduction V–1
V.1.1 Feedback synchronizer V–1
V.1.2 Feedforward synchronizer V–8

V.2 Feedback synchronizer acquisition performance V–11
V.2.1 Feedback carrier synchronization V–12
V.2.2 Feedback symbol synchronization V–14

V.3 Feedforward synchronizer acquisition performance V–16
V.3.1 Feedforward carrier synchronization V–16
V.3.2 Feedforward symbol synchronization V–19

V.4 Summary V–22

Figures V–23

PART VI: CYCLE SLIPPING PERFORMANCE

VI.1 Preliminaries VI–1
VI.1.1 Feedback synchronizer model VI–2
VI.1.2 Cycle slips and their effect on the BER VI–3
VI.1.3 Basic analytical results from Fokker–Planck theory VI–5
VI.1.4 Cycle slip rates of second–order loops VI–6

VI.2 Application of Fokker–Planck theory to feedback carrier and symbol synchronizers VI–8
VI.2.1 Discrete–time algorithms VI–9
VI.2.2 Self–noise at equilibrium points VI–10
VI.2.3 Approximation for small E_b/N_o VI–12

VI.3 Cycle slip results for feedback synchronizers VI–15
VI.3.1 Feedback carrier synchronizers VI–16
VI.3.2 Feedback symbol synchronizers VI–18
VI.3.3 Miscellaneous VI–21

VI.4 Feedforward synchronizers VI–23
VI.4.1 Introduction VI–23
VI.4.2 Feedforward carrier synchronizers VI–24
VI.4.3 Feedforward symbol synchronizers VI–29

VI.5 Conclusions VI–30

Figures VI–31

PART VII: TRACKING PERFORMANCE UNDER NONIDEAL CHANNEL CONDITIONS

VII.1 Preliminaries VII–1
VII.1.1 Introduction VII–1
VII.1.2 Linear distortions VII–2
VII.1.3 Nonlinear distortions VII–9
VII.1.4 Feedback and feedforward synchronizer VII–12
VII.1.5 Self noise measurement VII–12

VII.2 Tracking performance results for carrier synchronization – linear distortions VII–15
VII.2.1 Qualitative analysis VII–15
VII.2.2 Carrier synchronizer performance VII–17

VII.3 Tracking performance results for symbol synchronization
– linear distortions VII–18
VII.3.1 Qualitative analysis VII–18
VII.3.2 Symbol synchronizer performance VII–20

VII.4 Tracking performance results for carrier synchronization
– nonlinear distortions VII–24
VII.4.1 Mean phase shift VII–24
VII.4.2 Slopes VII–27
VII.4.3 Loop noise VII–28
VII.4.4 Self noise VII–28
VII.4.5 Conclusions VII–31

VII.5 Tracking performance results for symbol synchronizers
– nonlinear distortions VII–32
VII.5.1 Introduction VII–32
VII.5.2 Synchronizer performance neglecting self noise VII–32
VII.5.3 Self noise VII–35
VII.5.4 Conclusions VII–36

Figures VII–39

PART VIII: INFLUENCE OF DIGITAL SIGNAL PROCESSING ON TRACKING PERFORMANCE

VIII.1 Preliminaries VIII–1
VIII.1.1 Digital implementation aspects VIII–1
VIII.1.2 Digital number representation VIII–3
VIII.1.3 Analog to digital conversion (ADC) VIII–4
VIII.1.4 Digital filters VIII–14
VIII.1.5 Loop filter, NCO VIII–15
VIII.1.6 Simulation study VIII–20

VIII.2 Tracking performance results for carrier synchronization VIII–24
VIII.2.1 Preliminaries VIII–24
VIII.2.2 Phase rotator VIII–26
VIII.2.3 Carrier synchronization: Quantization at the matched filter output VIII–28
VIII.2.4 Carrier synchronization: Quantization before the matched filter VIII–31
VIII.2.5 Carrier synchronization : Conclusions VIII–33

VIII.3 Tracking performance results for symbol synchronization VIII–35
VIII.3.1 Preliminaries VIII–35
VIII.3.2 Interpolator VIII–36
VIII.3.3 Timing synchronization: Quantization at the matched filter output VIII–37
VIII.3.4 Timing synchronization: Quantization before the matched filter VIII–41
VIII.3.5 Feedforward timing synchronizers VIII–43
VIII.3.6 Symbol timing synchronizers : Conclusions VIII–44

VIII.4 Conclusions VIII–45

Figures VIII–47

TABLES T1

REFERENCES R1

GLOSSARY

The following is a list of symbols and abbreviations, frequently used in this report. The mathematical relation between some of these quantities is given in Table T1, 'Miscellaneous definitions'.

a_m	m–th bit (±1) in the in–phase channel
$\hat{a}_m$	receiver's decision about a_m
b_m	m–th bit (±1) in the quadrature channel
$\hat{b}_m$	receiver's decision about b_m
B_L	one–sided noise bandwidth of synchronizer
BER	bit error rate
BPSK	biphase shift keying
DD	decision–directed
DTTL	data transition tracking loop (symbol synchronizer)
E_b	energy per bit
E_s	energy per symbol (BPSK : $E_s = E_b$; (O)QPSK : $E_s = 2E_b$)
f	frequency variable
FB	feedback
FF	feedforward
g(t)	matched filter output pulse ($g(t) = h(t) \odot h(-t)$)
GA	Gardner symbol synchronizer
$h_I(t)$	received in–phase pulse (BPSK, (O)QPSK : $h_I(t) = h(t)$)
$h_Q(t)$	received quadrature pulse (BPSK : $h_Q(t) = 0$; QPSK : $h_Q(t) = h(t)$; OQPSK : $h_Q(t) = h(t - T/2)$)
$h_I(-t)$	matched filter impulse response (in–phase channel)
$h_Q(-t)$	matched filter impulse response (quadrature channel)
$h_I'(-t)$	derivative matched filter impulse response (in–phase channel)
$h_Q'(-t)$	derivative matched filter impulse response (quadrature channel)
Im[.]	imaginary part of complex quantity
M&M	Mueller and Müller symbol synchronizer
mM&M	modified Mueller and Müller symbol synchronizer
ML	maximum likelihood
MSK	minimum shift keying
N_o	one–sided power spectral density of additive noise at receiver input
n(t)	additive noise at receiver input
NDA	nondata–aided, nondecision–aided

$p(k;\tilde{\tau})$	matched filter output sample at $kT + \tilde{\tau}$ (in–phase channel)
$p'(k;\tilde{\tau})$	derivative matched filter output sample at $kT + \tilde{\tau}$ (in–phase channel)
$p_R(k;\tilde{\tau},\tilde{\theta})$	real part of phase–rotated (over $\tilde{\theta}$) matched filter output sample at $kT + \tilde{\tau}$ (in–phase channel)
$p_R'(k;\tilde{\tau},\tilde{\theta})$	real part of phase–rotated (over $\tilde{\theta}$) derivative matched filter output sample at $kT + \tilde{\tau}$ (in–phase channel)
OQPSK	offset QPSK
$q(k;\tilde{\tau})$	matched filter output sample at $kT + \tilde{\tau}$ (quadrature channel)
$q'(k;\tilde{\tau})$	derivative matched filter output sample at $kT + \tilde{\tau}$ (quadrature channel)
$q_R(k;\tilde{\tau},\tilde{\theta})$	real part of phase–rotated (over $\tilde{\theta}$) matched filter output sample at $kT + \tilde{\tau}$ (quadrature channel)
$q_R'(k;\tilde{\tau},\tilde{\theta})$	real part of phase–rotated (over $\tilde{\theta}$) derivative matched filter output sample at $kT + \tilde{\tau}$ (quadrature channel)
QPSK	quadri–phase shift keying
Re[.]	real part of complex quantity
sgn(.)	sign function
SNR	signal–to–noise ratio
SQ	squaring carrier or symbol synchronizer
t	time variable
T	symbol interval (BPSK : one symbol = one bit, 1/T = bitrate; (O)QPSK : one symbol = two bits, 2/T = bitrate)
T_s	mean time between slips (cycle slip rate = $1/T_s$)
V&V	Viterbi and Viterbi carrier synchronizer
$x_L(t)$	complex baseband representation of received signal
$\odot$	operator denoting convolution
*	operator denoting complex conjugation
α	rolloff parameter
θ	unknown carrier phase [rad]
$\hat{\theta}$	carrier phase esimate [rad]
$\tilde{\theta}$	trial value of carrier phase esimate [rad]

τ	unknown timing [s]
$\hat{\tau}$	timing estimate [s]
$\tilde{\tau}$	trial value of timing estimate [s]
ϕ	carrier phase error [rad] : $\phi = \theta - \hat{\theta}$
ϵ	normalized timing error [dimensionless] : $\epsilon = (\tau - \hat{\tau})/\mathrm{T}$

I. SYNCHRONIZATION ALGORITHMS AND ANALYSIS METHODS

I.1 PRELIMINARIES.

This study deals with carrier and symbol synchronization for narrowband (excess bandwidth not exceeding 100%) uncoded BPSK, QPSK and OQPSK signals in a satellite communication environment. The emphasis is on synchronizer algorithms suitable for digital implementation.

Many synchronizer algorithms can be related to the maximum–likelihood (ML) criterion. According to this criterion, the estimates of the carrier phase and the symbol timing jointly maximize the likelihood function, which is a function depending on the received signal and on trial values of the carrier phase and the symbol timing [Fra'80].

When deriving synchronization algorithms from the ML criterion, one usually assumes that the satellite channel behaves like an additive white Gaussian noise (AWGN) channel, which adds white Gaussian noise but does not distort the transmitted signal. In principle, a more realistic channel model (including linear and nonlinear distortions, and various kinds of interference) could be taken into account when applying the ML criterion, but it turns out that this approach is far too complicated. Instead, one derives synchronization algorithms from the ML criterion under the AWGN assumption, and investigates the performance of the resulting synchronizers for the AWGN channel and for more realistic channel models. We have adopted the same strategy in the present study.

I.1.1 Signal representation.

Assuming that the AWGN channel model holds and the received carrier frequency is known, the complex envelope of the received signal is given by

$$x_L(t) = \left[\sum_m a_m h_I(t-mT-\tau) + j \sum_m b_m h_Q(t-mT-\tau) \right] e^{j\theta} + n(t) \qquad \text{(I.1.1–1)}$$

where $\{a_m\}$ and $\{b_m\}$ are the in–phase and quadrature binary data symbols, which randomly take the values –1 and 1; $h_I(t)$ and $h_Q(t)$ are the in–phase and quadrature baseband pulses, normalized to unit energy; n(t) is a complex–valued Gaussian white noise process, with statistically independent real and imaginary parts, each having a two–sided spectral density of $N_o/2E_b$; E_b denotes the energy per bit (E_b is related to E_s, the energy per symbol, by $E_s = E_b$ for BPSK or $E_s = 2E_b$ for (O)QPSK); 1/T represents the symbol rate (BPSK : one symbol = one bit, (O)QPSK : one symbol = one in–phase bit + one quadrature bit); θ and τ are the unknown carrier phase and symbol timing. The representation (I.1.1–1) is valid for all three modulation formats considered in this study :

– BPSK : $h_I(t) = h(t)$, $h_Q(t) = 0$.

$$x_L(t) = \sum_m a_m h(t-mT-\tau) e^{j\theta} + n(t)$$

– QPSK : $h_I(t) = h_Q(t) = h(t)$.

$$x_L(t) = \sum_m (a_m + jb_m) h(t-mT-\tau) e^{j\theta} + n(t)$$

– OQPSK : $h_I(t) = h(t)$, $h_Q(t) = h(t - T/2)$

$$x_L(t) = \left[\sum_m a_m h(t-mT-\tau) + j \sum_m b_m h(t-mT-T/2-\tau) \right] e^{j\theta} + n(t)$$

In many cases, the representation (I.1.1–1) will allow us to write a single formula and to draw a single block diagram, which is valid for all three considered modulation formats.

We denote by g(t) the response of the matched filter (with impulse response h(–t)) when driven by the baseband pulse h(t), i.e.

$$g(t) = \int_{-\infty}^{+\infty} h(t+u)\, h(u)\, du = h(t) \odot h(-t) \qquad \text{(I.1.1–2)}$$

and $\odot$ denotes convolution. It is assumed that g(t) satisfies the first Nyquist criterion : $g(kT) = 0$ for $k \neq 0$. When deriving numerical results, a cosine rolloff pulse with rolloff α ($\alpha \in (0,1)$) will be taken for g(t); in this case, g(t) and its

Fourier transform G(f) are given by :

$$g(t) = \frac{\sin(\pi t/T)}{\pi t/T} \frac{\cos(\alpha \pi t/T)}{1 - (2\alpha t/T)^2} \tag{I.1.1–3}$$

$$G(f) = \begin{cases} T & |f| \leq \frac{1-\alpha}{2T} \\ \frac{T}{2}\left[1 - \sin\left[\frac{\pi T}{\alpha}\left(|f| - \frac{1}{2T}\right)\right]\right] & \frac{1-\alpha}{2T} < |f| \leq \frac{1+\alpha}{2T} \\ 0 & \frac{1+\alpha}{2T} < |f| \end{cases} \tag{I.1.1–4}$$

I.1.2 DD and NDA log–likelihood functions.

For trial values $\tilde{\theta}$ and $\tilde{\tau}$, the logarithm of the decision–directed (DD) likelihood function is given by [Fra'80]

$$L_{DD}(\tilde{\theta},\tilde{\tau}) = (2E_b/N_o) \left[\sum_m \hat{a}_m \, \mathrm{Re}[p(m;\tilde{\tau})\, e^{-j\tilde{\theta}}] + \sum_m \hat{b}_m \, \mathrm{Im}[q(m;\tilde{\tau})\, e^{-j\tilde{\theta}}] \right] \tag{I.1.2–1}$$

$L_{DD}(.,.)$ is called the DD log–likelihood function. In (I.1.2–1), the complex–valued quantities $p(m;\tilde{\tau})$ and $q(m;\tilde{\tau})$ are obtained by applying $x_L(t)$ to filters which are matched to the baseband pulses $h_I(t)$ and $h_Q(t)$, respectively, and sampling the filter outputs at the instants $mT+\tilde{\tau}$:

$$p(m;\tilde{\tau}) = [x_L(t) \odot h_I(-t)]\big|_{t=mT+\tilde{\tau}}$$
$$q(m;\tilde{\tau}) = [x_L(t) \odot h_Q(-t)]\big|_{t=mT+\tilde{\tau}}$$

For the three modulation formats considered in this study one obtains :

– BPSK : $q(m;\tilde{\tau}) = 0$

– QPSK : $q(m;\tilde{\tau}) = p(m;\tilde{\tau})$

– OQPSK : $q(m;\tilde{\tau}) = p(m;\tilde{\tau} + T/2) = p(m + 1/2;\tilde{\tau})$

The quantities $\hat{a}_m$ and $\hat{b}_m$ in (I.1.2–1) are the receiver's decisions, corresponding to the trial values of the carrier phase and the symbol timing :

$$\hat{a}_m = \mathrm{sgn}(\mathrm{Re}[p(m;\tilde{\tau})\, e^{-j\tilde{\theta}}]) \tag{I.1.2–2a}$$

$$\hat{b}_m = \mathrm{sgn}(\mathrm{Im}[q(m;\tilde{\tau})\, e^{-j\tilde{\theta}}]) \tag{I.1.2–2b}$$

These relations are illustrated in Fig. I.1–1, which holds for BPSK, QPSK and OQPSK. The complex–valued matched filter output signals are sampled at the instants $mT+\tilde{\tau}$, and phase–rotated clockwise over an angle $\tilde{\theta}$; the appropriate real and imaginary parts of the phase rotator outputs are applied to the decision devices. In the case of OQPSK, the in–phase and quadrature components of the received signal $x_L(t)$ are offset by half a symbol interval, i.e. by T/2. As the matched filter impulse responses $h_I(-t)$ and $h_Q(-t)$ are also offset by T/2, decisions about the in–phase bit a_m and the quadrature bit b_m are obtained from samples of the complex–valued matched filter output signals $x_L(t) \odot h_I(-t)$ and

$x_L(t) \odot h_Q(-t)$, respectively, taken at the same instant $mT+\tilde{\tau}$. Although Fig. I.1–1 shows two matched filters (with impulse responses $h_I(-t)$ and $h_Q(-t)$), only one matched filter (with impulse response $h(-t)$) needs to be implemented, as illustrated in Fig. I.1–2. In the cases of BPSK and QPSK, decisions are based upon samples of the complex–valued matched filter output signal $x_L(-t) \odot h(-t)$, taken at the instants $mT+\tilde{\tau}$. In the case of OQPSK, decisions about the in–phase bits and quadrature bits are obtained from samples of $x_L(-t) \odot h(-t)$, taken at the instants $mT+\tilde{\tau}$ and $mT + T/2 + \tilde{\tau}$, respectively; note that sampling $x_L(-t) \odot h(-t)$ at the instant $mT + T/2 + \tilde{\tau}$ is equivalent to sampling $x_L(-t) \odot h_Q(-t)$ at the instant $mT+\tilde{\tau}$.

Synchronizers which produce estimates that maximize $L_{DD}(.,.)$ are decision– directed (DD), because they make use of the receiver's decisions. As we will explain in section I.1.3, it is sometimes preferred to perform nondata–aided (NDA) synchronization, which operates independently of the receiver's decisions. The appropriate NDA likelihood function is obtained by replacing in the DD likelihood function the receiver's decisions by the actual data symbols, and averaging the resulting function with respect to the binary data symbols. Its logarithm is called the NDA log–likelihood function, and is given by [Fra'80]

$$L_{NDA}(\tilde{\theta},\tilde{\tau}) = \sum_m \ln\cosh\left[(2E_b/N_o)\, \mathrm{Re}[p(m;\tilde{\tau})\, e^{-j\tilde{\theta}}] \right] + \sum_m \ln\cosh\left[(2E_b/N_o)\, \mathrm{Im}[q(m;\tilde{\tau})\, e^{-j\tilde{\theta}}] \right] \qquad \text{(I.1.2–3)}$$

Maximization of $L_{NDA}(.,.)$ requires knowledge of E_b/N_o, which determines the operating point of the nonlinear function ln cosh(.). This can be avoided by using in (I.1.2–3) the approximation

$$\ln\cosh x \cong \frac{x^2}{2} - \frac{x^4}{12} \qquad |x| << 1$$

which yields the low E_b/N_o limit of $L_{NDA}(.,.)$:

$$L_{NDA}(\tilde{\theta},\tilde{\tau}) \cong \frac{1}{2}(2E_b/N_o)^2 \sum_m \left[(\mathrm{Re}[p(m;\tilde{\tau})\, e^{-j\tilde{\theta}}])^2 + (\mathrm{Im}[q(m;\tilde{\tau})\, e^{-j\tilde{\theta}}])^2 \right] - \frac{1}{12}(2E_b/N_o)^4 \sum_m \left[(\mathrm{Re}[p(m;\tilde{\tau})\, e^{-j\tilde{\theta}}])^4 + (\mathrm{Im}[q(m;\tilde{\tau})\, e^{-j\tilde{\theta}}])^4 \right] \qquad \text{(I.1.2–4)}$$

In the cases of BPSK and OQPSK, it is sufficient to retain only the first summation in (I.1.2–4); in the case of QPSK, the first summation does not depend on the carrier phase estimate (because $p(m;\tilde{\tau}) = q(m;\tilde{\tau})$), so that the second summation must be considered only when dealing with QPSK carrier synchronization. Approximating (I.1.2–3) by its high E_b/N_0 limit would yield the DD log–likelihood function (I.1.2–1) again [Fra'80]; this approach does not give rise to NDA algorithms.

There exists a large variety of carrier and symbol synchronization algorithms. Most of them produce estimates of the carrier phase and symbol timing which (approximately) maximize the DD or NDA log–likelihood function. A systematic overview of carrier and symbol synchronizers has been presented in [Gar'88]; we give a more extended classification in section I.2.

I.1.3 Feedback and feedforward synchronization; coupling between algorithms.

For both carrier and symbol synchronization, there exist feedback (FB) and feedforward (FF) algorithms. FB synchronizers are able to track slow variations of the carrier phase and symbol timing, such as variations due to a small frequency offset. A disadvantage of FB synchronizers is the extended acquisition time when the initial parameter estimate is close to an unstable equilibrium point (the "hangup" problem). NDA FF synchronizers do not exhibit hangup problems, because of the absence of feedback (note that DD FF synchronizers do suffer from hangup, because of the decision feedback !). A disadvantage of FF synchronizers is their rather high sensitivity to a frequency offset [Vit'83].

In this section, it is assumed that timing correction is obtained by adjusting the phase of the sampling clock, which is controlled by the timing estimate; this is called synchronized sampling. In section I.1.4 we will discuss the case of non–synchronized sampling, where timing correction is obtained by interpolating between samples, taken by a fixed clock; the operation of the interpolator is controlled by the timing estimate.

Assuming that the carrier synchronizer operates on a signal which has been sampled at instants related to the timing estimate $\hat{\tau}$, and that the symbol synchronizer operates on a signal which has been phase–rotated over an angle equal to the carrier phase estimate $\hat{\theta}$ (as we will show in section I.2, this holds for most algorithms), we now consider the four possible combinations of FB and/or FF synchronizers.

– FB carrier synchronization, FB symbol synchronization (Fig. I.1–3)
From the phase rotator output samples, the carrier and symbol synchronizers derive error signals which give an indication about the carrier phase error and symbol timing error, respectively. These error signals are fed back to the sampler and the phase rotator, which adjust their estimates accordingly.

– FF carrier synchronization, FB symbol synchronization (Fig. I.1–4)
As before, the FB symbol synchronizer feeds back to the sampler a timing error signal which is derived from the phase–rotated matched filter output samples. The FF carrier synchronizer makes an estimate of the carrier phase, from observing the matched filter output samples; this estimate is passed to the phase rotator.

– FB carrier synchronization, FF symbol synchronization (Fig. I.1–5)
The FF symbol synchronization algorithm makes use of the signal before the sampler, to make a timing estimate; this timing estimate determines the instants at which the sampler operates. As the symbol synchronizer is assumed to use the carrier phase estimate, the phase rotator must precede the sampler. As before, the FB carrier synchronizer derives from the phase–rotated and sampled matched filter output signal an error signal which is fed back to the phase rotator.
– FF carrier synchronization, FF symbol synchronization (Fig. I.1–6)
As we have assumed that the carrier and symbol synchronizer operation depends on the estimates of the timing and the carrier phase, respectively, the FF carrier synchronizer requires that the phase rotator follows the sampler, whereas for the FF symbol synchronizer the order must be reversed. This problem can be solved by using two phase rotators, so that the FF carrier synchronizer can operate on a signal which has been sampled but not phase rotated, and the FF symbol synchronizer can operate on a signal which has been phase rotated but not sampled.

Our assumption about the carrier and symbol synchronizer operation depending on the timing and carrier phase estimate, respectively, implies that both algorithms are coupled. An additional coupling arises when the algorithms are decision–directed. The mutual interaction between carrier and symbol synchronization may give rise to problems during acquisition, when the carrier phase error and the timing error can assume large values. Such problems can be avoided by using a symbol synchronization algorithm which does not depend on the carrier phase estimate, and/or a carrier synchronization algorithm which does not depend on the timing estimate.

When the symbol synchronization algorithm is independent of the carrier phase estimate, it can operate on the signal at the input of the phase rotator. Figs. I.1–7 and I.1–8 show the cases of carrier–independent FF and FB symbol synchronization, respectively. As the sampler precedes the phase rotator, phase rotation must be performed only at the rate of the sampler; this is in contrast with Fig. I.1–5, where the phase rotator must precede the sampler. Also, when using carrier–independent FF symbol synchronization in combination with FF carrier synchronization, the phase rotator in the upper branch of Fig. I.1–6 is no longer needed.

When the carrier synchronization algorithm is independent of the timing estimate, it can operate on the signal at the input of the sampler. Figs. I.1–9 and

I.1–10 show the cases of timing–independent FF and FB carrier synchronization, respectively. When combining timing–independent FF carrier synchronization with FF symbol synchronization, we can use the configuration of Fig. I.1–9, containing a single phase rotator; this is in contrast with Fig. I.1–6, where two phase rotators are needed.

I.1.4 Digital implementation

In a digital implementation, the receiver does not operate on the continuous–time received complex envelope, but rather on a sampled (and quantized) version of it. When this complex envelope is sampled at a rate above the Nyquist rate, the samples contain the same information as the continuous–time signal (assuming that the effect of quantization can be ignored). As the excess bandwidth of the received signal does not exceed 100%, the Nyquist sampling rate does not exceed 2/T. From these complex envelope samples, the digital receiver must derive the signal samples needed for the synchronization algorithms and for the symbol decisions. In the case of synchronized sampling, the received complex envelope is sampled at instants which are related to the symbol timing; the sampling clock is controlled by the timing estimate. In the case of non–synchronized sampling, the received complex envelope is sampled at instants determined by a fixed clock; samples at instants related to the symbol timing are obtained by means of an interpolator, whose operation is controlled by the timing estimate [Gar'90a]. When using synchronized sampling of the received complex envelope, carrier and symbol synchronizers consequently operate on synchronized samples, so that only feedback symbol synchronization can be realized. When using non–synchronized sampling of the received complex envelope, there is no such restriction on the type of synchronizers that can be implemented. Unless noted otherwise, we will concentrate on digital receivers with non–synchronized sampling. In the following, we investigate how the receiver structures from Figs. I.1–3 to I.1–6 can be implemented digitally.

In a digital receiver, the matched filter is normally implemented as a FIR filter; its output is a sequence of complex–valued non–synchronized samples of the underlying continuous–time matched filter output. In Fig. I.1–3, Fig. I.1–4 and in the lower branch of Fig. I.1–6, the continuous–time matched filter output is sampled at instants that are related to the receiver timing (synchronized sampling). In a corresponding digital receiver, these synchronized samples are obtained by interpolating between the non–synchronized matched filter output samples. The interpolator acts as an adjustable delay, and is implemented as a FIR filter, whose coefficients depend on the timing estimate [Gar'90a]. The phase rotator performs a memoryless operation (which can be implemented by means of a look–up table) on the interpolated matched filter output samples. In Fig. I.1–5, synchronized sampling is applied to the phase–rotated continuous–time matched filter output; in a corresponding digital implementation, the phase rotator

operates on the non–synchronized matched filter output samples, and interpolation between the non–synchronized phase rotator output samples yields the synchronized phase rotator output samples.

Figs. I.1–11 and I.1–12 show the block diagrams of the digital receiver, in the cases where interpolation precedes phase rotation and phase rotation precedes interpolation, respectively. It is indicated which signals must be passed to the inputs of the various types of carrier and symbol synchronizers. In Fig. I.1–11, the matched filter and the interpolator can be combined into a single FIR filter (interpolating matched filter), provided that both the carrier and the symbol synchronizer operate on interpolated (and possibly phase–rotated) samples.

The highest computational complexity of the digital receiver resides within the matched filter and the interpolator. Therefore, it is useful to limit the speed at which the matched filter and the interpolator must operate, by a judicious choice of the receiver configuration. The sampling rates at the input of the matched filter, at the output of the matched filter, and at the input of the interpolator are not necessarily the same, but must equal at least the Nyquist rate, in order to avoid frequency aliasing. This Nyquist rate equals $(1+\alpha)/T$, where $\alpha \in (0,1)$ denotes the rolloff and $1/T$ equals the symbol rate. When using timing–independent carrier synchronization and/or FF symbol synchronization, multiple samples per symbol might be required at the matched filter output (see section I.2). The sampling rate at the output of the interpolator depends on various factors :

– When using timing–independent carrier synchronization in combination with FF symbol synchronization, the synchronizer input signals are taken from before the interpolator, so that the sampling rate at the output of the interpolator is not affected by the synchronization algorithms. In this case, the interpolator only needs to deliver samples taken at the estimated decision instants, in order that the receiver can make the decisions (I.1.2–2). When the phase rotator follows the interpolator, one complex–valued sample per symbol is needed for BPSK and QPSK (i.e. the sample $p(k;\hat{\tau})$), and two complex–valued samples per symbol are needed for OQPSK (i.e. the samples $p(k;\hat{\tau})$ and $p(k+1/2;\hat{\tau})$). When the phase rotator precedes the interpolator, the interpolator must deliver to the decision device the quantities $\mathrm{Re}[p(k;\hat{\tau})\exp(-j\hat{\theta})]$ and $\mathrm{Im}[q(k;\hat{\tau})\exp(-j\hat{\theta})]$. This corresponds to one real–valued sample per symbol for BPSK, and two real–valued samples per symbol for (O)QPSK. As far as the computational complexity of the interpolator is concerned, the phase rotator preceding the interpolator is the preferred configuration for BPSK and OQPSK.

– When using FF symbol synchronization in combination with timing–dependent carrier synchronization, the sampling rate at the output of the interpolator is affected by the carrier synchronization algorithm. It will be shown in section I.2.1 that most timing–dependent carrier synchronizers need one complex–valued sample per symbol for BPSK and QPSK (i.e. the sample $p(k;\hat{\tau})$ or its phase–rotated version), and two complex–valued samples per symbol for OQPSK (i.e. the samples $p(k;\hat{\tau})$ and $p(k+1/2;\ \hat{\tau})$, or their phase–rotated versions). The computational complexity of the interpolator is independent of whether the phase rotator precedes or follows the interpolator.

– When using timing–independent carrier synchronization in combination with FB symbol synchronization, the sampling rate at the output of the interpolator is affected by the symbol synchronization algorithm. It will be shown in section I.2.2 that most FB symbol synchronizers make use of the quantities $\mathrm{Re}[p(k;\hat{\tau})\exp(-j\hat{\theta})]$ and $\mathrm{Im}[q(k;\hat{\tau})\ \exp(-j\hat{\theta})]$, which are already needed for the receiver's decisions, and in many cases also of the quantities $\mathrm{Re}[p(k+1/2;\hat{\tau})\exp(-j\hat{\theta})]$ and $\mathrm{Im}[q(k+1/2;\hat{\tau})\exp(-j\hat{\theta})]$; the latter quantities are obtained by sampling the phase–rotated matched filter output signal halfway between decision instants. When the FB symbol synchronizer uses only the quantities needed for the receiver's decisions, we obtain the same requirements on the interpolator output sampling rate as in the case of timing–independent carrier synchronization combined with FF symbol synchronization : one (for BPSK and QPSK) or two (for OQPSK) complex–valued samples per symbol when the phase rotator follows the interpolator, and one (for BPSK) or two (for (O)QPSK) real–valued samples per symbol when the phase rotator precedes the interpolator. When the FB symbol synchronizer in addition uses the samples halfway between decision instants, these sampling rates must be doubled, except for the case of OQPSK, where the sampling rate remains at two complex–valued samples per symbol when the phase rotator follows the interpolator (this is because $q(k+1/2;\hat{\tau}) = p(k+1;\hat{\tau})$ and $p(k+1/2;\hat{\tau}) = q(k;\hat{\tau})$ for OQPSK). As far as the computational complexity of the interpolator is concerned, the phase rotator preceding the interpolator is the preferred configuration for BPSK and OQPSK.

– When using FB symbol synchronization in combination with timing–dependent carrier synchronization, the sampling rate at the output of the interpolator is affected by both synchronization algorithms. When the phase rotator follows the interpolator, the interpolator output rate is determined by the symbol synchronization algorithm, because the complex–valued interpolator output samples needed for symbol synchronization include those samples needed for carrier synchronization and for the receiver's decisions. Hence, one (for BPSK and QPSK) or two (for OQPSK) complex–valued samples are needed when the FB

symbol synchronizer uses only samples at decision instants; two complex–valued samples are needed for BPSK and for (O)QPSK when the FB symbol synchronizer in addition uses samples halfway between decision instants. When the phase rotator precedes the interpolator, the carrier synchronizer requires an interpolator output sampling rate of one (for BPSK and QPSK) or two (for OQPSK) complex– valued samples per symbol; this corresponds to two (for BPSK and QPSK) or four (for OQPSK) real–valued samples per symbol. These real–valued samples include all those that are needed for the operation of a FB symbol synchronizer using samples at decision instants. For a FB symbol synchronizer using in addition samples halfway between decision instants, one (for BPSK) and two (for QPSK) additional real–valued samples per symbol must be provided by the interpolator; this yields a total sampling rate of three (for BPSK) or four (for (O)QPSK) real–valued samples per symbol. As far as the computational complexity of the interpolator is concerned, placing the phase rotator before the interpolator gives an advantage only in the case of BPSK, provided that the FB symbol synchronizer also uses samples halfway between decision instants.

The above results are summarized in the following tables.

Samples needed for decisions	
BPSK :	$\mathrm{Re}[p(k;\hat{\tau})\exp(-j\hat{\theta})]$
QPSK :	$\mathrm{Re}[p(k;\hat{\tau})\exp(-j\hat{\theta})]$ and $\mathrm{Im}[p(k;\hat{\tau})\exp(-j\hat{\theta})]$
OQPSK :	$\mathrm{Re}[p(k;\hat{\tau})\exp(-j\hat{\theta})]$ and $\mathrm{Im}[p(k+1/2;\hat{\tau})\exp(-j\hat{\theta})]$

Samples needed for timing–dependent carrier synchronization	
BPSK :	$p(k;\hat{\tau})$ or $p(k;\hat{\tau})\exp(-j\hat{\theta})$
QPSK :	$p(k;\hat{\tau})$ or $p(k;\hat{\tau})\exp(-j\hat{\theta})$
OQPSK :	$p(k;\hat{\tau})$ and $p(k+1/2;\hat{\tau})$, or $p(k;\hat{\tau})\exp(-j\hat{\theta})$ and $p(k+1/2;\hat{\tau})\exp(-j\hat{\theta})$

Samples needed for FB symbol synchronization	
A) Samples at decision instants	
BPSK :	$\mathrm{Re}[p(k;\hat{\tau})\exp(-j\hat{\theta})]$
QPSK :	$\mathrm{Re}[p(k;\hat{\tau})\exp(-j\hat{\theta})]$ and $\mathrm{Im}[p(k;\hat{\tau})\exp(-j\hat{\theta})]$
OQPSK :	$\mathrm{Re}[p(k;\hat{\tau})\exp(-j\hat{\theta})]$ and $\mathrm{Im}[p(k+1/2;\hat{\tau})\exp(-j\hat{\theta})]$
B) Additional samples halfway between decision instants	
BPSK :	$\mathrm{Re}[p(k+1/2;\hat{\tau})\exp(-j\hat{\theta})]$
QPSK :	$\mathrm{Re}[p(k+1/2;\hat{\tau})\exp(-j\hat{\theta})]$ and $\mathrm{Im}[p(k+1/2;\hat{\tau})\exp(-j\hat{\theta})]$
OQPSK :	$\mathrm{Re}[p(k+1/2;\hat{\tau})\exp(-j\hat{\theta})]$ and $\mathrm{Im}[p(k+1;\hat{\tau})\exp(-j\hat{\theta})]$

Interpolator output sampling rates (in real–valued samples per symbol)						
<u>Carrier synchronization</u>						
– timing–independent	x x		x x	x x		
– timing–dependent		x x			x x	x x
<u>Symbol synchronization</u>						
– FF	x x	x x				
– FB (at decision instants only)			x x		x x	
– FB (at and halfway between decision instants)				x x		x x
<u>Phase rotator</u>						
– follows interpolator	x	x	x	x	x	x
– precedes interpolator	x	x	x	x	x	x
<u>Interpolator output rate</u>						
– BPSK :	2 1	2 2	2 1	4 2	2 2	4 3
– QPSK :	2 2	2 2	2 2	4 4	2 2	4 4
– OQPSK :	4 2	4 4	4 2	4 4	4 4	4 4

I.2 CLASSIFICATION OF SYNCHRONIZATION ALGORITHMS

Scanning the synchronization literature, one soon finds out that there exists a very large variety of algorithms for carrier and symbol synchronization. In order to gain understanding about the differences and commonalities between the various algorithms, it is important to make a systematic classification according to their principles of operation.

It turns out that many synchronization algorithms can be related to the maximum–likelihood (ML) criterion. The synchronizers that are purely ML produce estimates which maximize either the decision–directed (DD) log–likelihood function (I.1.2–1) or the (small SNR limit of) the nondata–aided (NDA) log–likelihood function (I.1.2–4); this can be accomplished by means of feedback (FB) or feedforward (FF) algorithms. Many synchronizers that are not purely ML involve only small modifications of the purely ML synchronizers, for the sake of a simpler implementation. Most of the remaining ad hoc synchronizers, which cannot be related to the ML criterion, operate according to the spectral line generating principle : a (usually memoryless) nonlinear operation on the (filtered) received complex envelope gives rise to spectral lines, which can be used to extract the carrier phase and the symbol timing.

An excellent starting point for the systematic classification of synchronization algorithms is Dr. Gardner's study report [Gar'88], which contains an extensive overview of synchronization algorithms suited for digital implementation. We have considered not only the synchronizers from [Gar'88], but also other synchronizers of potential interest (such as timing–independent carrier synchronizers). Moreover, we have introduced a few new algorithms, which fill some gaps in the existing literature; among these new algorithms are :

– the extension of the Viterbi and Viterbi carrier synchronization algorithm to the case of OQPSK (see section I.2.1.3);

– a class of carrier–independent NDA FB symbol synchronizers for M–PSK, operating on matched filter output samples taken at decision instants only (see section I.2.2.6);

– the FF joint NDA ML carrier and symbol synchronization algorithms (see section I.2.3).

I.2.1 Carrier synchronization

Most carrier synchronization algorithms make use of the receiver's timing estimate : they operate on the complex–valued matched filter output samples, taken at the estimated decision instants. Some carrier synchronization algorithms do not require any knowledge about the timing : they operate on the continuous–time output of a (not necessarily matched) prefilter, driven by the complex envelope of the received signal (analog implementation), or on non–synchronized samples of the prefilter output signal (digital implementation).

The carrier synchronization algorithms investigated in this study can be categorized according to the following characteristics :
– decision–directed (DD) or nondata–aided (NDA);
– derived from the maximum–likelihood (ML) criterion or from some ad hoc principle; the ad hoc symbol synchronizers will be named after the operating principle (spectral line generating nonlinearity (NL) such as a squarer (SQ)), or the authors who introduced the synchronizer (Viterbi and Viterbi (V&V));
– feedforward (FF) or feedback (FB) phase correction;
– some carrier synchronizers are timing–independent ($\bar{T}$), but most symbol synchronizers make use of the timing estimate ();
– some carrier synchronizers are applicable only to BPSK (B), QPSK (Q) or OQPSK (O), but many carrier synchronizers apply to all three modulation formats ().
The above abbreviations between parentheses allow us to give compact names to the various carrier synchronization algorithms; for instance, NDASQFF$\bar{T}$O denotes a nondata–aided (NDA) carrier synchronizer which derives the carrier phase from a squaring (SQ) operation on the received signal, applies feedforward (FF) phase correction, operates independently of the timing estimate ($\bar{T}$) and applies to OQPSK (O) only.

I.2.1.1 Decision–directed (DD) maximum–likelihood (ML) carrier synchronization. Timing estimate is assumed to be known.

The DD ML carrier synchronizer looks for a carrier phase estimate $\hat{\theta}$, which maximizes the DD log–likelihood function (I.1.2–1) for a given value of the timing estimate $\hat{\tau}$; the receiver's decisions about the data symbols are available. The estimate $\hat{\theta}$ can be obtained by means of both feedback and feedforward (with decision–feedback (DFB)) algorithms [Kob'71; Fal'77; Sim'79; Fra'80; Mey'80a; Gar'88 (chaps. 8,9)].

– DDMLFB

The DD ML feedback synchronizer is shown in Fig. I.2–1. Its phase error detector output at instant kT is proportional to the derivative, with respect to the trial value of the carrier phase, of the terms with m=k in the DD log–likelihood function (I.1.2–1) :

$$u_{\theta}(k;\hat{\theta}.\hat{\tau}) = \mathrm{Im}[(\hat{a}_k p(k;\hat{\tau}) - j\hat{b}_k q(k;\hat{\tau}))\, e^{-j\hat{\theta}}] \qquad \text{(BPSK, (O)QPSK)}$$

– DDMLFF

The feedforward (plus DFB) algorithm, shown in Fig. I.2–2, produces either an estimate $\hat{\theta}$ of the carrier phase θ, or an estimate $\hat{C} + j\hat{S}$ of $e^{j\theta}$ which is not restricted to be on the unit circle. These estimates are given by

$$\hat{\theta} = \arg\left[\sum_{k=1}^{K} (\hat{a}_k p(k;\hat{\tau}) - j\hat{b}_k q(k;\hat{\tau}))\right] \qquad \text{(BPSK, (O)QPSK)}$$

$$\hat{C} + j\hat{S} = \frac{1}{K}\sum_{k=1}^{K} (\hat{a}_k p(k;\hat{\tau}) - j\hat{b}_k q(k;\hat{\tau})) \qquad \text{(BPSK, (O)QPSK)}$$

I.2.1.2 Nondata–aided (NDA) maximum–likelihood (ML) carrier synchronization. Timing estimate is assumed to be known.

The NDA ML carrier synchronizer looks for a carrier phase estimate which maximizes the low SNR limit (I.2.1–4) of the NDA log–likelihood function, for a given the timing estimate $\hat{\tau}$. The carrier phase estimate can be obtained by means of both feedback and feedforward algorithms [Sim'79; Fra'80; Mey'80a; Gar'88 (chaps. 8,9)].

– NDAMLFBB/O, NDAMLFBQ

The NDA ML feedback synchronizer is shown in Fig. I.2–3. For BPSK and OQPSK, the phase error detector output at instant kT is proportional with the derivative, with respect to the trial value of the carrier phase, of the terms with m=k in the first summation in (I.1.2–4); for QPSK, the first summation does not depend on the trial value of the carrier phase, so that the second summation must be considered :

$$u_\theta(k;\hat{\theta},\hat{\tau}) = \frac{1}{2}\,\mathrm{Im}[(p^2(k;\hat{\tau}) - q^2(k;\hat{\tau}))\,e^{-j2\hat{\theta}}] \qquad \text{(BPSK, OQPSK)}$$

or

$$u_\theta(k;\hat{\theta},\hat{\tau}) = \frac{1}{4}\,\mathrm{Im}[-p^4(k;\hat{\tau})\,e^{-j4\hat{\theta}}] \qquad \text{(QPSK)}$$

The minus sign in the expression for QPSK comes from the fact that the argument of the complex QPSK symbols $a_k + jb_k$ is supposed to be an odd multiple of $\pi/4$; removing the minus sign corresponds to QPSK symbols with arguments that are multiples of $\pi/2$.

– NDAMLFFB/O, NDAMLFFQ

The feedforward algorithm, shown in Fig. I.2–4, produces an estimate of the carrier phase given by

$$\hat{\theta} = \frac{1}{2}\arg\left[\sum_{k=1}^{K}(p^2(k;\hat{\tau}) - q^2(k;\hat{\tau}))\right] \qquad \text{(BPSK, OQPSK)}$$

or

$$\hat{\theta} = \frac{1}{4}\arg\left[-\sum_{k=1}^{K} p^4(k;\hat{\tau})\right] \qquad \text{(QPSK)}$$

The explanation of the presence of the minus sign in the expression for QPSK is the same as for the corresponding feedback algorithm.

I.2.1.3 Viterbi and Viterbi (V&V) carrier synchronization. Timing estimate is assumed to be known.

The original V&V carrier synchronization algorithm [Vit'83] makes use of a NDA feedforward algorithm for estimating the carrier phase of an M–PSK signal (M=2 and M=4 correspond to BPSK and QPSK, respectively). The algorithm is very similar to (and is in fact a generalization of) the NDA ML feedforward algorithm (see Fig. I.2–4); besides the squarer (BPSK) and the quadrupler (QPSK), which yield the NDA ML algorithm, other types of nonlinearities are considered as well. We have been able to extend the V&V algorithm to include the case of OQPSK too [Moe'90c]. Whereas the original V&V carrier synchronization algorithm is of the feedforward type, we have been able to define an equivalent feedback V&V carrier synchronization algorithm. The feedback algorithm is less sensitive to frequency errors then the feedforward version.

– <u>V&VFFB/Q, V&VFFO</u>

The carrier phase estimate is given by

$$\hat{\theta} = \frac{1}{M} \arg\left[\sum_{k=1}^{K} F(|p(k;\hat{\tau})|) \exp(jM \arg(p(k;\hat{\tau}))) \right] \qquad \text{(BPSK, QPSK)}$$

or

$$\hat{\theta} = \frac{1}{2} \arg\left[\sum_{k=1}^{K} F(|p(k;\hat{\tau})|) \exp(2j \arg(p(k;\hat{\tau}))) - \sum_{k=1}^{K} F(|q(k;\hat{\tau})|) \exp(2j \arg(q(k;\hat{\tau}))) \right] \qquad \text{(OQPSK)}$$

where F(.) is an arbitrary function.

– <u>V&VFBB/Q, V&VFBO</u>

The phase error detector output is given by

$$u_{\theta}(k;\hat{\theta},\hat{\tau}) = \frac{1}{M} F(|p(k;\hat{\tau})|) \sin(M \arg(p(k;\hat{\tau})) - M\,\hat{\theta}) \qquad \text{(BPSK, QPSK)}$$

or

$$u_{\theta}(k;\hat{\theta},\hat{\tau}) = \frac{1}{2} F(|p(k;\hat{\tau})|) \sin(2 \arg(p(k;\hat{\tau})) - 2\,\hat{\theta}) - \frac{1}{2} F(|q(k;\hat{\tau})|) \sin(2 \arg(q(k;\hat{\tau})) - 2\,\hat{\theta}) \qquad \text{(OQPSK)}$$

where F(.) is an arbitrary function.

I.2.1.4 Carrier synchronization independent of timing.

Carrier synchronizers which make use of the timing estimate are influenced by the symbol synchronization operation; this interaction may give rise to acquisition problems. One way to avoid these problems is to use carrier synchronizers which operate independently of the symbol timing. The following analog synchronizers are examples of such timing–independent carrier synchronizers.

- The Costas loop with passive arm filters and hard limiter, which can be used for BPSK, QPSK and OQPSK [Sim'78a; Sim'78b].
- The biphase (BPSK) and quadriphase ((O)QPSK) Costas loops with passive arm filters, which are equivalent with the squaring loop (BPSK) and the quadrupling loop ((O)QPSK) [Lin'73 (chap. 2); Sim'77; Sim'78b].
- The carrier synchronizer for MSK (which is a special case of OQPSK), considered in [deB'72; DAn'85; Moe'86].

These analog synchronizers derive a carrier phase estimate from a memoryless nonlinear operation on the continuous–time output of a (not necessarily matched) prefilter, driven by the received complex envelope. In a digital implementation, the carrier synchronizers operate on non–synchronized prefilter output samples. When these samples are processed in the same way as is the continuous–time prefilter output in the analog synchronization algorithm, the digital synchronizer is equivalent with the analog synchronizer only when the non–synchronized sampling rate at the prefilter output is sufficiently high that the frequency aliasing at the output of the nonlinearity (squarer, quadrupler, hard limiter) is negligible; this might require non–synchronized sampling rates exceeding a multiple of the Nyquist rate $(1+\alpha)/T$; the multiple depends on the specific nonlinearity. Strictly speaking, the digital synchronizer can be made equivalent with the analog synchronizer, as soon as the non–synchronized sampling rate exceeds the Nyquist rate $(1+\alpha)/T$, because then the samples at the prefilter output contain the same information as the underlying continuous–time signal. In this case, however, the non–synchronized samples must be processed differently from the continuous–time signal in order to obtain equivalence between the analog and the digital synchronizer; we expect that this processing performs some nonlinear filtering on the non–synchronized samples, which is computationally complex because of the memory involved in the filtering.

– <u>DDMLFB$\bar{T}$</u>

The structure of the Costas loop with passive arm filters and hard limiter can be

obtained from the DDMLFB carrier synchronizer by simply removing the sampling at decision instants, so that the algorithm operates on the (non–synchronized samples of the underlying) continuous–time matched filter output signal; possibly, the matched filter is replaced by some prefilter with real–valued impulse response $h_o(t)$. The resulting carrier synchronizer is denoted by DDMLFB$\bar{T}$, and shown in Fig. I.2–5. The phase error detector output is given by

$$u_\theta(t;\hat{\theta}) = \mathrm{Im}[(m_I(t) - jm_Q(t))\, y_L(t)\, e^{-j\hat{\theta}}] \qquad \text{(BPSK, (O)QPSK)}$$

where

$$m_I(t) = \mathrm{sgn}(\mathrm{Re}[y_L(t)\, e^{-j\hat{\theta}}])$$

$$m_Q(t) = \begin{cases} \mathrm{sgn}(\mathrm{Im}[y_L(t)\, e^{-j\hat{\theta}}]) & \text{((O)QPSK)} \\ 0 & \text{(BPSK)} \end{cases}$$

can be considered as continuous–time estimates of the binary data sequences, and $y_L(t)$ denotes the prefilter output signal : $y_L(t) = x_L(t) \odot h_o(t)$.

– <u>DDMLFF$\bar{T}$</u>

The DDMLFF$\bar{T}$ synchronizer can be obtained from the DDMLFF synchronizer by removing the sampling at the decision instants; this results in a feedforward (plus "decision" feedback) version of the Costas loop with passive arm filters and hard limiter. The DDMLFF$\bar{T}$ synchronizer is shown in Fig. I.2–6; its estimates are given by

$$\hat{\theta} = \arg\left[\int_0^{KT} (m_I(t) - jm_Q(t))\, y_L(t)\, dt \right] \qquad \text{(BPSK,(O)QPSK)}$$

$$\hat{C} + j\hat{S} = \frac{1}{KT} \int_0^{KT} (m_I(t) - jm_Q(t))\, y_L(t)\, dt \qquad \text{(BPSK,(O)QPSK)}$$

– <u>NDAMLFB$\bar{T}$B, NDAMLFB$\bar{T}$Q/O</u>

The structure of the squaring (BPSK) and quadrupling (QPSK) loop can be obtained by removing the sampling at the decision instants from the NDAMLFB carrier synchronizers for BPSK and QPSK, respectively. Applying the same

procedure to the (quadratic) NDAMLFB carrier synchronizer for OQPSK yields an algorithm which is completely inoperative, because the average phase error detector output is identically zero, irrespective of the carrier phase error [Gar'88]; instead, the quadrupling loop for QPSK can be used for OQPSK. These synchronizers are denoted by NDAMLFB$\bar{T}$ and are shown in Fig. I.2–7; the phase error detector output is given by

$$u_\theta(t;\hat{\theta}) = \frac{1}{2}\,\mathrm{Im}[y_L^2(t)\,e^{-2j\hat{\theta}}] \qquad \text{(BPSK)}$$

or

$$u_\theta(t;\hat{\theta}) = \frac{1}{4}\,\mathrm{Im}[y_L^4(t)\,e^{-4j\hat{\theta}}] \qquad \text{((O)QPSK)}$$

– <u>NDAMLFF$\bar{T}$B, NDAMLFF$\bar{T}$Q/O</u>

For BPSK and QPSK, the NDAMLFF$\bar{T}$ carrier synchronizers are obtained from the corresponding NDAMLFF algorithms by removing the sampling at the decision instants; the NDAMLFF$\bar{T}$ carrier synchronizer for OQPSK is the same as that for QPSK. The resulting synchronizers are feedforward versions of the squaring and quadrupling loops. They are shown in Fig. I.2–8; the carrier phase estimates are given by

$$\hat{\theta} = \frac{1}{2}\arg\left[\int_0^{KT} y_L^2(t)\,dt\right] \qquad \text{(BPSK)}$$

or

$$\hat{\theta} = \frac{1}{4}\arg\left[\int_0^{KT} y_L^4(t)\,dt\right] \qquad \text{((O)QPSK)}$$

– <u>NDASQFF$\bar{T}$O</u>

The carrier synchronizer for MSK, considered in [deB'72], makes use of the fact that squaring an MSK signal (or, more generally, an OQPSK signal) with carrier frequency f_0 yields spectral lines at $2f_0 + 1/T$ and $2f_0 - 1/T$. Each of these spectral lines is tracked by means of a PLL; mixing the VCO signals yields a spectral line at $4f_0$, which depends on the carrier phase θ but is independent of the time delay τ (the other mixer product is a spectral line at $2/T$, which depends on the time delay τ but is independent of the carrier phase θ; this spectral line can be used for symbol synchronization (see the NDASQFF$\bar{C}$O algorithm from section I.2.2.6)) [deB'72]. We have derived an equivalent feedforward carrier

synchronizer, operating on the complex envelope of the received OQPSK signal. This synchronizer is denoted by NDASQFFTO, and shown in Fig. I.2–9. The carrier phase estimate is given by

$$\hat{\theta} = \frac{1}{4}\arg\left[\int_0^{KT} y_L^2(t)\, e^{j2\pi t/T}\, dt\right] + \frac{1}{4}\arg\left[\int_0^{KT} y_L^2(t)\, e^{-j2\pi t/T}\, dt\right] \qquad \text{(OQPSK)}$$

It does not seem to be possible to derive an equivalent feedback carrier synchronizer operating on the complex envelope of the received OQPSK signal.

I.2.2 Symbol synchronization.

Deriving symbol synchronization for carrier–modulated signals by means of the ML criterion yields an algorithm which operates in the following way [Fra'80].

– The received complex envelope $x_L(t)$ is phase–rotated, using the carrier phase estimate $\hat{\theta}$. This yields $x_L(t)\exp(-j\hat{\theta})$, which will be interpreted by the symbol synchronizer as if it were the received complex envelope $x_L(t)\exp(-j\theta)$ with respect to the true carrier.
– In the case of BPSK, the in–phase component with respect to the true carrier is a noisy baseband PAM signal, whereas the quadrature component represents only noise. Therefore the symbol synchronizer for BPSK operates only on $\mathrm{Re}[x_L(t)\exp(-j\hat{\theta})]$, and derives timing by means of a symbol synchronization algorithm suitable for a baseband PAM signal, transmitted over an AWGN channel. In the case of (O)QPSK, the in–phase and quadrature components with respect to the true carrier are statistically independent noisy baseband PAM signals which exhibit either zero delay (QPSK) or a half symbol delay (OQPSK) with respect to each other. Therefore, the symbol synchronizer for (O)QPSK applies a ML symbol synchronization algorithm for baseband PAM to $\mathrm{Re}[x_L(t)\exp(-j\hat{\theta})]$, applies the same algorithm to $\mathrm{Im}[x_L(t)\exp(-j\hat{\theta})]$, and combines the information from both algorithms in order to obtain a symbol timing estimate.
Clearly, the ML symbol synchronization algorithms make use of the carrier phase estimate $\hat{\theta}$.

Inspection of the log–likelihood functions (I.1.2–1) and (I.1.2–4) reveals that, for a given carrier phase estimate $\hat{\theta}$, FF ML symbol synchronization algorithms will make use of the following quantities for maximizing the appropriate log–likelihood function:

$$p_R(k;\tilde{\tau},\hat{\theta}) = \mathrm{Re}[p(k;\tilde{\tau})\,e^{-j\hat{\theta}}] = \mathrm{Re}[x_L(t)\,e^{-j\hat{\theta}} \odot h_I(-t)]\big|_{t=kT+\tilde{\tau}}$$
$$q_I(k;\tilde{\tau},\hat{\theta}) = \mathrm{Im}[q(k;\tilde{\tau})\,e^{-j\hat{\theta}}] = \mathrm{Im}[x_L(t)\,e^{-j\hat{\theta}} \odot h_Q(-t)]\big|_{t=kT+\tilde{\tau}}$$

whereas the FB ML symbol synchronization algorithms, which try to make zero the derivative of the log–likelihood function with respect to $\tilde{\tau}$, also need the

following quantities :

$$p'_R(k;\tilde{\tau},\hat{\theta}) = \mathrm{Re}[p'(k;\tilde{\tau})\, e^{-j\hat{\theta}}] = \mathrm{Re}[x_L(t)\, e^{-j\hat{\theta}} \odot h'_I(-t)]\big|_{t=kT+\tilde{\tau}}$$

$$q'_I(k;\tilde{\tau},\hat{\theta}) = \mathrm{Im}[q'(k;\tilde{\tau})\, e^{-j\hat{\theta}}] = \mathrm{Im}[x_L(t)\, e^{-j\hat{\theta}} \odot h'_Q(-t)]\big|_{t=kT+\tilde{\tau}}$$

where $h'_I(-t)$ and $h'_Q(-t)$ are the impulse responses of the derivative matched filters. For the various modulation formats, we obtain
– BPSK : $h'_I(-t) = h'(-t)$; $h'_Q(-t) = 0$
– QPSK : $h'_I(-t) = h'_Q(-t) = h'(-t)$
– OQPSK : $h'_I(-t) = h'(-t)$; $h'_Q(-t) = h'(-t-T/2)$
and $h'(-t)$ is the derivative, with respect to t, of $h(-t)$.

Many ad hoc symbol synchronization algorithms operate in a similar way as the ML algorithms, in the sense that now an ad hoc (instead of a ML) symbol synchronization algorithm for baseband PAM signals is applied to the signal $\mathrm{Re}[x_L(t)\exp(-j\hat{\theta})]$ for BPSK or to the signals $\mathrm{Re}[x_L(t)\exp(-j\hat{\theta})]$ and $\mathrm{Im}[x_L(t)\exp(-j\hat{\theta})]$ for (O)QPSK. Evidently, these ad hoc algorithms also make use of the carrier phase estimate $\hat{\theta}$.

Finally, there are a few ad hoc symbol synchronization algorithms which operate independently of the carrier phase estimate. These carrier–independent symbol synchronizers can operate properly without prior carrier phase acquisition.

The symbol synchronization algorithms investigated in this study can be categorized according to the following characteristics :
– decision–directed (DD) or nondata–aided (NDA);
– derived from the maximum–likelihood (ML) criterion or from some ad hoc principle; the ad hoc symbol synchronizers will be named after the operating principle (early–late (EL), data transition tracking loop (DTTL), spectral line generating nonlinearity (NL) such as a squarer (SQ)), or the authors who introduced (the basic version of) the synchronizer (Gardner (GA), Mueller and Müller (M&M, mM&M));
– feedforward (FF) or feedback (FB) timing correction;
– some symbol synchronizers make use of feedforward or feedback self–noise cancellation ($\bar{S}\bar{N}\bar{F}\bar{F}$ or $\bar{S}\bar{N}\bar{F}\bar{B}$), but most do not ();
– some symbol synchronizers are carrier–independent ($\bar{C}$), but most symbol synchronizers make use of the carrier phase estimate ();

– some symbol synchronizers are applicable only to BPSK (B), QPSK (Q) or OQPSK (O), but most symbol synchronizers apply to all three modulation formats ().

The above abbreviations between parentheses allow us to give compact names to the various symbol synchronization algorithms; for instance, NDASQFF$\bar{C}$O denotes a nondata–aided (NDA) symbol synchronizer which derives timing from a squaring (SQ) operation on the received signal, applies feedforward (FF) timing correction, operates independently of the carrier phase estimate ($\bar{C}$) and applies to OQPSK (O) only.

I.2.2.1 Decision–directed (DD) maximum–likelihood (ML) symbol synchronization. Carrier phase estimate assumed to be known.

The DD ML symbol synchronizer looks for the timing estimate which maximizes the DD log–likelihood function (I.1.2–1) for a given value $\hat{\theta}$ of the carrier phase estimate; decisions about the data symbols are available. The timing estimate can be obtained either by means of a feedforward algorithm (search method), or a feedback algorithm. These algorithms have been considered in [Git'71; Kob'71; Fal'77; Fra'80; Moe'84b; Mey'80a; Gar'88 (chap. 10)].

– <u>DDMLFF</u>

The DD ML feedforward algorithm, shown in Fig. I.2–10, produces an estimate $\hat{\tau}$, which maximizes the DD log–likelihood function :

$$\hat{\tau} : \max_{\tilde{\tau}} \left[\sum_{k=1}^{K} (\hat{a}_k \, p_R(k;\tilde{\tau},\hat{\theta}) \; + \; \hat{b}_k \, q_I(k;\tilde{\tau},\hat{\theta})) \right]$$

The search is performed over all trial values $\tilde{\tau}$ in the interval (–T/2, T/2).

– <u>DDMLFB</u>

The feedback DD ML symbol synchronizer, shown in Fig. I.2–11, yields a timing error detector output, which at instant kT is proportional with the derivative, with respect to the trial value of the timing estimate, of the terms with m=k in the DD log–likelihood function (I.1.2–1) :

$$u_{\tau}(k;\hat{\theta},\hat{\tau}) = \hat{a}_k \, p_R'(k;\hat{\tau},\hat{\theta}) \; + \; \hat{b}_k \, q_I'(k;\hat{\tau},\hat{\theta})$$

Application of the DDMLFB symbol synchronizer requires the implementation of the derivative matched filter.

I.2.2.2 Nondata–aided (NDA) maximum–likelihood (ML) symbol synchronization. Carrier phase estimate assumed to be known.

The NDA ML symbol synchronizer looks for the timing estimate which maximizes the low SNR limit of the NDA log–likelihood function for a given value $\hat{\theta}$ of the carrier phase estimate; only the first summation in (I.1.2–4) needs to be considered. The timing estimate can be obtained either by means of a feedforward algorithm (search method), or a feedback algorithm. These algorithms have been considered in [Fra'80; Mey'80a; Moe'84b; Gar'88 (chap. 10)].

– NDAMLFF

The NDA ML feedforward algorithm, shown in Fig. I.2–10, produces an estimate $\hat{\tau}$, which maximizes the first summation in (I.1.2–4) :

$$\hat{\tau} : \max_{\tilde{\tau}} \left[\sum_{k=1}^{K} (p_R^2(k;\tilde{\tau},\hat{\theta}) + q_I^2(k;\tilde{\tau},\hat{\theta})) \right]$$

The search is performed over all trial values $\tilde{\tau}$ in the interval (–T/2, T/2).

– NDAMLFB

The feedback NDA ML symbol synchronizer, shown in Fig. I.2–11, yields a timing error detector output, which at instant kT is proportional with the derivative, with respect to the trial value of the timing estimate, of the terms with m=k in the first summation in (I.1.2–4) :

$$u_\tau(k;\hat{\theta},\hat{\tau}) = p_R(k;\hat{\tau},\hat{\theta})\, p_R'(k;\hat{\tau},\hat{\theta}) + q_I(k;\hat{\tau},\hat{\theta})\, q_I'(k;\hat{\tau},\hat{\theta})$$

Application of the NDAMLFB symbol synchronization algorithm requires the implementation of the derivative matched filter. In [Gar'88 (chap. 11)] the NDAMLFB algorithms for QPSK and OQPSK have been considered as ad hoc algorithms, although they are clearly based upon the maximization of the NDA log–likelihood function (I.1.2–4).

I.2.2.3 Feedback symbol synchronization without self–noise. Carrier phase estimate assumed to be known.

– DDMLFBS̄ÑF̄B̄

The self–noise in the DD ML feedback synchronizer can be eliminated by subtracting from the timing error detector output a decision feedback self–noise cancellation signal; the cancellation principle closely resembles that of a decision feedback equalizer. The corresponding timing error detector output is

$$u_\tau(k;\hat{\theta},\hat{\tau}) = \hat{a}_k\, p_R'(k;\hat{\tau},\hat{\theta}) + \hat{b}_k\, q_I'(k;\hat{\tau},\hat{\theta}) - \sum_m (\hat{a}_k\hat{a}_m + \hat{b}_k\hat{b}_m)\, g'(kT-mT)$$

where $g'(t)$ is the baseband pulse at the output of the derivative matched filter. This algorithm has been considered in [Mey'80a; Gar'88 (chap. 10)].

– DDMLFBS̄ÑF̄F̄, NDAMLFBS̄ÑF̄F̄

Forward self–noise cancellation is obtained for both the DD ML and NDA ML feedback symbol synchronizers, when the derivative matched filter, with impulse response $h'(-t)$, is substituted by a filter with transfer function $H_{mod}(f)$, given by

$$H_{mod}(f) = H^*(f)\left[j2\pi f - \sum_{m=-\infty}^{+\infty} \frac{g'(mT)}{g(0)} \exp(-j2\pi fmT) \right]$$

where $j2\pi f\, H^*(f)$ is the transfer function of the derivative matched filter, and $g(t)$ is the baseband pulse at the output of the matched filter. The forward self–noise cancellation for the DD ML symbol synchronizer has originally been introduced in [Moe'83], and has been mentioned in [Gar'88 (chap.10)]; note that a typing error has slipped into [Gar'88 (eq. 10.29)], where $h'(.)$ and $h(.)$ are used instead of $g'(.)$ and $g(.)$.

– DDMMSEFB

The minimum–mean–square–error (MMSE) symbol synchronizer uses a feedback algorithm for minimizing the mean square error between the input and the output of the receiver's decision device. The timing error detector output is given by

$$u_\tau(k;\hat{\theta},\hat{\tau}) = (p_R(k;\hat{\tau},\hat{\theta}) - \hat{a}_k)\, p_R'(k;\hat{\tau},\hat{\theta}) + (q_I(k;\hat{\tau},\hat{\theta}) - \hat{b}_k)\, q_I'(k;\hat{\tau},\hat{\theta})$$

The algorithm has been investigated in [Men'77; Moe'83]; the resulting tracking performance is free of self–noise. The implementation of the MMSE algorithm requires an AGC circuit, which controls the signal level at the input of the decision device.

I.2.2.4 Decision–directed (DD) and nondata–aided (NDA) early–late (EL) feedback symbol synchronization, and closely related algorithms. Carrier phase estimate assumed to be known.

The DDMLFB and NDAMLFB symbol synchronizers make use of a derivative matched filter, with impulse response h'(–t). This filter could be approximated by a filter with impulse response $h(-t+\lambda T) - h(-t-\lambda T)$. The advantage is that the samples at the output of the approximated derivative matched filter can be obtained by subtracting samples of the output of the matched filter (with impulse response h(–t)), taken at the appropriate instants of time (which are shifted by $-\lambda T$ and λT with respect to the estimated decision instants). When $\lambda=1$, symbol synchronization is derived from matched filter output samples taken at decision instants only; when $\lambda=1/2$, symbol synchronization is derived from matched filter output samples taken at and halfway between decision instants. The resulting symbol synchronizers are shown in Fig. I.2–12.

The ML synchronizer with the approximate derivative matched filter yields a timing error detector output given by

$$u_\tau(k;\hat{\theta},\hat{\tau}) = \hat{a}_k\,(p_R(k+\lambda;\hat{\tau},\hat{\theta}) - p_R(k-\lambda;\hat{\tau},\hat{\theta})) + \hat{b}_k\,(q_I(k+\lambda;\hat{\tau},\hat{\theta}) - q_I(k-\lambda;\hat{\tau},\hat{\theta})) \qquad \text{(DDEL)}$$

or

$$u_\tau(k;\hat{\theta},\hat{\tau}) = p_R(k;\hat{\tau},\hat{\theta})\,(p_R(k+\lambda;\hat{\tau},\hat{\theta}) - p_R(k-\lambda;\hat{\tau},\hat{\theta})) + q_I(k;\hat{\tau},\hat{\theta})\,(q_I(k+\lambda;\hat{\tau},\hat{\theta}) - q_I(k-\lambda;\hat{\tau},\hat{\theta})) \qquad \text{(NDAEL)}$$

These algorithms in fact describe early–late (EL) DD and NDA synchronizers [Lin'73 (chap. 9)].

– <u>DDEL(λ=1/2), DTTL</u>

Let us consider the DDEL symbol synchronization algorithm for $\lambda = 1/2$. The timing error detector output is given by

$$u_\tau(k;\hat{\theta},\hat{\tau}) = \hat{a}_k\,(p_R(k+\tfrac{1}{2};\hat{\tau},\hat{\theta}) - p_R(k-\tfrac{1}{2};\hat{\tau},\hat{\theta})) + \hat{b}_k\,(q_I(k+\tfrac{1}{2};\hat{\tau},\hat{\theta}) - q_I(k-\tfrac{1}{2};\hat{\tau},\hat{\theta})) \qquad \text{(DDEL}(\lambda=1/2))$$

A closely related algorithm is

$$u_\tau(k;\hat\theta;\hat\tau) = (\hat a_k - \hat a_{k+1})\, p_R(k+\tfrac{1}{2};\hat\tau,\hat\theta)$$
$$+\ (\hat b_k - \hat b_{k+1})\, q_I(k+\tfrac{1}{2};\hat\tau,\hat\theta) \qquad \text{(DTTL)}$$

which is equivalent with the data transition tracking loop (DTTL) with $\xi = 1/2$ [Lin'73 (chap. 9)]; in [Gar'88 (chap. 10)] this algorithm has been called the zero–crossing detector. The close relationship between the DD EL algorithm with $\lambda = 1/2$ and the DTTL algorithm follows from the fact that both timing error detector outputs yield the same long term arithmetical average :

$$\lim_{K\to\infty} \left[\frac{1}{K} \sum_{k=1}^{K} u_\tau(k;\hat\theta,\hat\tau) \Big|_{\text{DDEL}(\lambda=1/2)} - \frac{1}{K} \sum_{k=1}^{K} u_\tau(k;\hat\theta,\hat\tau) \Big|_{\text{DTTL}} \right] = 0$$

– <u>NDAEL(λ=1/2), GA</u>

The timing error detector output for the NDA EL algorithm with $\lambda = 1/2$ is given by

$$u_\tau(k;\hat\theta,\hat\tau) = p_R(k;\hat\tau,\hat\theta)\,(p_R(k+\tfrac{1}{2};\hat\tau,\hat\theta) - p_R(k-\tfrac{1}{2};\hat\tau,\hat\theta))$$
$$+\ q_I(k;\hat\tau,\hat\theta)\,(q_I(k+\tfrac{1}{2};\hat\tau,\hat\theta) - q_I(k-\tfrac{1}{2};\hat\tau,\hat\theta)) \qquad \text{(NDAEL}(\lambda=1/2))$$

A closely related algorithm is determined by

$$u_\tau(k;\hat\theta,\hat\tau) = p_R(k+\tfrac{1}{2};\hat\tau,\hat\theta)\,(p_R(k;\hat\tau,\hat\theta) - p_R(k+1;\hat\tau,\hat\theta))$$
$$+\ q_I(k+\tfrac{1}{2};\hat\tau,\hat\theta)\,(q_I(k;\hat\tau,\hat\theta) - q_I(k+1;\hat\tau,\hat\theta)) \qquad \text{(GA)}$$

This is the Gardner (GA) algorithm, originally introduced in [Gar'86]. The timing error detector outputs of the NDAEL(λ=1/2) algorithm and the GA algorithm yield the same long term arithmetical average.

– <u>DDEL(λ=1), M&M, mM&M</u>

The DDEL algorithm corresponding to λ=1 is determined by

$$u_\tau(k;\hat\theta,\hat\tau) = \hat a_k\,(p_R(k+1;\hat\tau,\hat\theta) - p_R(k-1;\hat\tau,\hat\theta))$$
$$+\ \hat b_k\,(q_I(k+1;\hat\tau,\hat\theta) - q_I(k-1;\hat\tau,\hat\theta)) \qquad \text{(DDEL}(\lambda=1))$$

A closely related algorithm is

$$u_\tau(k;\hat{\theta},\hat{\tau}) = \hat{a}_k\, p_R(k+1;\hat{\tau},\hat{\theta}) - \hat{a}_{k+1}\, p_R(k;\hat{\tau},\hat{\theta}) + \hat{b}_k\, q_I(k+1;\hat{\tau},\hat{\theta}) - \hat{b}_{k+1}\, q_I(k;\hat{\tau},\hat{\theta}) \qquad \text{(M\&M)}$$

This is the Mueller and Müller (M&M) algorithm, originally introduced in [Mue'76]. Another closely related algorithm is

$$u_\tau(k;\hat{\theta},\hat{\tau}) = (\hat{a}_{k-1} - \hat{a}_{k+1})\, p_R(k;\hat{\tau},\hat{\theta}) + (\hat{b}_{k-1} - \hat{b}_{k+1})\, q_I(k;\hat{\tau},\hat{\theta}) \qquad \text{(mM\&M)}$$

We call this the modified M&M algorithm; it has been introduced in [Moe'88b]. The timing error detector outputs of the DDEL(λ=1) algorithm, the M&M algorithm and the mM&M algorithm yield the same long term arithmetical average.

[– <u>NDAEL(λ=1)</u>
The NDA EL algorithm for $\lambda = 1$ is inoperative; the statistical average of the timing error detector output is identically zero, irrespective of the timing error.]

I.2.2.5 Non–search feedforward symbol synchronization. Carrier phase estimate assumed to be known.

The non–search feedforward symbol synchronizers found in the literature are of the NDA type. They are based upon the fact that a nonlinear operation on the (prefiltered) in–phase and quadrature components of the received signal give rise to a spectral line at frequency 1/T, whose phase is determined by the unknown time delay τ. The non–search feedforward symbol synchronizers operate on the continuous–time prefilter output (analog implementation), or on non–synchronized samples thereof (digital implementation). Similar remarks as for the timing–independent carrier synchronizers (see section I.2.1.4) are valid with respect to the non–synchronized sampling rate.

– NDANLFF

The non–search feedforward symbol synchronizer is shown in Fig. I.2–13. The prefilter impulse responses are determined by

– BPSK : $h_{oI}(t) = h_o(t)$; $h_{oQ}(t) = 0$

– QPSK : $h_{oI}(t) = h_{oQ}(t) = h_o(t)$

– OQPSK : $h_{oI}(t) = h_o(t)$; $h_{oQ}(t) = h_o(t-T/2)$

and the real–valued impulse response $h_o(t)$ is not necessarily matched to the baseband pulse h(t). The timing estimate is given by

$$\hat{\tau} = \frac{T}{2\pi} \arg \left[\int_0^{KT} \left[NL(Re[y_{LI}(t)e^{-j\hat{\theta}}]) + NL(Im[y_{LQ}(t)e^{-j\hat{\theta}}] \right] e^{j2\pi t/T} \, dt \right]$$

where NL(.) denotes the nonlinear operation. Often encountered in the literature is the case where the nonlinear operation is a squarer [Fra'74; Fra'80, Moe'84a]; this algorithm will be denoted as NDASQFF. The digitally implemented NDASQFF has been introduced and analyzed in [Oer'88].

I.2.2.6 Ad hoc carrier–independent symbol synchronization.

All symbol synchronization algorithms considered up till now make use of the carrier phase estimate $\hat{\theta}$. However, for QPSK it turns out that the quadratic symbol synchronizers become independent of $\hat{\theta}$; basically, this is because $p(k;\tilde{\tau}) = q(k;\tilde{\tau})$ for QPSK. For example, the NDAMLFF algorithm for QPSK becomes

$$\hat{\tau}: \max_{\tilde{\tau}} \left[\sum_{k=1}^{K} (p_R^2(k;\tilde{\tau},\hat{\theta}) + q_I^2(k;\tilde{\tau},\hat{\theta})) \right]$$

$$= \max_{\tilde{\tau}} \left[\sum_{k=1}^{K} ((\mathrm{Re}[p(k;\tilde{\tau})e^{-j\hat{\theta}}])^2 + (\mathrm{Im}[q(k;\tilde{\tau})e^{-j\hat{\theta}}])^2) \right]$$

$$= \max_{\tilde{\tau}} \left[\sum_{k=1}^{K} ((\mathrm{Re}[p(k;\tilde{\tau})e^{-j\hat{\theta}}])^2 + (\mathrm{Im}[p(k;\tilde{\tau})e^{-j\hat{\theta}}])^2) \right]$$

as $p(k;\tilde{\tau}) = q(k;\tilde{\tau})$ for QPSK

$$= \max_{\tilde{\tau}} \left[\sum_{k=1}^{K} \left|p(k;\tilde{\tau})e^{-j\hat{\theta}}\right|^2 \right]$$

$$= \max_{\tilde{\tau}} \left[\sum_{k=1}^{K} \left|p(k;\tilde{\tau})\right|^2 \right] \qquad \text{as } \left|e^{-j\hat{\theta}}\right| = 1$$

The quadratic algorithms, which become independent of the carrier phase, are NDAMLFF, NDAMLFB, NDAMLFB$\bar{S}\bar{N}\bar{F}\bar{F}$, NDAEL ($\lambda$=1/2), GA and NDASQFF; the carrier–independent QPSK version of these algorithms (which is mathematically equivalent with the original "carrier–dependent" QPSK version) will be denoted by its original name, with the extension $\bar{C}$Q added, where $\bar{C}$ and Q stand for carrier–independent and QPSK, respectively. An advantage of symbol synchronizers operating independently of the carrier phase estimate is that acquisition of timing does not require prior acquisition of the carrier phase. It is tempting to apply the carrier–independent QPSK version of the above algorithms as ad hoc carrier–independent algorithms for BPSK and OQPSK as well. This approach works well for BPSK; unfortunately it fails completely for OQPSK because the resulting average timing error detector output becomes identically zero, irrespective of the timing error [Gar'88]. When used as ad hoc carrier–independent algorithms for BPSK, the above quadratic algorithms will be denoted by their original name, with the extension $\bar{C}$B added, where $\bar{C}$ and B stand for carrier–independent and BPSK, respectively. These quadratic symbol synchronization algorithms are given in detail below.

– NDAMLFFC̄B/Q (see Fig. I.2–14)

$$\hat{\tau} : \max_{\tilde{\tau}} \left[\sum_{k=1}^{K} |p(k;\tilde{\tau})|^2 \right] \qquad \text{(BPSK, QPSK)}$$

– NDAMLFBC̄B/Q (see Fig. I.2–15)

$$u_{\tau}(k;\hat{\tau}) = \mathrm{Re}[p^*(k;\hat{\tau})\, p'(k;\hat{\tau})] \qquad \text{(BPSK, QPSK)}$$

– NDAMLFBS̄N̄F̄F̄C̄B/Q

Same as NDAMLFBC̄B/Q, but the derivative matched filter is replaced by a filter with transfer function $H_{mod}(f)$, given by

$$H_{mod}(f) = H^*(f) \left[j2\pi f - \sum_{m=-\infty}^{+\infty} \frac{g'(mT)}{g(0)} \exp(-j2\pi fmT) \right]$$

– NDAEL(λ=1/2)C̄B/Q (see Fig. I.2–16)

$$u_{\tau}(k;\hat{\tau}) = \mathrm{Re}[p^*(k;\hat{\tau})\, (p(k+1/2;\hat{\tau}) - p(k-1/2;\hat{\tau}))] \qquad \text{(BPSK, QPSK)}$$

– GAC̄B/Q (see Fig. I.2–16)

$$u_{\tau}(k;\hat{\tau}) = \mathrm{Re}[p^*(k+1/2;\hat{\tau})\, (p(k;\hat{\tau}) - p(k+1;\hat{\tau}))] \qquad \text{(BPSK, QPSK)}$$

– NDASQFFC̄B/Q (see Fig. I.2–17)

$$\hat{\tau} = \frac{T}{2\pi} \arg \left[\int_0^{KT} |y_L(t)|^2 e^{j2\pi t/T}\, dt \right] \qquad \text{(BPSK, QPSK)}$$

where $y_L(t) = x_L(t) \odot h_o(t)$.

None of the FB algorithms from the above list of carrier–independent symbol synchronizers operates solely on matched filter output samples, taken at the decsion instants : either a derivative matched filter is required (NDAMLFBC̄B/Q), or additional samples halfway between decision instants are needed (NDAEL(λ=1/2)C̄B/Q, GAC̄B/Q). As far as the computational complexity of the interpolator is concerned, we should prefer FB symbol synchronization algorithms using only matched filter output samples taken at

decision instants (see section I.1.4). The only such algorithms considered up till now are DDEL(λ=1), M&M and mM&M from section I.2.2.4; however, these algorithms make use of the carrier phase estimate. In [Moe'90b] we have introduced a class of carrier–independent FB symbol synchronization algorithms for M–PSK (M=2 and M=4 correspond to BPSK and QPSK, respectively), which use only one sample per symbol, taken at the decision instants; we denote these ad hoc synchronizers by NDANLFB$\bar{C}$B/Q.

– <u>NDANLFB$\bar{C}$B/Q</u> (see Fig. I.2–16)

$$u_\tau(k;\hat{\tau}) = (F(|p(k;\hat{\tau})|^2) - F(|p(k+1;\hat{\tau})|^2)\ \mathrm{Re}[p(k;\hat{\tau})p^*(k+1;\hat{\tau})] \quad \text{(BPSK,QPSK)}$$

where F(.) is an arbitrary function.

A carrier–independent symbol synchronization algorithm for OQPSK can be derived from the analog synchronizer for MSK, considered in [deB'72; DAn'85; Moe'86]. This analog synchronizer makes use of the fact that squaring an MSK signal (or, more generally, an OQPSK signal) with carrier frequency f_0 yields spectral lines at $2f_0 + 1/T$ and $2f_0 - 1/T$. Each of these spectral lines is tracked by means of a PLL; mixing the VCO signals yields a spectral line at 2/T, which depends on the time delay τ but is independent of the carrier phase θ (the other mixer product is a spectral line at $4f_0$, which depends on the carrier phase θ but is independent of the time delay τ; this spectral line can be used for carrier synchronization (see the NDASQFF$\bar{T}$O algorithm from section I.2.1.4)) [deB'72]. We have derived an equivalent feedforward symbol synchronizer, operating on the complex envelope of the received OQPSK signal. This synchronizer is denoted by NDASQFF$\bar{C}$O.

– <u>NDASQFF$\bar{C}$O</u> (see Fig. I.2–18)

$$\hat{\tau} = \frac{T}{4\pi}\arg\left[\int_0^{KT} y_L^2(t)\, e^{j2\pi t/T}\, dt\right] - \frac{T}{4\pi}\arg\left[\int_0^{KT} y_L^2(t)\, e^{-j2\pi t/T}\, dt\right] \quad \text{(OQPSK)}$$

where $y_L(t) = x_L(t) \odot h_0(t)$.

I.2.3 Joint carrier and symbol synchronization

The functions of carrier and symbol synchronization can be accomplished by combining any carrier synchronization algorithm from section I.2.1 with any symbol synchronization algorithm from section I.2.2. Two of these combinations deserve our special attention.

– Combination of the DDMLFF carrier synchronizer (section I.2.1.1) with the DDMLFF symbol synchronizer (section I.2.2.1). The estimates of the carrier phase and the symbol timing jointly maximize the DD log–likelihood function (I.1.2–1). This is called joint DDMLFF carrier and symbol synchronization (JDDMLFF).

– Combination of the NDAMLFF carrier synchronizer (section I.2.1.2) with the NDAMLFF symbol synchronizer (section I.2.2.2). The estimates of the carrier phase and the symbol timing jointly maximize the low SNR limit of the NDA log–likelihood function (I.1.2–4). This is called joint NDAMLFF carrier and symbol synchronization (JNDAMLFF).

The (DD or NDA) ML FF carrier synchronization algorithm tells us which carrier phase estimate maximizes the corresponding log–likelihood function, for a given timing estimate; the (DD or NDA) ML FF symbol synchronization algorithm yields the symbol timing estimate which maximizes the corresponding log–likelihood function, for a given carrier phase estimate. These carrier and symbol synchronization algorithms could be used in the following straightforward way to find the joint ML estimates :

(i) Take a starting value $\hat{\theta}(0)$ for the carrier phase estimate; set m=1.

(ii) Use $\hat{\theta}(m-1)$ in the symbol synchronizer to obtain $\hat{\tau}(m)$.

(iii) Use $\hat{\tau}(m)$ in the carrier synchronizer to obtain $\hat{\theta}(m)$.

(iv) Set m=m+1 and go to (ii).

The index m (m=1,2,3,...) indicates that the receiver is processing the m–th block of K symbol intervals. The above algorithm uses the carrier phase estimate from the previous block to obtain carrier phase and symbol timing estimates for the present block. As the starting value $\hat{\theta}(0)$ for the carrier phase estimate can be much different from the true carrier phase θ, it usually takes an acquisition period of several blocks of K symbol intervals before the estimates of the carrier phase and the symbol timing have converged towards the close vicinity of the true values.

The JDDMLFF and JNDAMLFF carrier and symbol synchronization algorithms find the joint ML estimates in a different way, yielding faster

acquisition. Let us denote by $L(\tilde{\theta},\tilde{\tau})$ the appropriate log–likelihood function to be maximized :

$$\hat{\theta},\ \hat{\tau} : \max_{\tilde{\theta},\tilde{\tau}} L(\tilde{\theta},\tilde{\tau})$$

The joint ML symbol timing estimate $\hat{\tau}$ is determined by

$$\hat{\tau} : \max_{\tilde{\tau}} \left[\max_{\tilde{\theta}} L(\tilde{\theta},\tilde{\tau}) \right] = \max_{\tilde{\tau}} L(\tilde{\theta}(\tilde{\tau}),\tilde{\tau})$$

where $\tilde{\theta}(\tilde{\tau})$ denotes the carrier phase estimate which maximizes $L(\tilde{\theta},\tilde{\tau})$ for a given trial value $\tilde{\tau}$ of the symbol timing; in other words, $\tilde{\theta}(\tilde{\tau})$ is the carrier phase estimate resulting from the (DD or NDA) ML FF carrier synchronizer corresponding to a given trial value $\tilde{\tau}$ of the symbol timing. The function $L(\tilde{\theta}(\tilde{\tau}),\tilde{\tau})$ is obtained by replacing in the log–likelihood function $L(\tilde{\theta},\tilde{\tau})$ the trial value $\tilde{\theta}$ by the expression for the (DD or NDA) ML FF carrier phase estimate that corresponds to a trial value $\tilde{\tau}$ of the symbol timing; this expression for the carier phase estimate is obtained by simply replacing $\hat{\tau}$ by $\tilde{\tau}$ in the carrier phase estimate expressions from sections I.2.1.1 and I.2.1.2. The joint ML symbol timing estimate $\hat{\tau}$ resulting from the maximization of $L(\tilde{\theta}(\tilde{\tau}),\tilde{\tau})$ is passed to the (DD or NDA) ML FF carrier synchronizer, which yields the joint ML carrier phase estimate $\hat{\theta}$ as $\hat{\theta} = \tilde{\theta}(\hat{\tau})$. Hence, the estimates $\hat{\theta}$ and $\hat{\tau}$ jointly maximize the log–likelihood function $L(\tilde{\theta},\tilde{\tau})$. For each block of K symbol intervals, the joint ML FF carrier and symbol synchronizer operates as follows.

(i) The symbol timing estimate $\hat{\tau}$ is obtained from the maximization of $L(\tilde{\theta}(\tilde{\tau}),\tilde{\tau})$.

(ii) The timing estimate $\hat{\tau}$ from (i) is passed to the ML FF carrier synchronizer, which yields the carrier phase estimate $\hat{\theta}$ as $\hat{\theta} = \tilde{\theta}(\hat{\tau})$.

As the joint ML carrier phase and symbol timing estimates for the present block are made independently of the estimates from previous blocks, the carrier phase and symbol timing are acquired already during the first considered block of K symbol intervals; basically, this is because symbol synchronization has become independent of the carrier phase estimate.

We have evaluated the functions $L(\tilde{\theta}(\tilde{\tau}),\tilde{\tau})$ for the JDDMLFF and JNDAMLFF carrier and symbol synchronizers. This results in the following algorithms, shown in Fig. I.2–19.

– JDDMLFF

$$\hat{\tau}: \max_{\tilde{\tau}} \left[\left| \sum_{k=1}^{K} (\hat{a}_k\, p(k;\tilde{\tau}) - j\hat{b}_k\, q(k;\tilde{\tau})) \right| \right] \qquad \text{(BPSK,(O)QPSK)}$$

$$\hat{\theta} = \arg \left[\sum_{k=1}^{K} (\hat{a}_k\, p(k;\tilde{\tau}) - j\hat{b}_k\, q(k;\tilde{\tau})) \right] \qquad \text{(BPSK,(O)QPSK)}$$

It is important to note that the timing estimate implicitly depends on the carrier phase estimate through the receiver's decisions; the timing estimate is strictly carrier–independent only when the data symbols are known in advance, i.e. during a known training sequence. The JDDMLFF algorithm has been considered in [Gar'88] for BPSK and QPSK only, but the extension to OQPSK is straightforward. The JDDMLFF algorithm has recently been implemented in a high data rate digital receiver [Asc'89].

– JNDAMLFFB/O, JNDAMLFFQ

The NDAMLFF carrier synchronization algorithm for QPSK is different from the NDAMLFF carrier synchronization algorithm for BPSK and OQPSK (see section I.2.1.2); this results in two JNDAMLFF algorithms : one for BPSK and OQPSK (denoted as JNDAMLFFB/O) and one for QPSK (denoted as JNDAMLFFQ). The estimates of the symbol timing and the carrier phase are determined by

$$\hat{\tau}: \max_{\tilde{\tau}} \left[\sum_{k=1}^{K} (|p(k;\tilde{\tau})|^2 + |q(k;\tilde{\tau})|^2) + \left| \sum_{k=1}^{K} (p^2(k;\tilde{\tau}) - q^2(k;\tilde{\tau})) \right| \right] \qquad \text{(BPSK,OQPSK)}$$

$$\hat{\theta} = \frac{1}{2} \arg \left[\sum_{k=1}^{K} (p^2(k;\hat{\tau}) - q^2(k;\hat{\tau})) \right] \qquad \text{(BPSK,OQPSK)}$$

or

$$\hat{\tau}: \max_{\tilde{\tau}} \left[\sum_{k=1}^{K} |p(k;\tilde{\tau})|^2 \right] \qquad \text{(QPSK)}$$

$$\hat{\theta} = \frac{1}{4} \arg \left[- \sum_{k=1}^{K} p^4(k;\hat{\tau}) \right] \qquad \text{(QPSK)}$$

As the timing estimate is strictly independent of the carrier phase estimate, the carrier phase and symbol timing can be acquired during the first block of K symbol intervals, without using a known preamble.

I.3 THEORETICAL ANALYSIS METHODS.

In this section we give a general overview of methods which are available for the theoretical analysis of digital feedback and feedforward synchronizers.

I.3.1 Feedback synchronization

Although there exists a large variety of feedback synchronization algorithms, they can all be analyzed by means of the same equivalent feedback model, whose parameters depend on the specific algorithm.

– The equivalent feedback model

Let us consider the digital receiver structure, shown in Fig. I.3–1, which operates on samples of the complex envelope $x_L(t)$ of the received waveform, with respect to the true carrier frequency f_0. Carrier and symbol synchronization are accomplished by means of two feedback loops. It is important to note that these loops are coupled :

– The phase error detector input signal has passed through the interpolator. Therefore, carrier synchronization is influenced by the operation of the symbol synchronizer.

– The timing error detector input signal is taken from the phase rotator output. Therefore, symbol synchronization is influenced by the operation of the carrier synchronizer.

When using decision–directed (DD) algorithms there is additional coupling, because the receiver's decisions depend on the estimates of carrier phase and the symbol timing. In some cases, there can be less coupling than in Fig. I.3–1. For instance, for feedback symbol synchronizers operating independently of the carrier phase, the timing error detector input can be taken from the input of the phase rotator, so that symbol synchronization is not influenced by the carrier synchronization.

Feedback synchronizers are analyzed by means of an equivalent model. The derivation of this feedback model is based upon the following decomposition of the phase error detector and timing error detector outputs :

$$u_\theta(k;\hat{\theta},\hat{\tau}) = g_\theta(\phi,\epsilon) + N_\theta(k;\phi,\epsilon) \qquad \text{[rad]}$$

$$u_\tau(k;\hat{\theta},\hat{\tau}) = g_\tau(\phi,\epsilon) + N_\tau(k;\phi,\epsilon) \qquad \text{[dimensionless]} \qquad \text{(I.3.1–1)}$$

where the synchronization surfaces $g_\theta(\phi,\epsilon)$ and $g_\tau(\phi,\epsilon)$ are the average phase error detector and timing error detector outputs, for given values of the phase error $\phi=\theta-\hat{\theta}$ and the normalized timing error $\epsilon=(\tau-\hat{\tau})/T$; the phase error detector and timing error detector noise terms $N_\theta(k;\phi,\epsilon)$ and $N_\tau(k;\phi,\epsilon)$ represent the statistical fluctuations with respect to the average phase error detector and timing error detector outputs, for given ϕ and ϵ. The spectra of $N_\theta(k;\phi,\epsilon)$ and $N_\tau(k;\phi,\epsilon)$ are denoted by $S_\theta(fT;\phi,\epsilon)$ [rad^2] and $S_\tau(fT;\phi,\epsilon)$ [dimensionless]. The synchronization surfaces and the phase and timing error detector noise spectra are periodic in ϕ (with period $2\pi/M$, where M=2 for BPSK and OQPSK, and M=4 for QPSK) and in ϵ (with period equal to one). Once the synchronization surfaces and the loop noise spectra have been determined, the equivalent feedback model, shown in Fig. I.3–2, can be derived [Men'77]. The stable equilibrium points of the feedback synchronizers satisfy the equations $g_\theta(\phi,\epsilon) = g_\tau(\phi,\epsilon) =0$. These stable equilibrium points are shown in Fig. I.3–3 for M–PSK (M=2 for BPSK, M=4 for QPSK) and OQPSK.

The equivalent feedback model is the starting point for all theoretical analysis of digital synchronizers using feedback. In the literature, a similar equivalent model has been used for analog synchronizers, the only difference being the signals and the filters, which are in continuous time rather than discrete time. Therefore, the analysis methods for digital feedback synchronizers closely resemble those for analog feedback synchronizers.

– Tracking analysis

During the tracking mode, the estimation errors ϕ and ϵ exhibit small random fluctuations about a stable equilibrium point (say, $\phi=0$, $\epsilon=0$), so that linearization about the stable equilibrium point applies. This involves the following approximations [Men'77] :

$$g_\theta(\phi,\epsilon) = K_{\theta\theta}\phi + K_{\theta\tau}\epsilon \qquad g_\tau(\phi,\epsilon) = K_{\tau\theta}\phi + K_{\tau\tau}\epsilon$$

$$S_\theta(fT;\phi,\epsilon) = S_\theta(fT;0,0) \qquad S_\tau(fT;\phi,\epsilon) = S_\tau(fT;0,0) \qquad \text{(I.3.1–2)}$$

where $K_{\theta\theta}$, $K_{\theta\tau}$, $K_{\tau\theta}$ and $K_{\tau\tau}$ are the slopes of the synchronization surfaces at the equilibrium point :

$$K_{\theta\theta} = \left.\frac{\partial g_\theta(\phi,\epsilon)}{\partial\phi}\right|_{\substack{\phi=0\\ \epsilon=0}} \qquad K_{\theta\tau} = \left.\frac{\partial g_\theta(\phi,\epsilon)}{\partial\epsilon}\right|_{\substack{\phi=0\\ \epsilon=0}}$$

$$K_{\tau\theta} = \left.\frac{\partial g_\tau(\phi,\epsilon)}{\partial\phi}\right|_{\substack{\phi=0\\ \epsilon=0}} \qquad K_{\tau\tau} = \left.\frac{\partial g_\tau(\phi,\epsilon)}{\partial\epsilon}\right|_{\substack{\phi=0\\ \epsilon=0}} \tag{I.3.1.–3}$$

The corresponding linearized equivalent model is shown in Fig. I.3–4. For many synchronizer types and modulation formats of practical interest, it can be shown that $K_{\theta\tau} = K_{\tau\theta} = 0$, so that the resulting linearized equivalent model exhibits no coupling (see Fig. I.3–5). When $K_{\theta\tau} = 0$, the tracking performance of the carrier synchronizer can be analyzed as if symbol timing were perfect; when $K_{\tau\theta} = 0$, the tracking performance of the symbol synchronizer can be analyzed as if carrier phase estimation were perfect. When coupling is absent, the linearized equivalent model can be further simplified, as shown in Fig. I.3–6, where the closed loop transfer functions $H_\theta(z)$ and $H_\tau(z)$ are defined as

$$H_\theta(z) = \frac{K_{\theta\theta}F_\theta(z)}{z - 1 + K_{\theta\theta}F_\theta(z)}, \quad H_\tau(z) = \frac{K_{\tau\tau}F_\tau(z)}{z - 1 + K_{\tau\tau}F_\tau(z)} \tag{I.3.1–4}$$

It follows from Fig. I.3–6 that the estimation errors ϕ and ϵ consist of two terms each. The first term is a lowpass filtered version of the phase error detector or timing error detector noise (which is due to additive noise and self– noise); its variance easily follows from standard linear filter theory, and is given by

$$\begin{aligned} \mathrm{Var}[\phi] &= \frac{1}{K_{\theta\theta}^2}\int_{-1/2}^{+1/2} S_\theta(fT;0,0)\,|H_\theta(\exp(j2\pi fT)|^2\,d(fT) \\ \mathrm{Var}[\epsilon] &= \frac{1}{K_{\tau\tau}^2}\int_{-1/2}^{+1/2} S_\tau(fT;0,0)\,|H_\tau(\exp(j2\pi fT)|^2\,d(fT) \end{aligned} \tag{I.3.1–5}$$

The second term is a highpass filtered version of the carrier phase θ or the normalized time delay τ/T, and is identically zero when θ or τ are constant. This second term models the effect of small phase noise, Doppler shifts and frequency uncertainties.

The absence of coupling between the (linearized) carrier and symbol synchronizer not only simplifies the tracking performance analysis, but is also of practical importance. As the carrier phase fluctuations are in general more rapid than the time delay fluctuations, the carrier synchronizer should be designed as to have a larger bandwidth than the symbol synchronizer. However, when coupling is present, the carrier synchronizer dynamics are determined not only by the time

constants of the carrier synchronizer, but also by the much larger time constants of the clock synchronizer. The overall effect is that coupling slows down the dynamics of the carrier synchronizer [Men'77; Fal'77; Fra'80]. Finally, we briefly mention that the coupling between the (linearized) carrier and symbol synchronizer in the case of an off–centered receiver for MSK–like signals can be exploited to minimize the effects of a static phase error by making sampling timing adjustments [Rya'80, Ana'86].

In Part II of this report, we have evaluated the tracking error variances (I.3.1–5) of most feedback synchronizers from the classification in section I.2.

– Cycle slip analysis

A cycle slip occurs when the estimation error (carrier phase error or timing error) moves from a stable equilibrium point to a neighbouring stable equilibrium point. Although cycle slips are rare phenomena, they could have a detrimental effect on receiver operation, because each cycle slip gives rise to many decision errors.

The theoretical analysis of cycle slips starts from the nonlinear equivalent model from Fig. I.3–1, which shows that in general carrier and symbol synchronization are coupled. As the bandwidth of the symbol synchronizer is in general much smaller than the carrier synchronizer bandwidth, cycle slips of the carrier synchronizer have only a short duration with respect to the inverse of the symbol synchronizer loop bandwidth, and, therefore, are not likely to give rise to cycle slips of the symbol synchronizer. Hence, cycle slips of the symbol synchronizer can be studied by ignoring the coupling with the carrier synchronizer; this corresponds to assuming that $\phi = 0$ (perfect carrier recovery). On the other hand, cycle slips of the symbol synchronizer have a long duration with respect to the inverse of the carrier synchronizer bandwidth, and, therefore, are likely to give rise to cycle slips of the carrier synchronizer. However, as the cycle slip rate of the symbol synchronizer is expected to be much smaller than the cycle slip rate of the carrier synchronizer (because of considerably different synchronizer bandwidths), most of the carrier synchronizer slips are not caused by symbol synchronizer slips. Hence, carrier synchronizer slips can be studied by ignoring the coupling with the symbol synchronizer; this corresponds to assuming that $\epsilon = 0$ (perfect symbol synchronization). The resulting models for cycle slip analysis of carrier and symbol synchronizers are shown in Fig. I.3–7.

Although cycle slips in symbol synchronizers are usually much more infrequent than in carrier synchronizers, they are worth studying, because they have a larger impact than carrier slips. Indeed, whereas both types of slips give

rise to an error burst, a symbol synchronizer slip also yields a loss of word synchronization because of the omission/repetition of a data symbol.

Similar models have been used in the literature for investigating cycle slips in analog synchronizers. Cycle slipping in analog synchronizers is studied by applying Fokker–Planck techniques to the stochastic differential equations which describe the operation of the synchronizer [Mey'75; Lin'77; Ryt'78]. A closed–form expression for the cycle slip rate can be obtained only for a first–order loop. An experimental study of cycle slips in second–order loops has shown that, in order to avoid bursts of cycle slips, the loop damping factor should be large [Asc'82]. For a large damping factor, the dynamical behaviour of a second–order loop is nearly the same as for a first–order loop. Hence, when a synchronizer operates at a low E_b/N_o where cycle slips become important, a large damping factor should be chosen in order to avoid bursts of slips; the resulting cycle slip rate is closely approximated by the expression for first–order loops.

The closed–form expression for the cycle slip rate in an analog first–order loop is difficult to evaluate numerically, because it contains a double integral. In [Moe'85a; Moe'85b], a simple but accurate approximation for the cycle slip rate in a first–order loop has been derived, and was found to agree satisfactorily with simulation results.

Digital synchronizers are modelled by means of stochastic difference equations, because the estimates are updated in discrete time. Strictly speaking, Fokker–Planck techniques, which are restricted to stochastic differential equations, cannot be applied. However, since the synchronizer bandwidth is much smaller than the rate at which the estimates are updated (this rate usually equals the symbol rate), the stochastic difference equations can be accurately approximated by stochastic differential equations (differences are replaced by differentials). Then cycle slips in digital synchronizers are studied by applying Fokker–Planck techniques to these stochastic differential equations.

The cycle slip performance of various feedback synchronizers is studied in Part VI.

– Acquisition analysis

The theoretical analysis of acquisition performance (acquisition time, hangups, sideband lock, spurious lock) is extremely difficult, because the coupling between carrier recovery and clock recovery cannot be ignored during acquisition, and because of the nonlinearities involved. Moreover, an extra coupling with frequency estimation arises when frequency–aided acquisition is used.

The literature does not provide us with any general method of theoretical analysis for acquisition in analog or digital synchronizers; most investigations are based upon computer simulations.

In this report, acquisition of feedback synchronizers is treated in Part V.

I.3.2 Feedforward synchronizers.

There are two categories of feedforward synchronizers. The first category produces an estimate by direct computation : the estimate is an explicit function of the received complex envelope. This is the case for all feedforward carrier synchronizers (DDMLFF, NDAMLFF, V&VFF, DDMLFF$\bar{T}$, NDAMLFF$\bar{T}$ and NDASQFF$\bar{T}$O), and for some feedforward symbol synchronizers (NDANLFF, NDASQFF$\bar{C}$B/Q and NDASQFF$\bar{C}$O). The second category produces an estimate by performing a search over a range of trial values. This is the case for some feedforward symbol synchronizers (DDMLFF, NDAMLFF and NDAMLFF$\bar{C}$B/Q). The literature does not present an equivalent feedforward model that can be used for both categories.

Let us concentrate on the feedforward carrier synchronizer structure, shown in Fig. I.3–8. The phase detector performs a nonlinear operation on the interpolator output samples. Hence, the feedforward carrier synchronizer is influenced by the symbol synchronization. Possibly, decision feedback is used, which introduces additional coupling. The phase detector output passes through an averaging lowpass filter (with transfer function $H_\theta(z)$), which reduces the effect of phase detector noise. The carrier phase estimate $\hat{\theta}$ is obtained by applying to the filtered phase detector output signal a nonlinear "measurement" function; depending on the type of synchronizer and on the modulation format, this measurement function equals arg(.), arg(.)/2 or arg(.)/4. The carrier phase estimate is passed to the phase rotator.

Non–search feedforward symbol synchronizers (see Fig. I.3–9) operate in a similar way as feedforward carrier synchronizers. In general, the timing detector operates on the phase–rotated matched filter output signal. Hence, the feedforward symbol synchronizer is influenced by the carrier synchronizer; additional coupling is introduced when using decision feedback. The timing detector output is averaged by a lowpass filter with transfer function $H_\tau(z)$, and then applied to a measurement function (involving the function arg(.)), which delivers a timing estimate to the interpolator.

Some feedforward symbol synchronizers determine the timing estimate by means of a search over a set of trial values (see Fig. I.3–10). The timing detector must produce an output signal for each considered trial value of the timing. A search over all considered trial values yields the timing estimate, which

maximizes the output of the averaging lowpass filter.

– Tracking analysis

Any feedforward synchronizer considered in this study produces an estimate (either by direct computation or by means of a search) which maximizes, with respect to a trial value of the carrier phase or the symbol timing, a function that depends on the received signal; in most cases this function is (related to) the DD or NDA log–likelihood function. An alternative approach to obtain a carrier phase or symbol timing estimate is to use a feedback synchronizer, whose error signal is the derivative of the function to be maximized. In the tracking mode, this feedback synchronizer yields the same estimates as the feedforward synchronizer, provided that the closed loop transfer function of the feedback synchronizer is the same as the transfer function of the averaging lowpass filter of the feedforward synchronizer; basically, this is because the feedback and feedforward estimates maximize the same function. Hence, the tracking performance of the feedforward synchronizer is obtained by analyzing the corresponding feedback synchronizer, by means of the simplified linear model from Fig. I.3.1–6. When the parameters to be estimated do not fluctuate in time, the resulting estimation error variances are given by (I.3.1–5).

In Part II of this report, we have evaluated the tracking error variances (I.3.1–5) of most feedforward synchronizers from the classification in section I.2.

– Cycle slip analysis

Feedforward carrier synchronizers are well–suited for TDMA applications, because of their ability of rapid phase acquisition during a short signal burst [Vit'83]. As there is no a priori relation between the carrier phases of successive bursts, the carrier synchronizer must acquire the carrier phase of each individual burst. Hence, cycle slips can occur only when the carrier phase error exhibits large fluctuations (in the order of $2\pi/M$, with M=2 for BPSK and OQPSK, and M=4 for QPSK) within a single burst. When TDMA bursts are short, the carrier phase error fluctuations within a burst are very small (in many cases, even one carrier phase estimate per burst might be sufficient, so that cycle slipping becomes impossible). Therefore, cycle slips are not a concern in short burst TDMA applications.

In the case of continuous transmission or long TDMA bursts, feedforward carrier and symbol synchronizers systematically slip cycles when small frequency offsets are present. When the signal–to–noise ratio is not too small, these slips can be corrected by observing the recent evolution of the estimates [Asc'89]. However, occasional cycle slips due to the phase (or timing) detector noise cannot

be corrected. Cycle slips in feedforward synchronizers have not been dealt with in the literature. Fokker–Planck theory does not seem to be the appropriate tool to tackle the problem; maybe level–crossing theory, which has successfully been applied in a different context [Moe'88a], could lead to a solution.

Cycle slipping in feedforward synchronizers is briefly considered in Part VI of this report.

– Acquisition analysis

As for feedback synchronizers, the literature does not provide us with a general method for theoretical analysis of acquisition in analog or digital feedforward synchronizers, so that computer simulations are the only way to obtain numerical results.

Acquisition in feedforward synchronizers is dealt with in Part V of this report.

I.4 Simulation Tools and Models

In this section a description of the methods and the software tools used in the simulation studies is given. The following introductory subsection provides a survey of the functions of the simulation system COSSAP used in the study. Also, the various levels of documentation available in COSSAP are presented to ease verification of the simulation results. In the second subsection, subblocks common to the simulation of all clock and carrier recovery schemes are described, i.e. the models of transmitter and channel as well as the statistical evaluation. Finally, in subsections I.4.3 and I.4.4 carrier and clock synchronizer models are given.

I.4.1 Introduction

For the simulations the Communication System Simulation and Analysis Package (COSSAP) developed by CADIS GmbH has been used. In COSSAP, communication systems are modelled for simulation by placing and routing blocks as they are used in common blockdiagram representations of systems. Each block corresponds to a COSSAP model. There are two types of models: 1) primitive models which directly correspond to a Fortran or C subroutine and 2) hierarchical models which consist of several primitive models (and/or hierarchical models). A primitive model (referenced as 'module' in the subsequent text) implements an elementary function like squaring or a higher level function like filtering or decoding. Most of the models must be parameterized, e.g. the bandwidths of a filter or the signal-to-noise ratio of a noisy channel must be specified. Signals may be integer, real (complex signals are represented by two real signals but do not form an explicit signal type), bit_vector (for bit true simulation of digital signal processing) or double_precision.

Top level of the description of a COSSAP simulation is the blockdiagram set up during the configuration in the COSSAP frontend. For each primitive model of the system model libraries a documentation is included in the manual but also is available on-line. Among other entries this documentation gives a short and a long description of the primitive model function, the model parameters and the input/output ports. For user developed models such a documentation must be generated as well, before this model can be used in COSSAP. Appendix I.A shows a cross reference list of primitive

model names and short descriptions. Full documentation sheets of the primitive models used in this study are given in a separate volume (Annex A).

Blockdiagram and primitive model description form a complete functional description of the simulated system. For full information about a particular simulation, the parameter settings must also be known. In this report only the relevant parameters will be given for the individual simulations, i.e. signal-to-noise ratio, filter bandwidth, roll-off factor, back-off, etc. Of course, the complete parameter settings are stored with the simulations actually run. To understand this, a more detailed description of the COSSAP simulation file system may be useful.

Whenever a system configuration is completed and a system shall be simulated, the first step will be to "create" a simulation. Then, COSSAP assigns an unique identification number ('the simulation number') to this particular simulation and generates a set of files which contain the complete configuration information, including all parameter settings and, eventually, data for input files. This set of files is referenced by the simulation number. From these files the complete configuration may be restored at any time. After creation, a simulation can be submitted. When the simulation is finished the above mentioned set of configuration files are stored together with the simulation results in a result directory, again completely referencable by the simulation number. Therefore, it is possible to review the configuration belonging to a particular result or even to re-run this simulation. This information is also available as print out ("report") together with additional run time information. To be precise, the description of a simulation is complete only when the source code itself is available. With COSSAP the system module source code is part of the system.

The blockdiagrams given in this part all show the lowest level of hierarchy, i.e. primitive models only. In the simulations it is of course more practical to use hierarchical blocks for basic functions like symbol timing or carrier synchronization algorithms. But to understand how the alghorithms have been simulated the primitive model representation is more helpful. Although COSSAP allows to use arbitrary naming, the primitive models mainly use Fortran compatible names for "historical" reasons. The individual functions should become clear from the attached short descriptions (see Appendix I.A).

I.4.2 General models

Several subsystems are common to many simulations, i.e. the transmitter, various channel models, and the statistical evaluation. These subsystems and the corresponding COSSAP models will be described in the following subsections.

I.4.2.1 Transmitter

Three modulation schemes are considered in this report: BPSK, QPSK and offset QPSK (OQPSK). In Tab. T.1 a unique representation of the received signal is given for all three schemes. The corresponding unique representation of the transmitted signal is

$$s(t) = \sqrt{E_B} \left[\sum_m a_m h_I(t-mT) + j \sum_m b_m h_Q(t-mT) \right]$$

where

BPSK: $h_I(t) = h(t)$, $h_Q(t) = 0$ (or $b_m = 0$ for all m)

QPSK: $h_I(t) = h_Q(t) = h(t)$

OQPSK: $h_I(t) = h(t)$, $h_Q(t) = h(t-T/2)$

The same model, shown in Fig. I.4.1, is used to generate the three different transmitted signals. The individual modulation schemes are selected by the appropriate parameter settings. Module BSRC2 is a pseudo random binary source based on a 31 bit maximum period, i.e. length $2^{31}-1$, linear feedback shift register. The next module SYMGCG combines N = NUM_BIT bits into one symbol (represented by an integer value between 0 and 2^N-1 where NUM_BIT is a module parameter, N = 1 for BPSK and N = 2 for (O)QPSK. SYMGCG uses a Gray code, i.e. symbols differing by one differ by only one input bit.

The MPSK modulator MPSKSM maps the input symbols K = 1,...,M-1 to complex MPSK symbols $\exp(j2\pi k/M)\cdot\exp(j\pi/M)$. The only parameter of the module is M, with M = 2 for BPSK and M = 4 for (O)QPSK. The impulse modulator IMPMOD increases the sampling rate to

N = RATE_FACTOR samples per symbol by inserting N-1 zeroes after each symbol. In addition, the samples are divided by the second module parameter SAMPLING_TIME so that the output samples (in the average) have unit power. The quadrature component (i.e. the imaginary part) of the signal is fed to the module DELAYR. In this module the quadrature component is delayed by DELAY samples. The module parameter is DELAY = 0 for BPSK and QPSK and DELAY = RATE_FACTOR/2 for OQPSK. Finally, the signal is passed through a pulse form filter. In this study a root cosine rolloff filter, module RCRFQC, has been used. The transfer function of this filter is the square root of a cosine rolloff transfer function $H(f) = \sqrt{G(f)}$ (see also eq. I.1.1-4)

$$G(f) = \begin{cases} \dfrac{1}{2f_0} & 0 \le |f| \le (1-\alpha)f_0 \\ \dfrac{1}{2f_0}\left[\dfrac{1}{2} - \dfrac{1}{2}\sin\left[\pi \dfrac{f - f_0}{2\alpha f_0}\right]\right] & (1-\alpha)f_0 \le |f| \le (1+\alpha)f_0 \\ 0 & (1+\alpha)f_0 < |f| \end{cases}$$

H(f) is shown in Fig. I.4.2.

Note that f_0 is the 3dB cutoff frequency of the root cosine rolloff filter and α is the rolloff factor. The spectrum is zero for $|f| \ge (1+\alpha)f_0$, the rolloff may vary between $0 < \alpha \le 1$.

Parameters of RCRFQC are CUTOFF_FREQUENCY_3DB (f_0), ROLLOFF (α), SAMPLING_TIME, GAIN, and FFT_LENGTH. Where not otherwise stated $f_0 = 1/(2T)$ and GAIN = 1 (to maintain unit signal power) have been used. RCRFQC works in the frequency domain. I.e. input signal sections of length FFT_LENGTH are transformed into the frequency domain, multiplied with the filter transfer function (represented by FFT_LENGTH samples), and the result is inversely transformed to the time domain. The first FFT_LENGTH/2 samples then are the output samples of this section. Since the second half of the input signal section must be used as first half of the next transform section, this method is called "overlap-save". Note that for correct operation the filter impulse response must be limited to FFT_LENGTH/2 samples. Therefore, FFT_LENGTH

determines the length of the simulated impulse response and, thus, the precision of the filter modelling. FFT_LENGTH = 2048 was found to be fully sufficient for most simulations.

I.4.2.2 Channel

Channels used in this study are additive, white, Gaussian noise (AWGN) channels, non-linearities (travelling wave tube, hardlimiter), carrier frequency offset and clock frequency offset.

AWGN

White Gaussian noise may be added by passing the signal through the module AWGNQC. This module adds pseudorandom noise with Gaussian distribution to the input signal. Like all COSSAP modules AWGNQC does not use the internal random number generator of the operating system but has its own software generator selected for optimum statistical properties of the noise sequence. Since AWGNQC is parameterized by the signal-to-noise ratio (SNR), the signal power must be known. By the switch SIGNAL_POWER_KNOWN the module can be programmed to measure the signal power first. Since in this study the signal power is known to be unity, this feature has not been used. The parameter FACTOR was set to (number of samples per symbol)/2 so that SNR = E_s/N_0.

Travelling wave tube amplifier (TWTA)

The TWTA model is shown in Fig. I.4.3. The complex input signal first is converted to a polar representation (MPOLAR). The amplitude is attenuated, i.e. multiplied by a constant factor ≤ 1, in the module MVSMUL to adjust the input backoff. TWT5AP models the influence of a TWT on the signal by a memoryless amplitude non-linearity (AM-AM conversion) and a memory less amplitude dependent phase shift (AM-PM conversion). Parameters of TWT5AP are the coefficients of rational polynomial approximations of the AM-AM and AM-PM conversions. The characteristics used in this study correspond to a Ku-band TWT amplifier from AEG Telefunken and are shown in Fig. I.4.4. Note that TWT5AP has been developed under the frame of this contract by adaption of an existing TWT module with polynomial approximation of lower order. After the TWT

the signal power is adjusted to unity power. APLC automatically adjusts the signal power to the level specified by the parameter OUTPUT-POWER (set to 1 here). The input power is determined by measurement and assumed to be constant. Finally, the complex signal is converted to a quadrature component representation (real and imaginary part) by MRECT and fed to the AWGN channel model (AWGNQC).

Hardlimiter

The influence of a hardlimiter, followed by a zonal filter, on a carrier signal (carrier frequency ω_0) is modelled by HLBP. The zonal filter is assumed to extract the n-th harmonic of the limiter output signal (around frequency $n\omega_0$). Input and output signal are given by the complex equivalent baseband representation. In this study the first harmonic, i.e. around input carrier frequency, is used (OUT_HARMONIC = 1). The threshold is set to THRESHOLD = 0. The limiter output amplitudes are set to $\pm\pi/4$ to obtain a unity power first harmonic output signal. Note that the hardlimiter function in this case is given by

$$w = \frac{x + jy}{\sqrt{x^2+y^2}}$$

where x,y are the quadrature components of the input signal and w is the output signal.

Carrier phase offset

A carrier phase offset is introduced by multiplying the signal with a complex constant of unit amplitude. This is easily implemented with the module MCVROT. The only parameter THETA corresponds to the phase offset.

Carrier frequency offset

A carrier frequency offset is introduced by multiplying the signal with a complex function of unit amplitude but with constantly increasing angle. This is easily implemented with the module FDEVC. The only parameter

LAMBDA equals the frequency offset Δf normalized to the sampling rate $1/T_S$, LAMBDA = $\Delta f T_S$.

Clock phase offset and clock frequency offset

The most straightforward method to implement a clock phase offset is the insertion or deletion of signal samples in the beginning of the simulation. This method is mainly used in the analysis of the timing error detectors.

Since the clock phase resolution is determined by the number of samples per symbol, a different method is used in some cases, e.g. in the closed loop analysis of feedback symbol synchronizers. The module FDELSO introduces a (time varying) shift of the sampling instants, "a delay". The output signal sampling instants may be at arbitrary positions w.r.t the input sampling instants, the module uses sin(x)/x interpolation to determine samples which are not at input signal sampling instants. The sin(x)/x function is windowed, i.e. time limited, to have a realizable system. A window length of 1024 input samples was found to be fully sufficient. The delay is controlled by an additional input and may vary in time. Even constantly increasing or decreasing delays can be handled. Then, output sampling rate and input sampling rate are different. With FDELSO delay control values are read synchronously to the output sampling rate. The number of output samples for which one delay control value applies is set by a module parameter (SHIFTED_SAMPLES_PER_DELAY_VALUE). Because of the sin(x)/x interpolation the time continuous signal from which the input signal samples were taken must be bandlimited to 1/2 of the sampling rate. For another method to introduce a clock frequency offset see section I.4.3.1.

I.4.2.3 Statistical evaluation

The basic statistical parameters evaluated in the simulation were mean, variance and power spectral density.

The estimation of mean and variance is straightforward and is implemented in the module STATP. Evaluation may start after a delay to suppress the influence of transients. In general, the first block of samples equal to the FFT length of preceding filters is not used.

For the power spectral density estimation the module PSD1 is used, which is based on the Bartlett-periodogram-method. As with STATP evaluation is started with a delay. Output of the power spectral density estimation module is the double sided power spectral density as a function of fT_s, i.e. the frequency normalized to the sampling rate of the module. Thus, the power in a frequency band $0 \leq f \leq f_0$ is given by

$$P = \int_{-f_0T_s}^{f_0T_s} S(f)\ d(fT_s) = 2 \int_{0}^{f_0T_s} S(f)\ d(fT_s)$$

(For the sake of simplicity a continuous frequency representation is used rather than the actual discrete frequency sum).

For narrowband synchronizers the power spectral density of the carrier phase or timing error detector plays an important role in the analysis. Since a large number of samples is required for a sufficiently precise power spectral density estimate, PSD1 is used to analyse the power spectral density qualitatively. As will be demonstrated later the influence of the shape of the power spectral density can be reduced to a term which is independent of the frequency f and a term which is proportional to f^2 (for calculation of the synchronizer performance the power spectrum itself is not very useful). The constant term is given by the power spectral density at $f = 0$. Instead of using PSD1 which estimates the complete power spectrum, the signal to be analysed therefore is passed through a narrowband ideal lowpass filter, the module ILPF. Then, the variance of the filtered signal is estimated with STATP. For instance, with a cutoff frequency of the ideal lowpass filter of 5% of the sampling rate $1/T_s$ the variance σ_0^2 measured with STATP is

$$\sigma_0^2 = 2 \int_{0}^{0.05} S(f)\ d(fT_s) \simeq 0.1\ S(0)$$

Thus, $S(0) \simeq 10\sigma_0^2$.

The complete statistical evaluation model is shown in Fig. I.4.5.

I.4.3 Carrier synchronizer models

In the following subsections we use the naming conventions as discussed in subsection I.2.1, for example, DDMLFBB refers to the Decision Directed Maximum Likelihood FeedBack carrier synchronizer for BPSK which requires symbol timing to be known.

I.4.3.1 Common components

Components occuring repeatedly in the carrier synchronizer models are shown in Fig. I.4.6a-e. These common subblocks will be explained in the following text.

Sampled matched filter

Fig. I.4.6a shows the matched filter (RCRFQC) followed by a sampling module (SRD). The matched filter parameters are the same as for the transmit filter (the transfer function H(f) is real, thus, $H^*(f) = H(f)$. The matched filter output is sampled at 1 sample per symbol in the optimum sampling instant by SRD. With N samples per symbol SRD outputs every N-th input sample starting with the first input sample for BPSK and QPSK signals. For OQPSK, SRD outputs every (N/2)-th sample (i.e. 2 samples per symbol), thus, alternately providing the samples $p(k;\hat{\tau})$ and $q(k;\hat{\tau})$ ("serial implementation").

Phase rotator

Fig. I.4.6b shows the carrier phase correction. The received signal (i.e. the sampled matched filter output) is multiplied by $\exp(-j\hat{\theta})$. MVSIN and MVCOS provide $\sin\hat{\theta}$ and $\cos\hat{\theta}$, respectively. MCVMUL is a complex multiplier whose parameter COMPLEX_CONJUGATE_FLAG is set to -1 so that the received signal is multiplied by the complex conjugate of $\cos\hat{\theta} + j \sin\hat{\theta}$, i.e. by $\exp(-j\hat{\theta})$.

Integrator and phase rotator

Fig. I.4.6c implements a carrier phase rotator as in Fig. I.4.6b preceded by the discrete time equivalent of an integrator (which plays a similar role to the VCO, voltage controlled oscillator, in an analog implementation). DELAYR operates as a one sample delay (z^{-1}). Thus MVADD and DELAYR form a filter with transfer function $F(z) = \frac{1}{1-z^{-1}} = \frac{z}{z-1}$.

Decision feedback

Fig. I.4.6d shows the decision feedback component. MPSKSD is the optimum demodulator for MPSK (with an AWGN channel). Output of MPSKSD are the detected symbols (i.e. integer numbers between 0 ... 2^{N-1}, as they are input to the modulator MPSKSM, see section I.4.2.1). These symbols are fed to the MPSK modulator MPSKSM (the parameter M of MPSKSD and MPSKSM is M = 2 for BPSK and M = 4 for (O)QPSK). The input signal to this subblock is multiplied with the complex conjugate of the remodulated symbols (COMPLEX_CONJUGATE_FLAG = -1 in MCVMUL).

Accumulator filter

In principle, we may implement the accumulator filter recursively

$$s(k+1) = s(k) + x(k) - x(k-M)$$
$$y(k) = s(k)/M$$

where x(k) is the filter input and y(k) is the filter output. With integer arithmetic which is preferable with a digital synchronizer implementation this works well and is computationally efficient. With floating point arithmetic mainly used in the simulations we experience rounding error accumulation. Therefore,

$$y(k) = \frac{1}{M} \sum_{m=k-M+1}^{k} x(m)$$

must be computed for each k. Without frequency deviation we use a simpler implementation for analysis purposes. The module INTDMP (INTegrate and DuMP) implements

$$\frac{1}{M} \sum_{m=0}^{M-1} x(m)$$

Since only one result is output per M input samples each output sample is repeated M times by the module HOLD.

Separation of $p(k;\hat{\tau})$ and $q(k;\hat{\tau})$ for OQPSK

Finally, in Fig. I.4.6e the separation of $p(k;\hat{\tau})$ and $q(k;\hat{\tau})$ is shown. The real part of the received signal samples (2 samples per symbol) is fed to the upper DEMUX module which alternately directs the input samples to its output ports. In the same way, the lower DEMUX module separates the imaginary parts of the $p(k;\hat{\tau})$ and $q(k;\hat{\tau})$ sequences. For clarification the signal names are shown.

Note that the two DEMUX modules perform a serial to parallel conversion. Therefore, the sampling rate at the outputs is 1/2 of the sampling rate at the inputs.

Since the root cosine rolloff filters do not introduce a delay, the first matched filter output sample is $p(0;\hat{\tau})$. Both DEMUX modules output the first sample to their first output port. Therefore, proper alignment of the two DEMUX modules is achieved by appropriate routing of the signals.

Clock frequency offset

To determine the performance of carrier synchronizers independent of timing a simple way to average over different timing errors is used, see Fig. I.4-6f. The signal is sampled for instance with 16 samples per symbol. Then, by a demultiplexer (DEMUX) one of for instance 1000 samples is omitted (send to KILL). Thus, for the receiver the timing error is advanced by T/16.

I.4.3.2 Decision directed maximum likelihood carrier synchronization. Timing estimate is assumed to be known

FeedBack

The structure of the DDMLFB is shown in Fig. I.4.7a for BPSK and QPSK. The matched filter output $(p(k;\hat{\tau}))$ is sampled at one sample per

symbol and multiplied by $\exp(-j\hat{\theta})$ in MCVMUL. After decision feedback the imaginary part

$$\mathrm{Im}\left[\ \hat{c}_k^{*}\ p(k;\hat{\tau})\ \exp(-j\hat{\theta})\ \right]$$

is amplified by MVSMUL (multiplication with a constant factor) and fed to the discrete time integrator.

For OQPSK (Fig. I.4.7b) the matched filter output is sampled twice per symbol to obtain $p(k;\hat{\tau})$ and $q(k;\hat{\tau})$ alternately. The real part of $p(k;\hat{\tau})$ (which is the in-phase component sampled at the optimum instant) and the imaginary part of $q(k;\hat{\tau})$ (which is the quadrature component sampled at the optimum instant) are fed to the decision module. The imaginary part of $p(k;\hat{\tau})$ and the real part of $q(k;\hat{\tau})$ are multiplied with the remodulated decisions. The difference computed in MVSUB is the phase detector output signal as given in Tab. T.4

$$\begin{aligned} u_{\theta}(k;\hat{\theta}) &= \mathrm{Im}\left[[\hat{a}_k p(k;\hat{\tau}) - j\hat{b}_k q(k;\hat{\tau})]e^{-j\hat{\theta}}\right] \\ &= \hat{a}_k \mathrm{Im}\left[\ p(k;\hat{\tau})e^{-j\hat{\theta}}\ \right] - \hat{b}_k \mathrm{Re}\left[\ q(k;\hat{\tau})e^{-j\hat{\theta}}\ \right] \end{aligned}$$

The result is amplified by MVSMUL (multiplication with a constant factor) and fed to the discrete time integrator. After the integrator the sampling rate is doubled to two samples per symbol (OQPSK) by duplicating each sample in the HOLD module.

Note that both Fig. I.4.7 show first order loops.

<u>Feedforward</u>

The structures of the two DDMLFF are shown in Fig. I.4.8a, Fig. I.4.9a for BPSK and QPSK, and in Fig. I.4.8b, Fig. I.4.9b for OQPSK. In the feedforward structures the carrier phase is detected immediately from the sampled matched filter output.

In Fig. I.4.9a the sampled matched filter output is multiplied with the remodulated decisions in MCVMUL. After multiplication with 1/K in MCVCSM the signal is fed to the accumulator filter. The result, $\hat{C} + j\hat{S}$,

finally is used for correction in MCVMUL which multiplies the sampler output by the complex conjugate of $\hat{C} + j\hat{S}$ before decision. The only difference in Fig. I.4.8a is that from $\hat{C} + j\hat{S}$ in MATAN2 the carrier phase estimate $\hat{\theta} = \arg(\hat{C} + j\hat{S})$ is determined. The sampler output in this case is multiplied by $\exp(-j\hat{\theta})$ before decision.

In the two OQPSK synchronizers Fig. I.4.8b, Fig. I.4.9b the matched filter output, now sampled at two samples per symbol, is demultiplexed to obtain the real and imaginary parts of $p(k;\hat{\tau})$ and $q(k;\hat{\tau})$ separately. On the right hand side $p(k;\hat{\tau})$ is multiplied by $\hat{a}_k$ and on the left hand side $q(k;\hat{\tau})$ is multiplied by $\hat{b}_k$. The imaginary part of $\hat{a}_k p(k;\hat{\tau}) - j\hat{b}_k q(k;\hat{\tau})$ is computed by MVSUB and the real part in the MVADD module. The result is processed as in the corresponding BPSK/QPSK synchronizers.

I.4.3.3 Non data-aided synchronizer. Timing estimate is assumed to be known

<u>Maximum Likelihood, FeedBack</u>

The NDAMLFB synchronizers are shown in Fig. I.4.10a-c. For BPSK (a), OQPSK (b), and QPSK (c) the structure is slightly different.

With BPSK (Fig. I.4.10a) the sampled matched filter output (1 sample per symbol) is squared in the second complex multiplier MCVMUL after phase rotation in the first MCVMUL module. The imaginary part which is the phase detector output given in Tab. T.4 is amplified (multiplied with a real, constant factor) and fed to the discrete time integrator.

With OQPSK (Fig. I.4.10b) the $p(k;\hat{\tau})$ and $q(k;\hat{\tau})$ samples (1 sample per symbol each) after phase rotation are separated by the two demultiplexer modules and squared each in the two complex multipliers. The imaginary part of the difference (computed in MCVSUB) is the phase detector output given in Tab. T.4. Further processing is as for BPSK.

Note that according to

$$\mathrm{Im}\{p^2\} = \mathrm{Im}\{(x+jy)^2\} = 2xy$$

the squaring of a complex signal for BPSK and OQPSK could be replaced by multiplying the imaginary part with the real part of the signal. Squaring

has been used here to be closer to the formula given in Tab. T.4 (but this has little impact on the required computing time).

With QPSK (Fig. I.4.10c) the sampled matched filter output (1 sample per symbol) is raised to the fourth power by the two consecutive complex multipliers (MCVMUL) after phase rotation in the first MCVMUL module. The imaginary part is the phase detector output signal given in Tab. T.4. Further processing is as for BPSK and OQPSK.

Maximum likelihood, FeedForward

The NDAMLFF synchronizers are shown in Fig. I.4.11a-c. As in the feedback synchronizers the structure slightly differs for BPSK (a), OQPSK (b), and QPSK (c).

With BPSK (Fig. I.4.11a) the sampled matched filter output is squared by the complex multiplier MCVMUL. After multiplication with 1/K (MCVCSM multiplies a complex signal by a (complex) constant) the signal is fed to the accumulator filter. MATAN2 computes the argument of its complex input signal and after multiplication by 1/2 (in MVSMUL) the estimate $\hat{\theta}$ is obtained as given in Tab. T.4. Finally, $\hat{\theta}$ is used in the phase rotator to correct the received signal phase prior to detection.

With OQPSK (Fig. I.4.11b) the $p(k;\hat{\tau})$ and $q(k;\hat{\tau})$ samples (1 sample per symbol each) after phase rotation are separated by the two demultiplexer modules and squared each in the two complex multipliers. Further processing is as for BPSK. Again, after MVSMUL the estimate $\hat{\theta}$ is obtained as given in Tab. T.4. Finally, $\hat{\theta}$ is used in the phase rotator to correct the phase of $\mathrm{Re}\{p(k;\hat{\tau})\} + j\ \mathrm{Im}\{q(k;\hat{\tau})\}$ prior to detection.

With QPSK (Fig. I.4.11c) the sampled matched filter output (1 sample per symbol) is raised to the fourth power by the two consecutive complex multipliers (MCVMUL) after phase rotation in the first MCVMUL module. Further processing is as for BPSK and QPSK, except for the MVSMUL module where the signal now is divided by 4 to obtain the estimate $\hat{\theta}$ as given in Tab. T.4.

Viterbi & Viterbi, FeedForward

The V&VFFB/Q is shown in Fig. I.4.12a. The sampled matched filter output (1 sample per symbol) is converted to a polar representation by MPOLAR. MVSMUL multiplies the phase by 2 for BPSK and by 4 for QPSK to remove the data. The amplitude is passed through a nonlinearity f(x) (see Fig. I.4.12a). The result is converted back to a cartesian representation by MRECT. Further processing is as for NDAMLFFB/Q.

The V&VFFO is shown in Fig. I.4.12b. Both, p and q samples are converted to a polar representation by MPOLAR. The phase is multiplied by 2 in MVSMUL and the amplitude is passed through a nonlinearity f(x) (see Fig. I.4.12b). The results are converted back to a cartesian representation by MRECT and the q-path output is subtracted from the p-path output. Further processing is as for V&VFFB.

I.4.3.4 Carrier synchronization independent of timing

Except for OQPSK the corresponding COSSAP representations are as in the case where the symbol timing is assumed to be known. Now the matched filter output is sampled at rates in general above 1 sample per symbol which is implemented by appropriately setting the parameter RATE_FACTOR in SRD and, possibly, using a higher sampling rate on the channel. Note that this also means that as prefilter a matched filter is assumed, i.e. $h(t) = h_0(t)$.

For OQPSK the synchronizers which are independent of timing are different from the synchronizers when the timing is assumed to be known. The DDMLT̄ for OQPSK is identical to the DDMLT̄ for QPSK and the NDAMLT̄ for OQPSK is identical to the NDAMLT̄ for QPSK, i.e. OQPSK is treated like a QPSK signal in both cases. Note that unlike in the DDMLT and in the NDAMLT there is no timing alignment, i.e. no T/2 time shift in the quadrature channel.

The Algorithms are summarized in Tab. T.4, for the definition of the variables see Tab. T.1.

I.4.4 Symbol synchronizer models

In the following subsections we use the naming conventions as discussed in subsection I.2.2, for example, DDMLFF refers to the Decision Directed Maximum Likelihood FeedForward symbol synchronizer.

Note that in the figures showing the simulation models the symbol synchronizers for QPSK are shown (which make use of both the inphase and the quadrature signal for symbol synchronization).

I.4.4.1 Decision directed maximum likelihood symbol synchronisation

FeedForward

The DDMLFF symbol synchronizer is shown in Fig. I.4.13. The maximum search required in this type of synchronizer can only be carried out for a limited number of clock phase hypotheses. For ideal case analysis this number should be chosen so large, that performance measurement is not influenced by this parameter. For the measurement range used here 16 has been found appropriate. This case is shown in Fig. I.4.13. On the channel only four samples per symbol (which is still above the Nyquist rate of the root cosine rolloff filters) have been used to increase the computational efficiency. Then, prior to the received matched filter, a sampling rate expansion is inserted (IMPMOD) to obtain 16 samples per symbol. In the tracking mode the synchronizer operates close to the correct timing. Therefore, in the simulation the samples at the optimum sampling instant are fed to the decision feedback (MPSKD, MPSKM). Each symbol is replicated 16 times (HOLD) to get a symbol sample for each symbol timing hypothesis. Since in COSSAP filters like RCRFQC are implemented such that the maximum of the impulse response is the first output sample, i.e. without a delay that eventually would have to be compensated in a bit error measurement, the very first output sample of the matched filter is in the middle of the first symbol. Thus, transmission starts with half a symbol and, for proper alignment, half of the symbol samples of the very first symbols have to be omitted (DELETE) in the decision feedback path. The received signal samples are multiplied with the complex conjugate of the decisions (MCVMUL). Note that with this type of an implementation QPSK symbol synchronisers make use of both channels. With the subsequent DELETE module the range bounds of the maximum

search can be set with respect to the symbol bounds. In the serial to parallel conversion (S2P) the received symbol samples are separated with respect to the clock phase value. Values belonging to the same hypothesis are summed up over an interval of length K in the integrate and dump module (INTDMP), interleaved back in the parallel to serial conversion module (P2S) in the same position order as before the block. Finally, MAXRP determines the relative position of the maximum within each block of 16 samples. The rate is 16 samples per K symbols before the MAXRP module and one position sample for each K symbols after this module. The relative position is fed to the statistical analysis section.

Feedback

The feedback symbol synchronizer operates with the channel sampling rate that is generally used (i.e. in most cases 4 samples per symbol). The matched filter output is fed to the sampling module directly and via the differentiating filter (LPDIQC) to obtain p'(), the derivative of p() (see Fig. I.4.14). After sampling at the optimum position (SRD), the undifferentiated signal is fed to the decision feedback unit. The sampled differentiated signal is multiplied with the complex conjugate of the decisions (MCVMUL). The real part of the multiplied output signal is the phase detector output signal as given in Tab. T.5. Note that with this type of an implementation QPSK symbol synchronisers make use of both channels.

I.4.4.2 Non data-aided symbol synchronization

Maximum Likelihood, FeedForward

In the NDAMLFF synchronizer, shown in Fig. I.4.15, the sampling rate is increased in the same way as in the decision directed feedforward symbol synchronizer. Now, after the matched filter the signal is squared with a complex conjugate multiplier (MCVMUL). With the DELETE module the bounds of the estimation interval can be set with respect to the symbol bounds. As for the decision directed feedforward symbol synchronizer, the 16 samples are split up with respect to the phase (S2P), summed up over K symbols (INTDMP), and converted back to serial in the same order as before this block (P2S). Finally, in MAXRP the relative position of the

maximum in each block of 16 output values is determined and fed to the statistical evaluation section.

Maximum Likelihood, FeedBack

In the NDAMLFB synchronizer shown in Fig. I.4.16, the received signal is sampled in the optimum sampling instant (SRD), multiplied (MCVMUL) with the complex conjugate of the derivative of the received signal (LPDIQC) to obtain the phase detector output as given in Tab. T.5. Note that with the complex conjugate multiplier both channels are used with QPSK.

I.4.4.3 Symbol synchronization without self-noise

DDMLFB with decision feedback self-noise cancellation

To achieve self-noise cancellation an additional decision feedback path is included in the decision directed maximum likelihood feedback symbol synchronizer as shown in Fig. I.4.17. The impulse response of the two root cosine rolloff filters (RCRFQC) with an ideal differentiator in between (LPDIQC) is the derivative of a cosine rolloff pulse. To have at least Nyquist rate (with respect to the root cosine rolloff filters), the sampling rate is increased by the impulse modulator (IMPMOD). In SRD the output signal is sampled at kT. Thus, this part implements the discrete time convolution between the decisions and g'(kT). In a complex subtraction module (MCVSUB) the result is subtracted from the sampled derivative of the received signal prior to the normal decision feedback as in the standard decision directed maximum likelihood feedback symbol synchronizer.

SNFF, FeedForward Self-Noise cancellation

The modified derivative matched filter for feedforward self noise cancellation is shown in Fig. I.4.18. This filter replaces the differentiating filter (LPDIQC) in the feedback symbol synchronizers. Together with the preceding matched filter and the subsequent sampler Fig. I.4.18 implements the modified matched filter transfer function given in Tab. T.5. The right path of the derivative matched filter uses the same principle as the decision feedback self-noise cancellation path in Fig. I.4.17. The matched filter output is sampled (SRD) at the optimum sampling instant with one sample

per symbol. Then, with an impulse modulator (IMPMOD) the sampling rate is increased above the Nyquist rate of the root cosine rolloff filter. The two root cosine rolloff filters (RCRFQC) followed by the ideal differentiator (LPDIQC) have a total impulse response which is the derivative of the cosine rolloff filter impulse response. Together with the sampler which is not shown in Fig. I.4.18 the right path implements the discrete convolution between received matched filter samples and the derivative cosine rolloff filter impulse response sampled at kT. The left path is the ideal differentiator with transfer function $j2\pi f$. Note that by interchanging differentiating and subtraction one differentiator filter could be omitted to speed up simulation. Here, two differentiators have been used to clarify the correspondence between simulation and terms in the filter equation in Tab. T.5. Note that for the root cosine rolloff filters the same parameters as for the matched filters are used.

Decision directed MMSE, Feedback

The decision directed MMSE feedback symbol synchronizer is shown in Fig. I.4.19. The difference to the NDAML feedback symbol synchronizer is that the decisions (MPSKD, MPSKM) are subtracted from the matched filter output samples before multiplication with the derivative matched filter samples. The real part of the complex conjugate muliplier (MCVMUL) output is the phase dectector output signal as given in Tab. T.5. Note that with the complex conjugate multiplication both channels are used in QPSK transmission.

I.4.4.4 Decision Directed and Non Data Aided Early Late FeedBack symbol synchronisation and closely related algorithms

DDEL, Feedback

The COSSAP model of the decision directed early late feedback symbol synchronizer is shown in Fig. I.4.20. The delete module after the received matched filter omits the first NUM_ITEMS samples from the received signal. Thus, the output signal is advanced to the future with respect to the left decision feedback part. After sampling at a rate of one sample per symbol (SRD), the difference (MCVSUB) between the samples and the delayed samples is determined.

The result is multiplied with the complex conjugate of the remodulated decisions (MPSKM, MPSKD). The real part of the multiplier output corresponds to the phase detector output signal as given in Tab. T.5.

When one symbol is deleted in the DELETE module and DELAY = 2 in the DELAYR the EL symbol synchronisation algorithm for $\lambda = 1$ is obtained and when half a symbol is deleted and DELAY = 1 is selected we have the EL symbol synchronisation algorithm for $\lambda = 1/2$.

Note that with the complex conjugate multiplier (MCVMUL) both channels are used with QPSK modulation.

DTTL, Feedback

The data transition tracking loop feedback symbol synchronizer is shown in Fig. I.4.21. In the left path after sampling the difference between consecutive remodulated decisions, i.e. $(\hat{a}_{k-1}+j\hat{b}_{k-1}) - (\hat{a}_k+j\hat{b}_k)$, is determined. In the right path the matched filter output is delayed by half a symbol before sampling to get p((k-1/2)T). With the substitution k -> k+1 the real part of the complex conjugate multiplier output corresponds to phase detector output for the DTTL algorithm as given in Tab. T.5.

M&M, Feedback

The Mueller&Müller feedback symbol synchronizer is shown in Fig. I.4.22. On the lower right undelayed decisions are multiplied with the matched filter output samples, delayed by one symbol. The result is subtracted from the product between delayed (by one symbol) decisions and undelayed received matched filter output samples. The real part of the subtracting module (MCVSUB) output is the phase detector output signal as given in Tab. T.5.

Note that with the complex subtraction module (MCVSUB) the synchronizer makes use of both channels for QPSK.

mM&M, Feedback

The modified Mueller&Müller feedback symbol synchronizer is shown in Fig. I.4.23. On the left side the decisions are subtracted from the decisions delayed by two symbols. The result is multiplied by the sampled matched filter output, delayed by one symbol. The real part of the multiplier output is the phase detector output signal as given in Tab. T.5.

Note that with the complex conjugate multiplier (MCVMUL) shown in Fig. I.4.23 both channels are used for QPSK.

NDAEL, Feedback

In Fig. I.4.24 the NDA Early-Late feedback symbol synchronizer is shown. The difference to the decision directed early-late symbol synchronizer (Fig. I.4.20) is that decision and remodulation is omitted, i.e. the decisions are replaced by the matched filter output samples. Note that only $\lambda = 1/2$ is used with the NDAEL.

GArdner, Feedback

GArdner's symbol synchronizer (Fig. I.4.25) is similar to the DTTL. The only difference is that decision and remodulation are omitted. Instead of the decisions the original matched filter output are used.

NDANL, FeedForward

In Fig. I.4.26 a particular implementation of a non data aided nonlinear feedforward symbol synchronizer is shown, the digital filter and square symbol synchronizer. After squaring (MCVMUL) the DFT of the real part at 1/T is determined. Note that with the complex conjugate multiplier used here, both channels are used with QPSK signals.

With four samples per symbol (i.e. $t = kT/4$) implementation is very simple - no multiplication is required. The algorithm then reads

$$\hat{\tau} = \frac{T}{2\pi} \arg \left[\sum_{k=0}^{K-1} y_L(kT/4) y_L^*(kT/4)\ e^{-j\pi kfT/2} \right] \Big|_{f=1/T}$$

$$= \frac{T}{2\pi} \arg \left[\sum_{k=0}^{K-1} y_L(kT/4) y_L^*(kT/4)\ j^k \right]$$

The S2P module demultiplexes the four samples per symbol. Real and imaginary part of the DFT are determined as the difference between first and third sample and between second and fourth sample, respectively. [I,32]. The accumulation is implemented by a discrete time integrate-and-dump (INTDMP). The phase of the DFT coefficient (determined by MATAN2) is the symbol phase estimate.

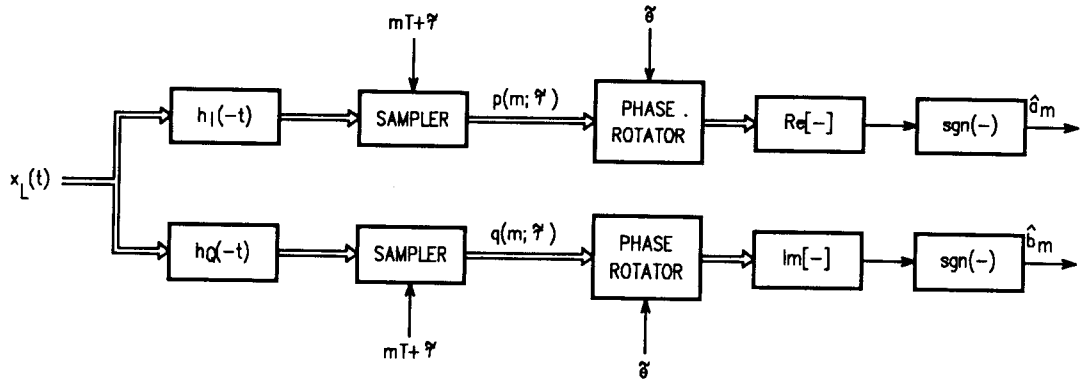


Fig. I.1–1 : Definition of $p(m;\tau)$, $q(m;\tau)$, $\hat{a}_m$ and $\hat{b}_m$

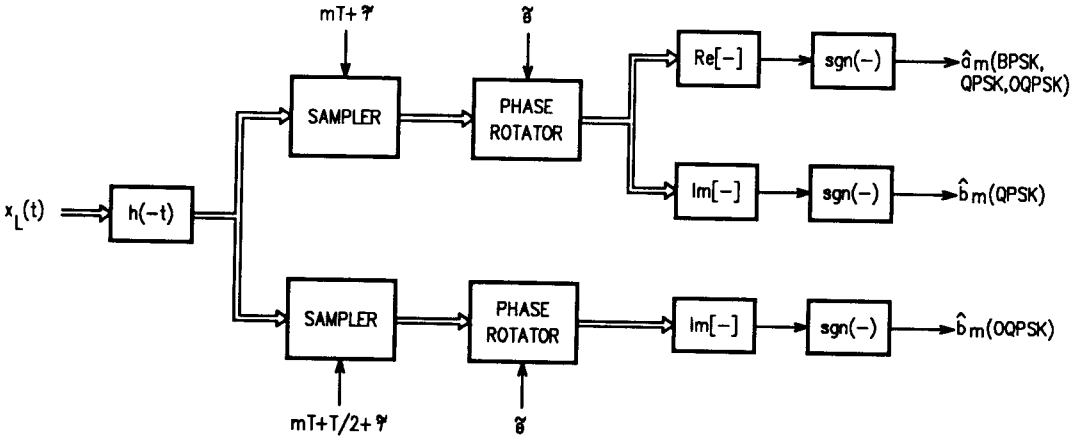


Fig. I.1–2 : implementation with a single matched filter

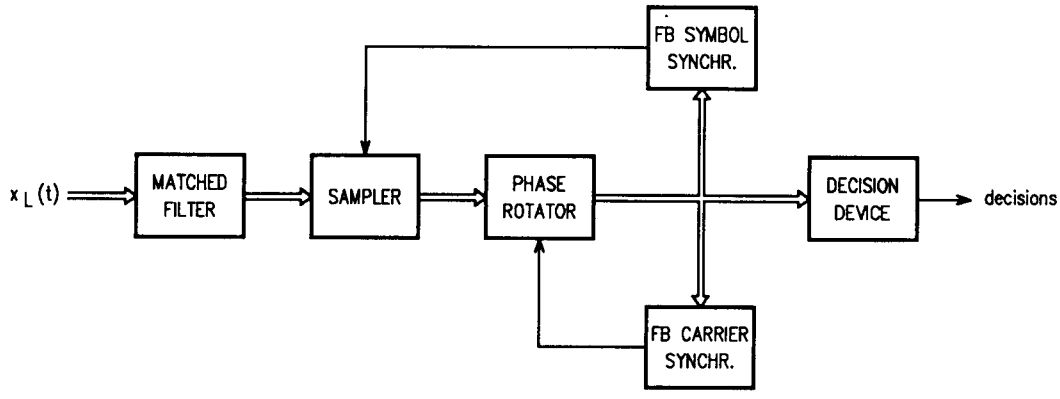


Fig. I.1–3: FB carrier synchronization, FB symbol synchronization

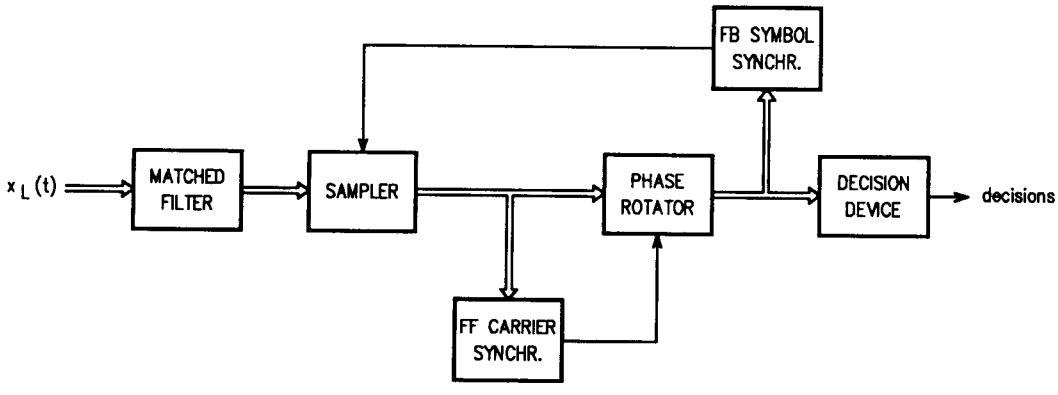


Fig. I.1–4: FF carrier synchronization, FB symbol synchronization

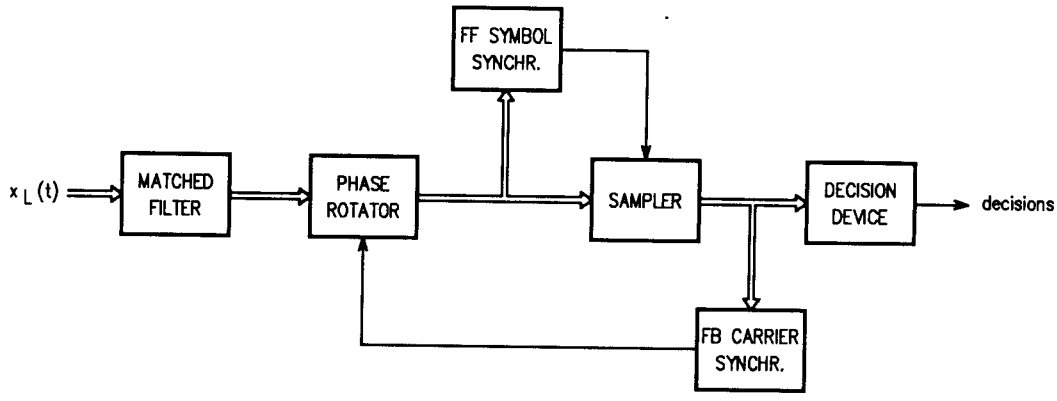


Fig. I.1–5: FB carrier synchronization, FF symbol synchronization

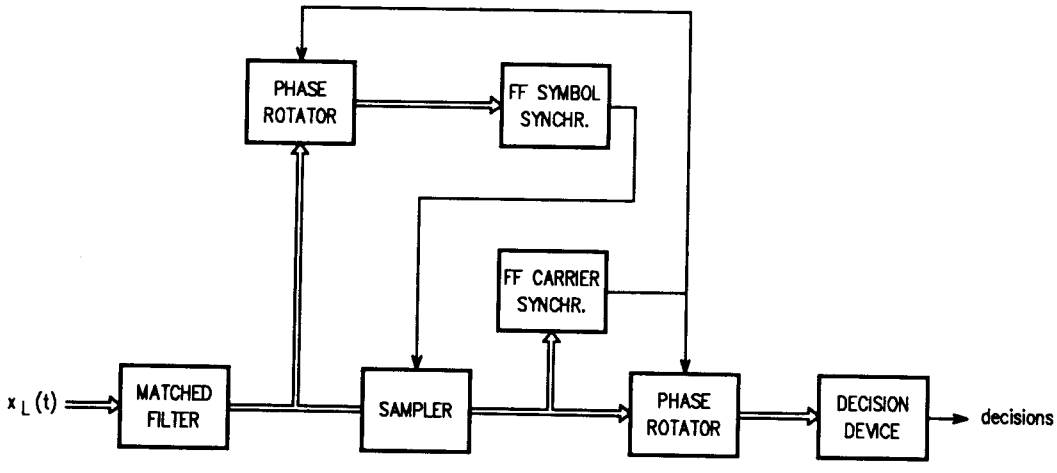


Fig. I.1–6: FF carrier synchronization, FF symbol synchronization

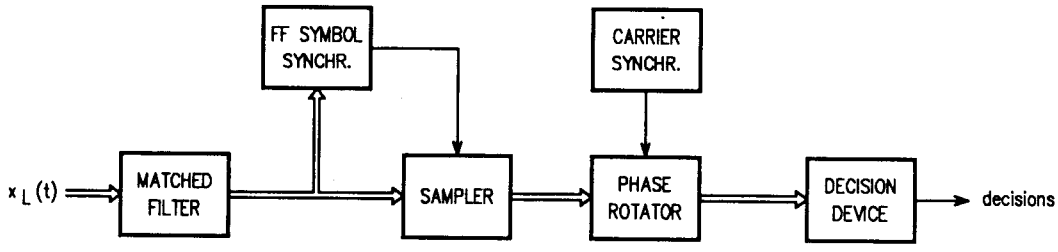


Fig.I.1–7: Carrier–independent FF symbol synchronization

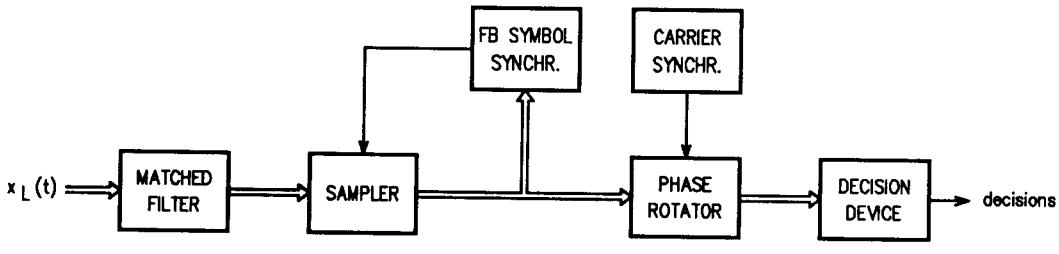


Fig.I.1–8: Carrier–independent FB symbol synchronization

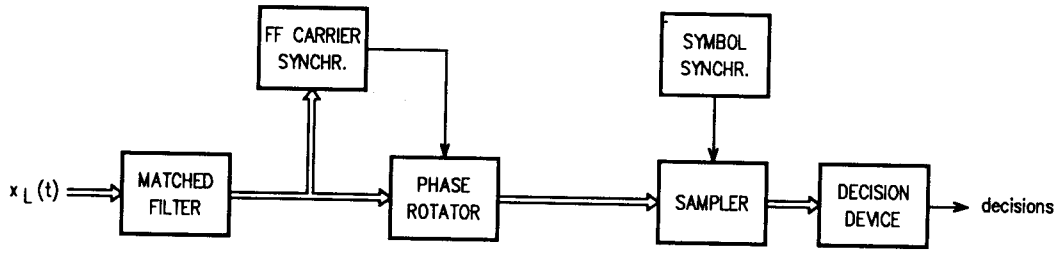


Fig.I.1–9: Timing–independent FF carrier synchronization

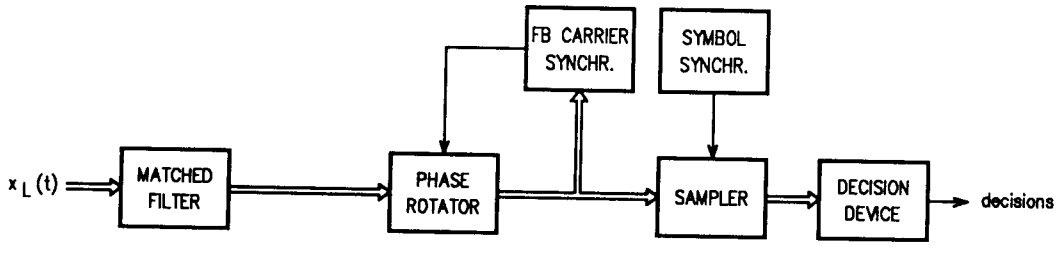


Fig.I.1–10:Timing–independent FB carrier synchronization

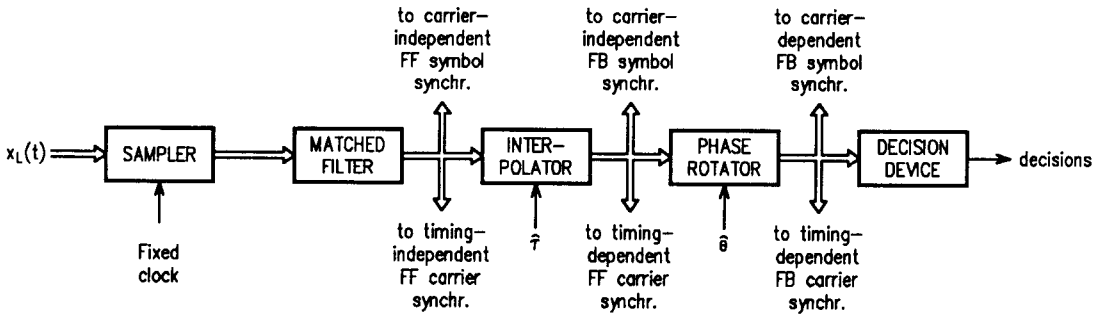


Fig.: I.1–11: digital implementation, with phase rotator following interpolator

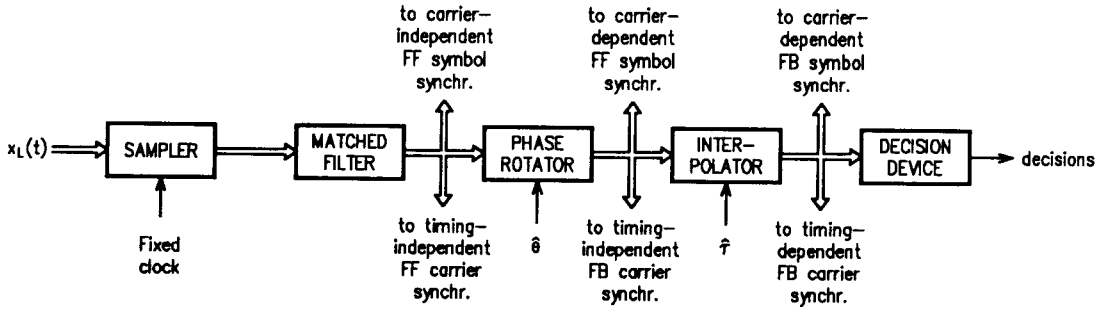


Fig.: I.1–12: digital implementation, with phase rotator preceding interpolator

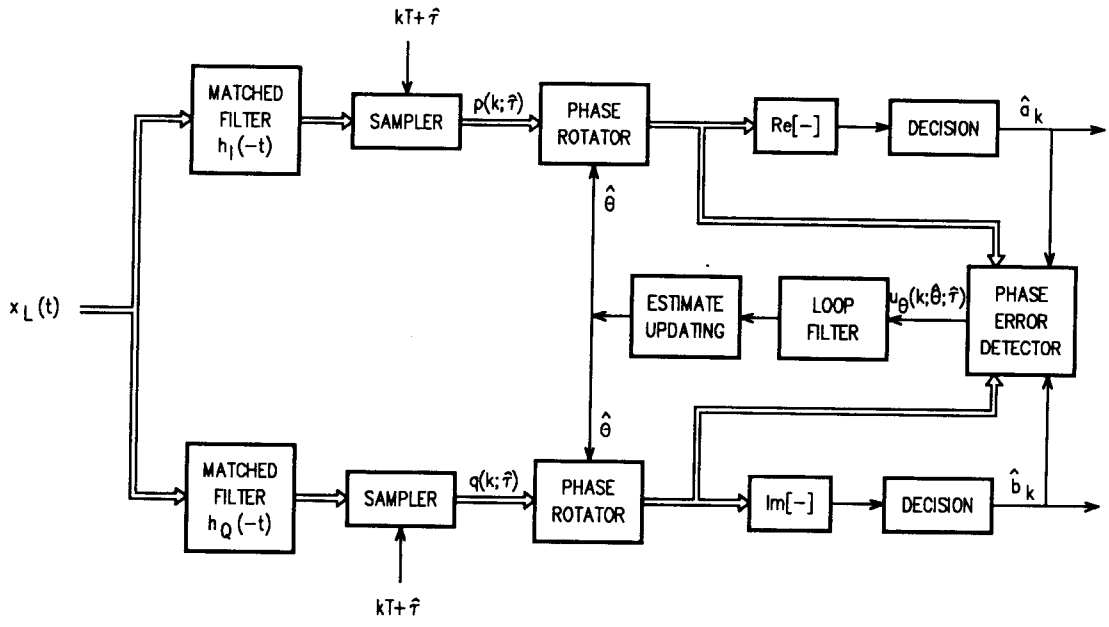


Fig.: I.2–1: DD ML feedback carrier synchronization (DDMLFB)

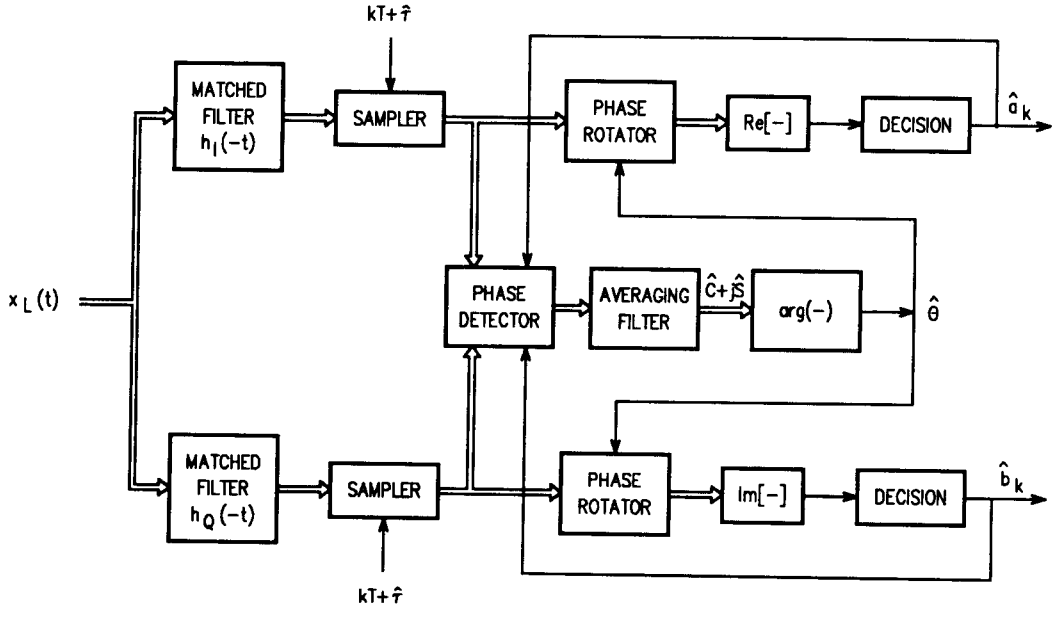


Fig.: I.2–2: DD ML Feedforward (plus DFB) carrier synchronization (DDMLFF)

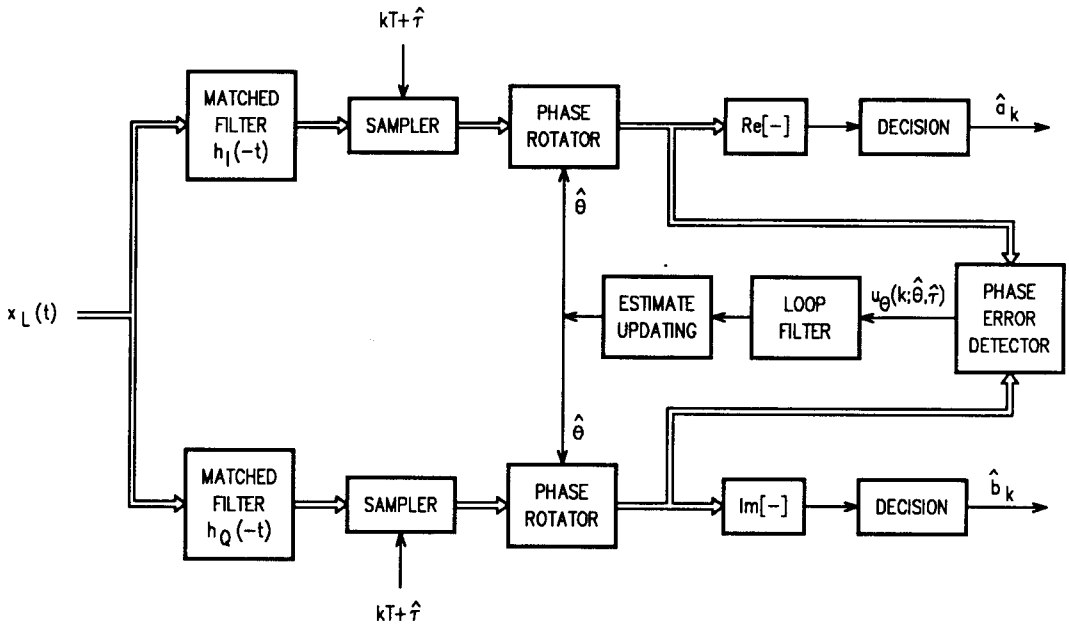


Fig. I.2–3 NDA ML Feedback carrier synchronization (NDAMLFB)

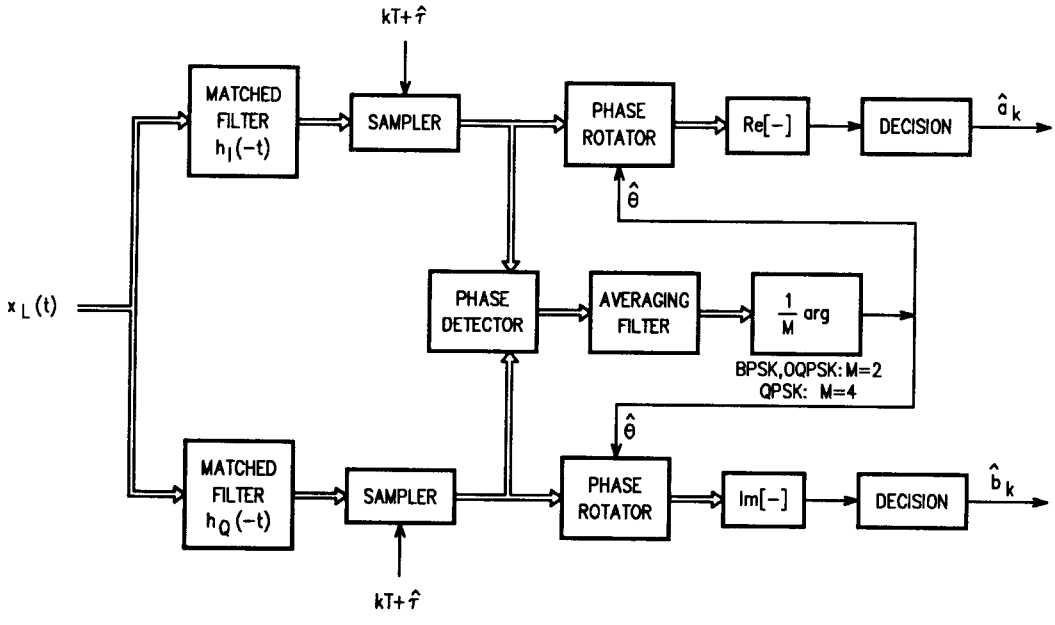


Fig. I.2–4: NDA ML Feedforward carrier synchronization (NDAMLFF)

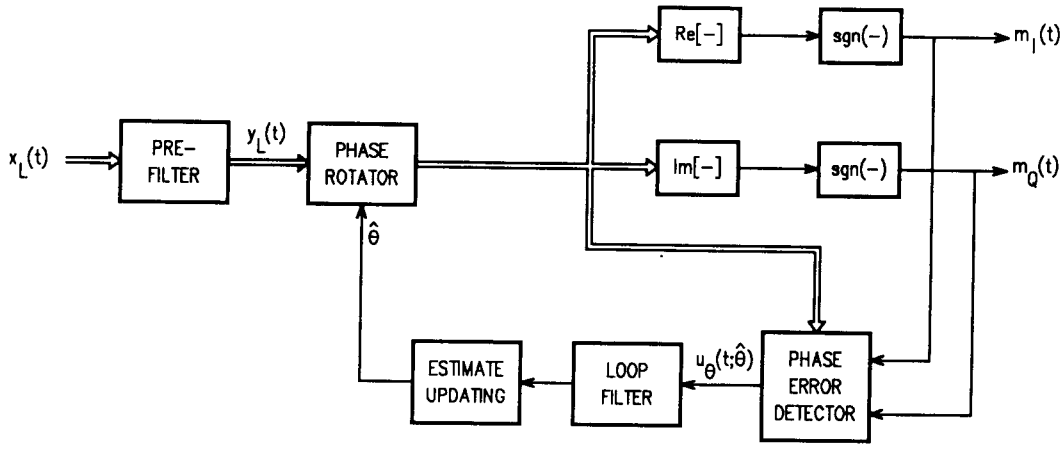


Fig. I.2–5 : Timing independent feedback carrier synchronization (DDMLFBT)

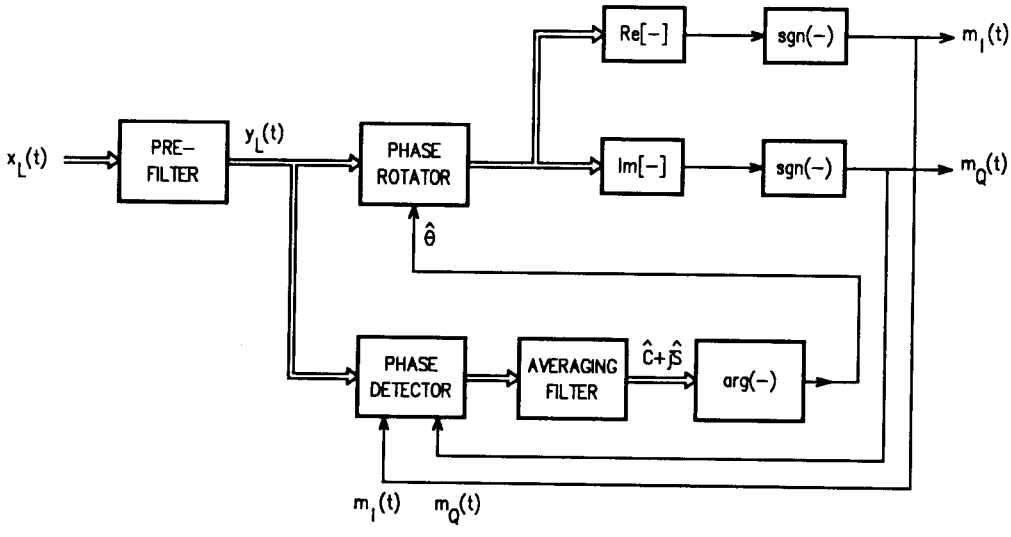


Fig. I.2–6 : Timing independent feedforward (plus DFB) carrier synchronization (DDMLFFT)

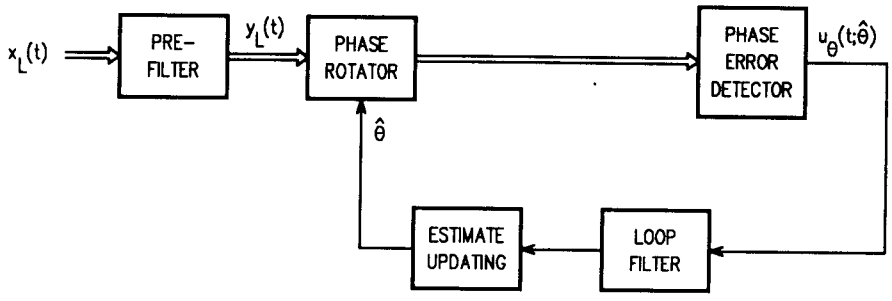


Fig. I.2–7 : Timing independent feedback carrier synchronization (NDAMLFBT)

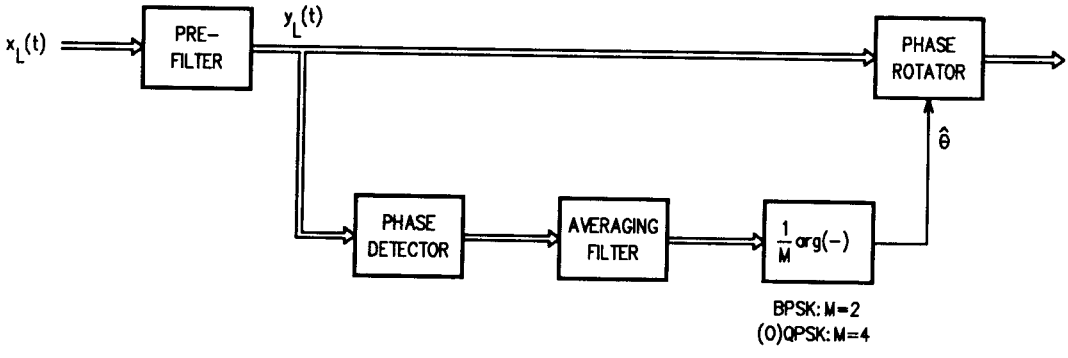


Fig. I.2–8 : Timing independent feedforward carrier synchronization (NDAMLFFT)

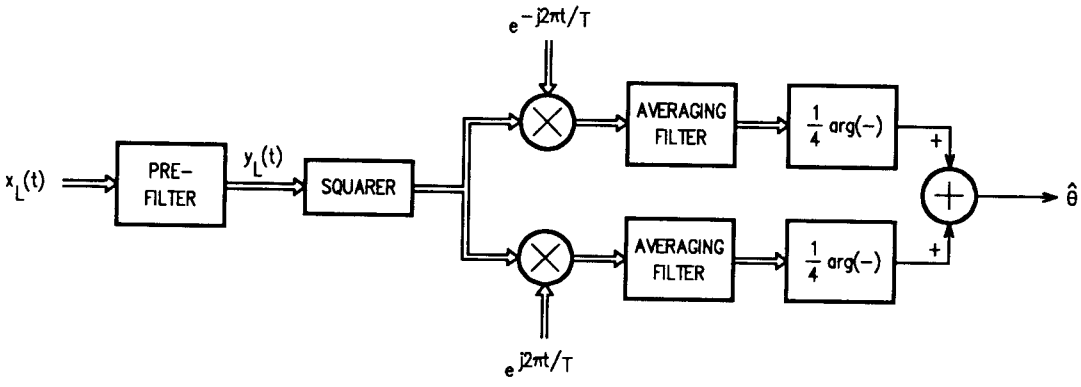


Fig. I.2.9 Timing independent feedforward carrier synchronization for MSK

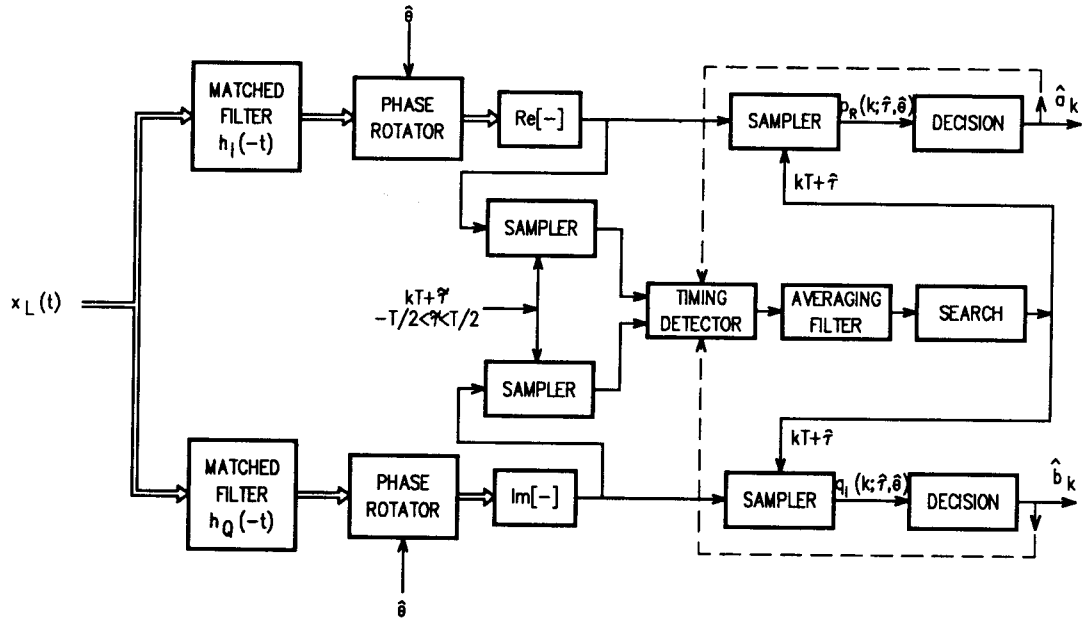


Fig. I.2–10 : ML feedforward symbol synchronization (DDMLFF, NDAMLFF)

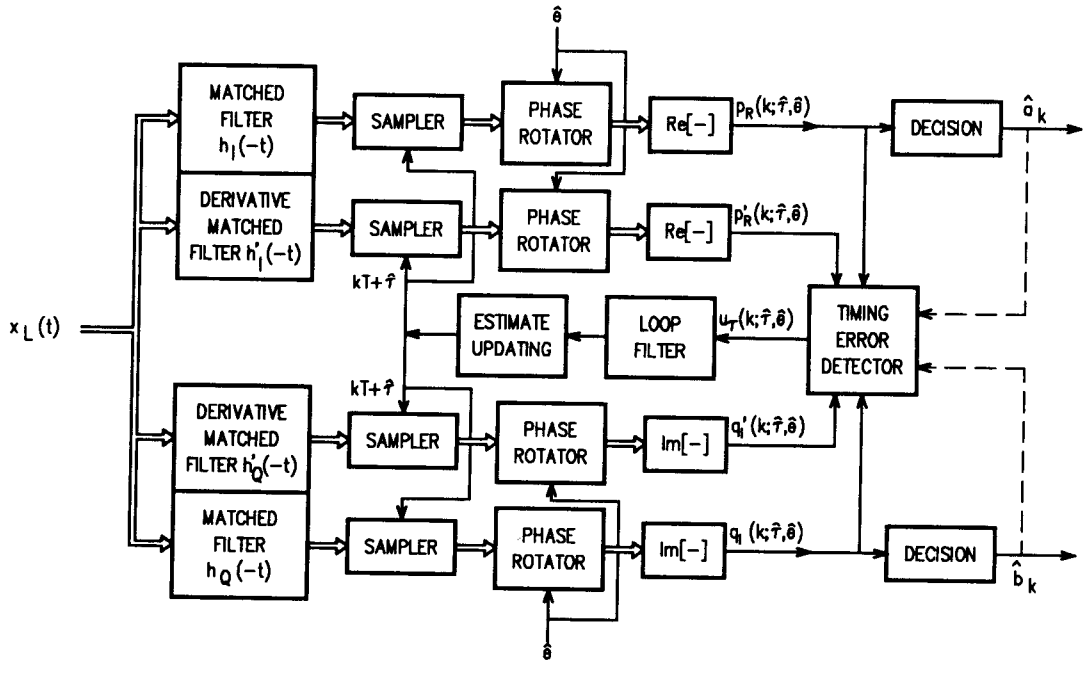


Fig. I.2–11: feedback symbol synchronization with derivative matched filter (DDMLFB, NDAMLFB, DDMLFB$\overline{\text{SNFB}}$, DDMMSEFB)

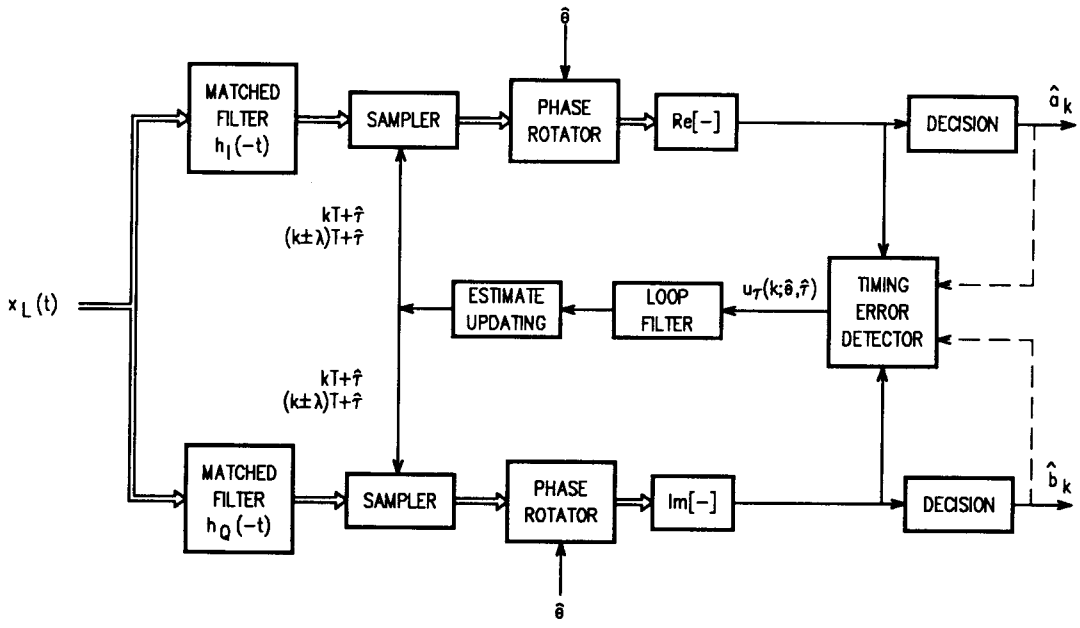


Fig. I.2–12: feedback symbol synchronization with approximate derivative matched filter (DDEL, NDAEL, DTTL, M&M, mM&M, GA)

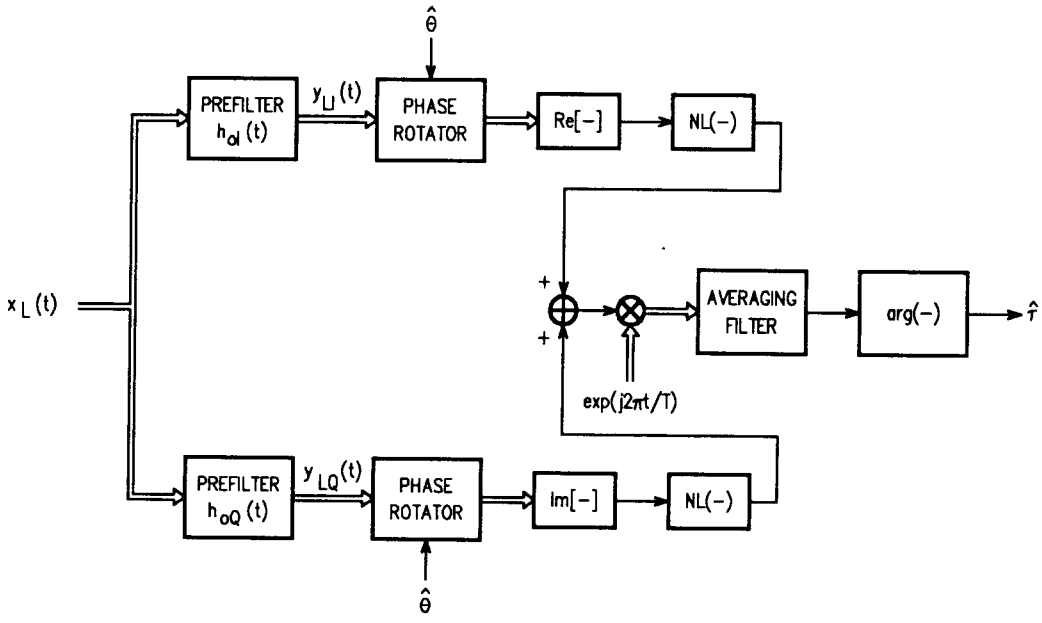


Fig. I.2–13: Non–search feedforward symbol synchronization (NDANLFF)

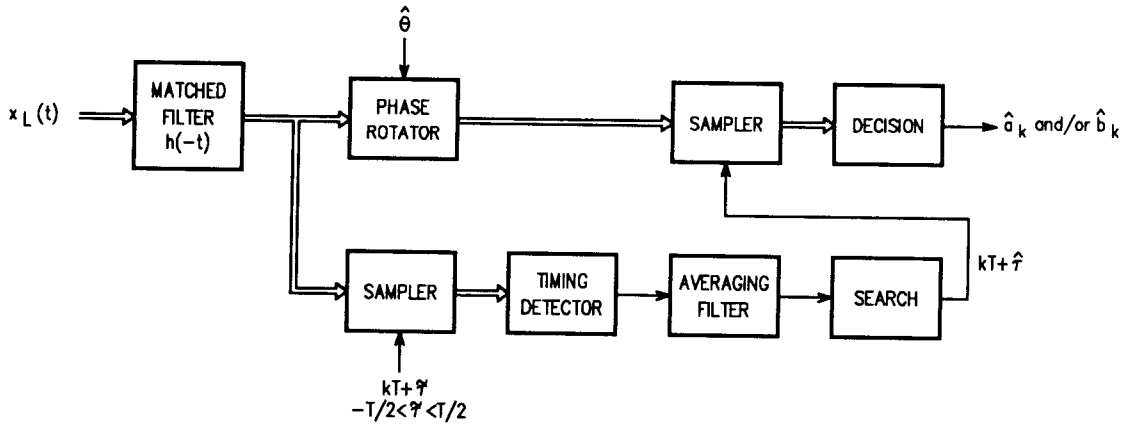


Fig. I.2–14: Carrier–independent feedforward (search) symbol synchronization (NDAMLFFCB/Q)

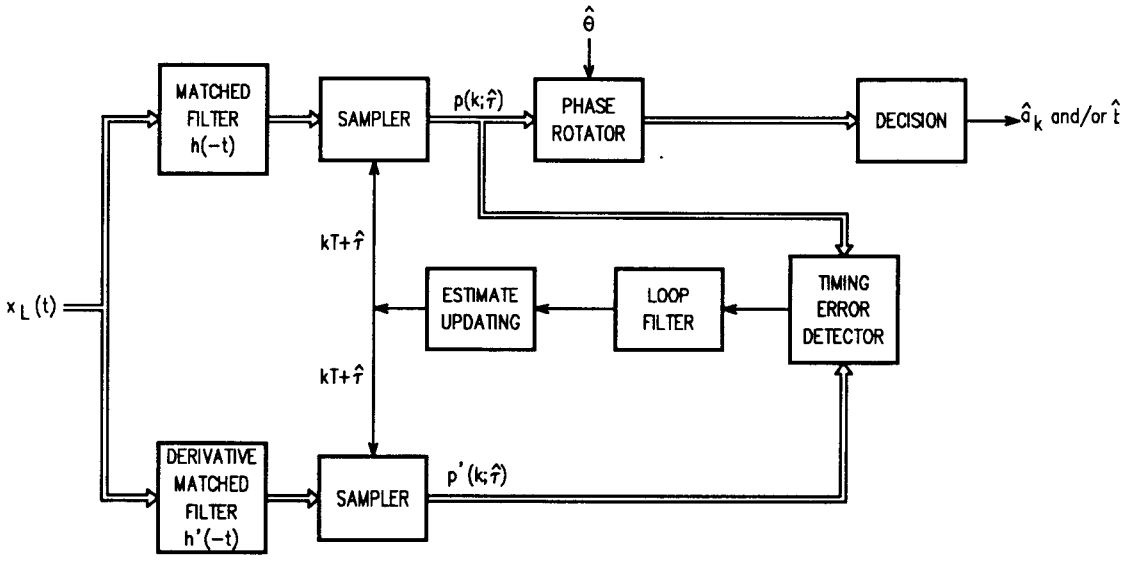


Fig. I.2–15: Carrier–independent feedback symbol synchronization with derivative matched filter (NDAMLFB$\overline{C}$B/Q)

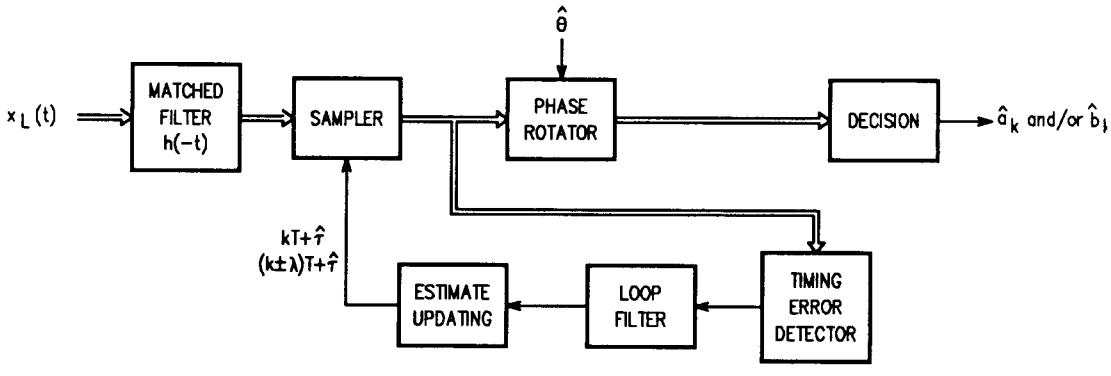


Fig. I.2–16: Carrier–independent feedback symbol synchronization without derivative matched filter (NDAEL(=1/2) $\overline{C}$B/Q, GA$\overline{C}$B/Q, NDANLFB$\overline{C}$B/Q)

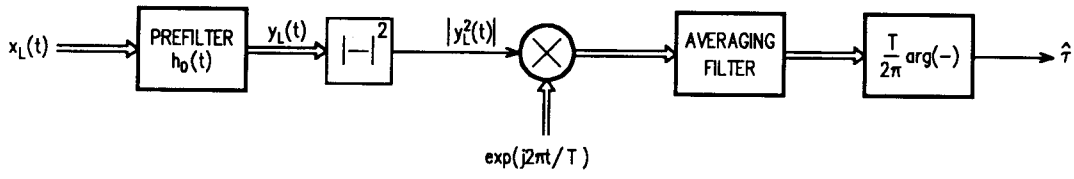


Fig. I.2–17: carrier– independent non–search feedforward symbol synchronization (NDASQFFC̄B/Q)

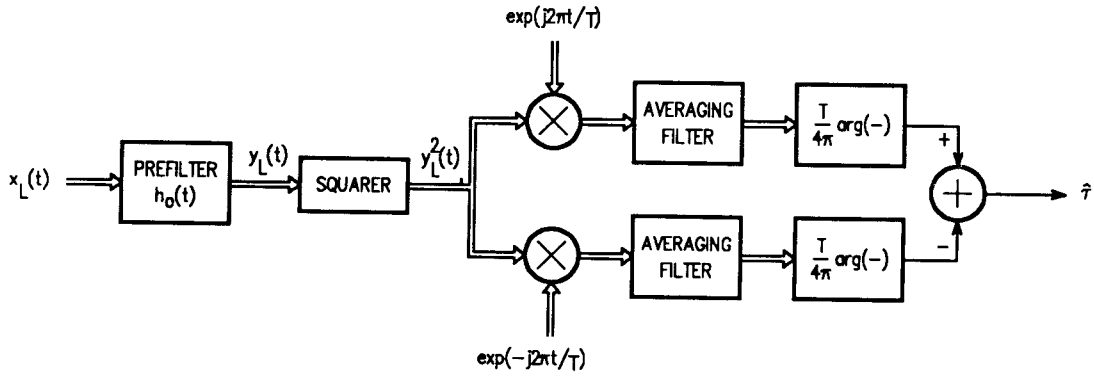


Fig. I.2–18 : Carrier independent non–search feedforward symbol synchronization (NDASQFFC̄O)

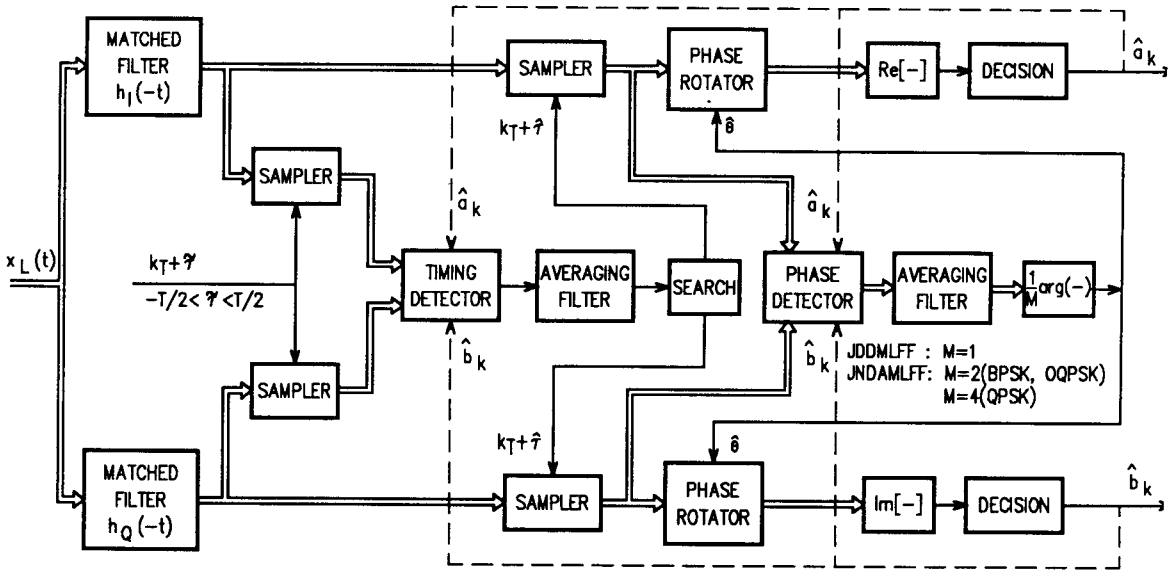


Fig. I.2–19: joint carrier and symbol synchronization (JDDMLFF, JNDAMLFF)

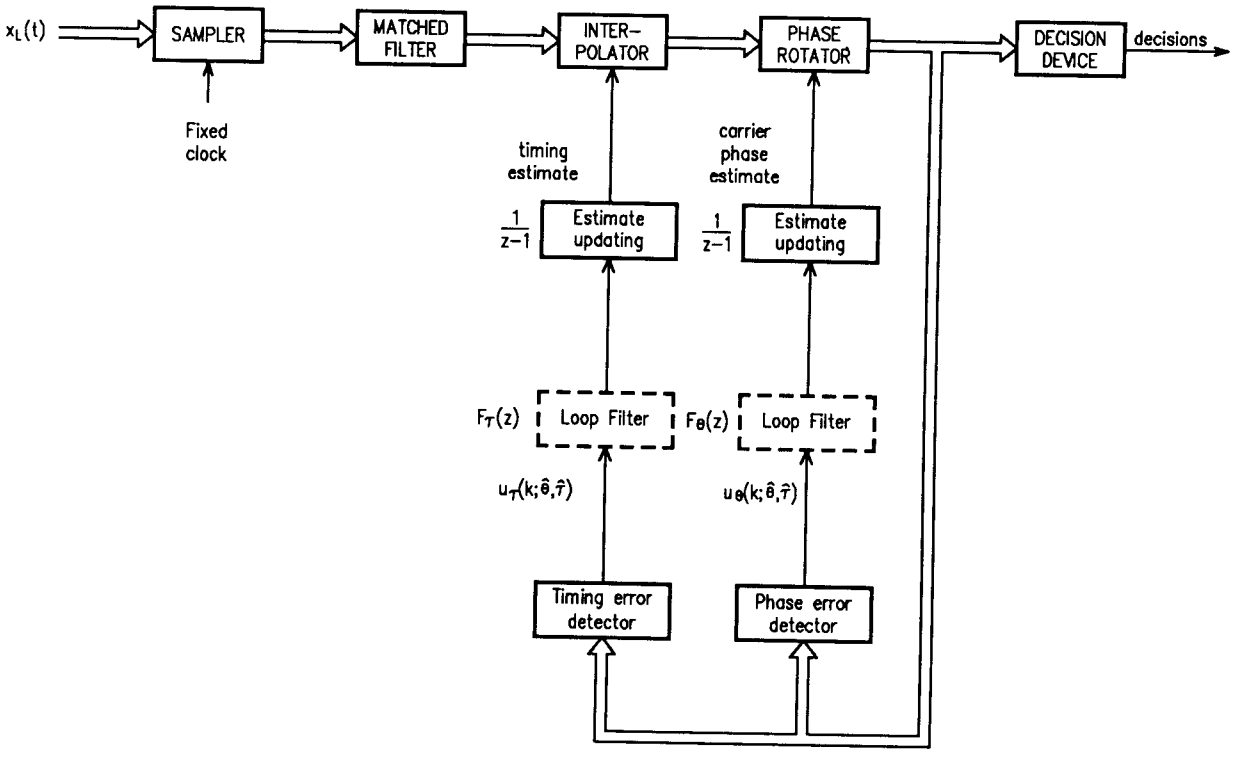


Fig. I.3–1 : Example of digital receiver structure (frequency correction not shown)

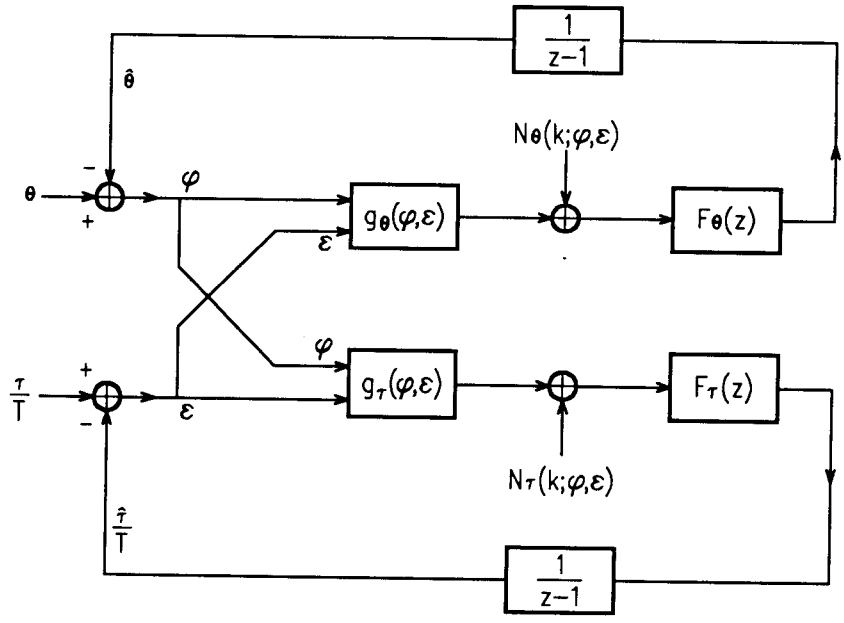


Fig. I.3–2: Equivalent feedback synchronizer model

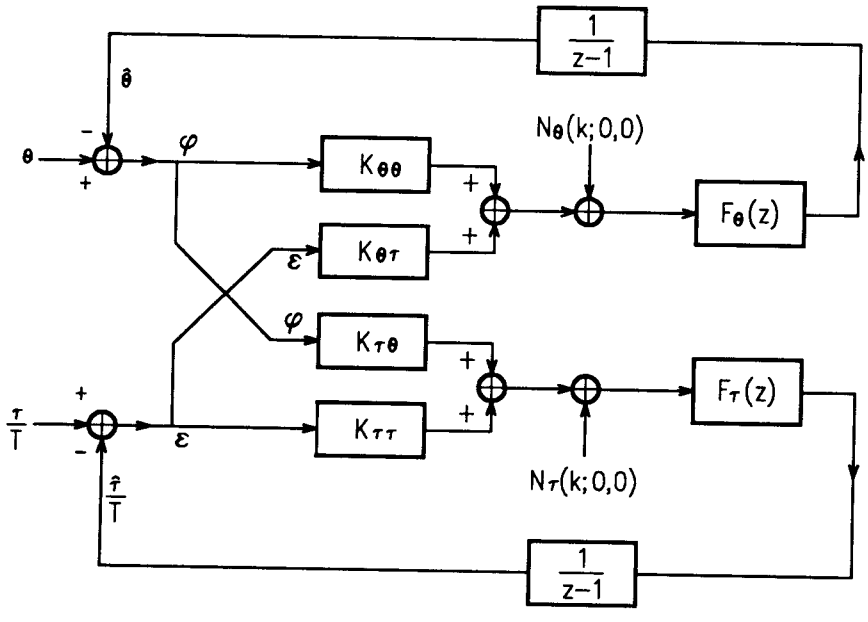


Fig.1.3–4: Linearized equivalent model

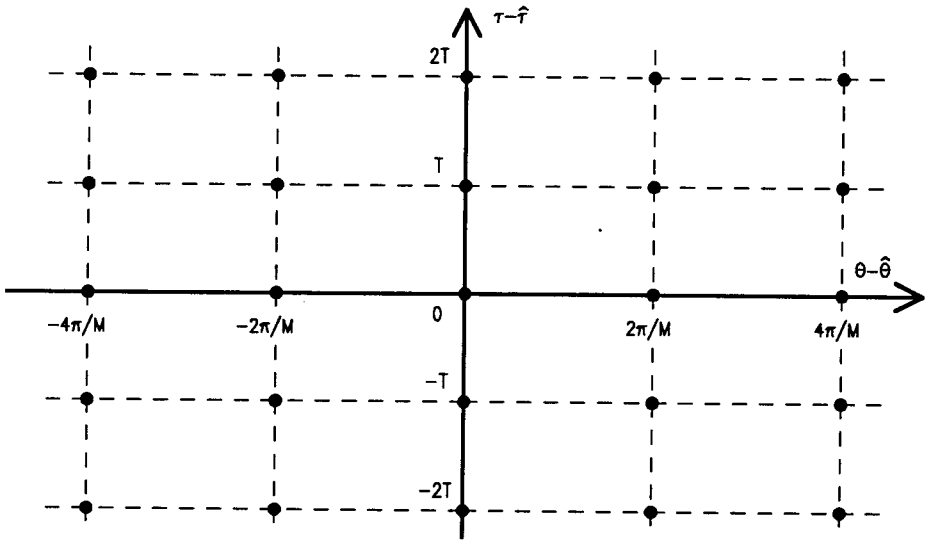


Fig. I.3–3a : equilibrium points for BPSK (M=2) and QPSK (M=4)

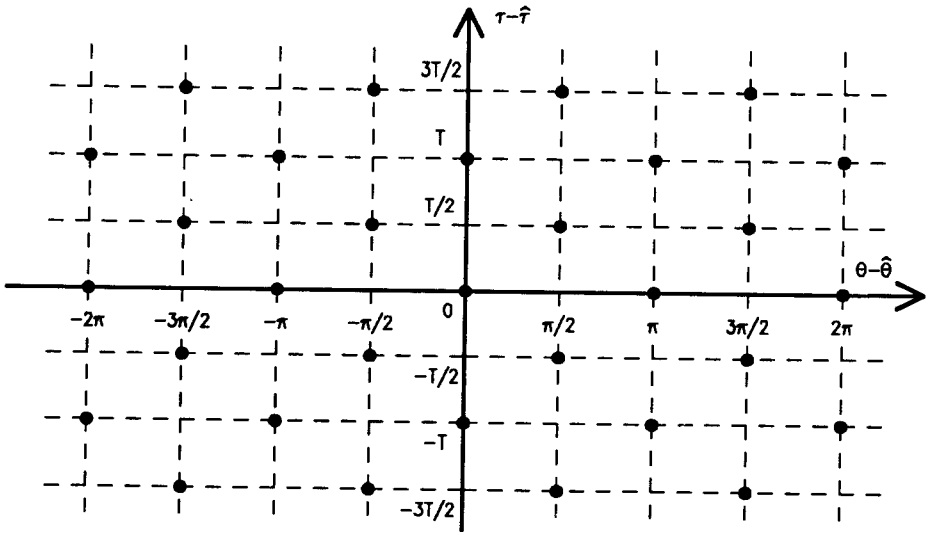


Fig. I.3–3b : equilibrium points for OQPSK

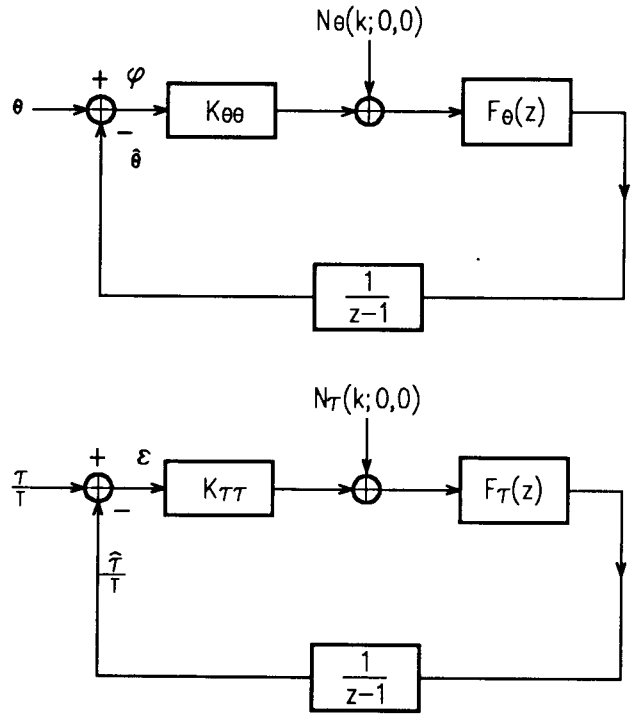


Fig. I.3–5 : Linearized equivalent model for $K_{\theta T}=K_{T\theta}=0$

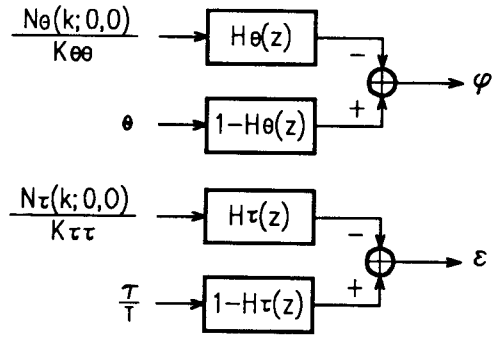


Fig. I.3–6: Simplified linear equivalent model for $K_{\theta T}=K_{T\theta}=0$

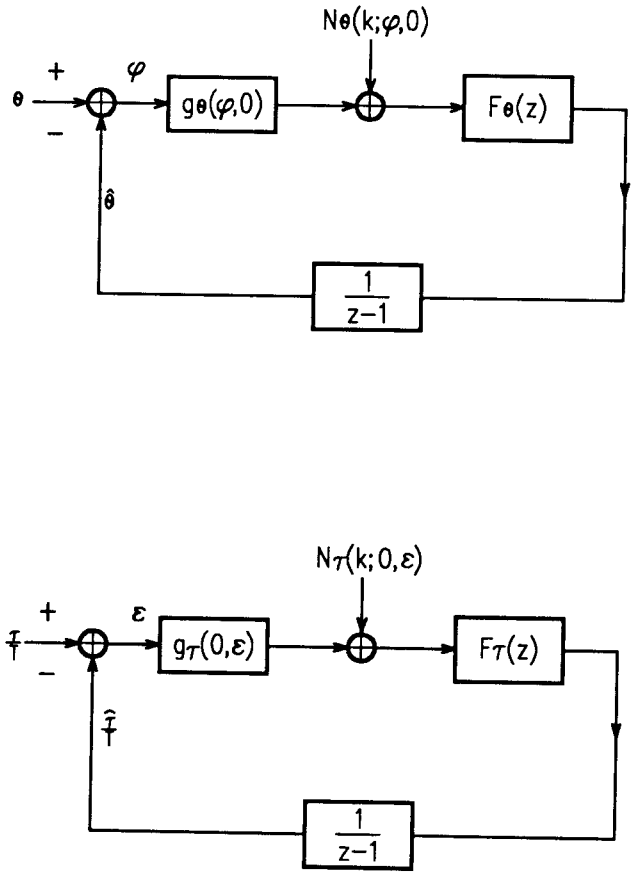


Fig.I.3–7 : Equivalent model for study of cycle slips

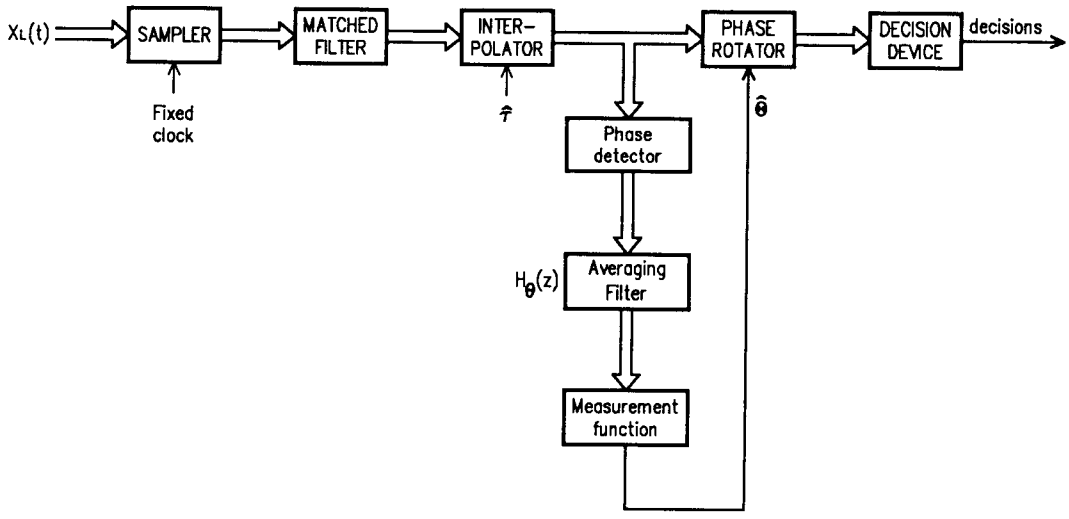


Fig. I.3-8 : Feedforward carrier synchronisation

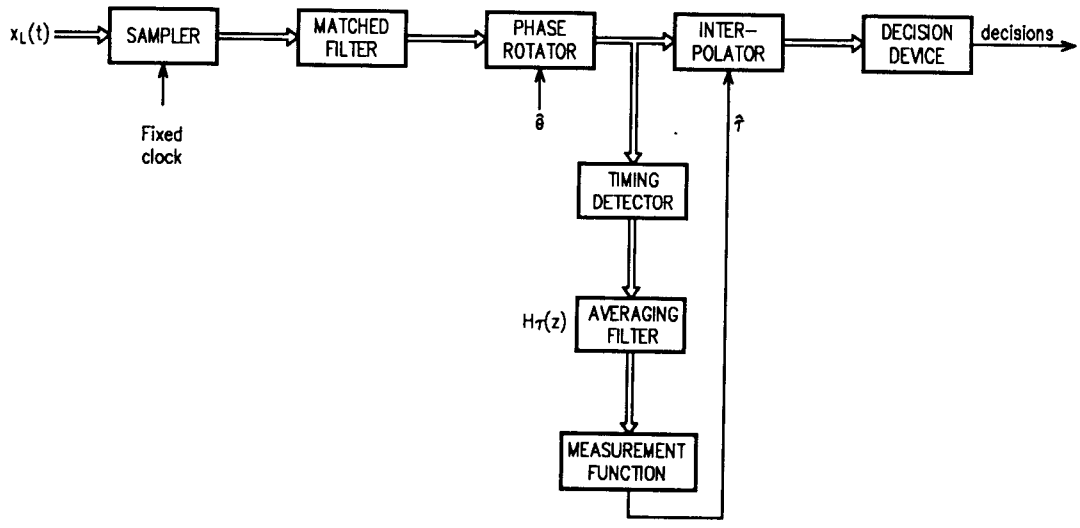


Fig. I.3–9 : Feedforward symbol synchronizer (non–search)

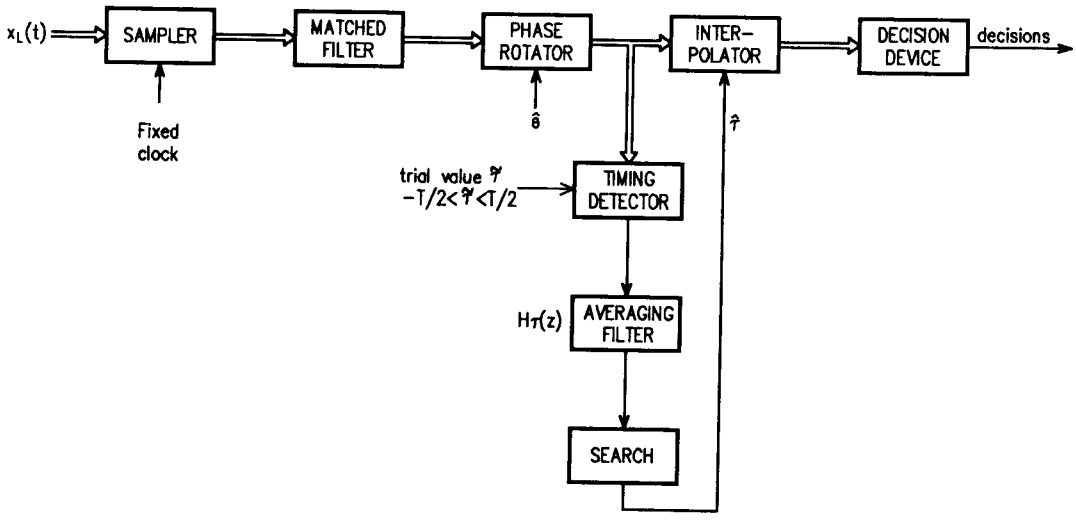


Fig. I.3–10 : Feedforward symbol synchronizer (search)

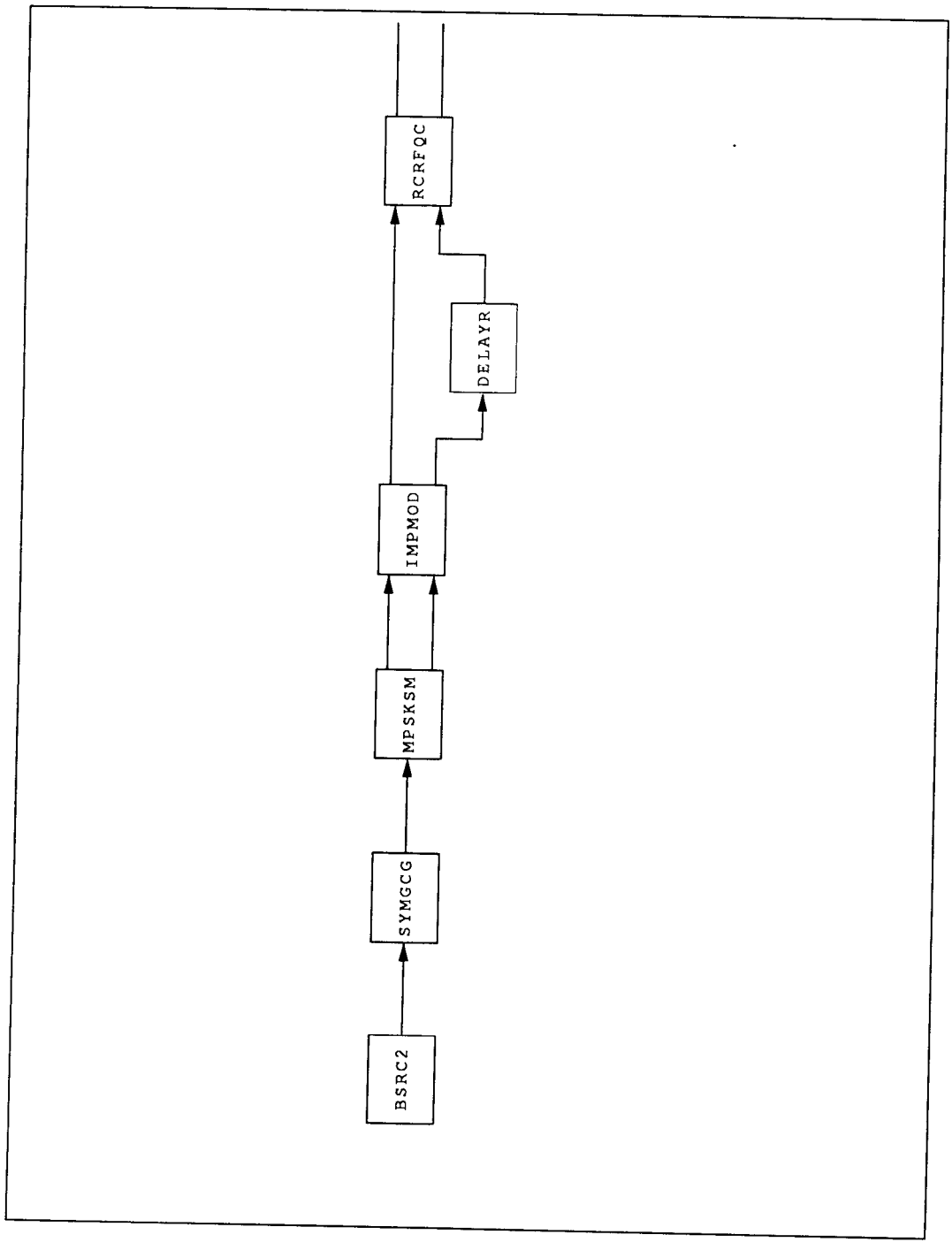


Fig. I.4-1: Transmit model

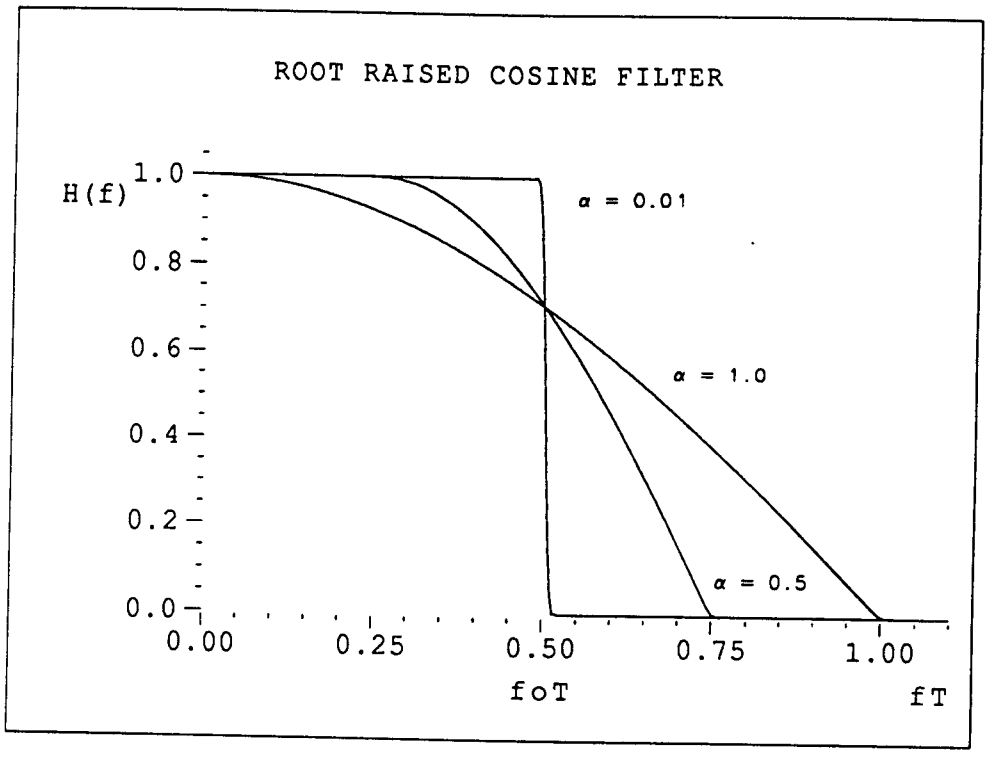


Fig. I.4-2: Transfer function of root raised cosine filter with rolloff (α) as parameter

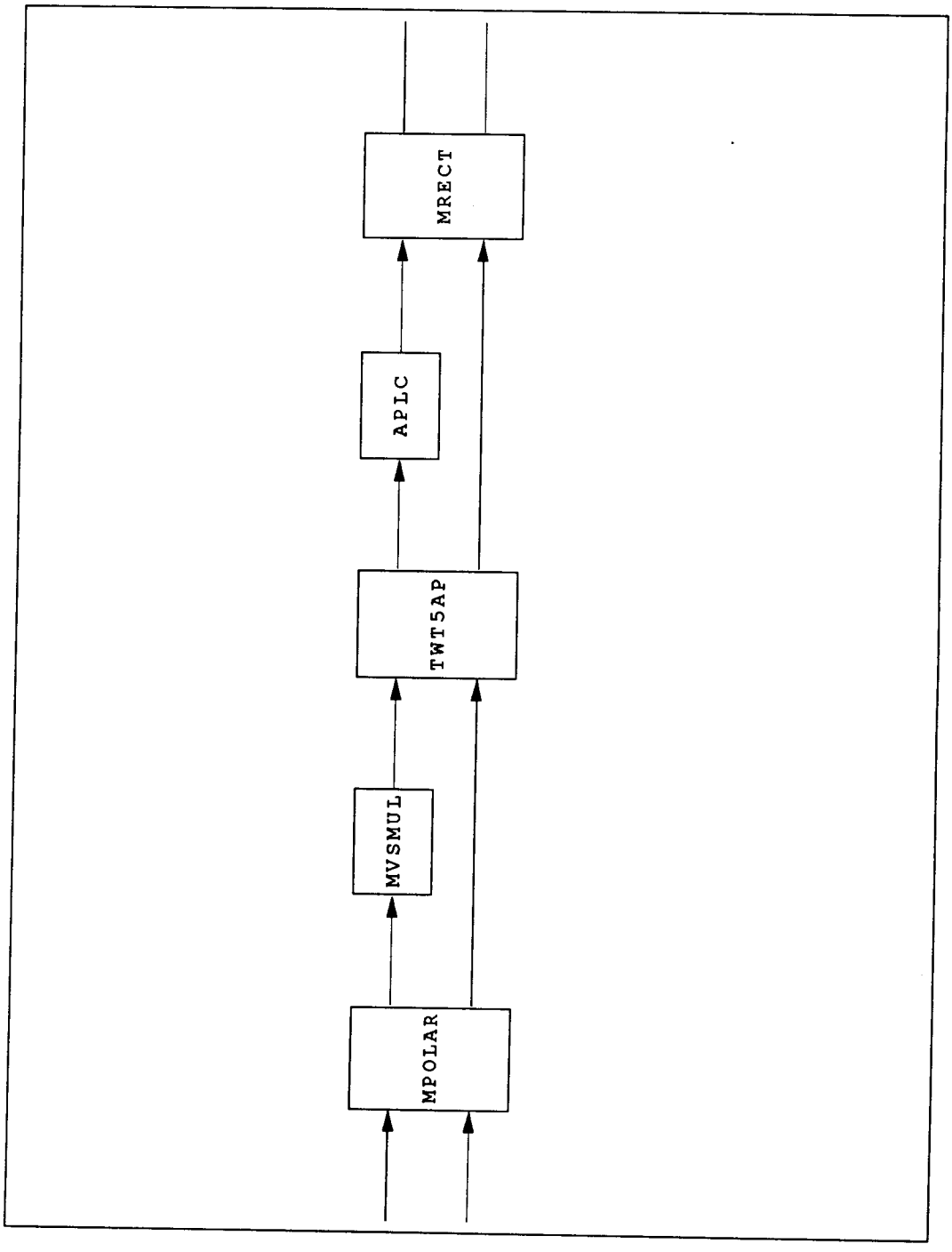
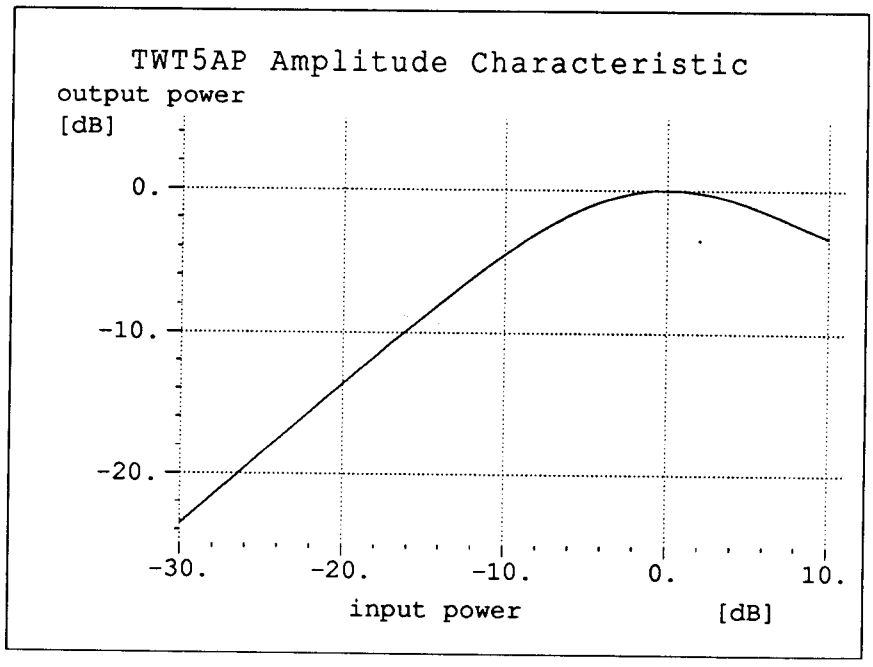


Fig. I.4-3: Travelling Wave Tube Amplifier (TWTA) model



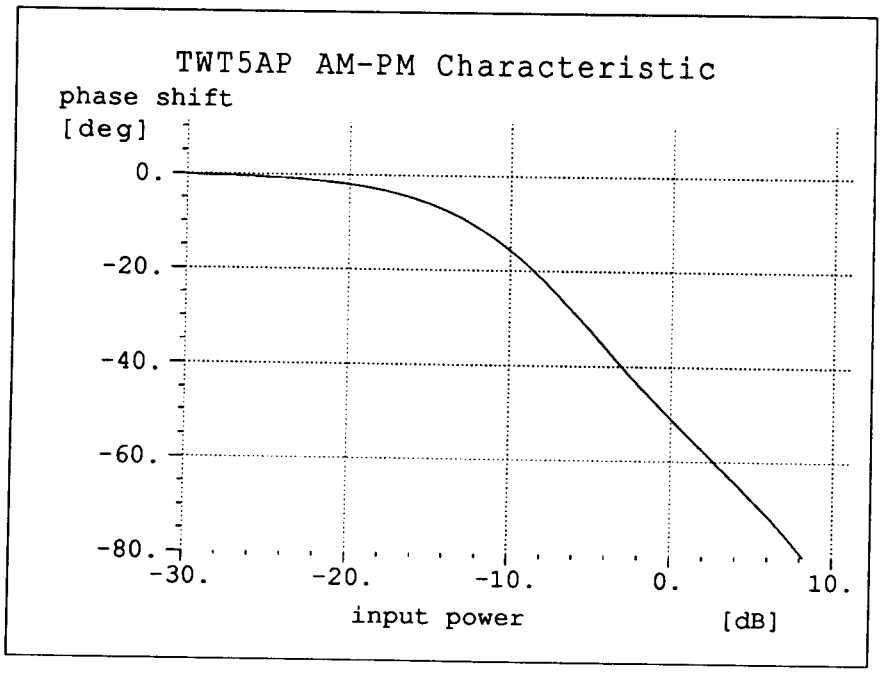


Fig. I.4-4: AM-AM and AM-PM conversion characteristics assumed in this study. Input and output power are normalized to the corresponding saturation levels, i.e. saturation occurs at 0dB input power and 0dB output power

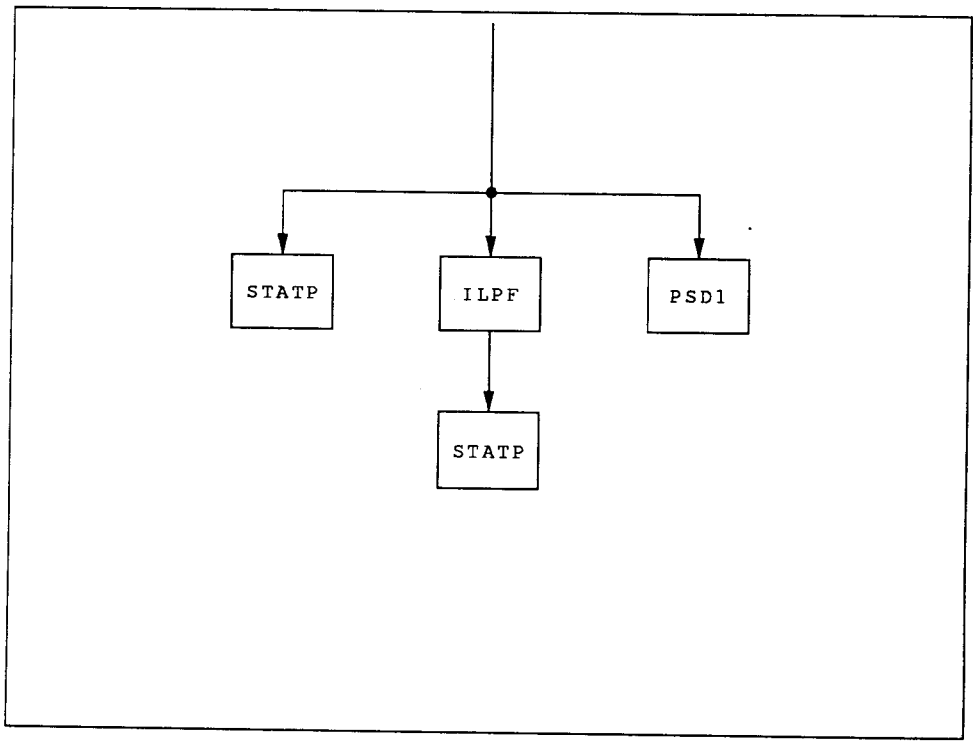


Fig. I.4-5: Statistical evaluation model

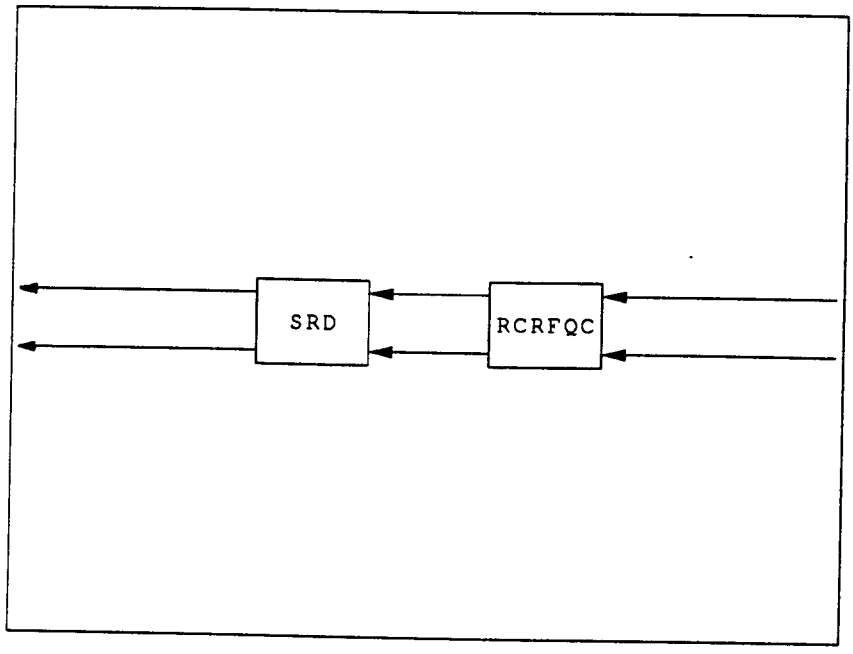


Fig. I.4-6a: Sampled matched filter

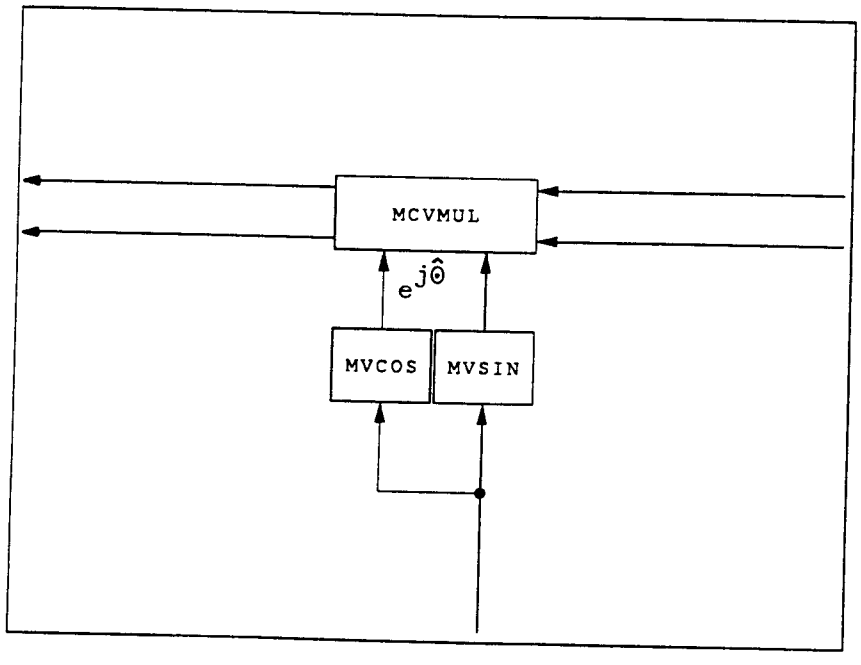


Fig. I.4-6b: Phase rotator

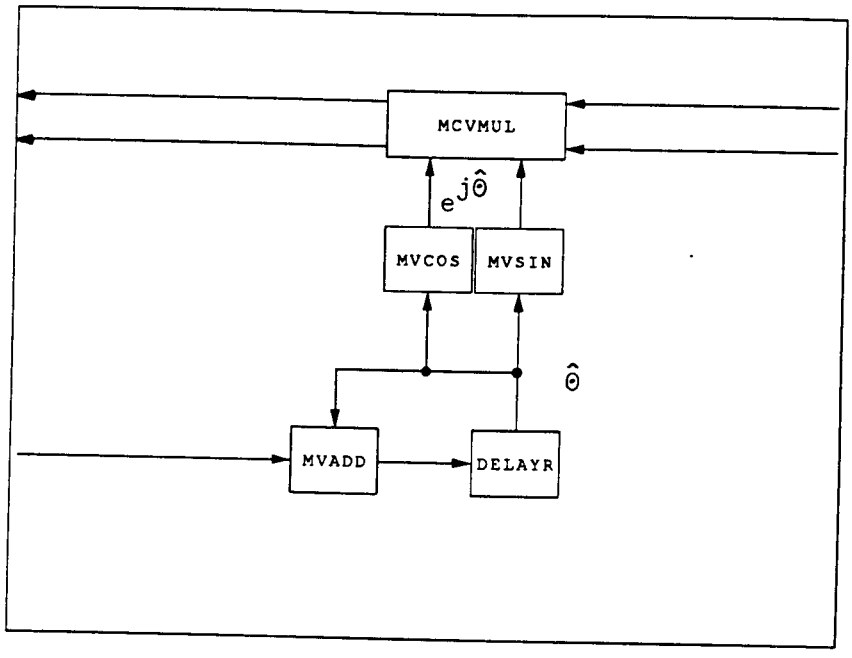


Fig. I.4-6c: Integrator and phase rotator

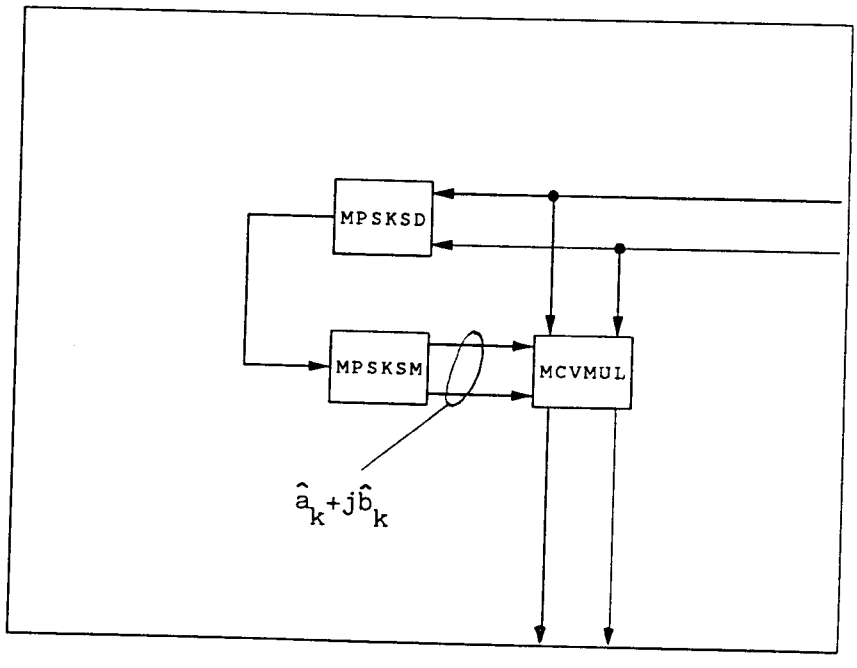


Fig. I.4-6d: Decision Feedback

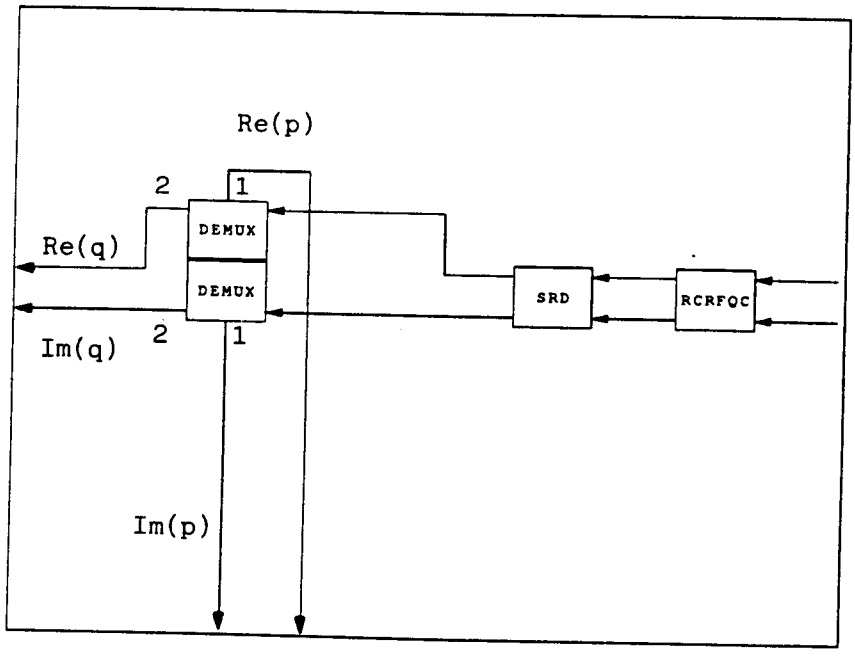


Fig. I.4-6e: Separation of p(k;τ) and q(k;τ)

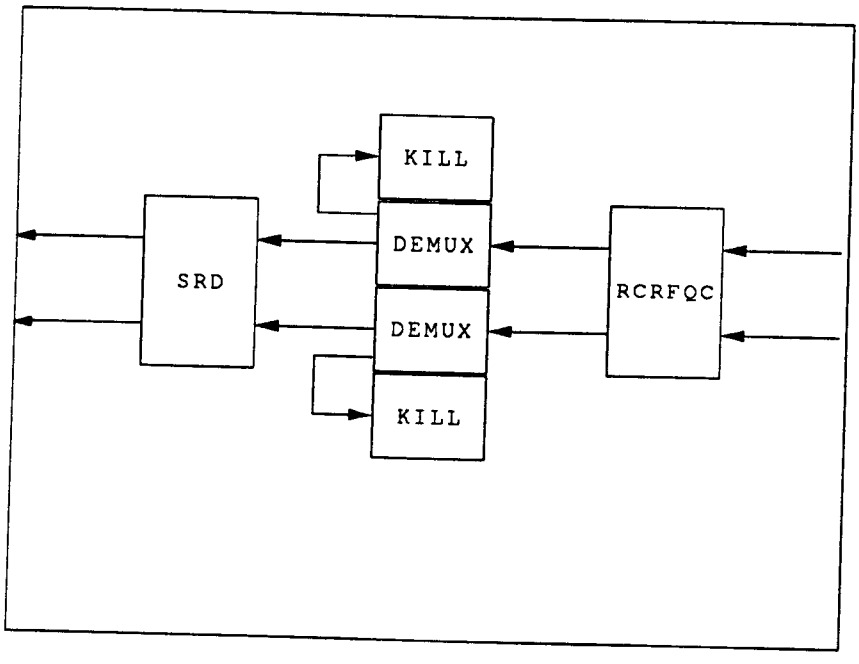


Fig. I.4-6f: Clock frequency deviation

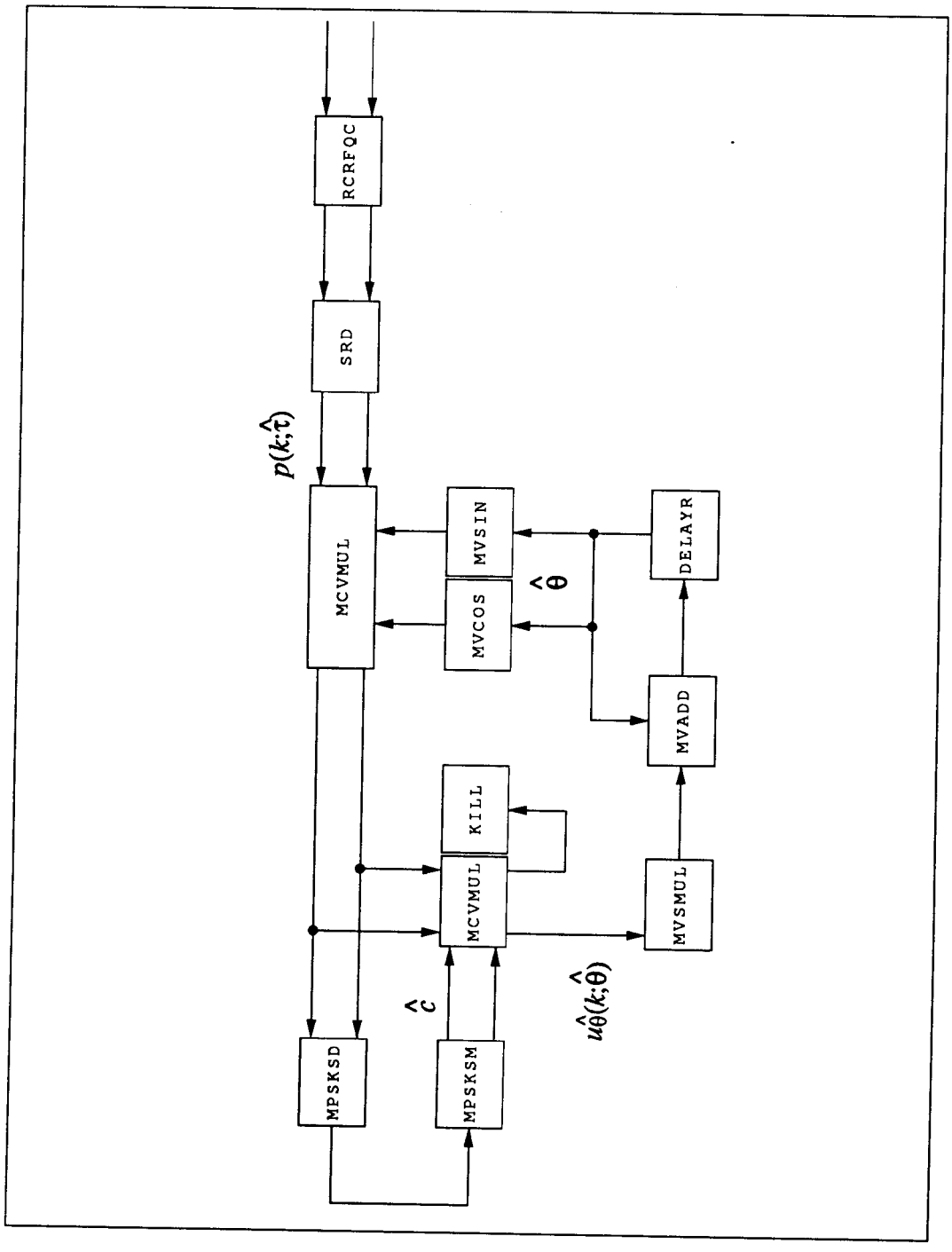


Fig. I.4-7a: DDMLFB, Decision Directed Maximum Likelihood FeedBack carrier synchronizer (BPSK, QPSK)

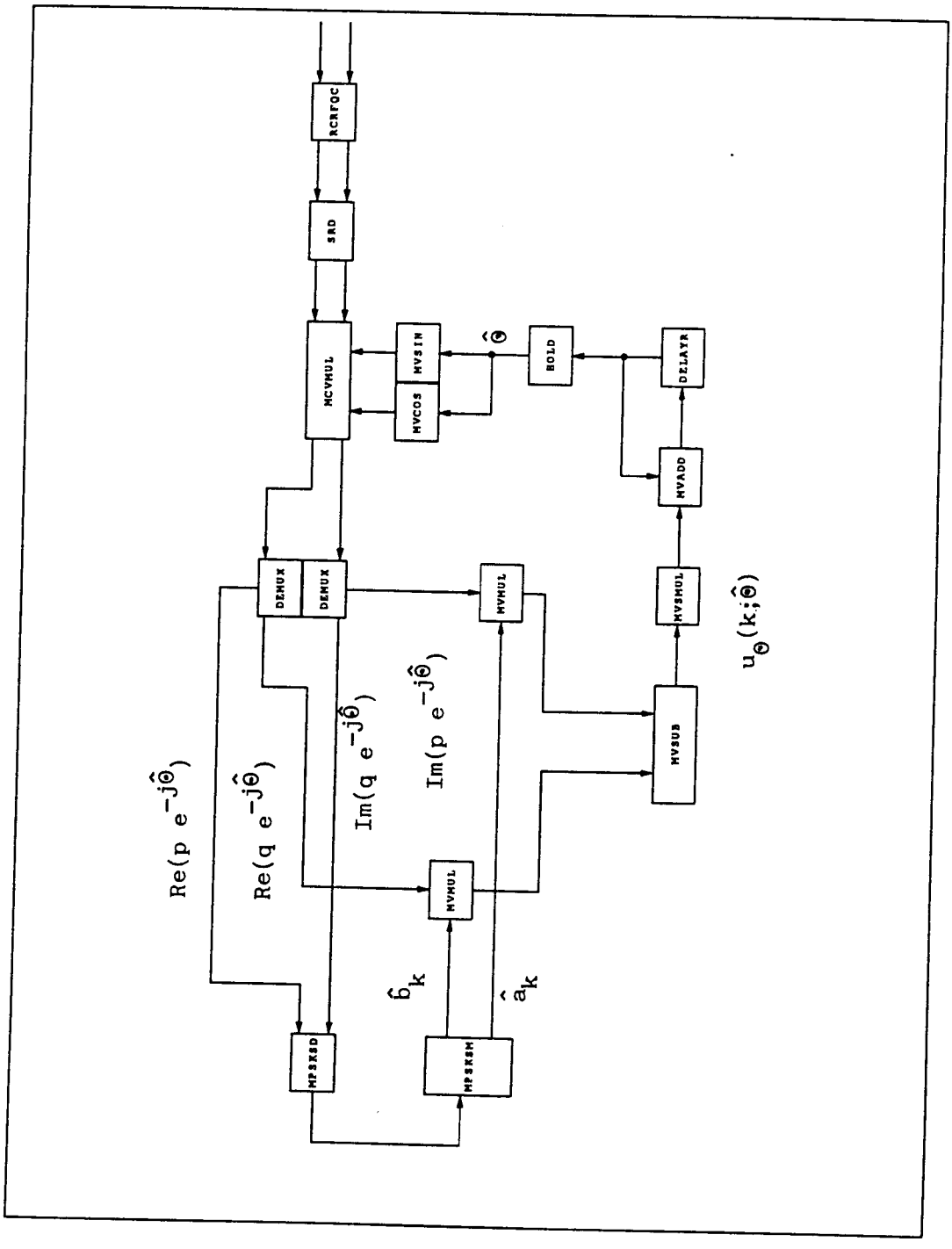


Fig. I.4-7b: DDMLFB, Decision Directed Maximum Likelihood FeedBack carrier synchronizer (OQPSK)

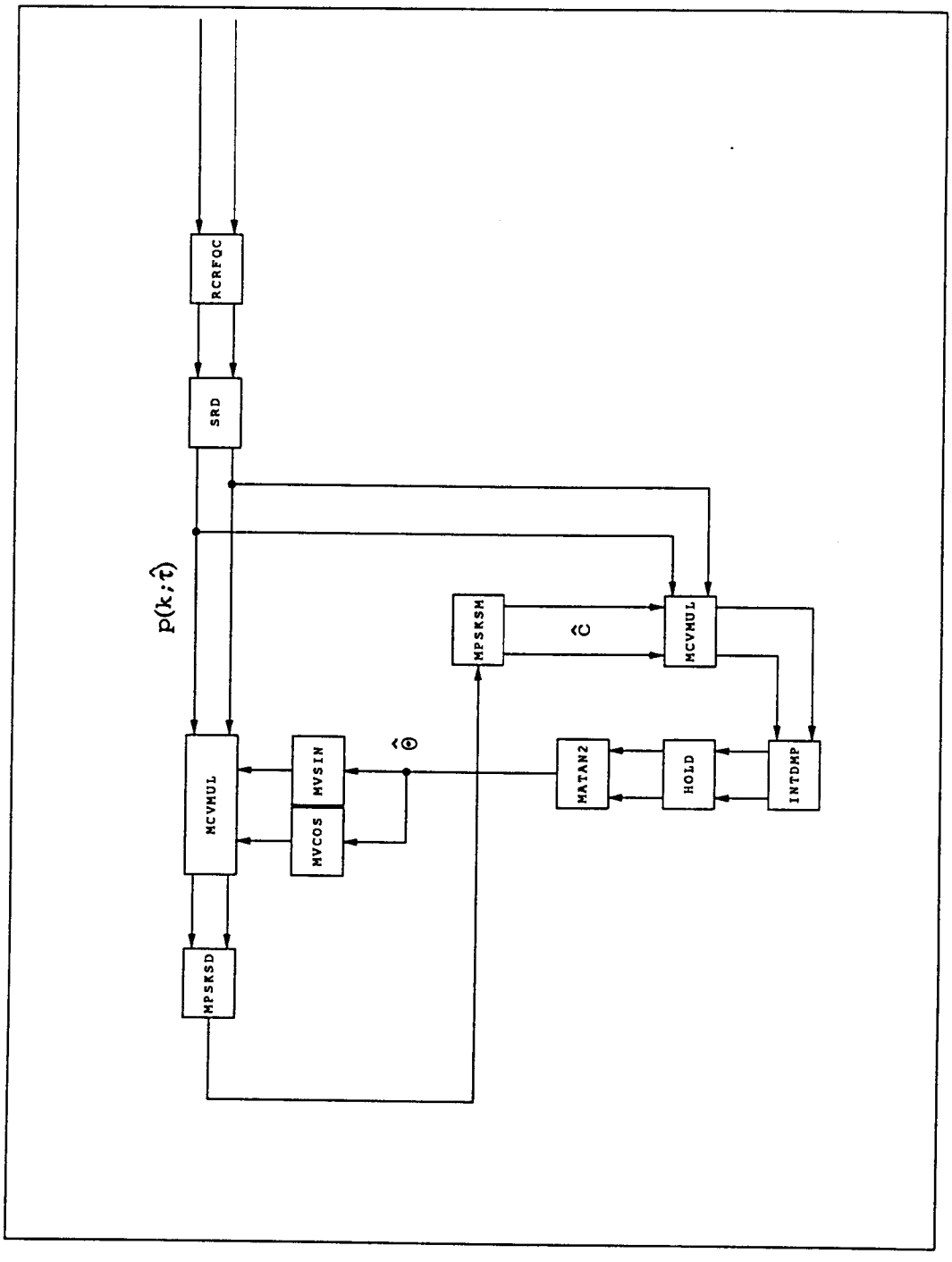


Fig. I.4-8a: DDMLFF, Decision Directed Maximum Likelihood FeedForward carrier synchronizer, θ (BPSK, QPSK)

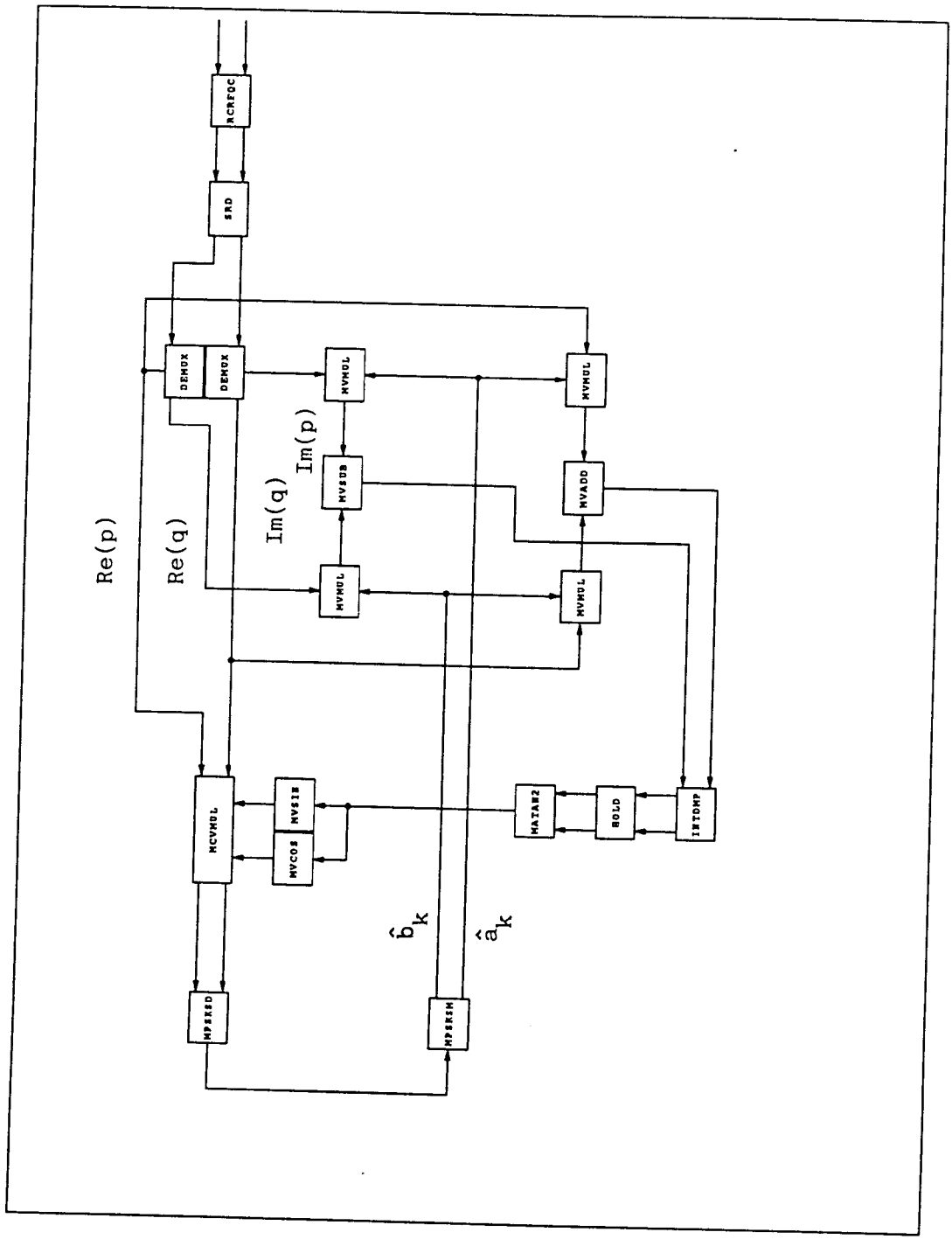


Fig. I.4-8b: DDMLFF, Decision Directed Maximum Likelihood FeedForward carrier synchronizer, θ (OQPSK)

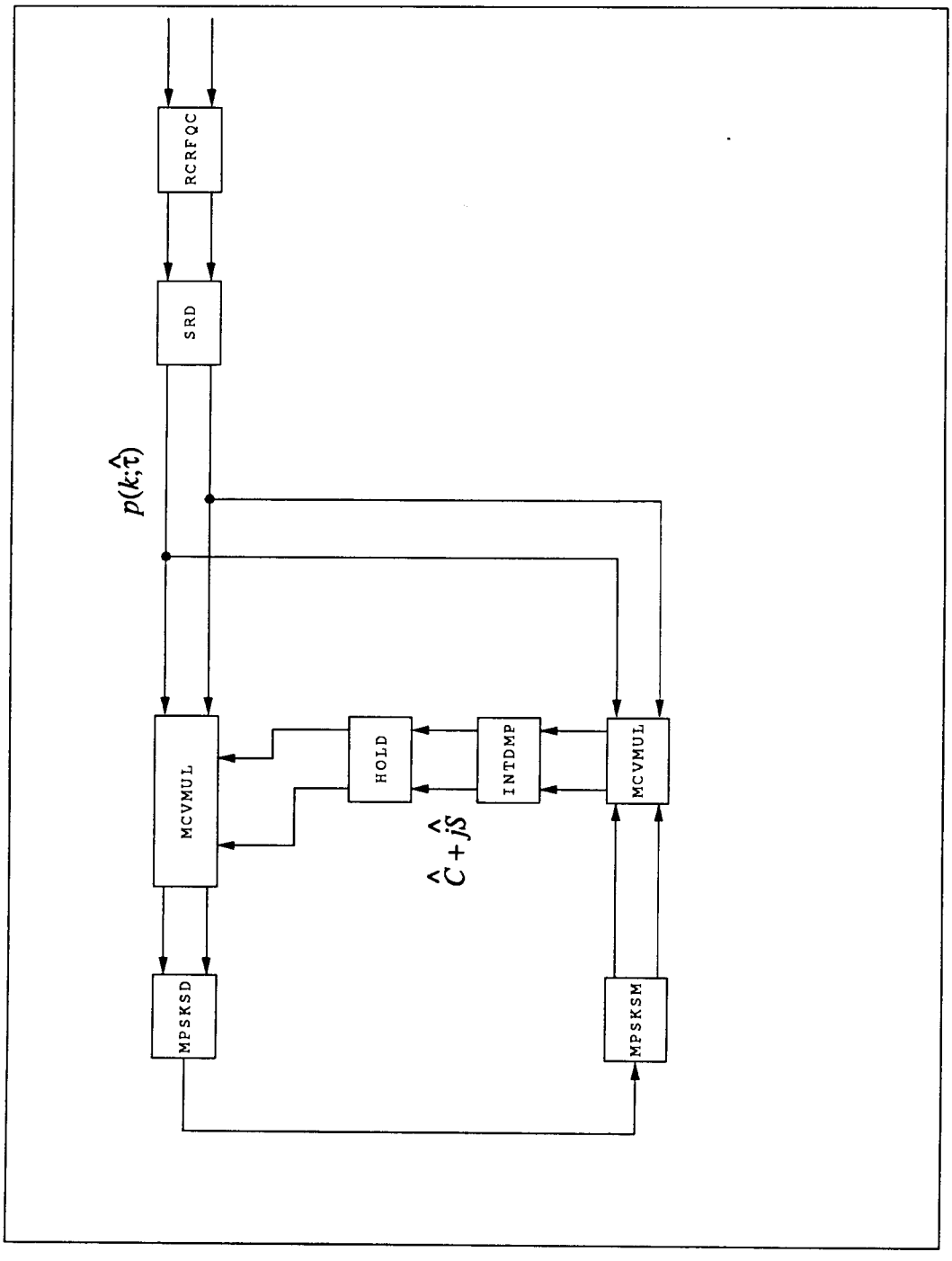


Fig. I.4-9a: DDMLFF, Decision Directed Maximum Likelihood FeedForward carrier synchronizer, C+jS (BPSK, QPSK)

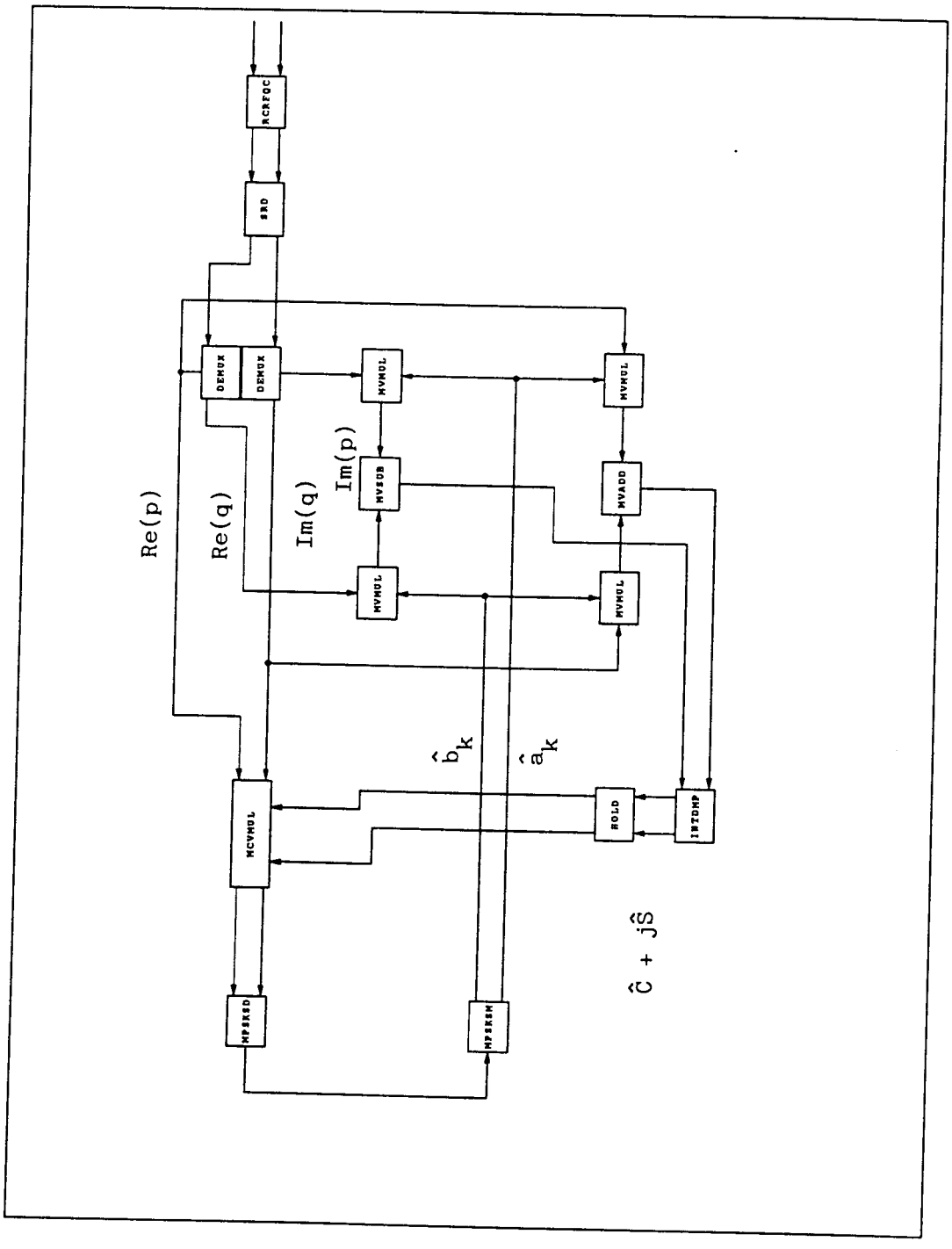


Fig. I.4-9b: DDMLFF, Decision Directed Maximum Likelihood FeedForward carrier synchronizer, C+jS (OQPSK)

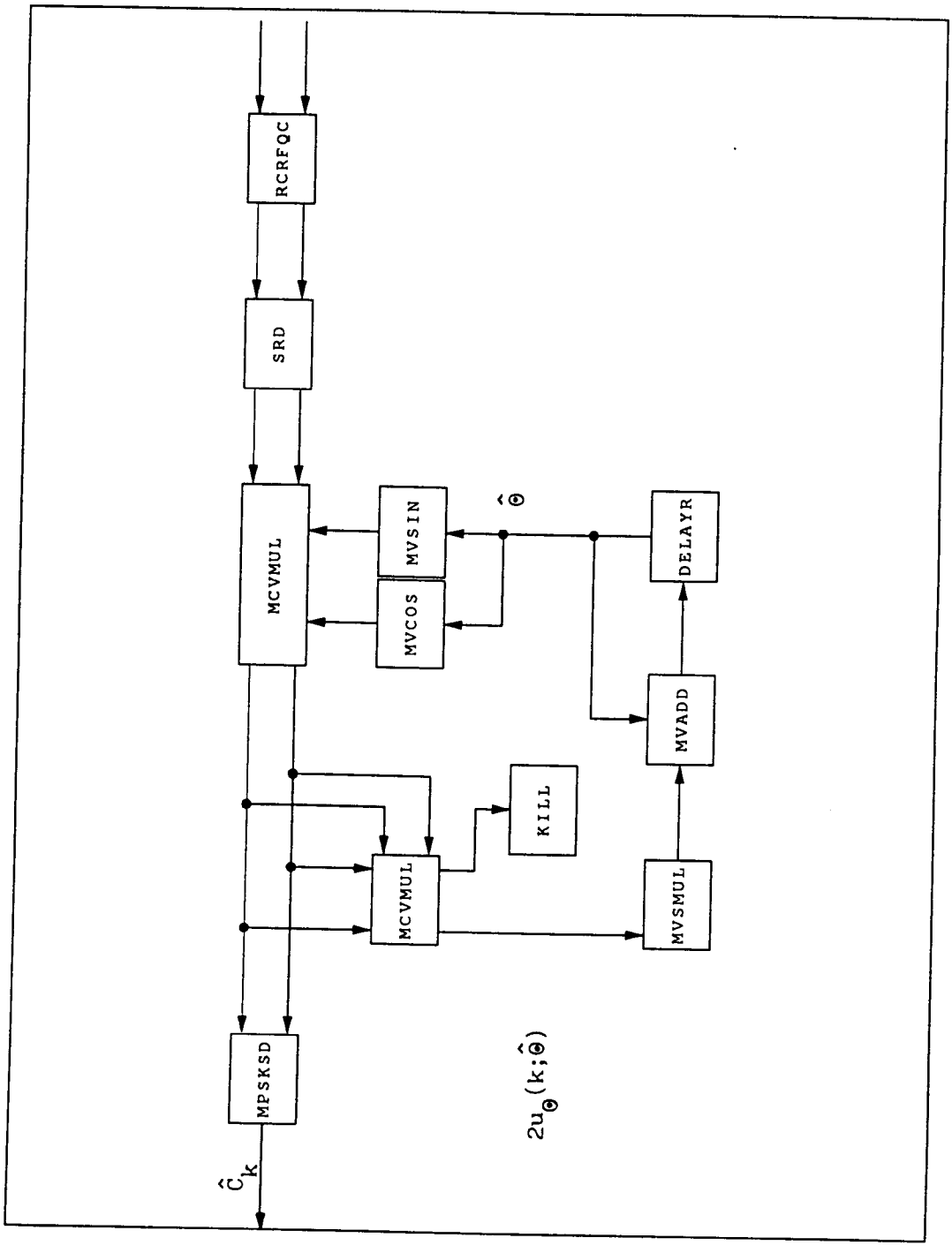


Fig. I.4-10a: NDAMLFB, Non Data-Aided Maximum Likelihood FeedBack carrier synchronizer (BPSK)

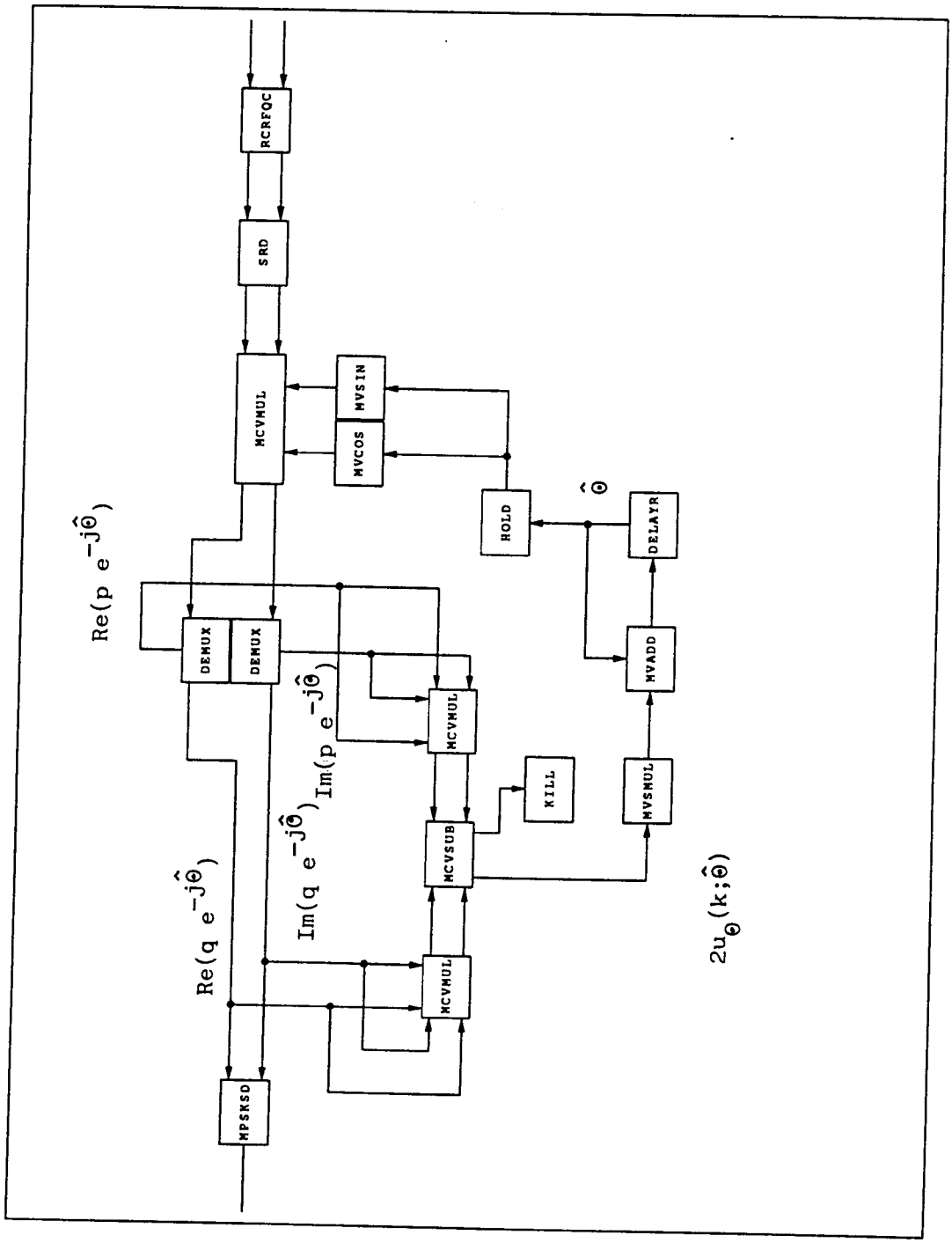


Fig. I.4-10b: NDAMLFB, Non Data-Aided Maximum Likelihood FeedBack carrier synchronizer (OQPSK)

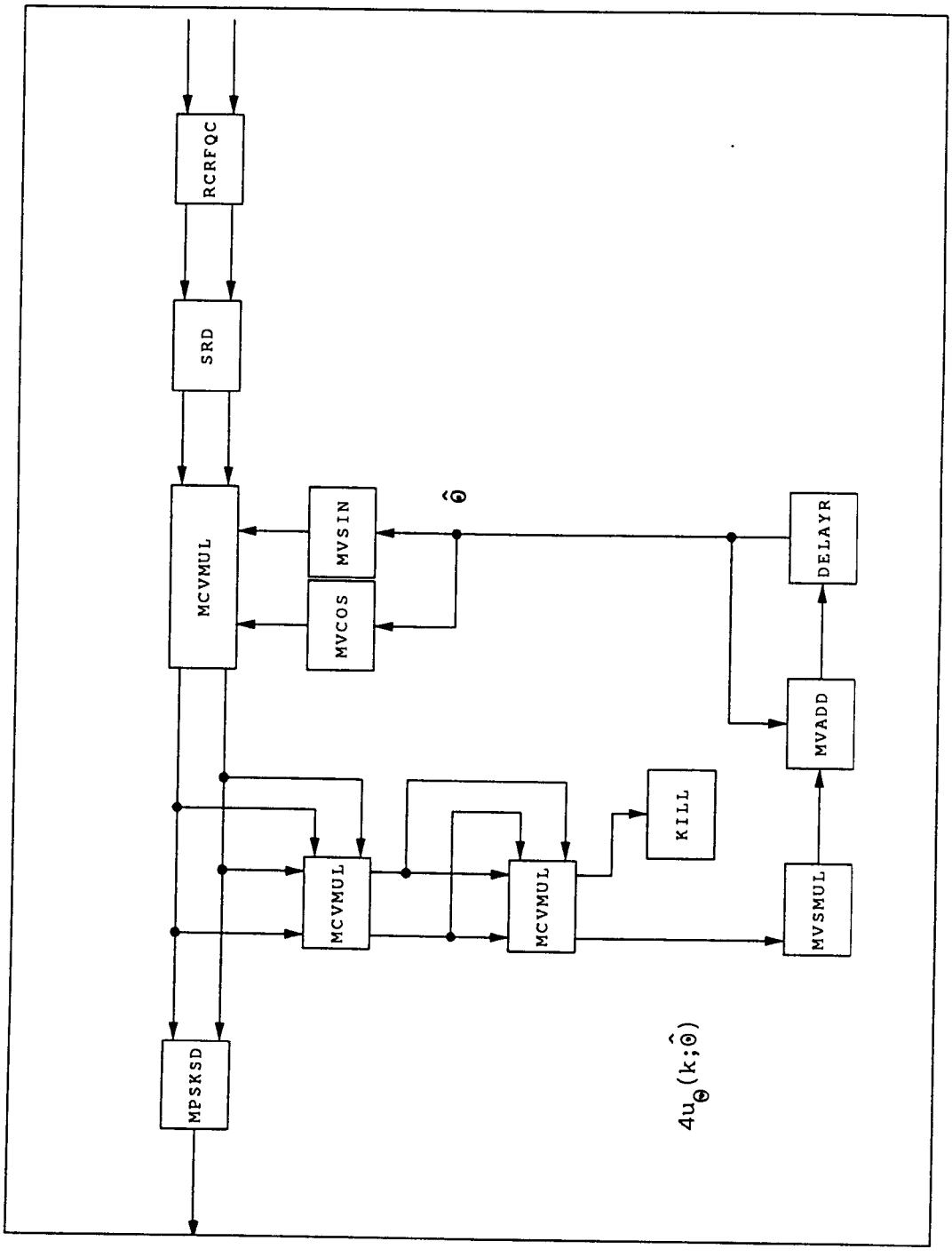


Fig. I.4-10c: NDAMLFB, Non Data-Aided Maximum Likelihood FeedBack carrier synchronizer (QPSK)

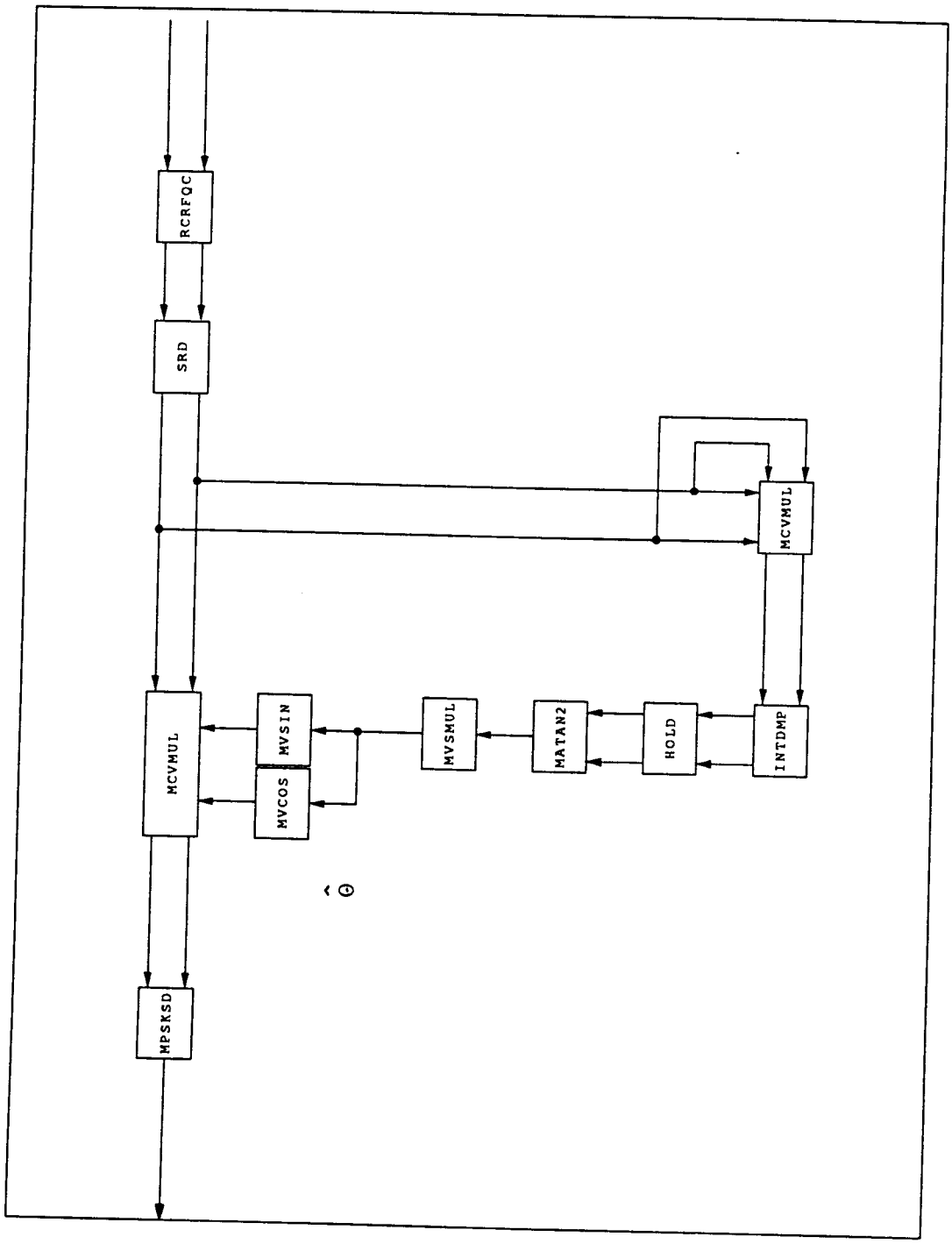


Fig. I.4-11a: NDAMLFF, Non Data-Aided Maximum Likelihood Feedforward carrier synchronizer (BPSK)

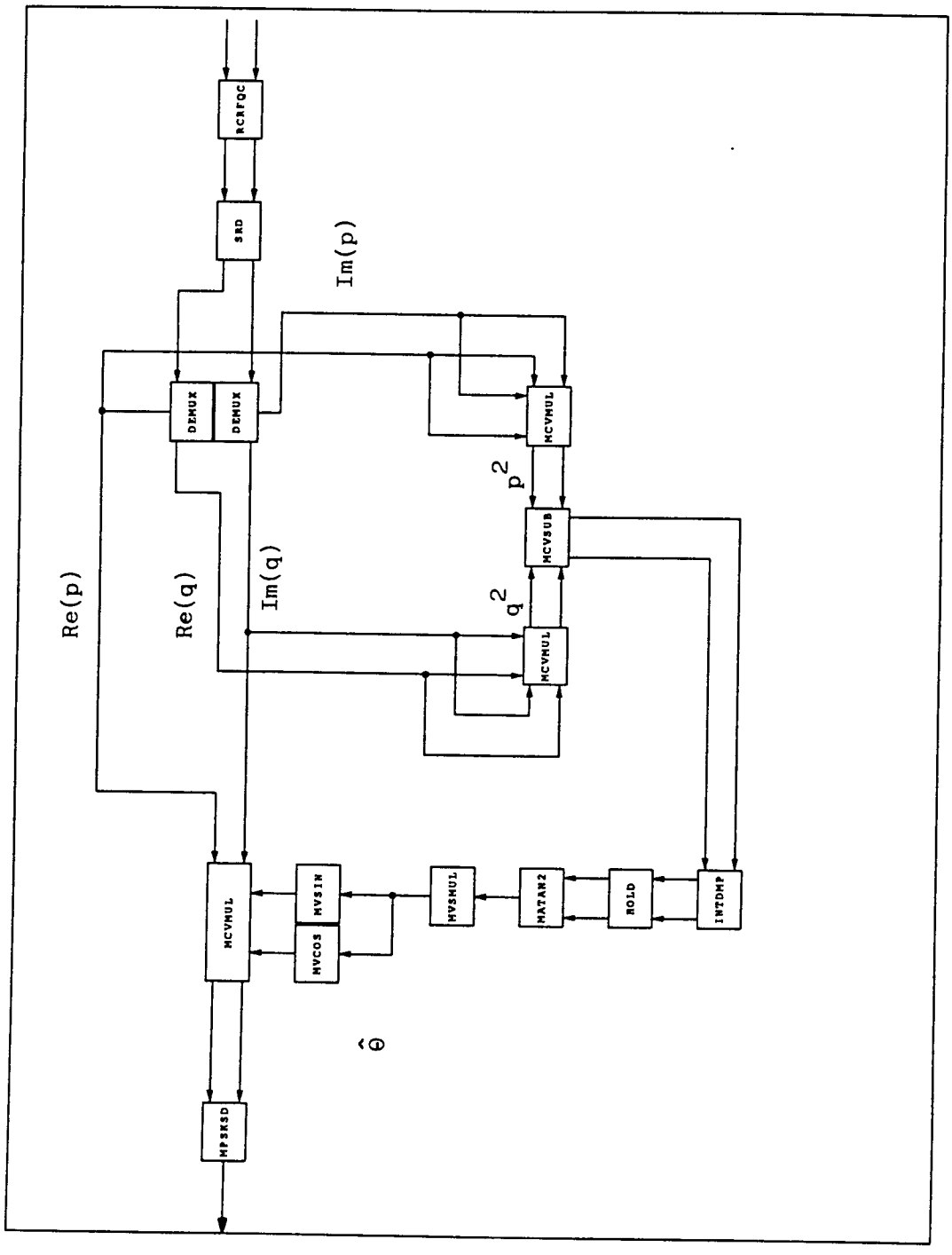


Fig. I.4-11b: NDAMLFF, Non Data-Aided Maximum Likelihood Feedforward carrier synchronizer (OQPSK)

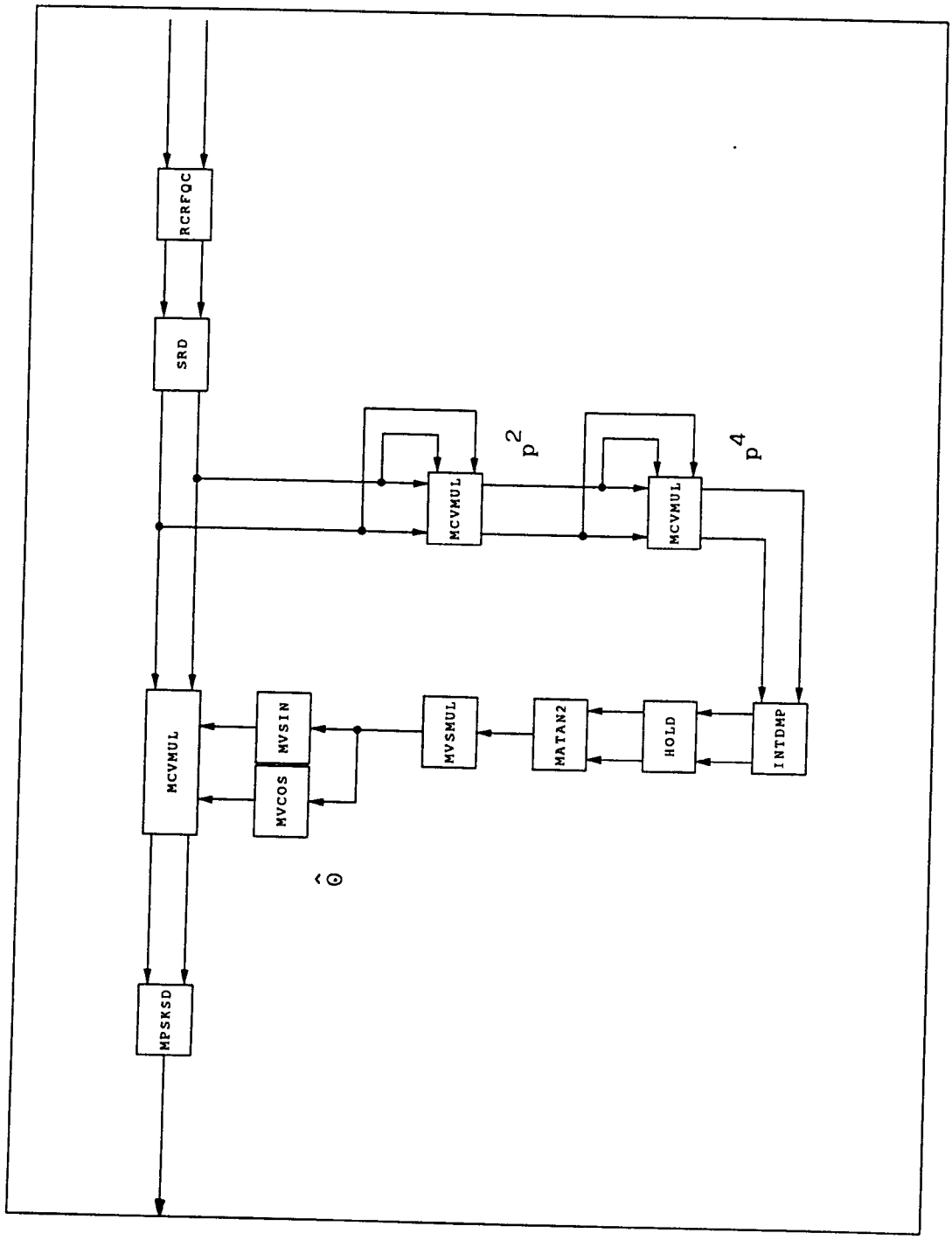


Fig. I.4-11c: NDAMLFF, Non Data-Aided Maximum Likelihood Feedforward carrier synchronizer (QPSK)

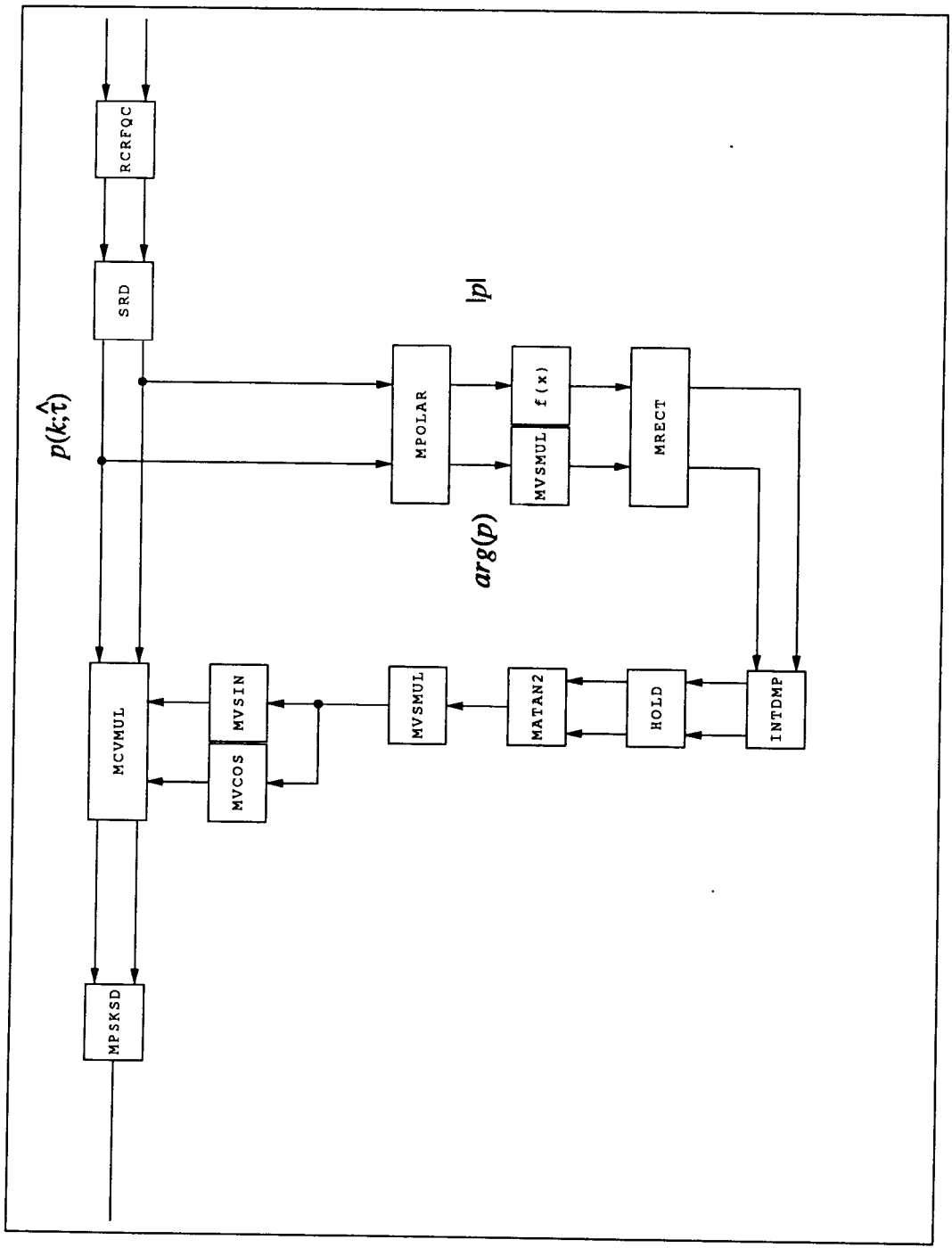


Fig. I.4-12a: V&VFF, Viterbi&Viterbi Feedforward carrier synchronizer for BPSK and QPSK

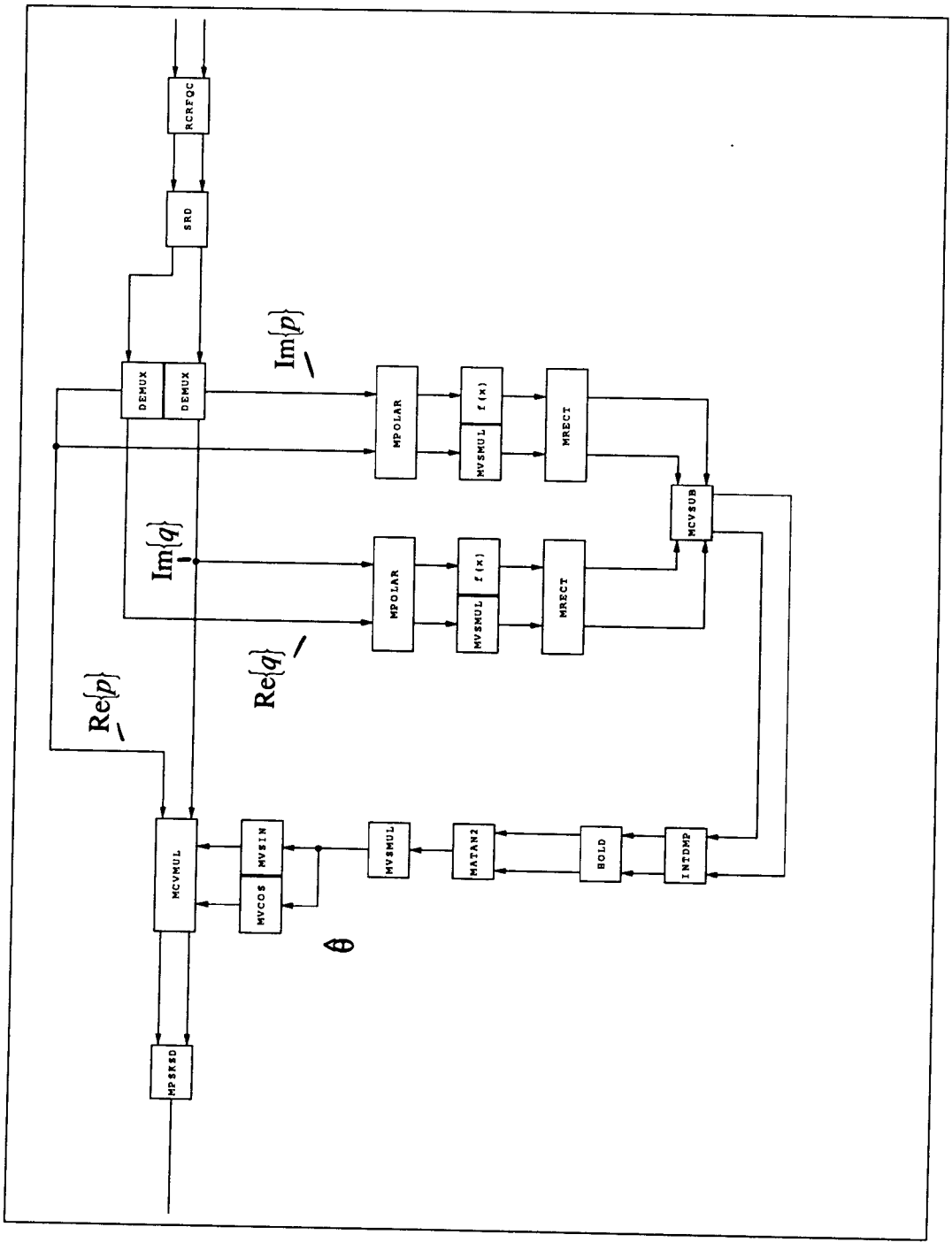


Fig. I.4-12b: V&VFF, Viterbi&Viterbi Feedforward carrier synchronizer for OQPSK

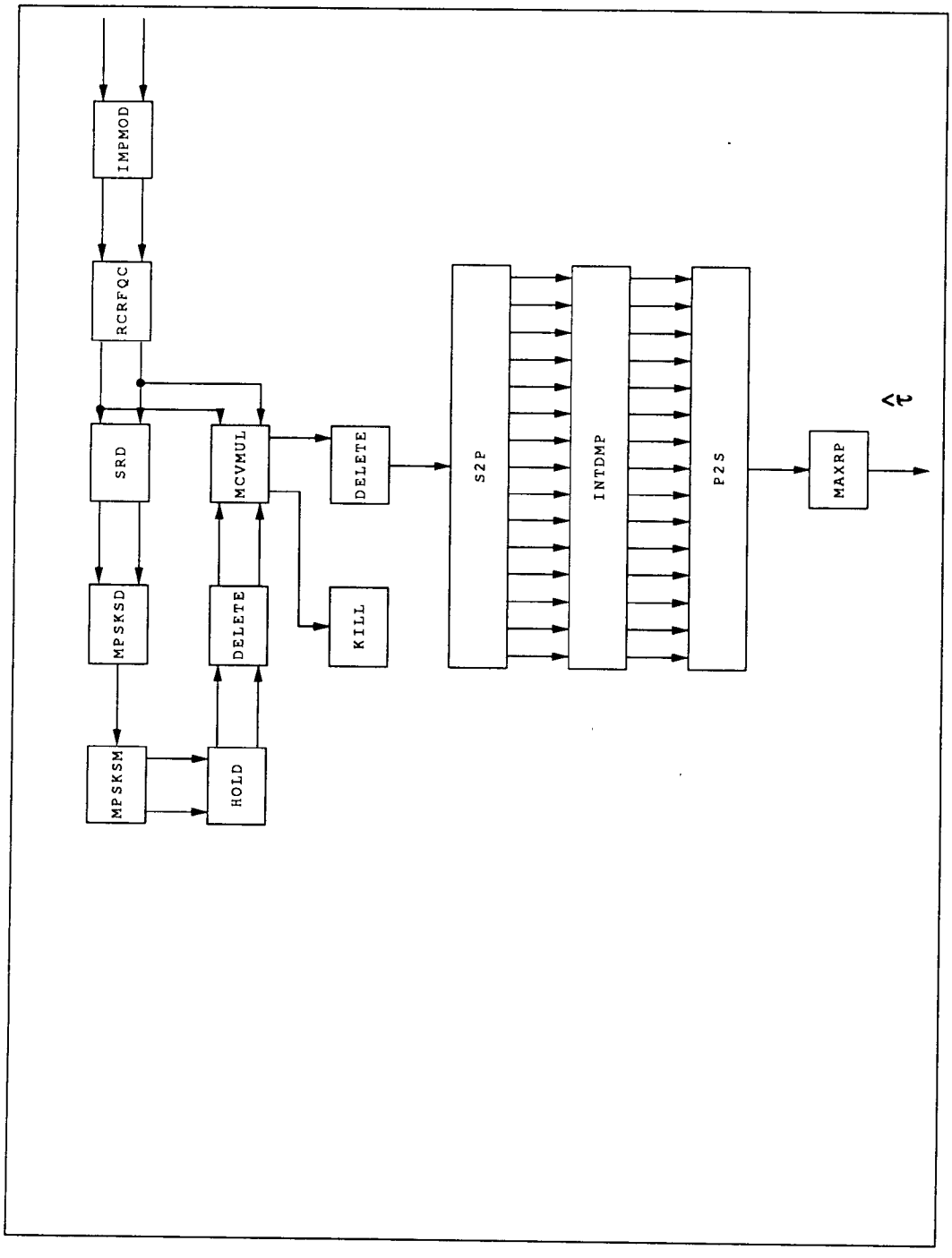


Fig. I.4-13: DDMLFF, Decision Directed Maximum Likelihood FeedForward symbol synchronizer

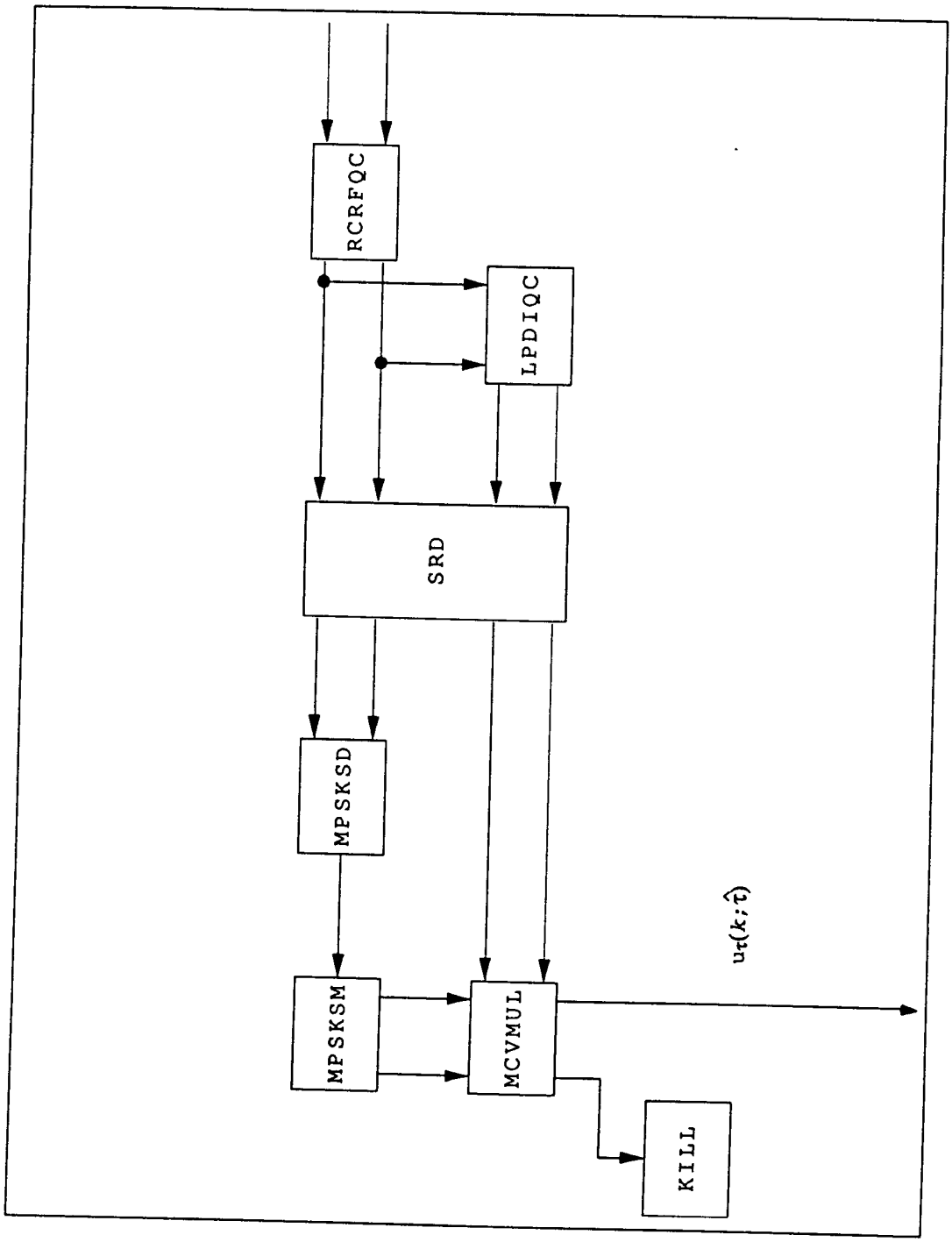


Fig. I.4-14: DDMLFB, Decision Directed Maximum Likelihood FeedBack symbol synchronizer

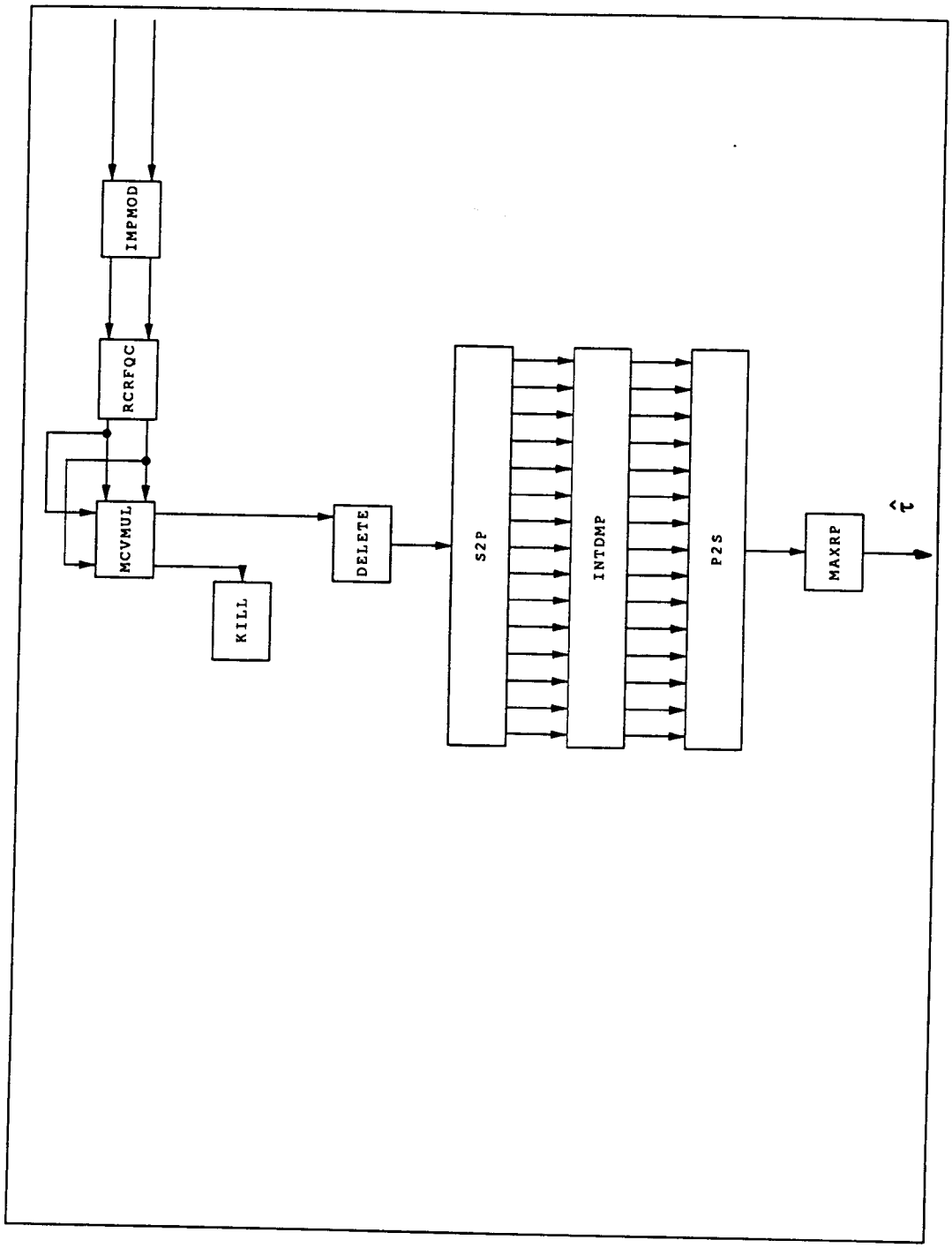


Fig. I.4-15: NDAMLFF, Non Data-Aided Maximum Likelihood FeedForward symbol synchronizer

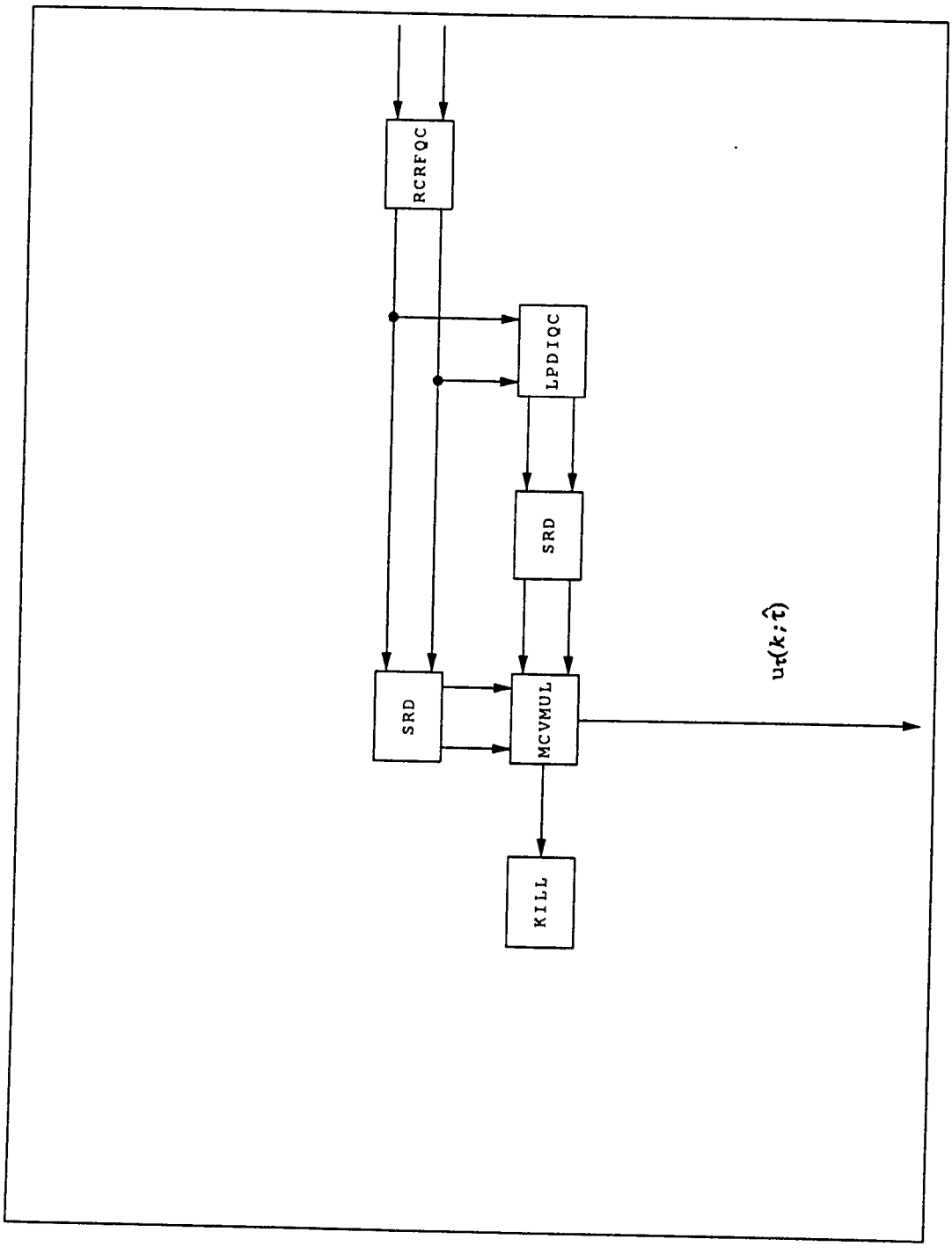


Fig. I.4-16: NDAMLFB, Non Data-Aided Maximum Likelihood FeedBack symbol synchronizer

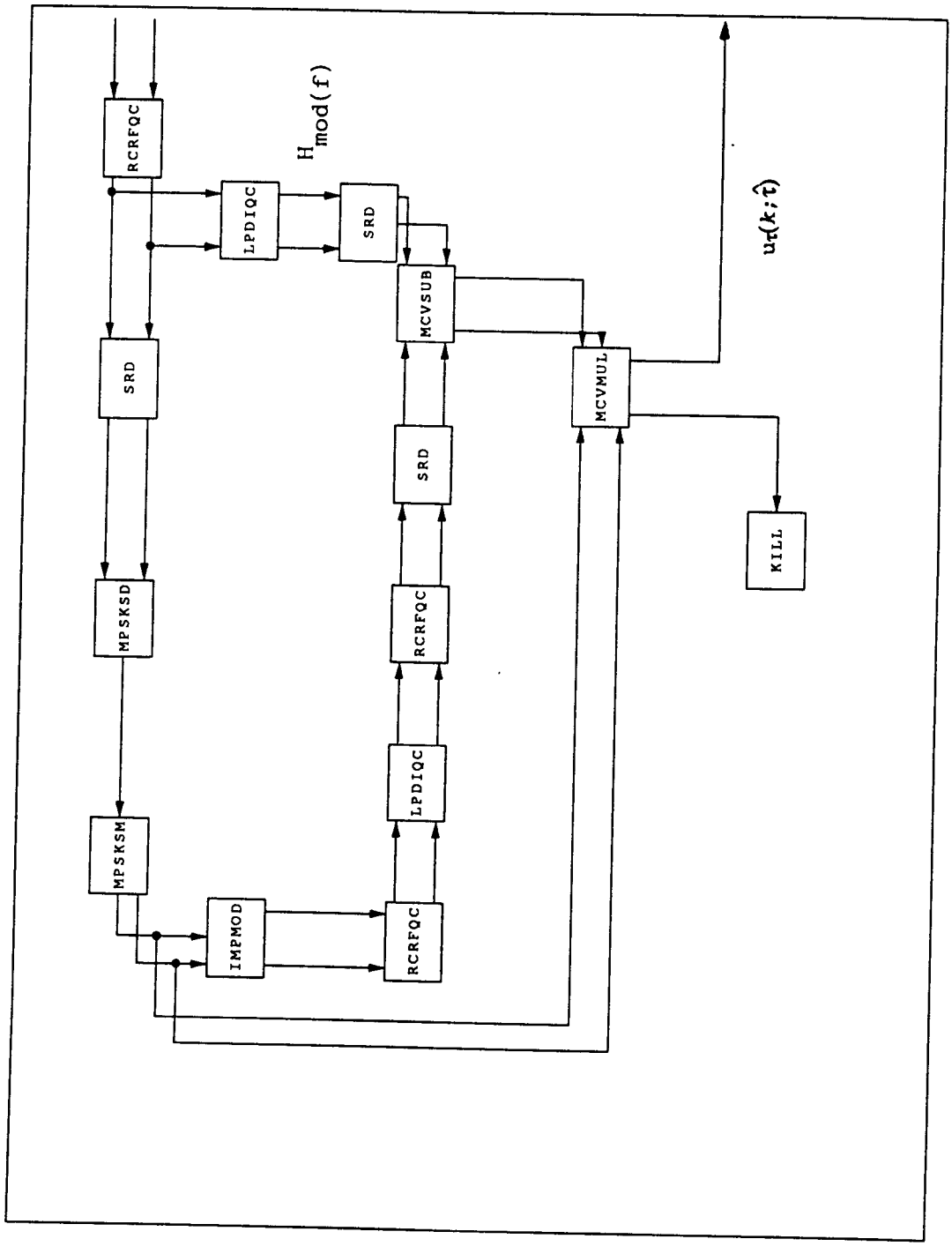


Fig. I.4-17: NDAMLFB$\overline{\text{SNFB}}$, Non Data-Aided Maximum Likelihood FeedBack symbol synchronizer with decision FeedBack SelfNoise cancellation

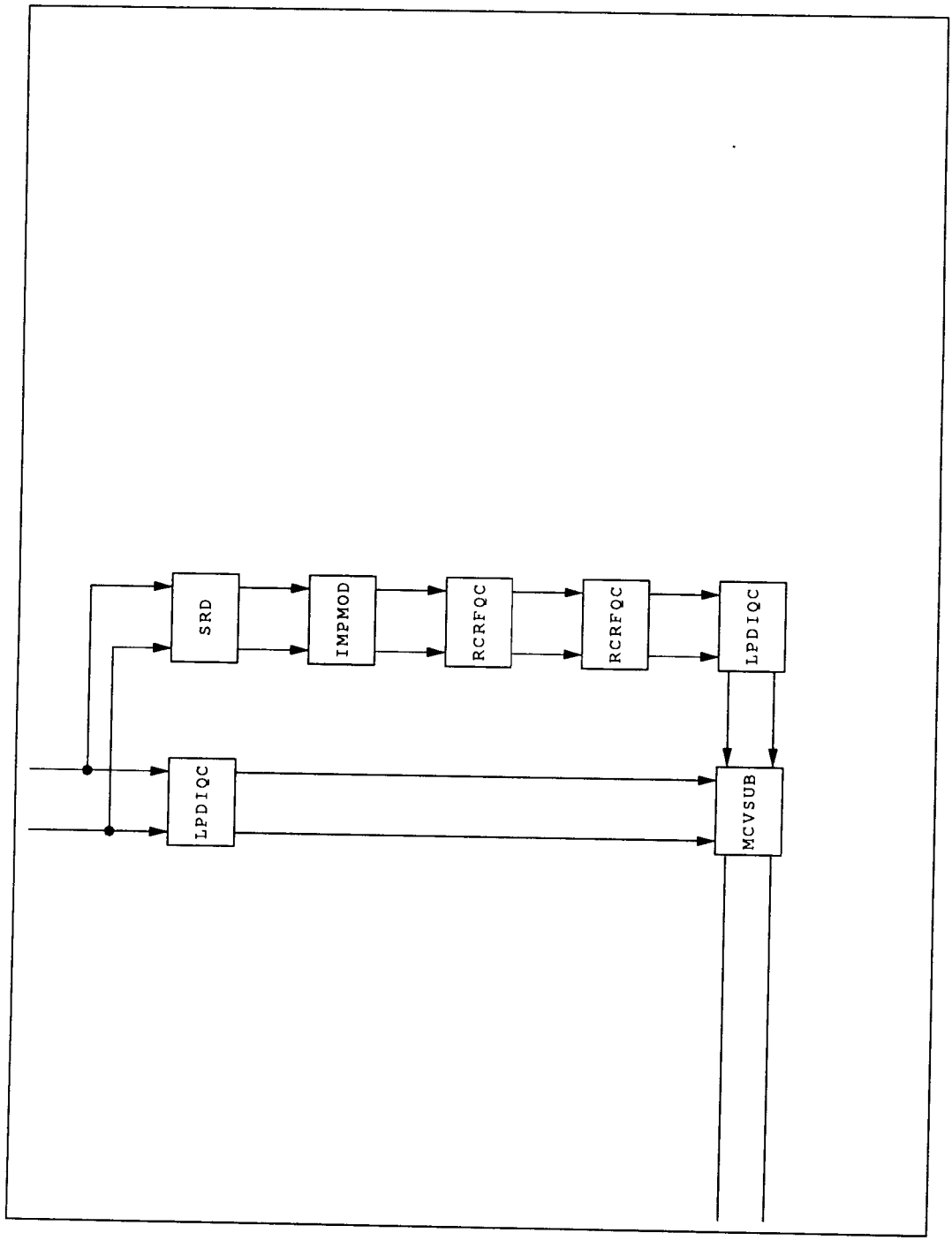


Fig. I.4-18: $\overline{\text{SNFF}}$, Modified derivative matched filter for FeedForward SelfNoise cancellation

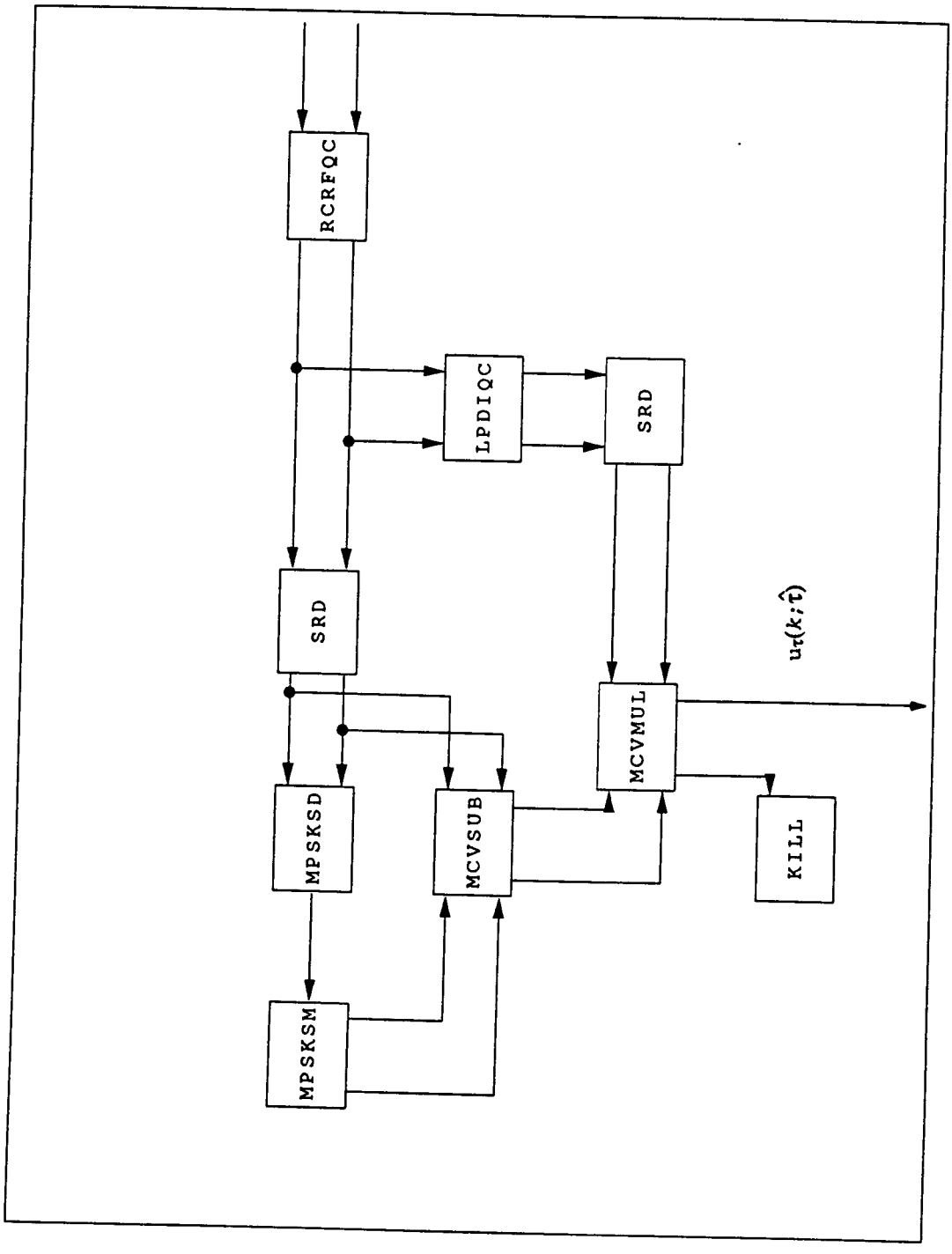


Fig. I.4-19: DDMMSEFB, Decision Directed MMSE FeedBack symbol synchronizer

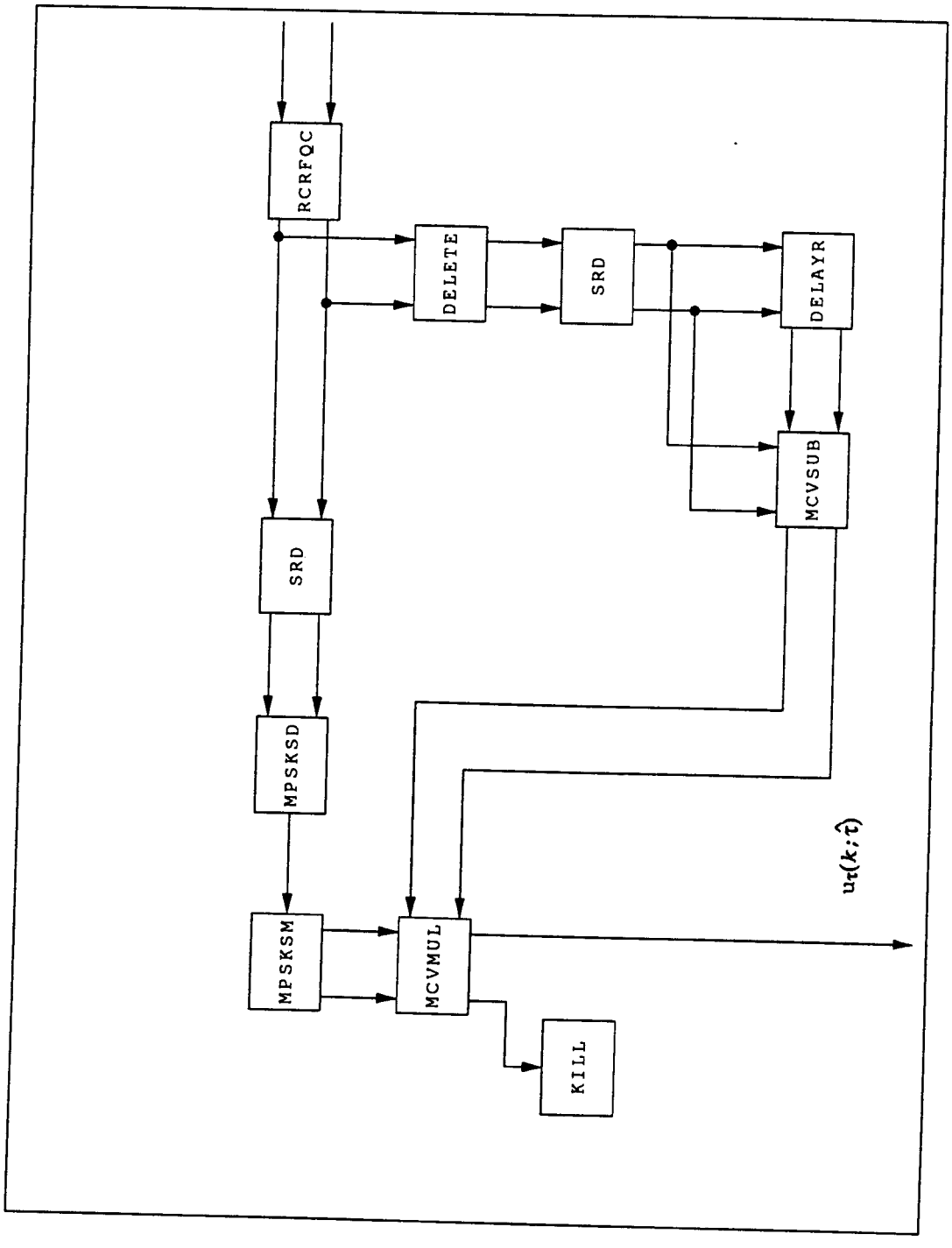


Fig. I.4-20: DDEL(FB), Decision Directed Early Late (FeedBack) symbol synchronizer

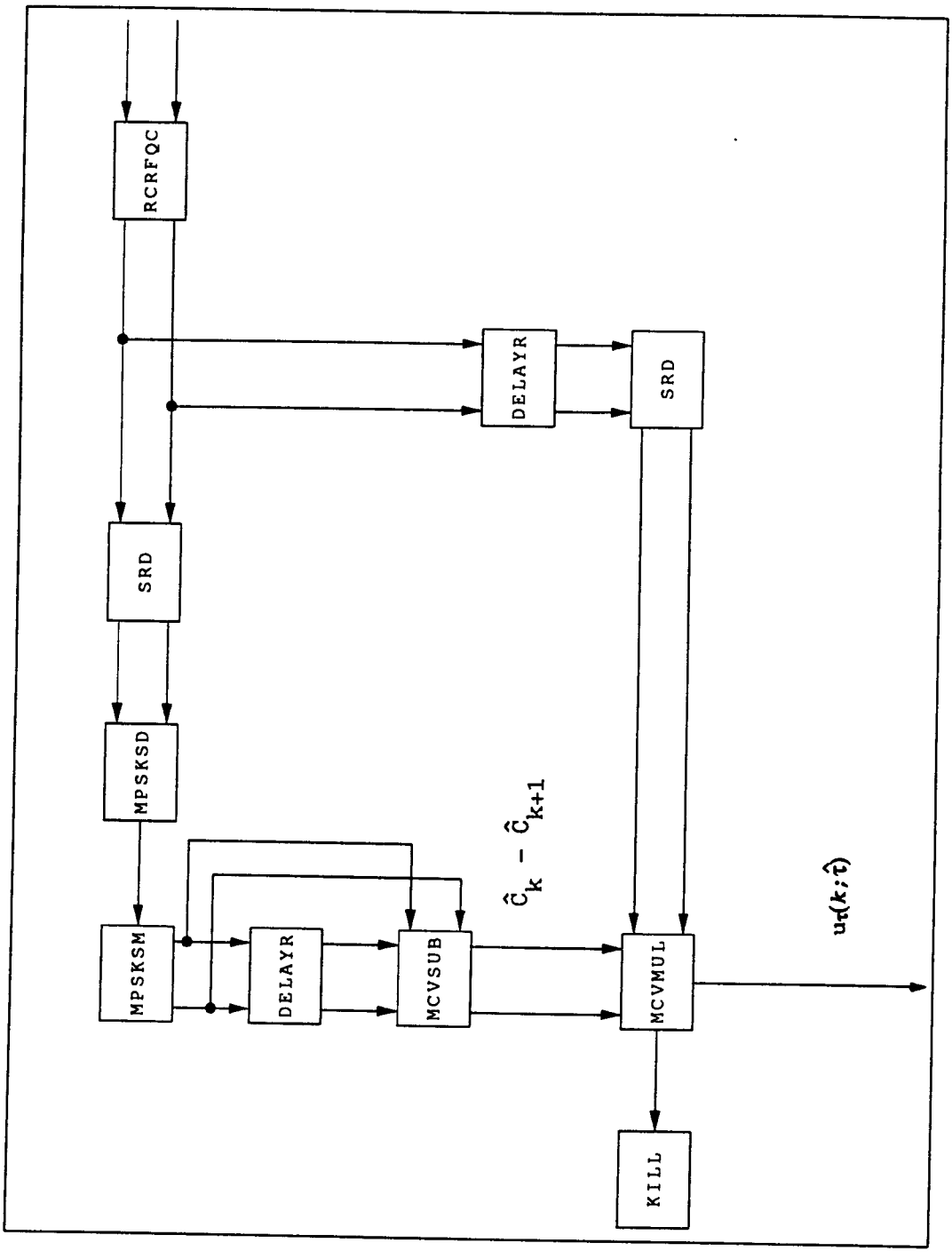


Fig. I.4-21: DTTL(FB), Data Transition Tracking Loop (FeedBack) symbol synchronizer

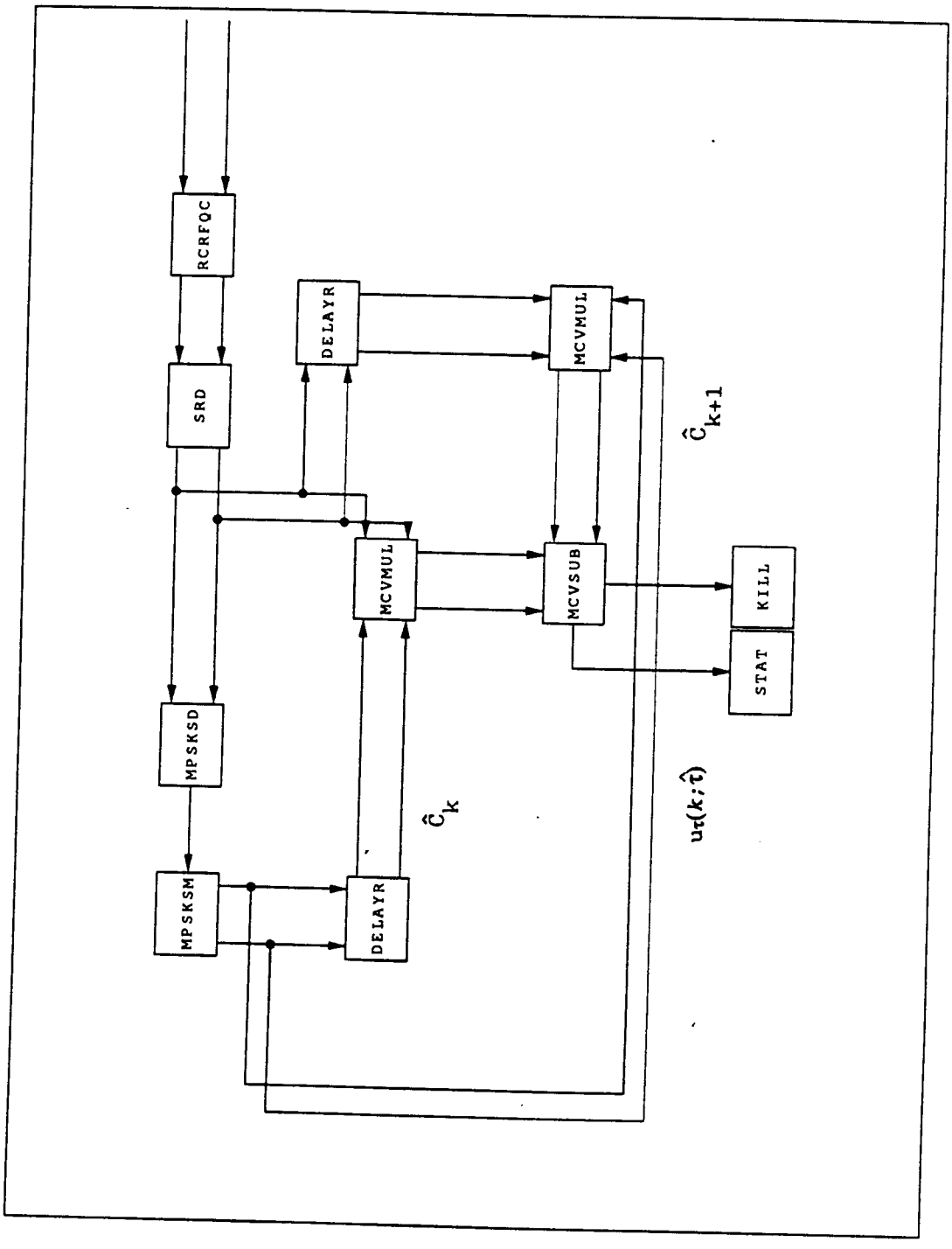


Fig. I.4-22: M&M(FB), Mueller&Müller (FeedBack) symbol synchronizer

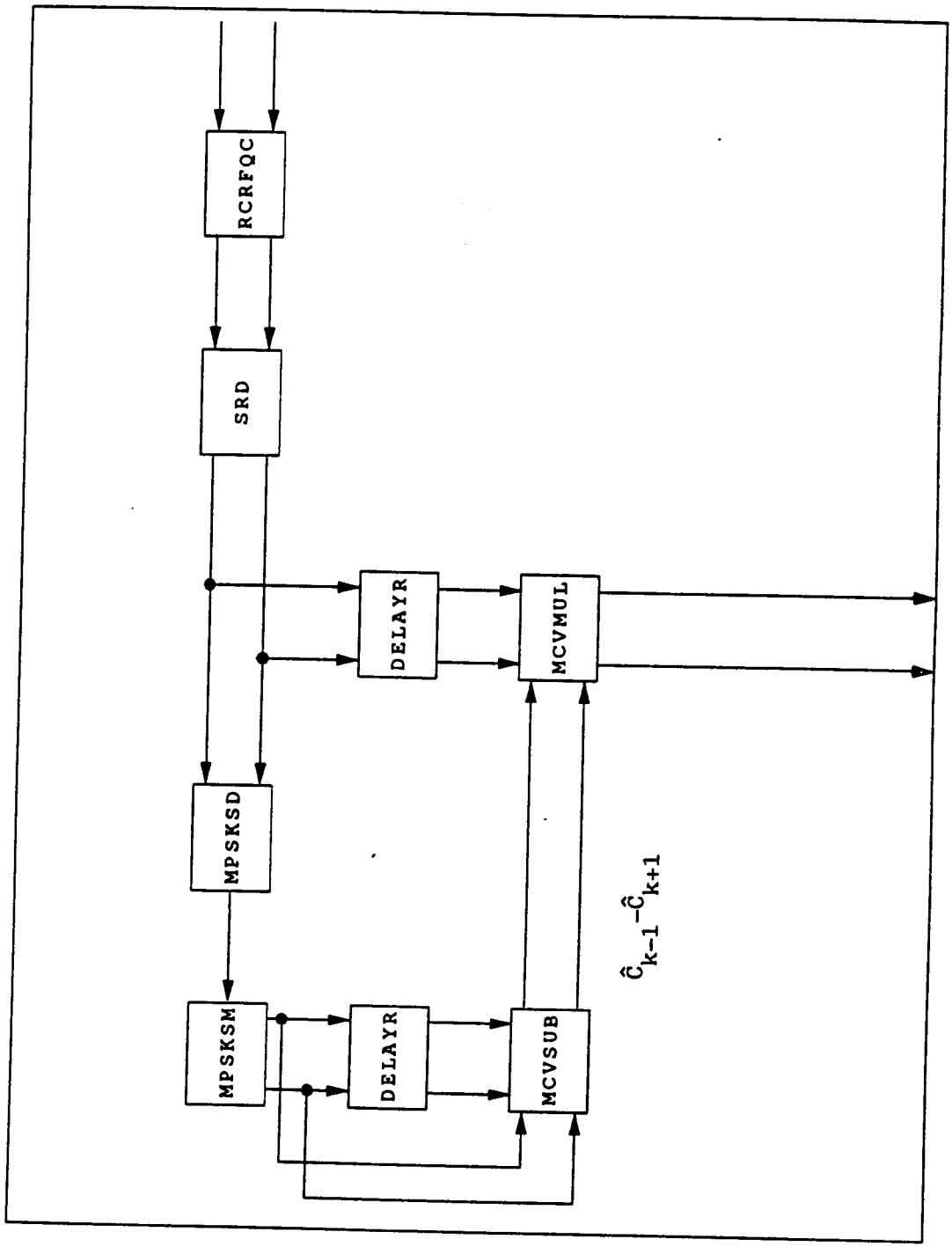


Fig. I.4-23: mM&M(FB), modified Mueller&Müller (FeedBack) symbol synchronizer

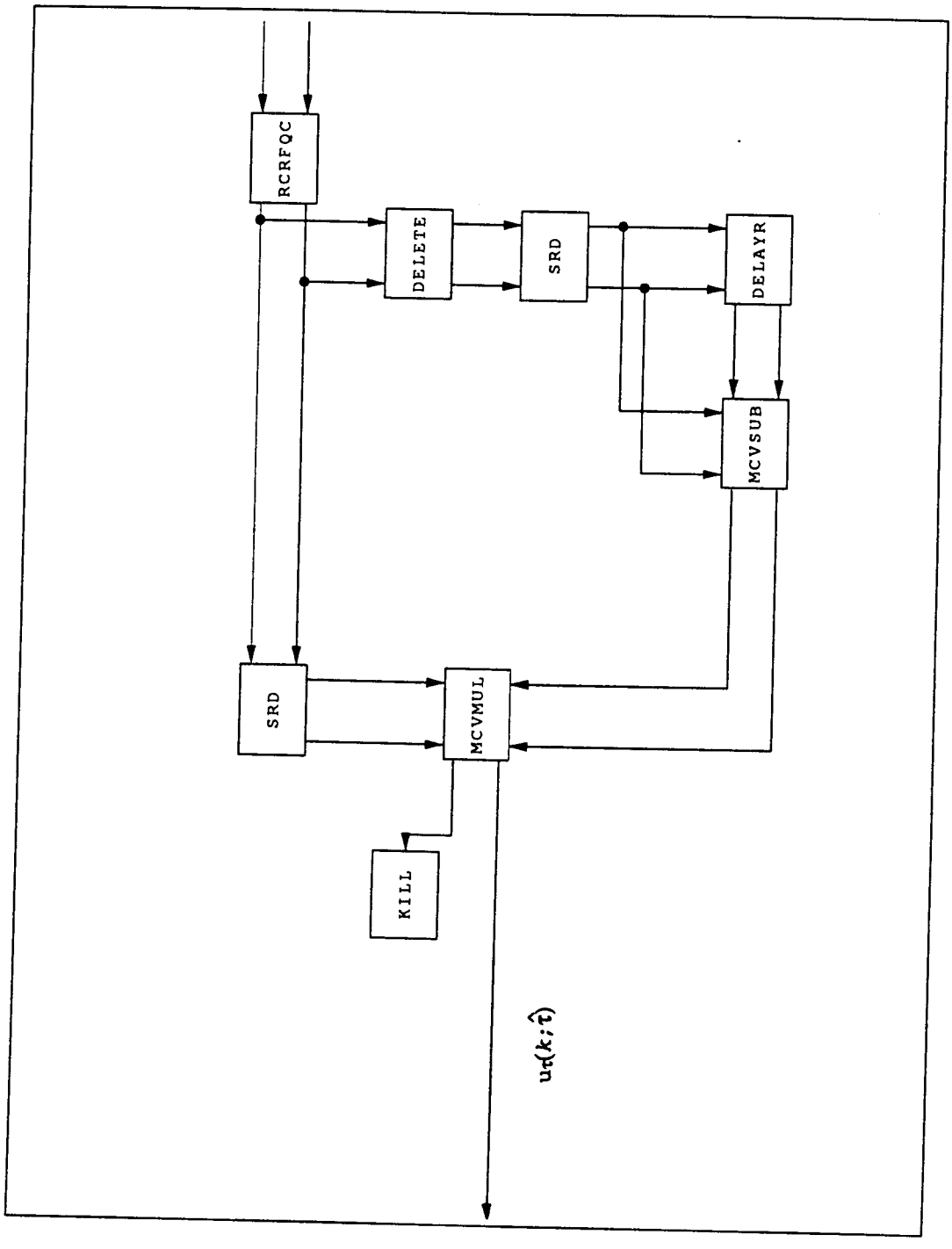


Fig. I.4-24: NDAEL(FB), Non Data-Aided Early Late (FeedBack) symbol synchronizer

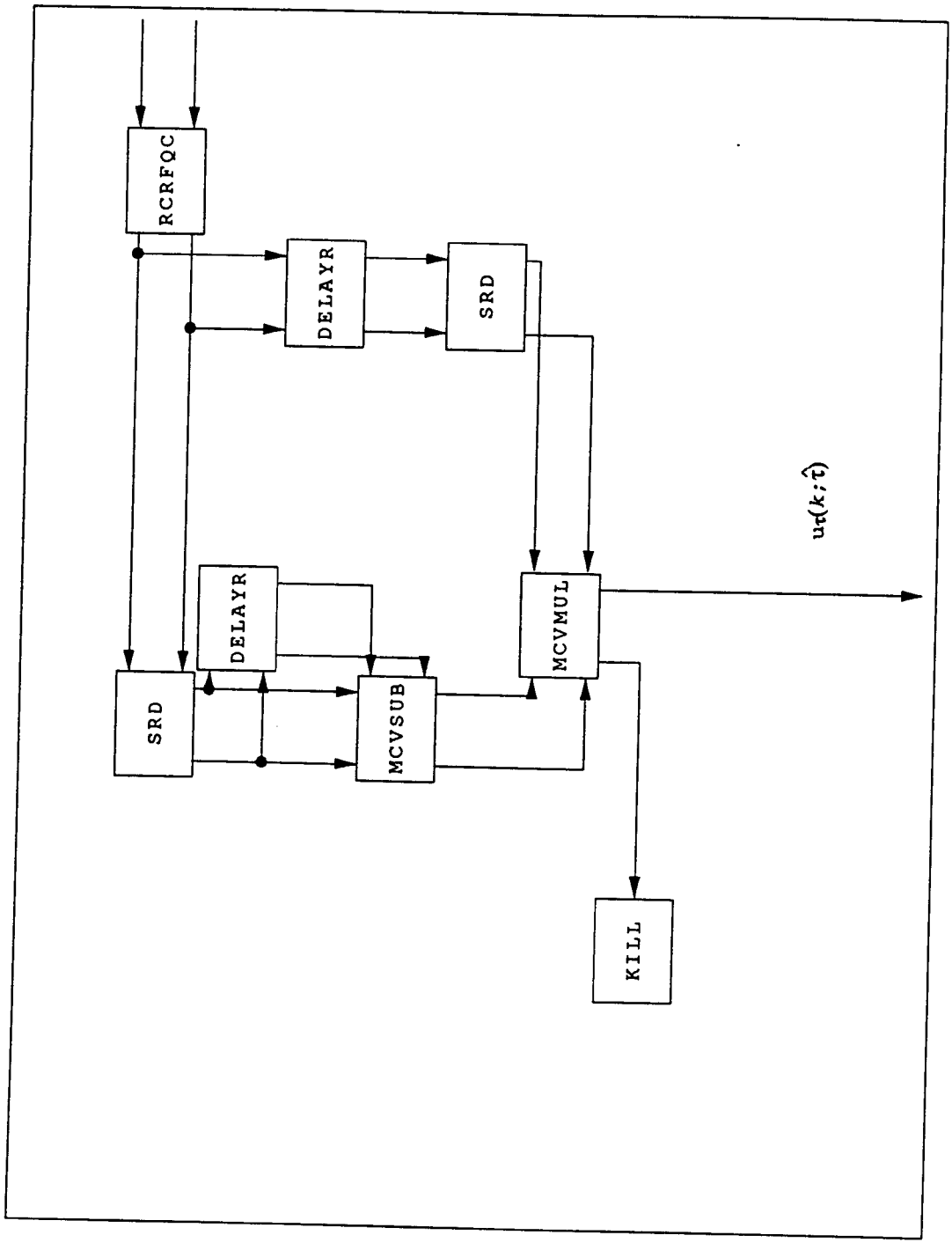


Fig. I.4-25: GA(FB), GArdner (FeedBack) symbol synchronizer

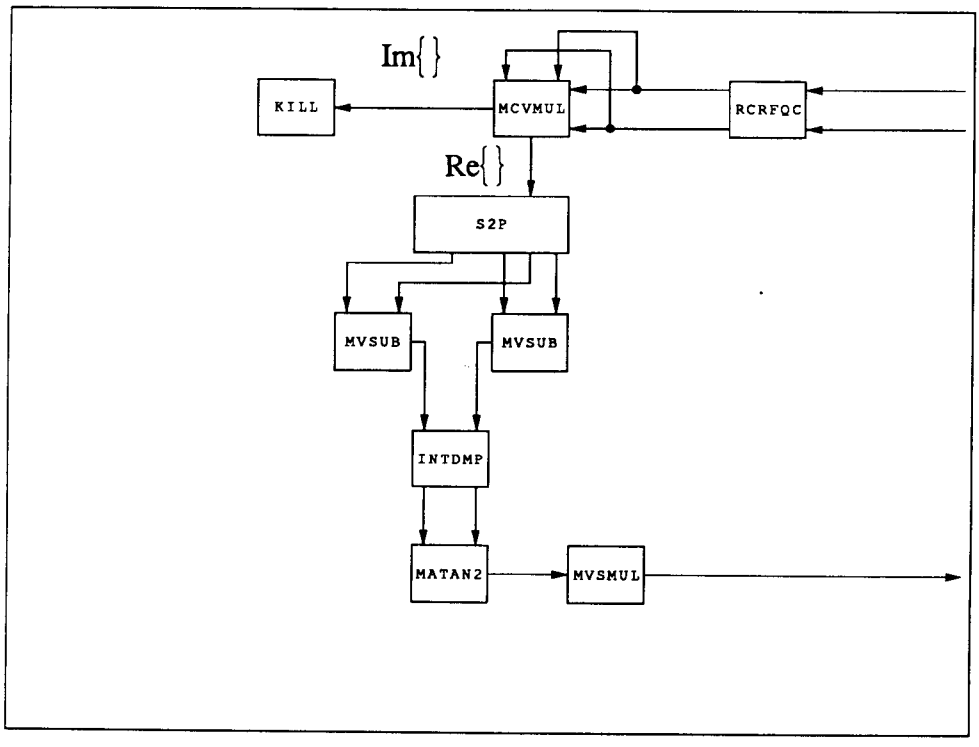


Fig. I.4-26: NDASQFF, Non Data-Aided Filter and Square FeedForward symbol synchronizer

Appendix I.A: List of Models in Alphabetical Order

This section contains a list of all models sorted in alphabetical order. For each model a short description is given.

Note that the following models have been renamed in COSSAP release 6.4:

Old Name	New Name
DELETE	DELR
DEMUX	DMUXR
HOLD	HOLDR
KILL	KILLR
P2S	P2SR
S2P	S2PR
SRD	SRDR

ACF1	Autocorrelation Function Estimation (Bartlett Periodogram Method)
ACF2	Autocorrelation Function Estimation (Welch Periodogram Method)
ADCONR	A/D Converter with Rounding Switching Levels
ADCONT	A/D Converter with Truncating Switching Level
ADIFI	Computes Absolute Difference between Two Integer Input Signals
AGCFF	Automatic Gain Control (feedforward)
APD	Avalanche Photodiode
APLC	Automatic Power Level Control
AWGN	Additive White Gaussian Noise Channel (Real)
AWGNQC	Additive White Gaussian Noise Channel (Complex)
BIAGVD	Computation of standard deviation factors (Variance Mod. Gaussian RV)
BIAPW	Piecewise Biasing of Input Signal Distribution
BIAPWD	Computation of interval bounds (Piecewise Const. Mod. Gaussian RV)
BSRC1	Pseudo Random Binary Source (1-32 Bit Shift Register)
BSRC2	Pseudo Random Binary Source (31 Bit Shift Register)
BUTF	Butterworth Filter (Real Input Signal)
BUTFQC	Butterworth Filter (Complex Input Signal)
CH2F	Chi-Square Fitting Test (Arbitrary Distribution)
CH2FG1	Chi Square Fitting Test of Gaussian RV (Known Mean and Variance)
CH2FG2	Chi Square Fitting Test of Gaussian RV (Unknown Mean and Variance)
CHBFQC	Chebychev Filter (Complex Input Signal)
CMPEQI	Compares Two Integer Signals (1 on Match)
CMPNEI	Compares Two Integer Signals (0 on Match)
CONSTI	Constant Integer Signal Source
CONSTR	Constant Real Signal Source
CONV	Convolution with Real Impulse Response (Real Input Signal)
CONVQC	Convolution with Complex Impulse Response (Complex Input Signal)
COUNT1	Counter (Error Counter)
COUNTP	Counter (Error Counter - Line Chart)
CPMM1	CPM Modulator (CPM: Continuous Phase Modulation)
CSYMIN	Complex Symbol Insert
CSYMRM	Complex Symbol Remove
DACONV	D/A Converter (Digital to Analog Converter)
DBGINI	Signal Input from Keyboard (INTEGER)
DBGINR	Signal Input from Keyboard (REAL)
DBGOUI	Signal Output to Screen (INTEGER)
DBGOUR	Signal Output to Screen (REAL)

DCOD1	Differential Encoder
DDCOD1	Differential Decoder
DELAYI	Delays Integer Signals (by multiples of the sampling time)
DELAYR	Delays Real Signals (by multiples of the sampling time)
DELCMP	Limiter with Data Compression
DELI	Deletes First Signal Elements (Integer Signals)
DELR	Deletes First Signal Elements (Real Signals)
DIGADD	Digital Adder
DIGMUL	Digital Multiplier
DIGSUB	Digital Subtractor
DIST1	Signal Power Distribution Output (Rayleigh paper)
DIST2	Peak Length Distribution Output
DMPC	Dump Complex Signal (graphics and file output)
DMPNI	Dump Integer Signals (graphics and file output)
DMPNR	Dump Real Signals (graphics and file output)
DMUXI	Demultiplexer (1 in 2, blockwise alternating, Integer Signals)
DMUXR	Demultiplexer (1 in 2, blockwise alternating, Real Signals)
DMUXSI	Demultiplexer (1 in N, blockwise, controlled by 2nd input signal, Integer Signals)
DMUXSR	Demultiplexer (1 in N, blockwise, controlled by 2nd input signal, Real Signals)
DPSKD1	Differential M-ary PSK Detector
DQ1	Computes Differential Quotient (Discrete Time Approximation)
DSC1	Hardlimiter
EKMSM0	Extended Kalman Smoother/Filter
EPSID	Maximum Likelihood Symbol Timing Estimation
EPSRE	Symbol Timing Estimation (by correlation)
ESTVA1	Variance Estimator for a Mean Value Estimator (known mean)
ESTVA2	Variance Estimator for a Mean Value Estimator
EYE	Eye Pattern (graphics and file output)
FDELSI	Time Varying (fractional) Delay - Synchronous to Input
FDELSO	Time Varying (fractional) Delay - Synchronous to Output
FDEVC	Rotates a Complex Signal by a Constantly Incrementing Angle (Constant Frequency Deviation)
FORKB	Forks a Bit_Vector Signal
FORKC	Forks a Complex Signal
FORKDI	Forks a Double Integer Signal
FORKDR	Forks a Double Real Signal
FORKI	Forks an Integer Signal
FORKR	Forks a Real Signal
GMSK	GMSK Modulator (Gaussian Minimum Shift Keying)
GSTAIL	Source for Random Data with Singlesided Gaussian Distribution
GSTLVT	Source for Random Data with Gaussian Tail Distr., Variable Threshold

GWSMR	Gliding Window Smoother (short term mean, N real input signals)
HIST1	Histogram of Input Signal Level (graphics and file output)
HLBP	HardLimiter (BandPass Signals, Constant Amplitude, n-th Harmonic)
HOLDI	Repeats Each Signal Element n Times (Integer Signals)
HOLDR	Repeats Each Signal Element n Times (Real Signals)
HYST	Hysteresis
IIRC	Infinite Impulse Response Filter (Real Inputs)
IIRCV	Infinite Impulse Response Filter with Variable Cutoff Frequency (Real Input Signals)
ILPF	Ideal Lowpass Filter (Real Input Signal)
ILPFQC	Ideal Lowpass Filter (Complex Input Signal)
IMPMOD	Impulse Modulator
INT1	Integrator (Discrete Time Approximation)
INTDMP	Integrate and Dump
INTRLI	Blockinterleaver / -deinterleaver (Integer Signals)
INTRLR	Blockinterleaver / -deinterleaver (Real Signals)
INVMOD	Inverse Modulo Operation
ISIGI	Input SIGnal Integer (Integer Input Signal Read From File)
ISIGR	Input SIGnal Real (Real Input Signal Read From File)
KFILS	Linear Kalman Filter
KILLI	Signal Sink (Integer Signals)
KILLR	Signal Sink (Real Signals)
LININT	Linear Interpolator (Real Input Signal)
LNR1	Lognormal Fading
LPDI	Differentiating Filter (Real Input Signal)
LPDIQC	Differentiating Filter (Complex Input Signal)
LTI	Linear Time Invariant Filter (Real Input Signal)
LTIQC	Linear Time Invariant Filter (Complex Input Signal)
LTV	Linear Time Variant Filter (Real Input Signal and Coefficients)
LTVQC	Linear Time Variant Filter (Complex Input Signal and Coefficients)
MATAN2	Computes Arcus Tangens of a Complex Input Signal
MAXI	Searches Maximum in an Integer Input Signal Block
MAXR	Searches Maximum in a Real Input Signal Block
MC1	Manchester Encoder
MCFFT	Computes the Complex (Inverse) DFT of a Complex Input Signal
MCVADD	Adds Two Complex Input Signals
MCVCSM	Multiplies a Complex Input Signal by a Complex Scalar
MCVDIV	Divides Two Complex Input Signals
MCVMGS	Computes the Squared Absolute Value of a Complex Input Signal
MCVMUL	Computes the Product of Two Complex Input Signals

MCVROT	Rotates Complex Input Signal by Constant Angle
MCVSUB	Subtracts Two Complex Input Signals
MFTAPC	Computation of Complex Matched Filter Coefficients (Time Variant)
MI2R	Integer Signal to Real Signal Conversion
MINI	Searches Minimum in an Integer Input Signal Block
MINR	Searches Minimum in a Real Input Signal Block
MLIND	M-ary Linear Demodulator (Programmable)
MLINM	M-ary Linear Modulator (Programmable)
MMSEQC	Minimum Mean Square Error Adaptive Equalizer (Complex Signals)
MPAMD	M-ary PAM Demodulator
MPAMM	M-ary PAM Modulator
MPOLAR	Converts Complex Signal from Cartesian to Polar Representaion
MPSKD	M-ary PSK Demodulator
MPSKM	M-ary PSK Modulator
MPSKSD	M-ary PSK Demodulator
MPSKSM	M-ary PSK Modulator
MR2I	Real Signal to Integer Signal Conversion (Rounding to Nearest Integer)
MRECT	Converts Complex Signal from Polar to Cartesian Representation
MSKD1	Minimum Shift Keying Detector
MUXAI	Multiplexer (N in 1, asynchronous, Integer Signals))
MUXAR	Multiplexer (N in 1, asynchronous, Real Signals))
MUXI	Multiplexer (2 in 1, blockwise alternating, Integer Signals)
MUXR	Multiplexer (2 in 1, blockwise alternating, Real Signals)
MVABS	Computes Absolute Value of a Real Input Signal
MVADD	Adds Two Real Input Signals
MVCLIP	Softlimiter (signal clipping)
MVCOS	Computes Cosine of a Real Input Signal
MVDIV	Divides Two Real Input Signals
MVEX	Exponentiate Real Input Signal (Base e)
MVEX10	Exponentiate Real Input Signal (Base 10)
MVIABS	Computes Absolute Value of an Integer Input Signal
MVIADD	Adds Two Integer Input Signals
MVIDIV	Divides Two Integer Input Signals
MVIMUL	Multiplies Two Integer Input Signals
MVISAD	Adds an Integer Scalar to an Integer Input Signal
MVISDI	Divides an Integer Input Signal by an Integer Scalar
MVISMU	Multiplies an Integer Input Signal by an Integer Scalar
MVISSU	Subtracts an Integer Scalar from an Integer Input Signal
MVISUB	Subtracts Two Integer Input Signals
MVMUL	Multiplies Two Real Input Signals

MVRG1	Pseudo Random Source (Uniform Distribution)
MVRG2	Pseudo Random Source (Uniform Distribution, Whichman/Hill Alg.)
MVSADD	Adds a Real Scalar to a Real Input Signal
MVSIN	Computes Sine of a Real Input Signal
MVSMUL	Multiplies a Real Input Signal by a Real Scalar
MVSQ	Computes Square of a Real Input Signal
MVSQRT	Computes the Square Root of a Real Input Signal
MVSSUB	Subtracts a Real Scalar from a Real Input Signal
MVSUB	Subtracts Two Real Input Signals
OHTRIR	Orthogonal Householder Transformation on Real Input Signals
P2SI	Parallel to Serial Conversion (N in 1, Sample by Sample Multiplexer, Integer Signals)
P2SR	Parallel to Serial Conversion (N in 1, Sample by Sample Multiplexer, Real Signals)
PEAKAA	Area Under Signal Peak (one output sample per peak)
PEAKAS	Area Under Signal Peak (output signal is running area)
PEAKLA	Length of Signal Peak (one output sample per peak)
PEAKLS	Length of Signal Peak (output signal is running length)
PPGENI	Periodic Integer Pattern Source
PPGENR	Periodic Real Pattern Source
PROD	Computes the Product of an Arbitrary Number of Real Input Signals
PSD1	Power Spectral Density Estimation (Bartlett Method, Real Signal)
PSD1QC	Power Spectral Density Estimation (Bartlett Method, Complex Signal)
PSD2	Power Spectral Density Estimation (Welch Method, Real Signal)
PSD2QC	Power Spectral Density Estimation (Welch Method, Complex Signal)
PULS	Periodic Puls Generator for Double-Trapezoidal Pulses
PULSLG	Periodic Puls Generator for Long Double-Trapezoidal Pulses
QUADCO	Balanced Quadricorrelator (frequency estimator)
QUANT	Quantizer
QUASI1	Quasi Analytical Estimation of Bit Error Probability (BPSK, AM)
QUASI2	Quasi Analytical Estimation of Bit Error Probability (QPSK)
RCIN	Interpolator with Raised Cosine Filter (Real Input Signal)
RCRF	Root Raised Cosine Filter (Real Input Signal)
RCRFQC	Root Raised Cosine Filter (Complex Input Signal)
REFGEN	Reference Signal Generation
RMEAN	Running Mean of Input Signal
RMIX1I	Random Multiplexer (2 in 1, no data lost, Integer Signals)
RMIX1R	Random Multiplexer (2 in 1, no data lost, Real Signals)
RMIX2I	Random Multiplexer (2 in 1, data partially lost, Integer Signals)

RMIX2R	Random Multiplexer (2 in 1, data partially lost, Real Signals)
RMIXI	Random Multiplexer (N in 1, no data lost, Integer Signals)
RMIXR	Random Multiplexer (N in 1, no data lost, Real Signals)
RSTI	Restore Dumped Integer Signals
RSTR	Restore Dumped Real Signals
RWALK1	Second Order Gauss-Markov Random Walk
S2PI	Serial to Parallel Conversion (1 in N, sample by sample Demultiplexer, Integer Signals)
S2PR	Serial to Parallel Conversion (1 in N, sample by sample Demultiplexer, Real Signals)
SAW	Sawtooth Nonlinearity
SCAT	Scatterdiagram (graphics and file output)
SCLBP	Symmetric Clipper (BandPass Signals, const. amplitude, n-th harmonic)
SFRAME	Symbol Framing (Insertion of Input Symbols into Frame Structure)
SIIN	Sin(x)/x - Interpolator (Real Input Signal)
SIINQC	Sin(x)/x - Interpolator (Complex Input Signal)
SRDI	Sampling Rate Decimation (Integer Signals)
SRDR	Sampling Rate Decimation (Real Signals)
SRXI	Sampling Rate Expansion (Integer Signals)
SRXR	Sampling Rate Expansion (Real Signals)
STAT	Signal Statistics - Mean, Power, Variance (Text Chart)
STATP	Signal Statistics - Mean, Power, Variance (Line Chart)
STOPI	Stops Simulation (by counting integer signal elements)
STOPR	Stops Simulation (by counting real signal elements)
SUMI	Computes the Sum of an Arbitrary Number of Integer Input Signals
SUMR	Computes the Sum of an Arbitrary Number of Real Input Signals
SYMDEC	M-ary to Binary Conversion
SYMEL1	Complex Nonlinearity (for M-ary PSK symbol elimination)
SYMGCD	M-ary to Binary Conversion, Gray Code
SYMGCG	Binary to M-ary Conversion, Gray Code
SYMGEN	Binary to M-ary Conversion
TAPFIL	Estimates Equiv. Channel Filter Coefficients (Time Variant)
TEBRS	Timing (Clock) Error Detector (Mueller&Mueller)
TERD	Timing (Clock) Error Detector for BPSK or QPSK (Gardner)

TIMEXT	Digital Filter&Square Timing (Clock) Error Detector for QAM or PSK
TRELL1	Trellis Encoder (programmable, 1 dimensional codes)
TWT1	Traveling Wave Tube (Scaled Amplitude Model by Thomas)
TWT2	Traveling Wave Tube (Quadrature Model by Hetrakul/Taylor)
TWT3	Traveling Wave Tube (General Amplitude/Phase Model, by Thomas et al.)
TWT4AP	Traveling Wave Tube (Amplitude-Phase Model by Saleh)
TWT4Q	Traveling Wave Tube (Quadrature Model by Saleh)
VITRB1	Viterbi Decoder (programmable, Euclidian Distance)
WGN	White Gaussian Noise Source (Real)
WGNQC	White Gaussian Noise Source (Complex)
WTSGAM	Computation of Unbiasing Weights (Abs. & Mean Mod. Gaussian RV)
WTSGBM	Computation of Unbiasing Weights (Bimod. Mean Mod. Gaussian RV)
WTSGM	Computation of Unbiasing Weights (Mean Mod. Gaussian RV)
WTSGMV	Computation of Unbiasing Weights (Mean Mod. Gaussian RV)
WTSGU	Computation of Unbiasing Weights (Unif. RV for Gaussian RV)
WTSGV	Computation of Unbiasing Weights (Variance Mod. Gaussian RV)
WTSGV	Computation of Unbiasing Weights (Variance Mod. Gaussian RV)
WTSGVM	Computation of Unbiasing Weights (Var. & Mean Mod. Gaussian RV)
WTSGVV	Computation of Unbiasing Weights (Variance Mod. Gaussian RV)
WTSPW	Computation of Unbias Weights (piecewise biased distribution)
WTTCM	Compute Total Weights from Single Weights

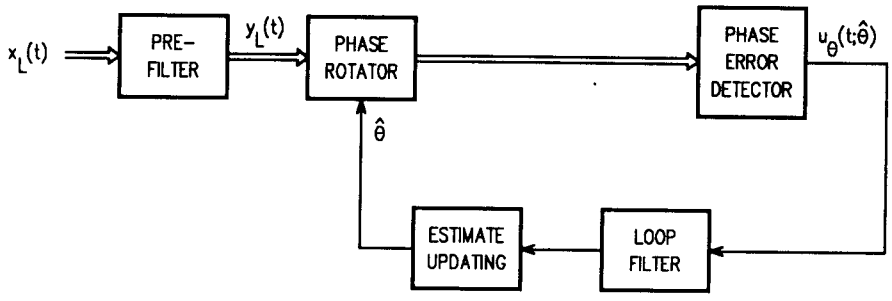


Fig. I.2–7 : Timing independent feedback carrier synchronization (NDAMLFBT)

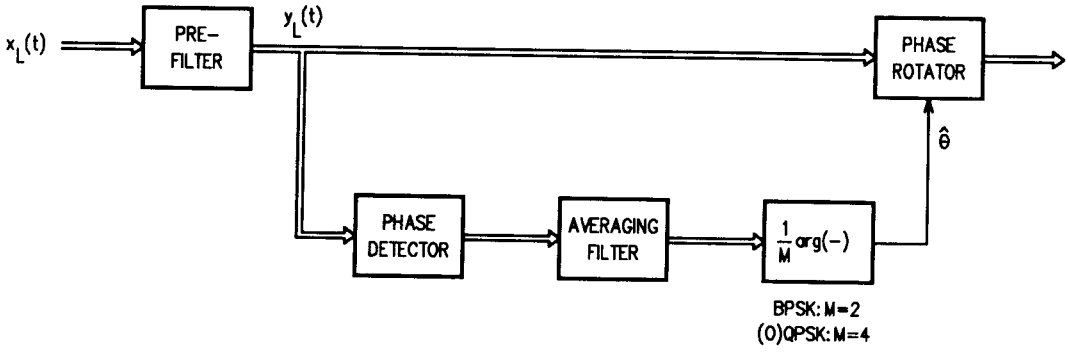


Fig. I.2–8 : Timing independent feedforward carrier synchronization (NDAMLFFT)

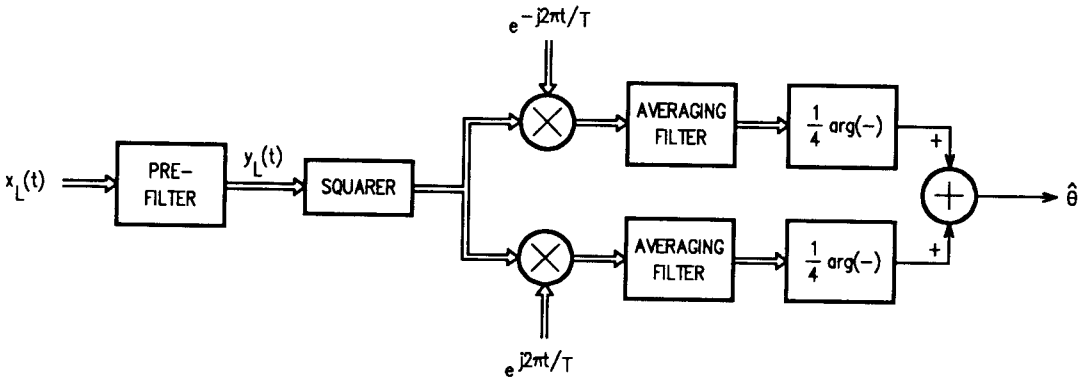


Fig. I.2.9 Timing independent feedforward carrier synchronization for MSK

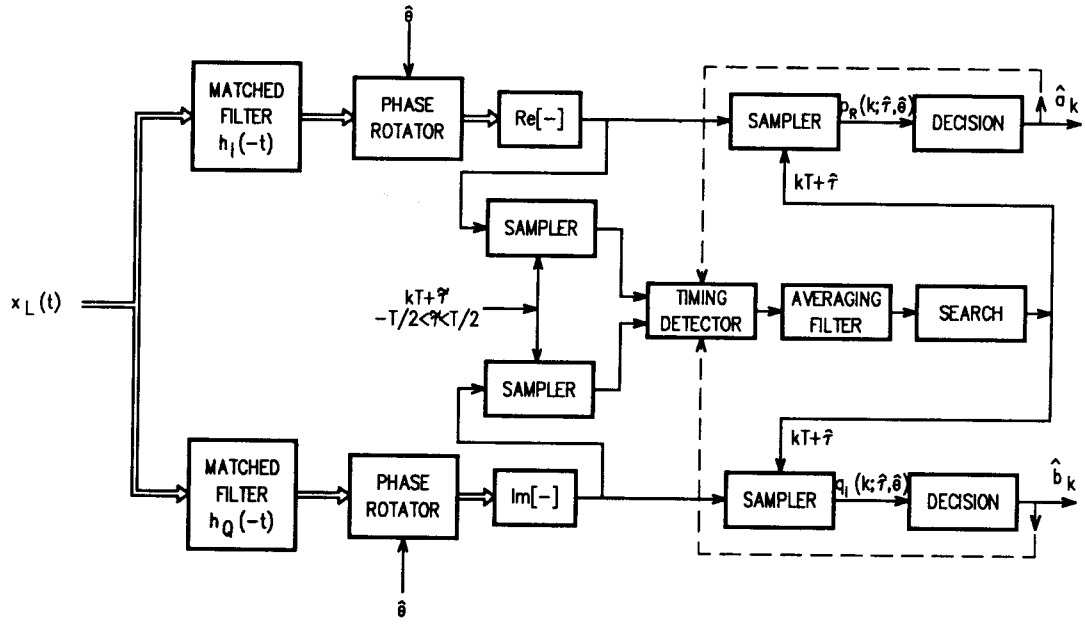


Fig. I.2–10 : ML feedforward symbol synchronization (DDMLFF, NDAMLFF)

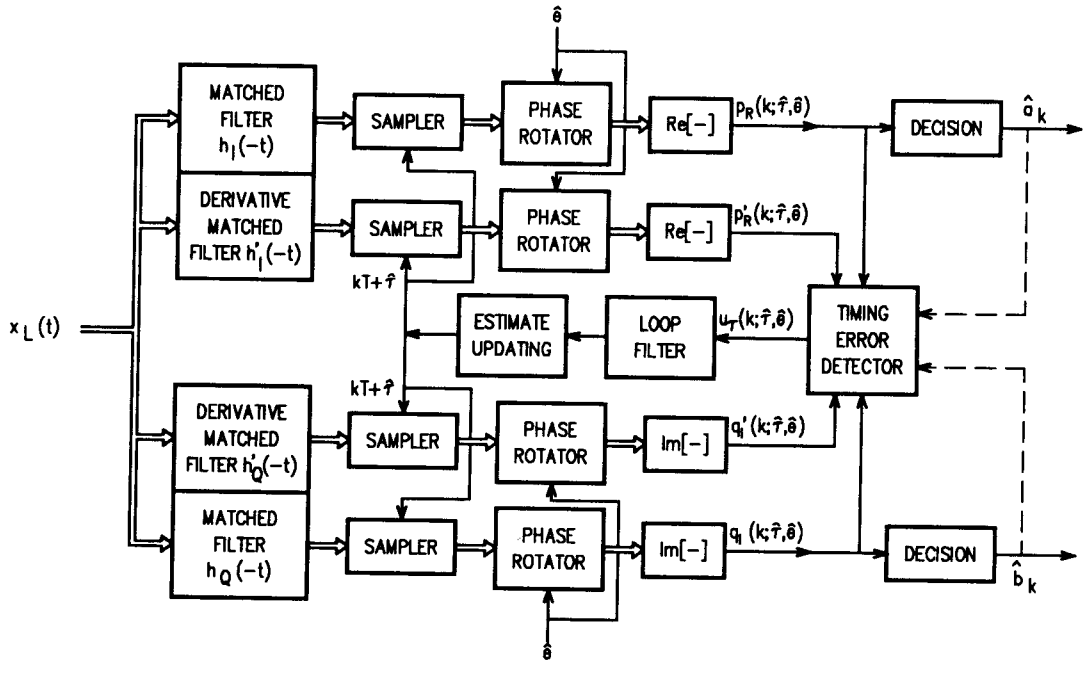


Fig. I.2–11: feedback symbol synchronization with derivative matched filter (DDMLFB, NDAMLFB, DDMLFB$\overline{\text{SNFB}}$, DDMMSEFB)

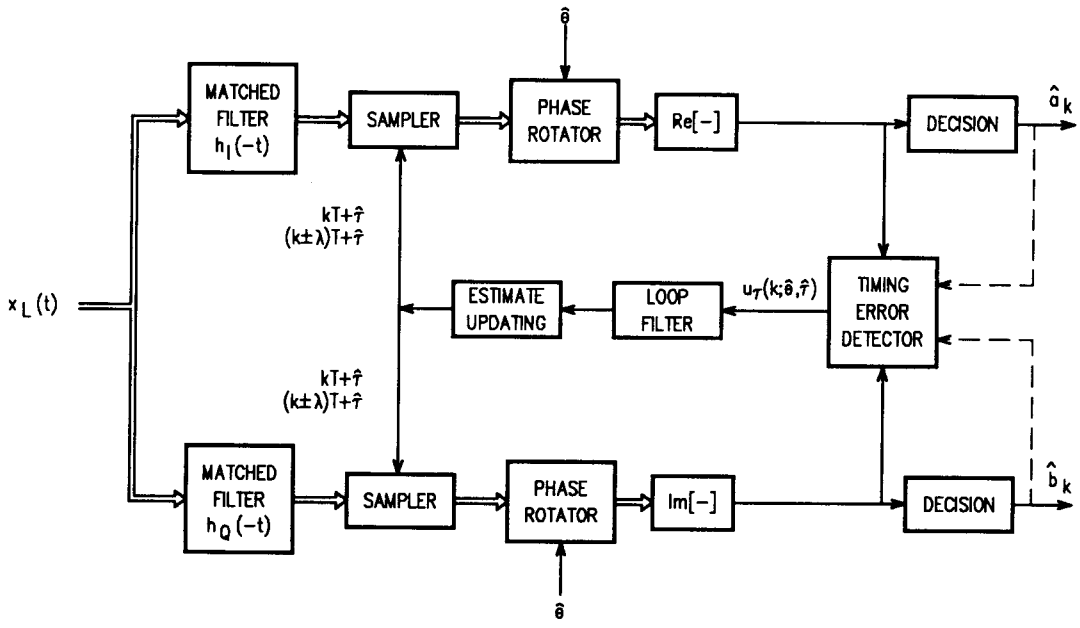


Fig. I.2–12: feedback symbol synchronization with approximate derivative matched filter (DDEL, NDAEL, DTTL, M&M, mM&M, GA)

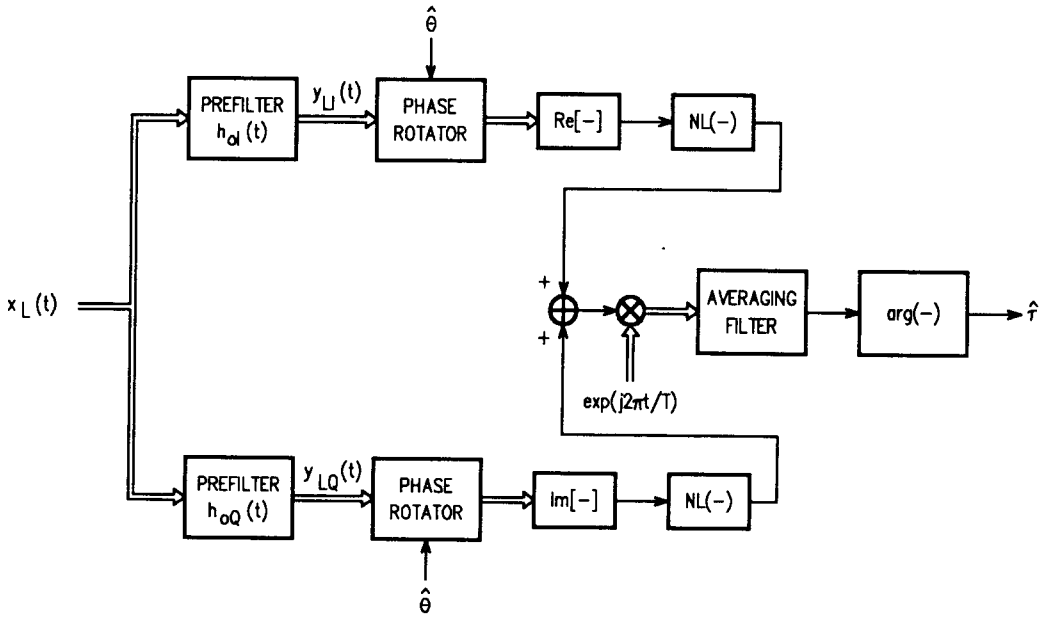


Fig. I.2–13: Non–search feedforward symbol synchronization (NDANLFF)

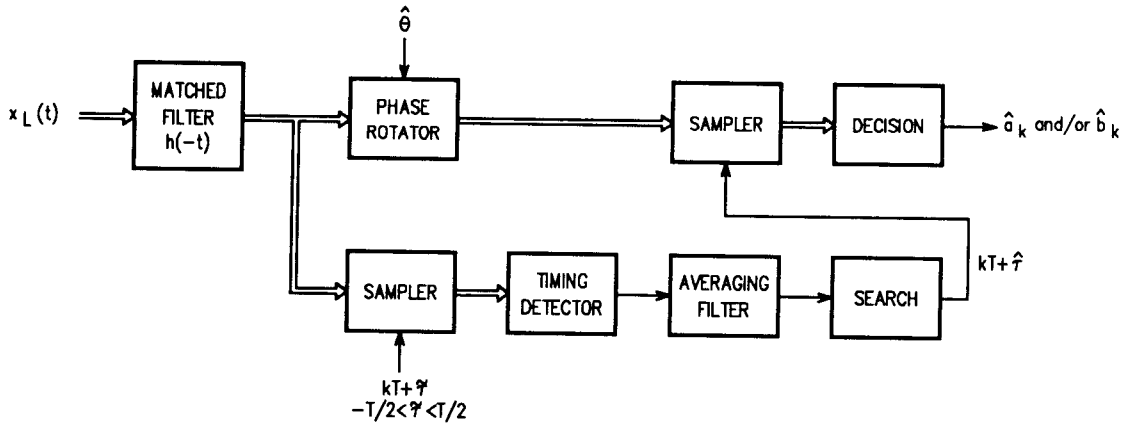


Fig. I.2–14: Carrier–independent feedforward (search) symbol synchronization (NDAMLFFCB/Q)

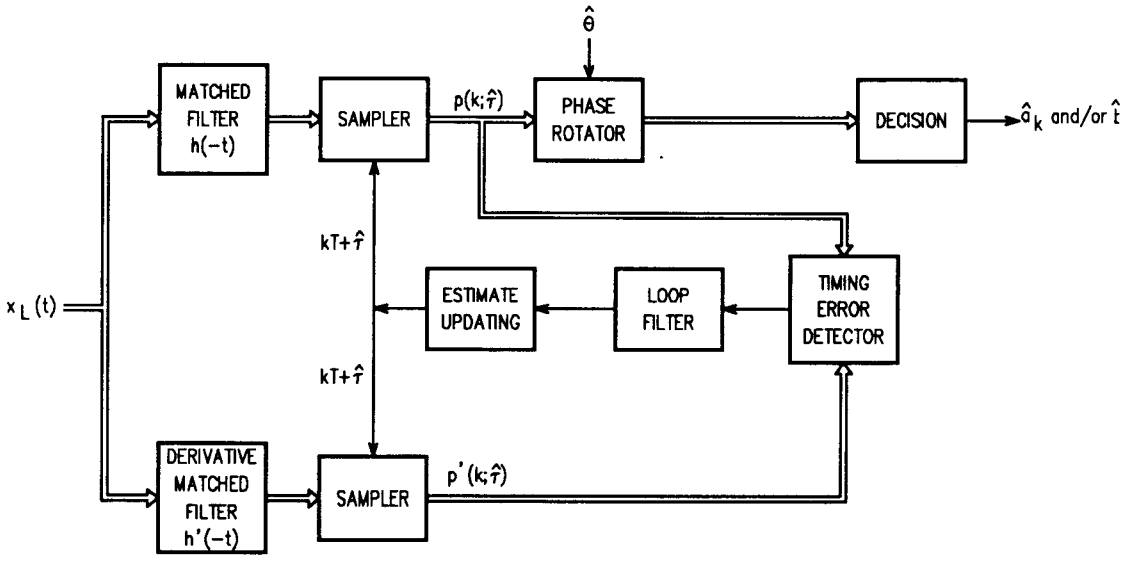


Fig. I.2–15: Carrier–independent feedback symbol synchronization with derivative matched filter (NDAMLFB$\overline{C}$B/Q)

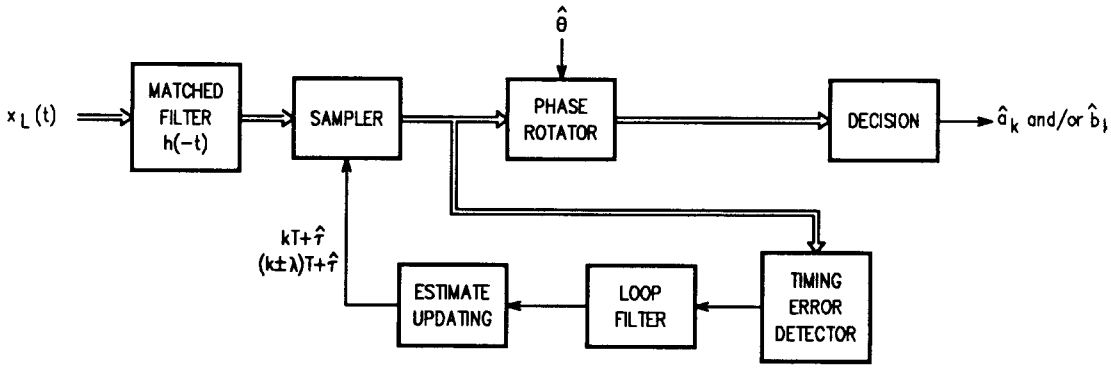


Fig. I.2–16: Carrier–independent feedback symbol synchronization without derivative matched filter (NDAEL(=1/2) $\overline{C}$B/Q, GA$\overline{C}$B/Q, NDANLFB$\overline{C}$B/Q)

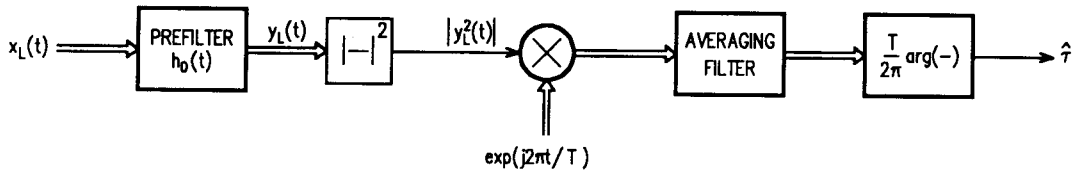


Fig. I.2–17: carrier– independent non–search feedforward symbol synchronization (NDASQFFC̄B/Q)

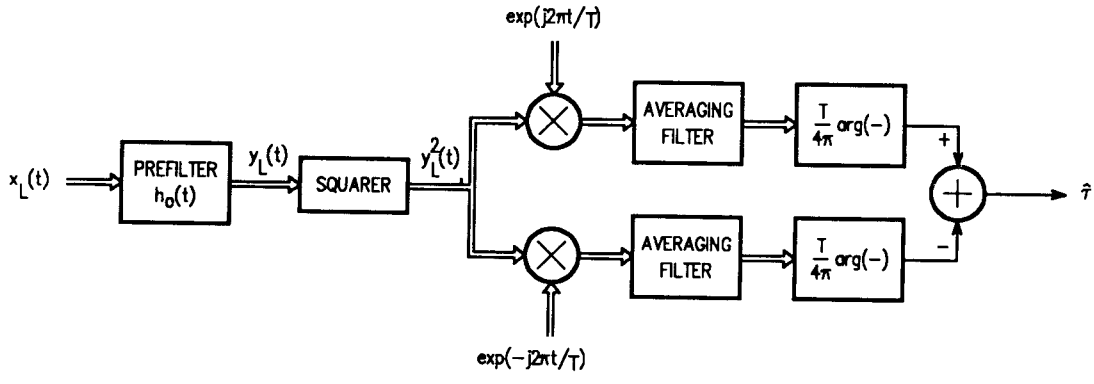


Fig. I.2–18 : Carrier independent non–search feedforward symbol synchronization (NDASQFFC̄O)

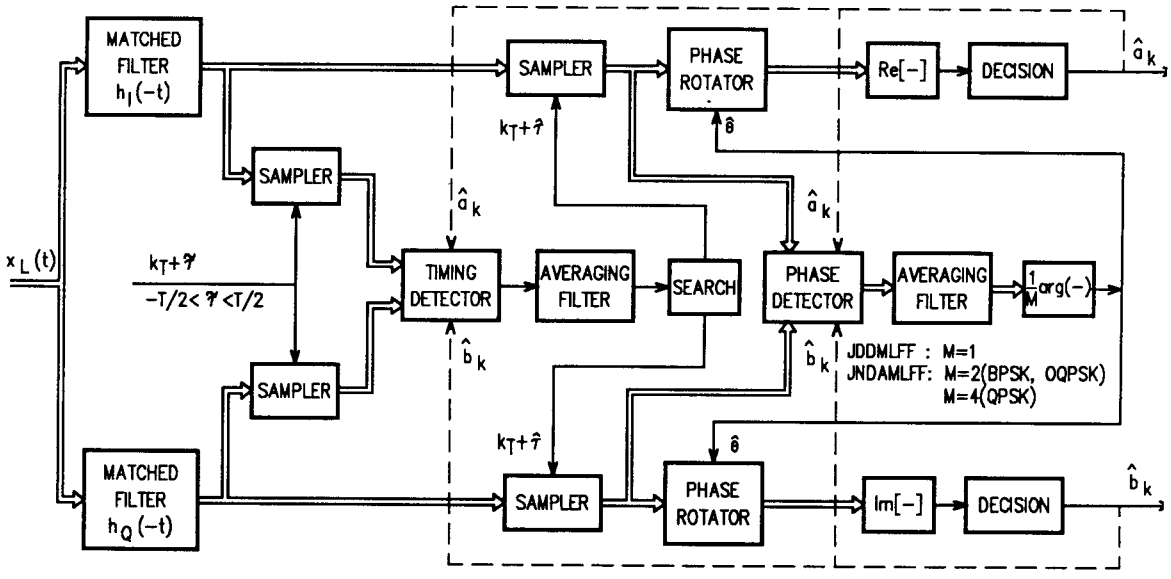


Fig. I.2–19: joint carrier and symbol synchronization (JDDMLFF, JNDAMLFF)

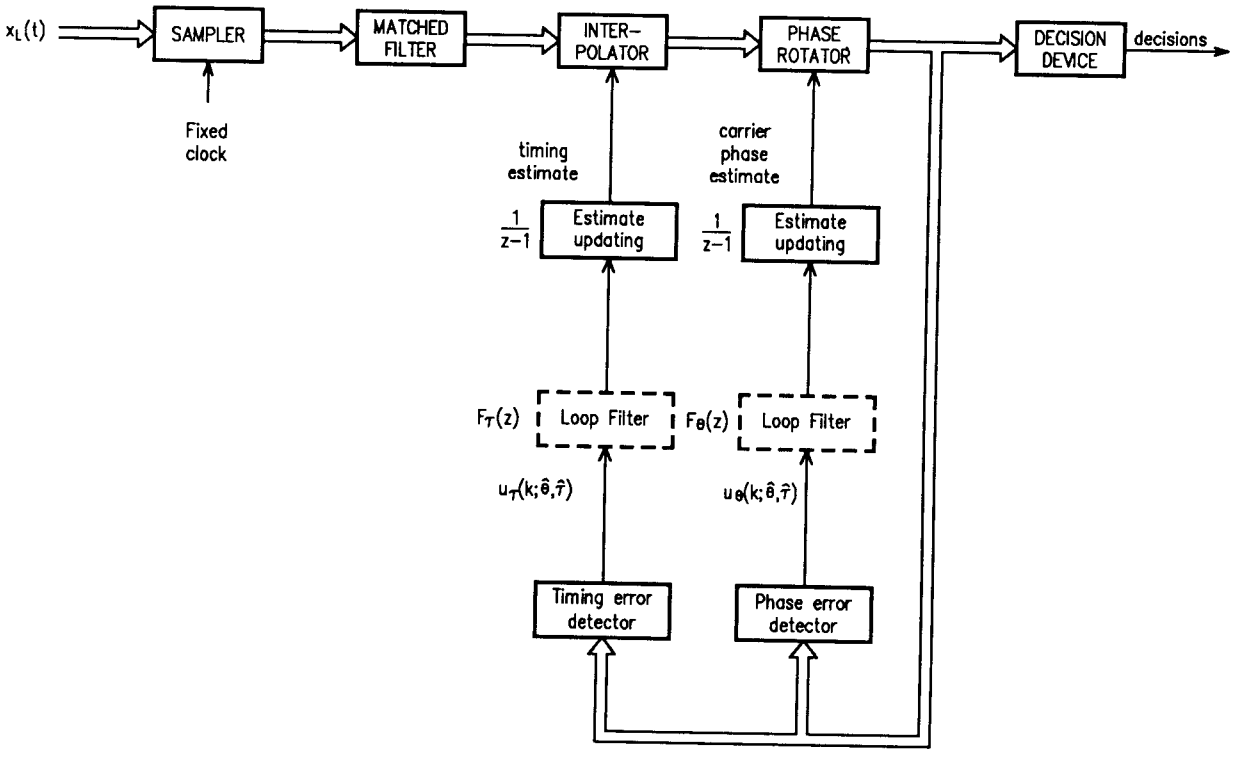


Fig. I.3–1 : Example of digital receiver structure (frequency correction not shown)

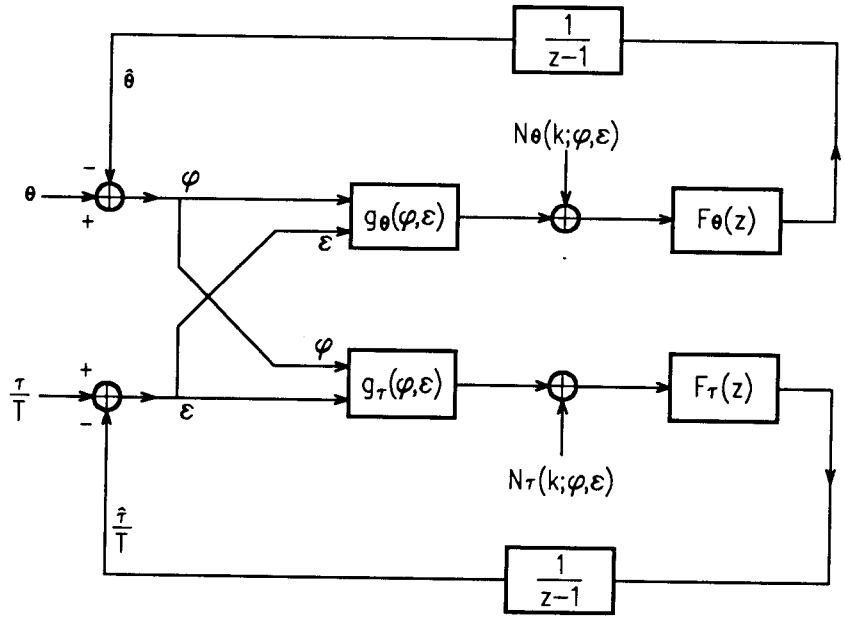


Fig. I.3–2: Equivalent feedback synchronizer model

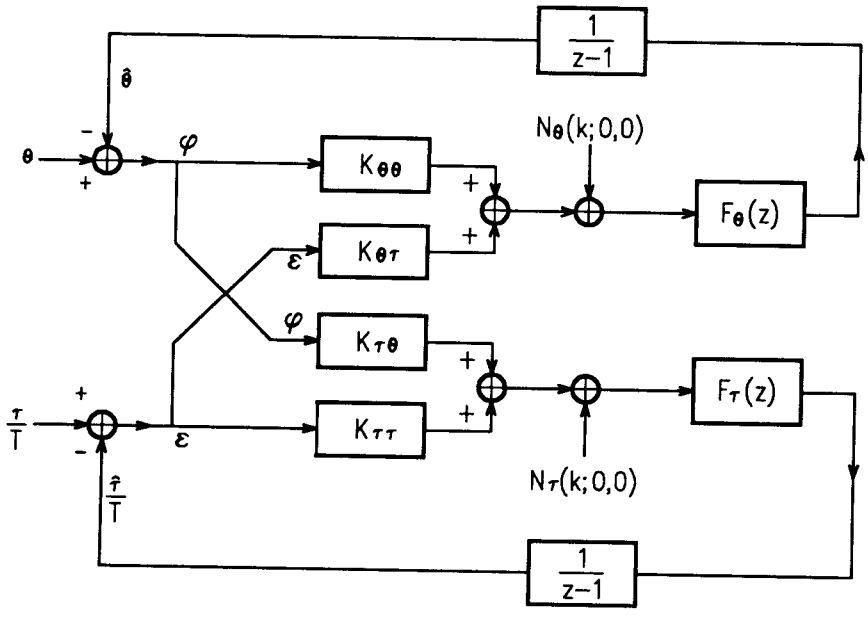


Fig.1.3–4: Linearized equivalent model

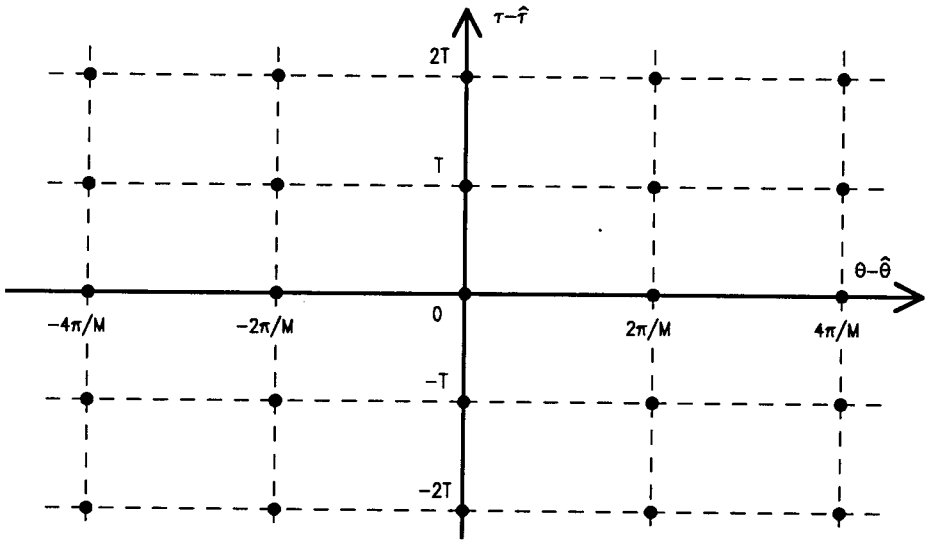


Fig. I.3–3a : equilibrium points for BPSK (M=2) and QPSK (M=4)

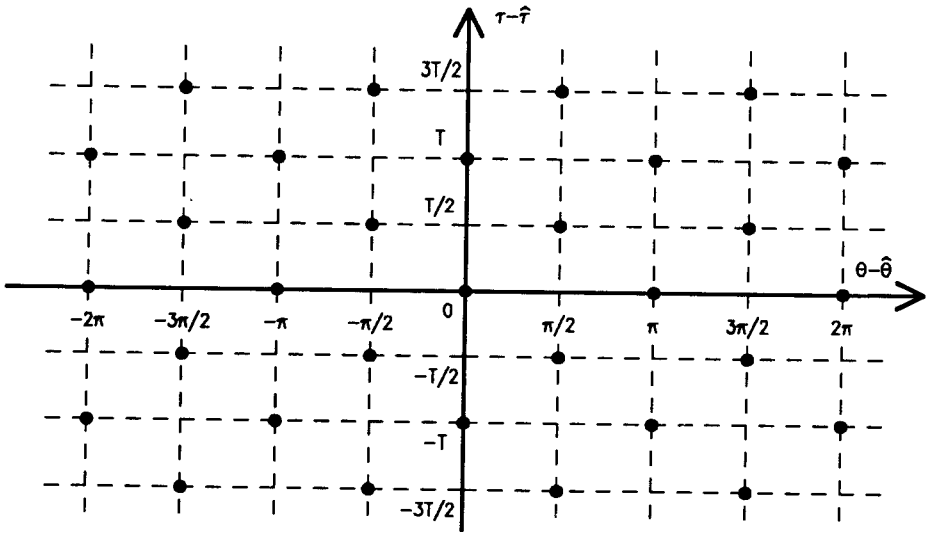


Fig. I.3–3b : equilibrium points for OQPSK

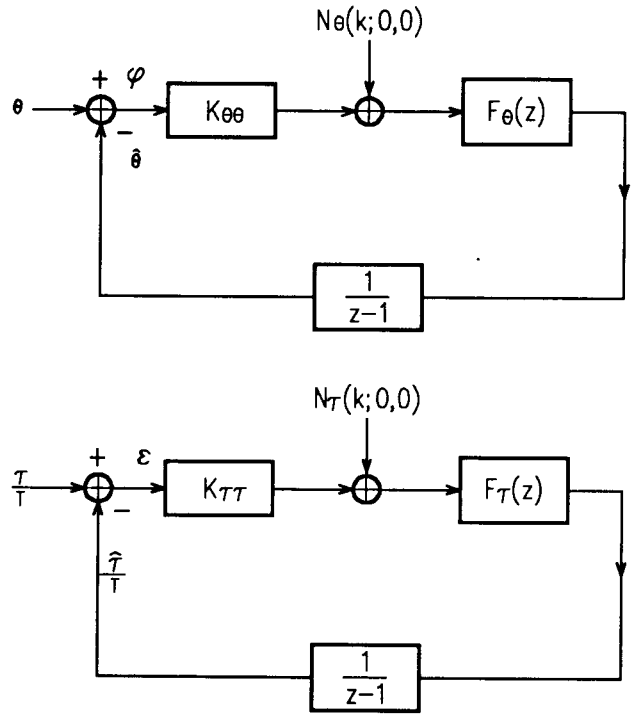


Fig. I.3–5 : Linearized equivalent model for $K_{\theta T}=K_{T\theta}=0$

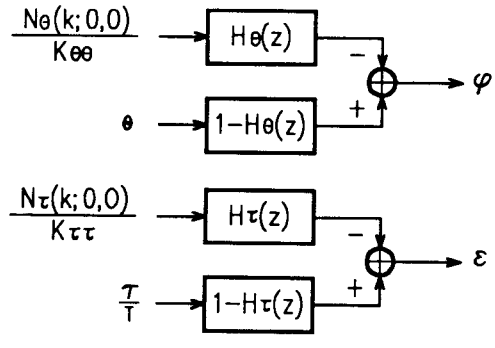


Fig. I.3–6: Simplified linear equivalent model for $K_{\theta T}=K_{T\theta}=0$

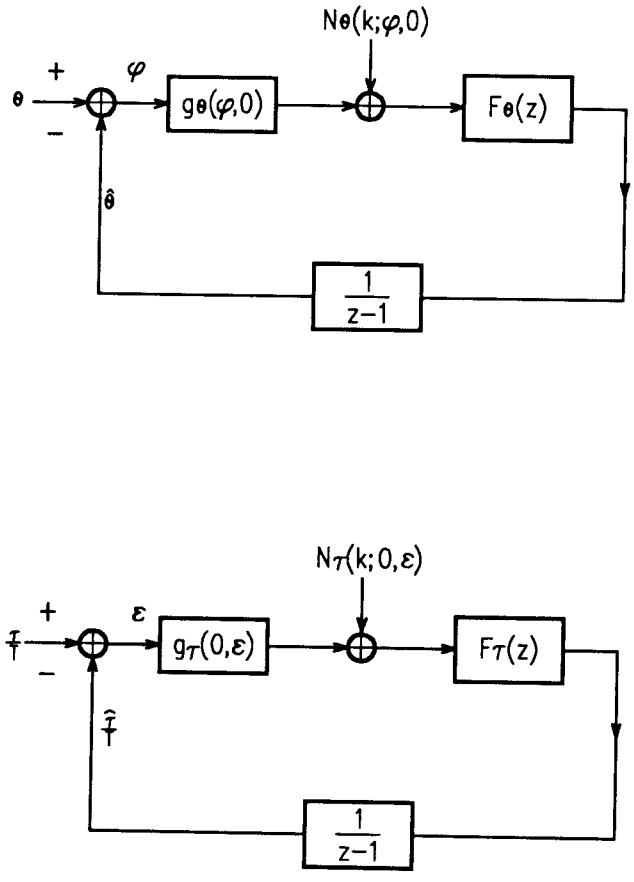


Fig.I.3–7 : Equivalent model for study of cycle slips

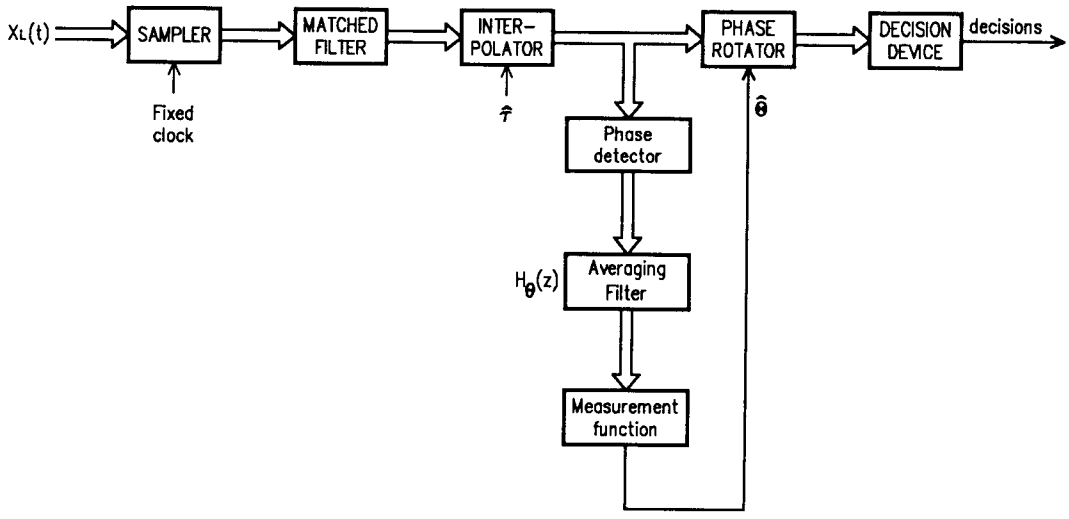


Fig. I.3-8 : Feedforward carrier synchronisation

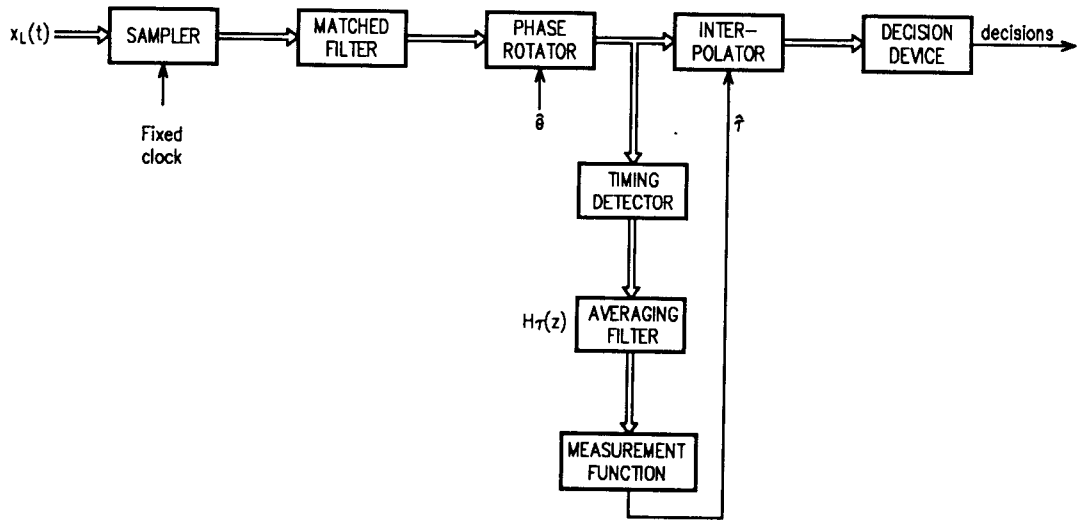


Fig. I.3–9 : Feedforward symbol synchronizer (non–search)

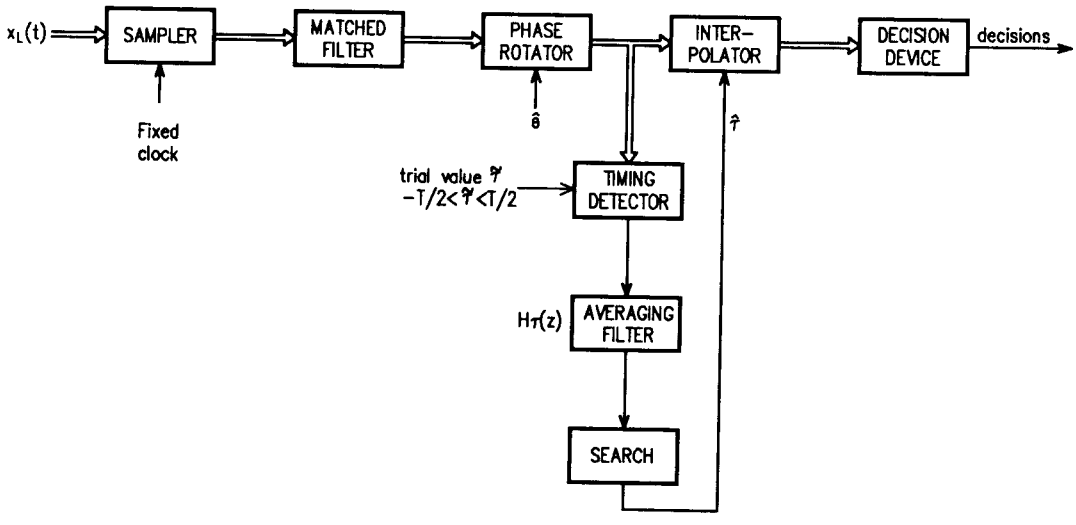


Fig. I.3–10 : Feedforward symbol synchronizer (search)

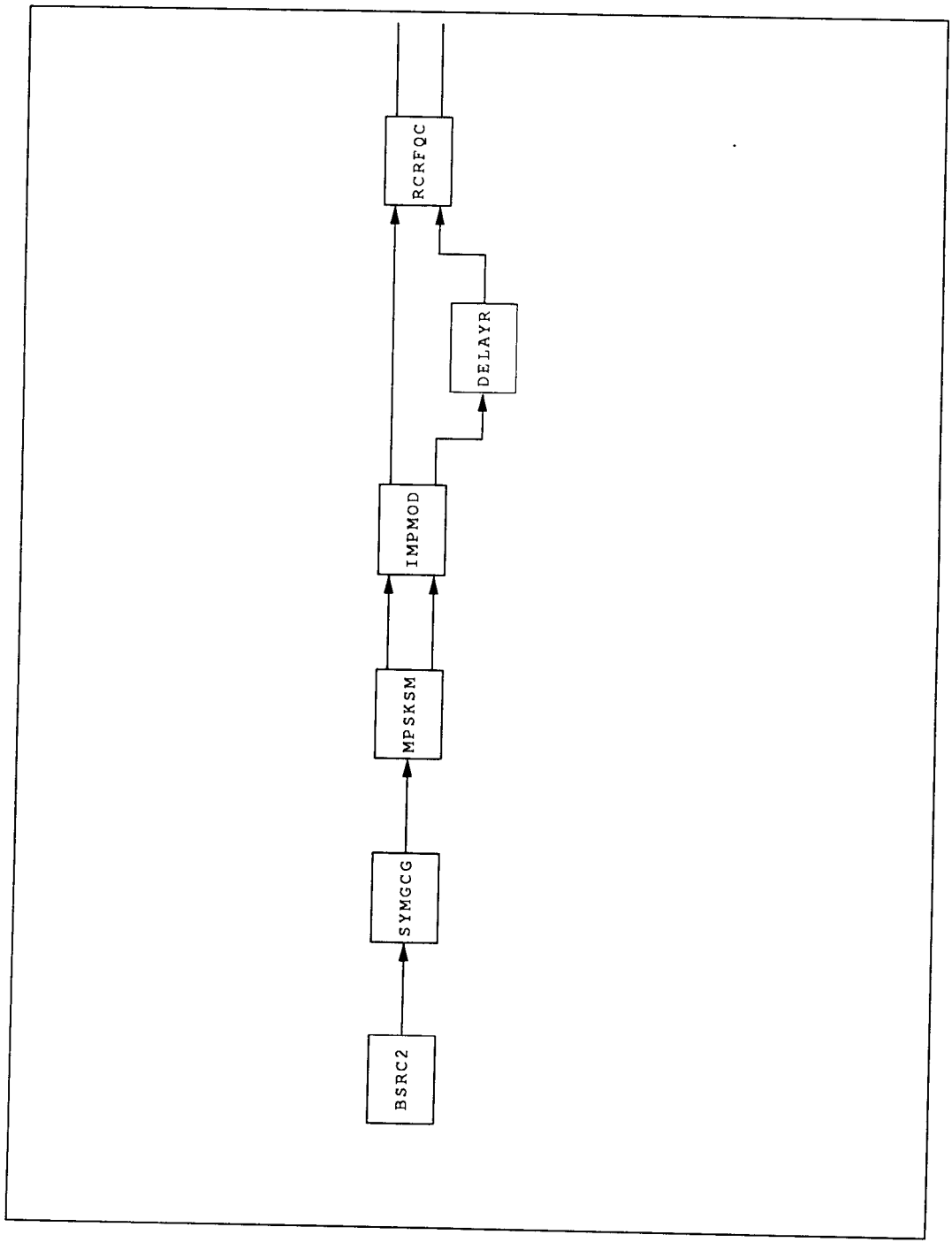


Fig. I.4-1: Transmit model

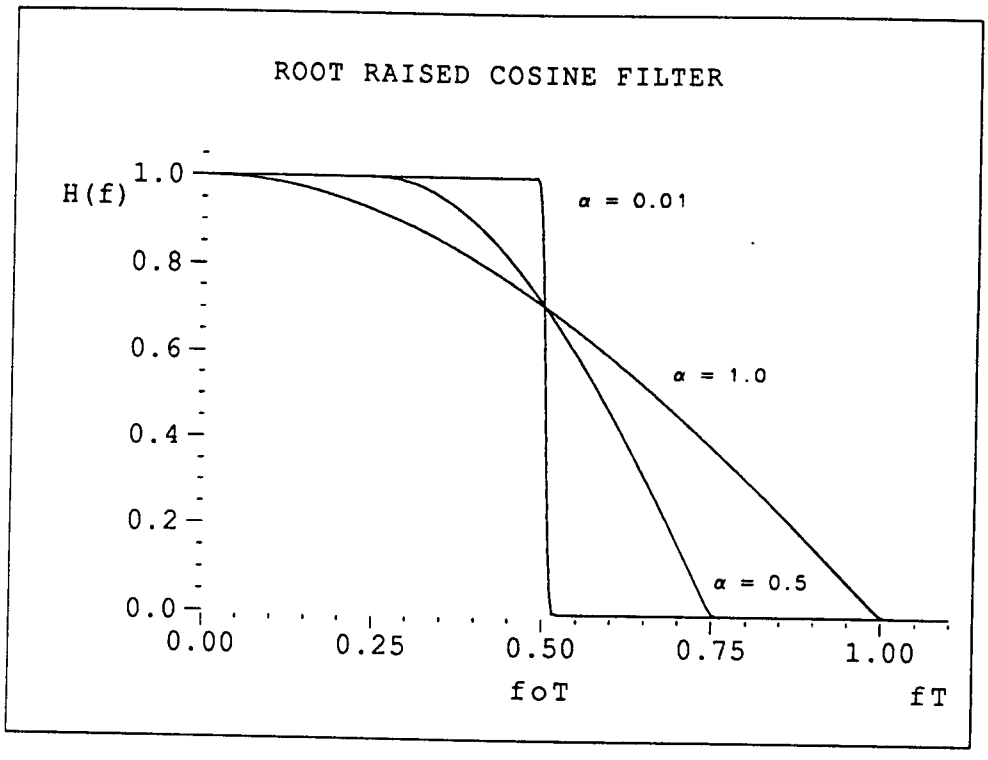


Fig. I.4-2: Transfer function of root raised cosine filter with rolloff (α) as parameter

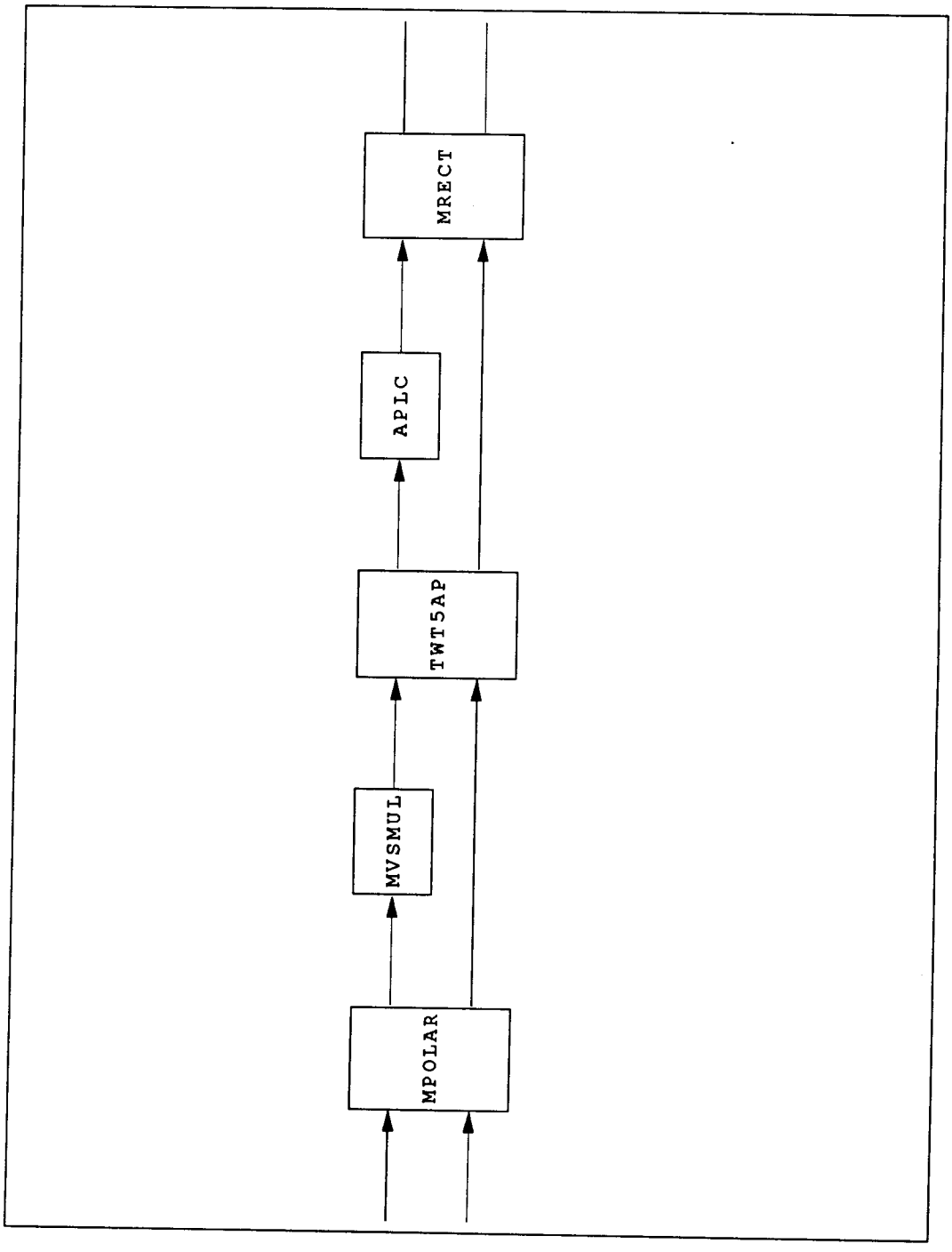
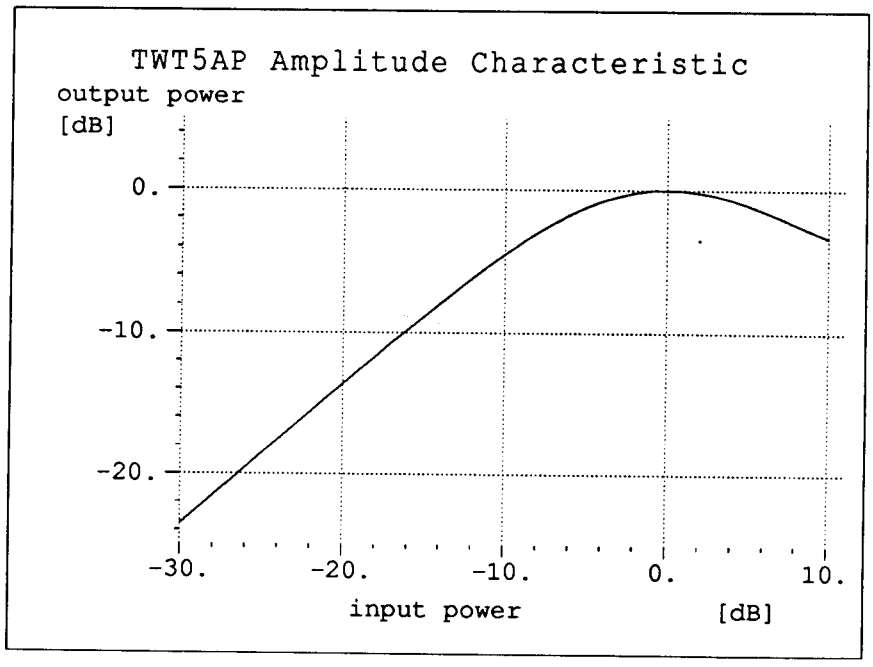


Fig. I.4-3: Travelling Wave Tube Amplifier (TWTA) model



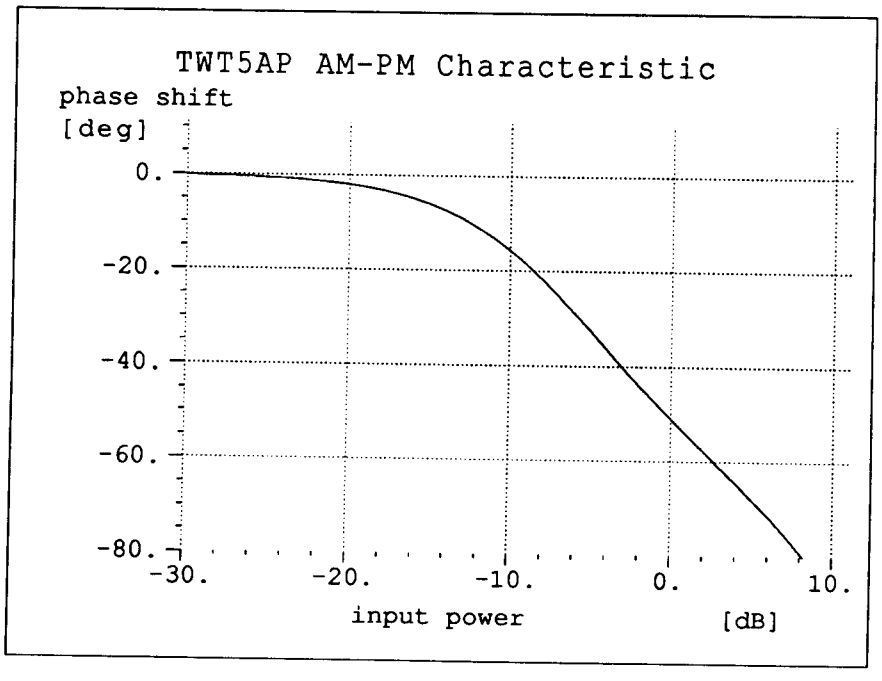


Fig. I.4-4: AM-AM and AM-PM conversion characteristics assumed in this study. Input and output power are normalized to the corresponding saturation levels, i.e. saturation occurs at 0dB input power and 0dB output power

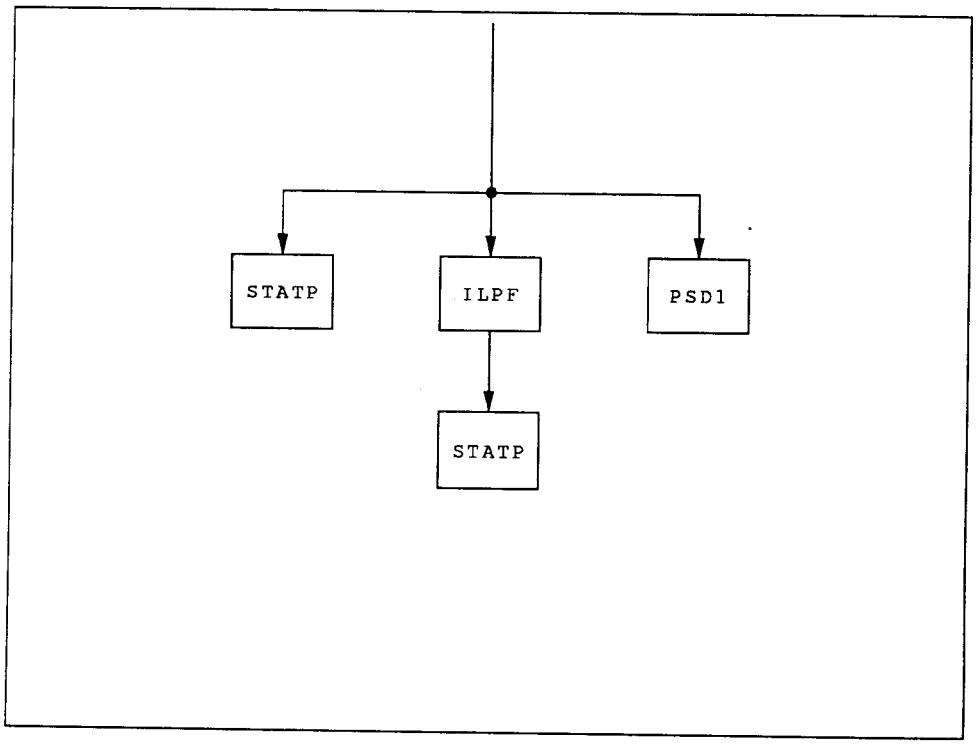


Fig. I.4-5: Statistical evaluation model

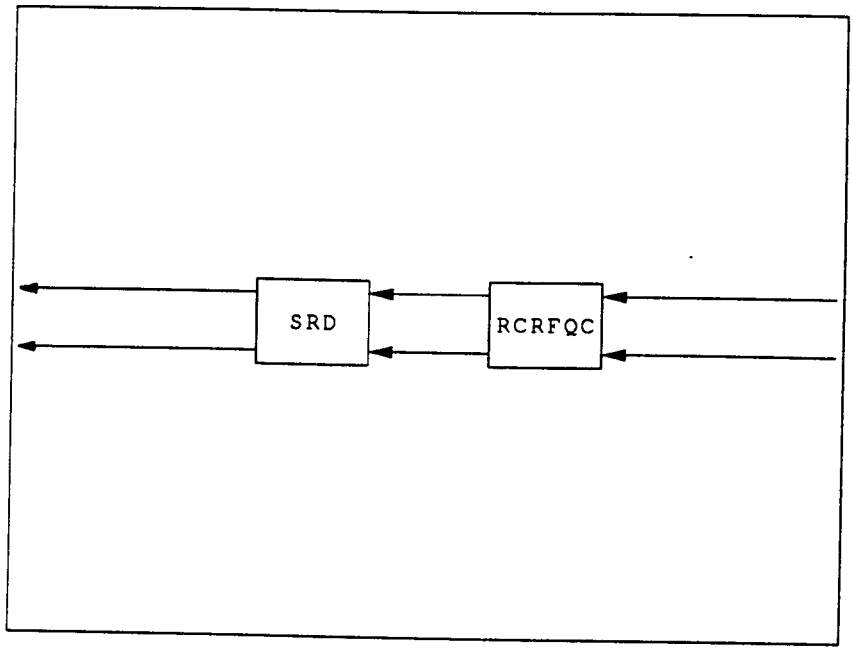


Fig. I.4-6a: Sampled matched filter

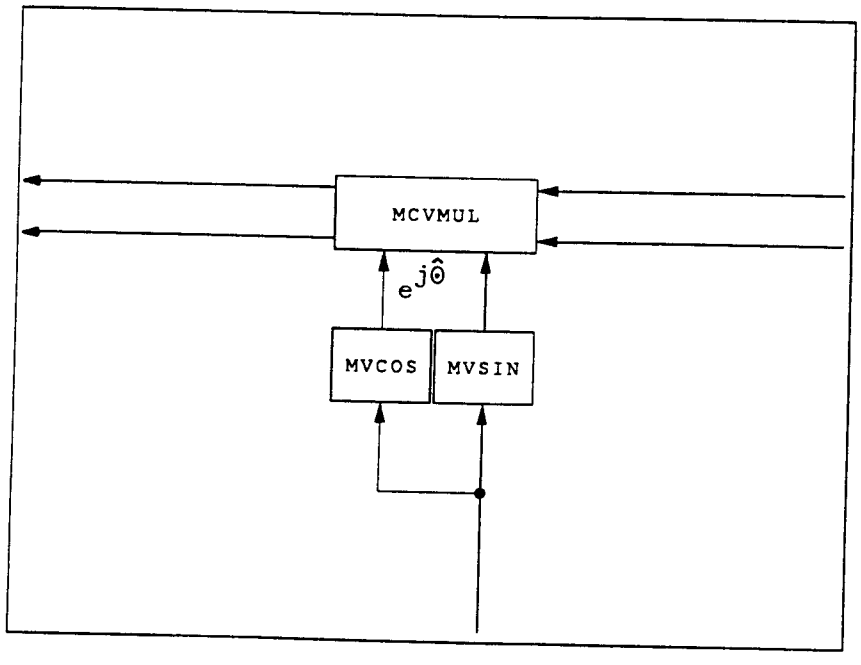


Fig. I.4-6b: Phase rotator

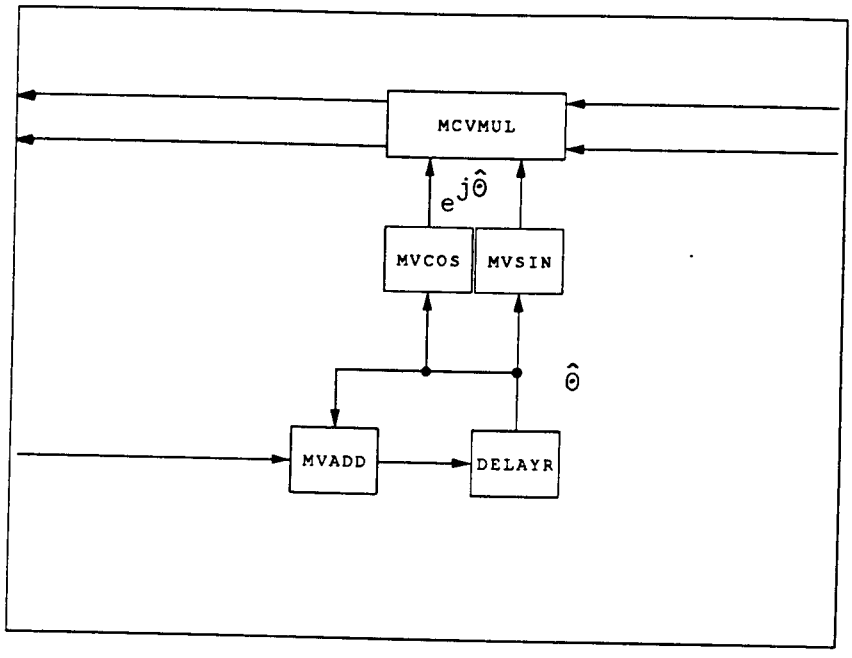


Fig. I.4-6c: Integrator and phase rotator

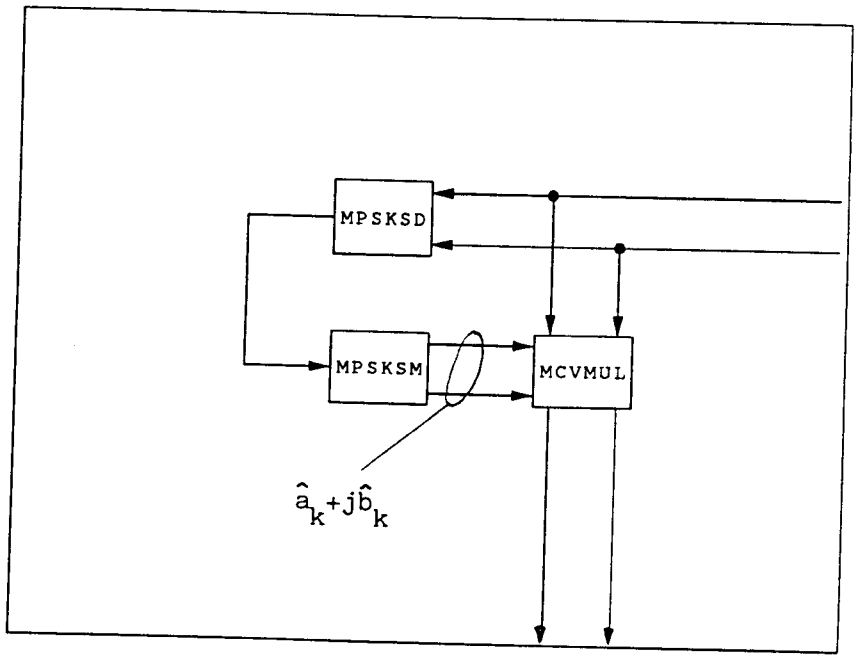


Fig. I.4-6d: Decision Feedback

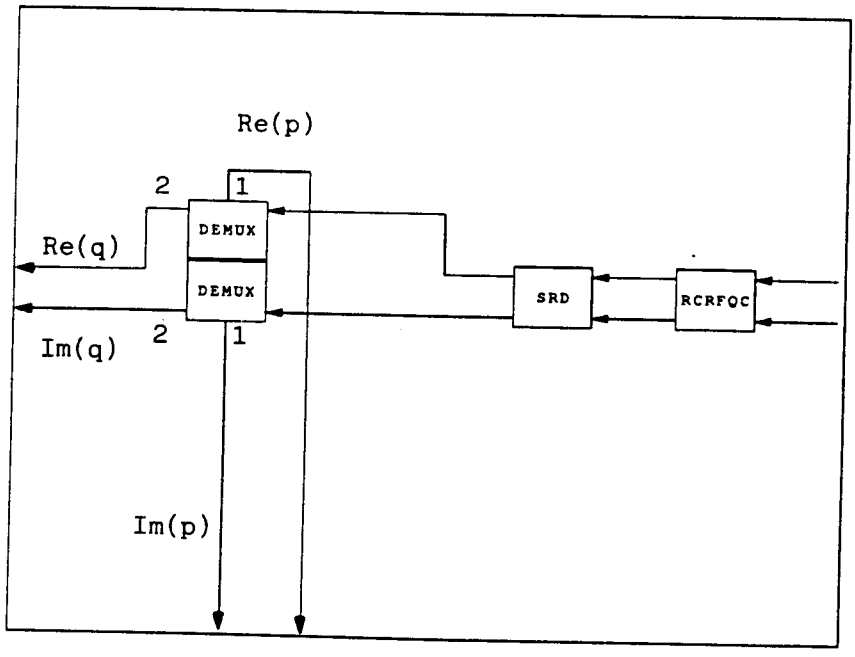


Fig. I.4-6e: Separation of p(k;τ) and q(k;τ)

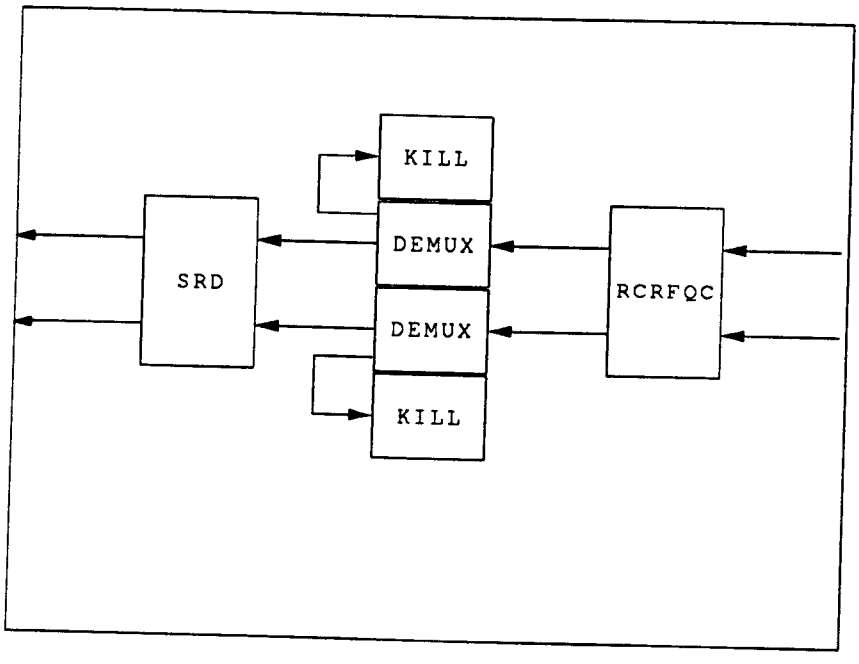


Fig. I.4-6f: Clock frequency deviation

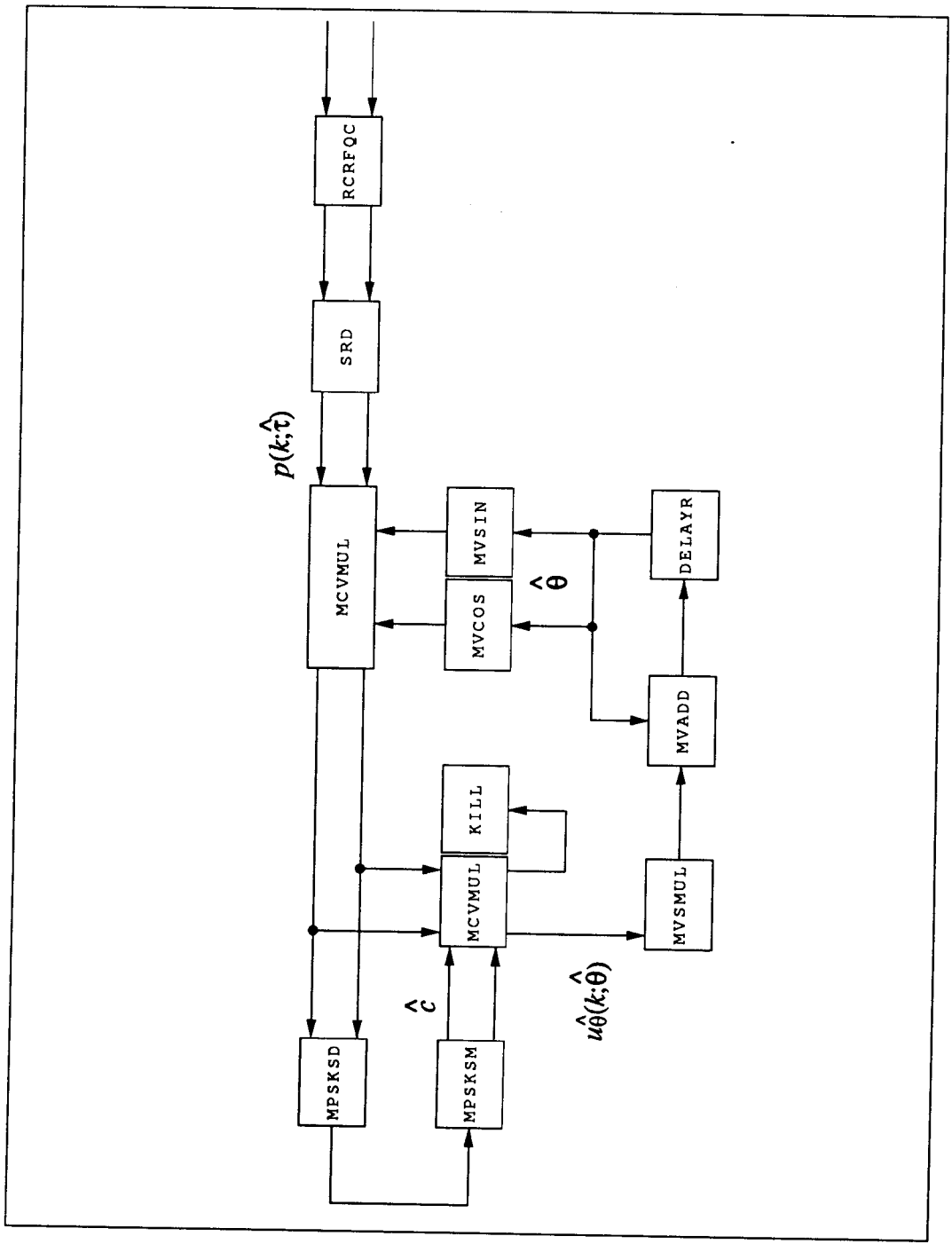


Fig. I.4-7a: DDMLFB, Decision Directed Maximum Likelihood FeedBack carrier synchronizer (BPSK, QPSK)

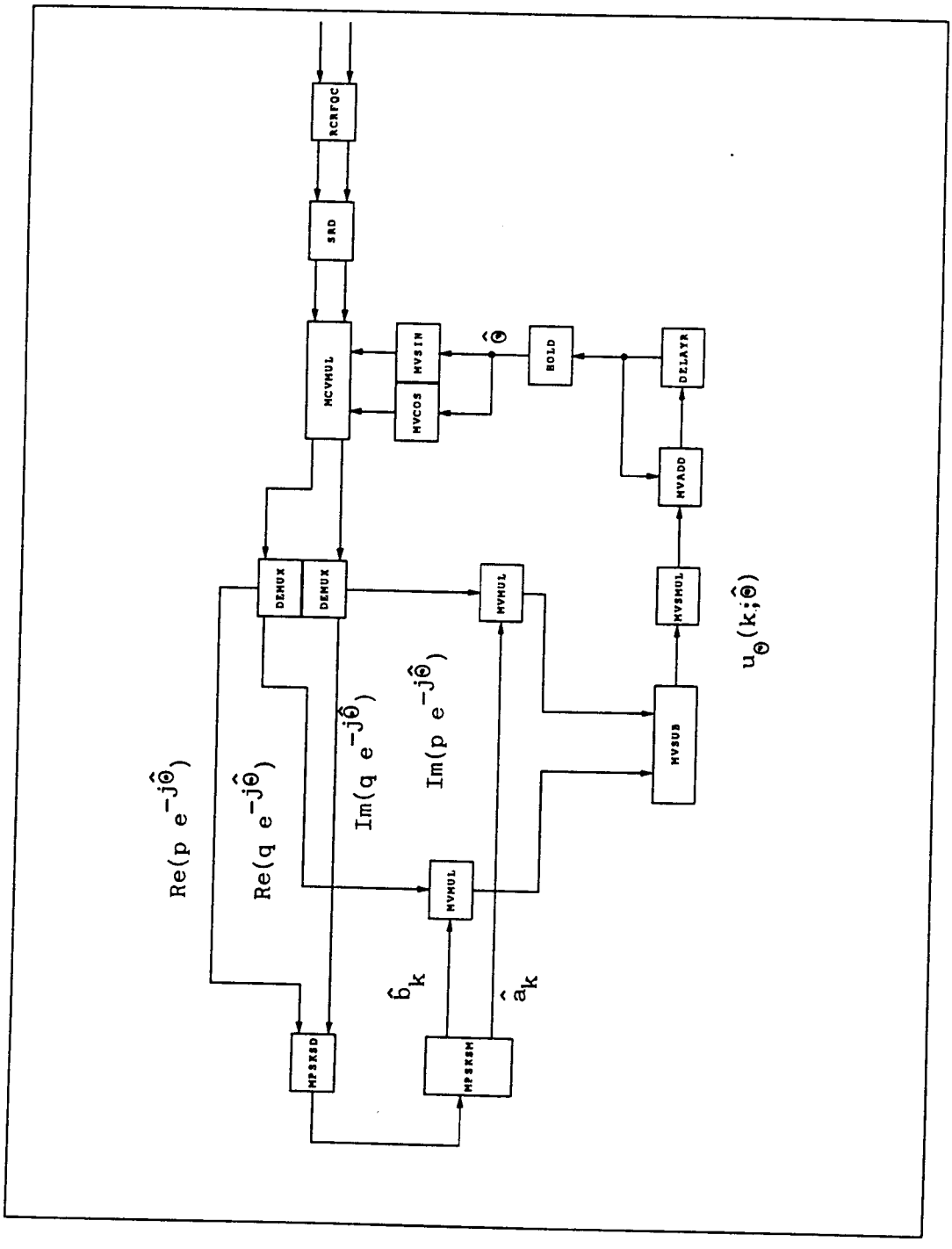


Fig. I.4-7b: DDMLFB, Decision Directed Maximum Likelihood FeedBack carrier synchronizer (OQPSK)

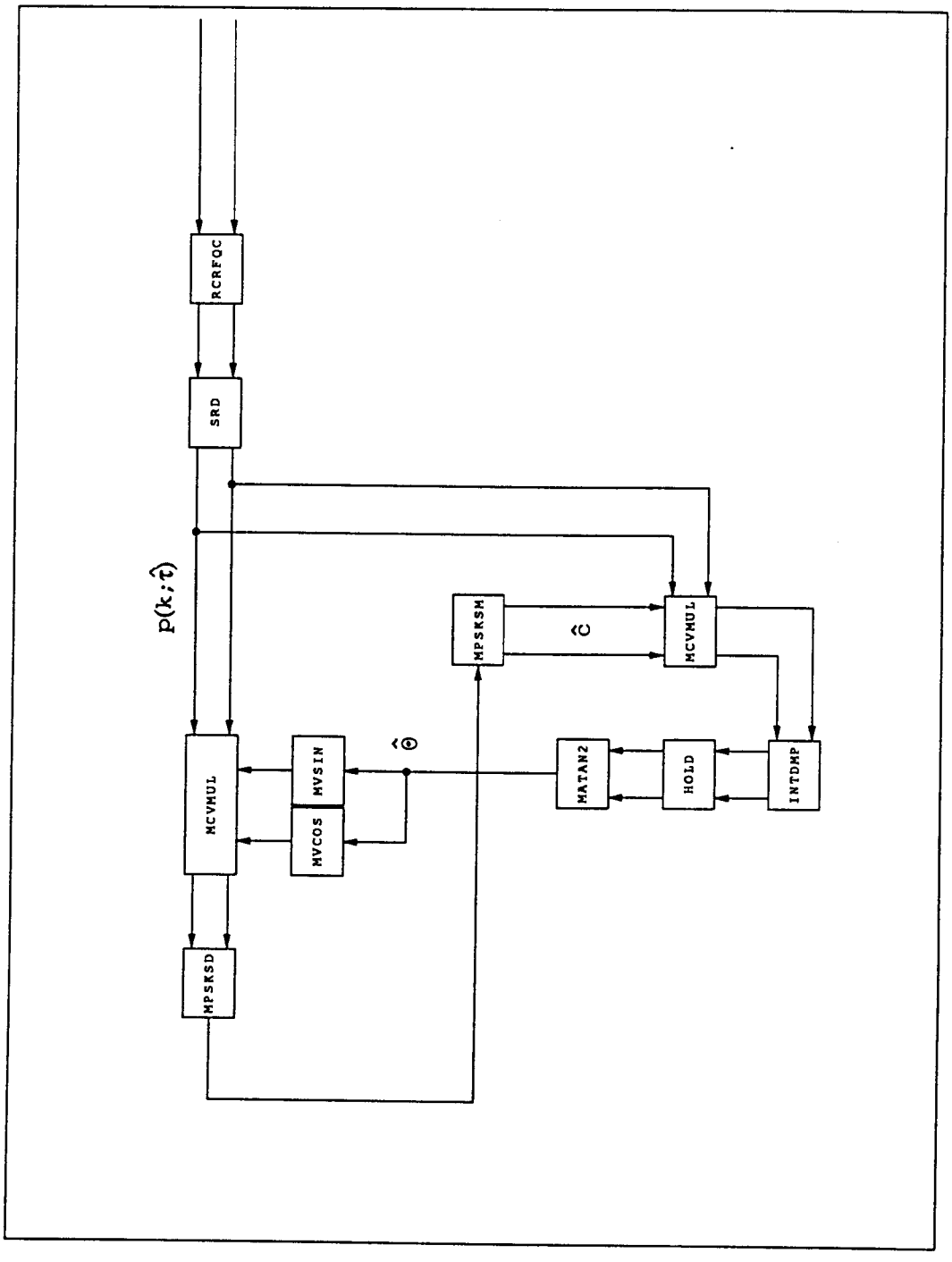


Fig. I.4-8a: DDMLFF, Decision Directed Maximum Likelihood FeedForward carrier synchronizer, θ (BPSK, QPSK)

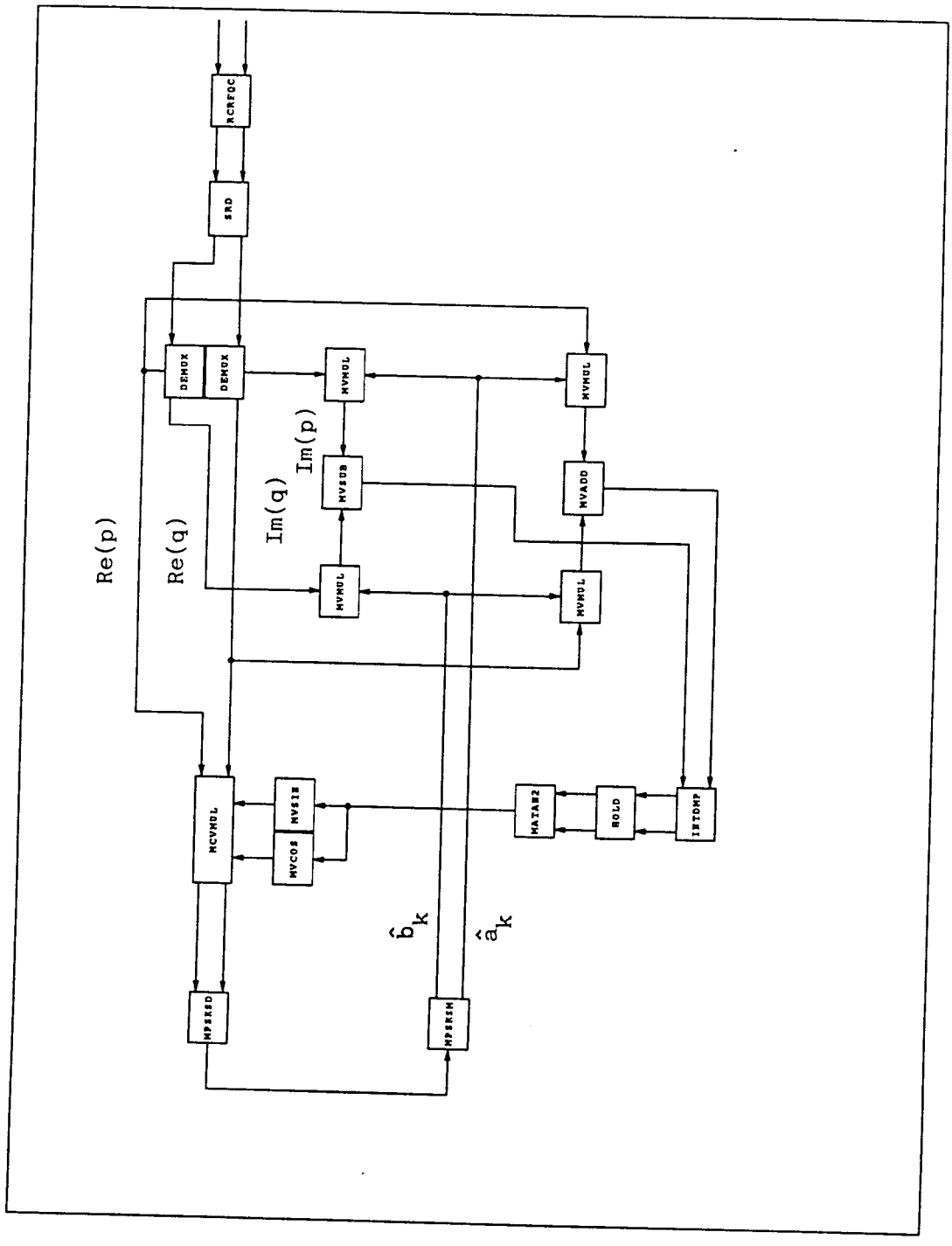


Fig. I.4-8b: DDMLFF, Decision Directed Maximum Likelihood FeedForward carrier synchronizer, θ (OQPSK)

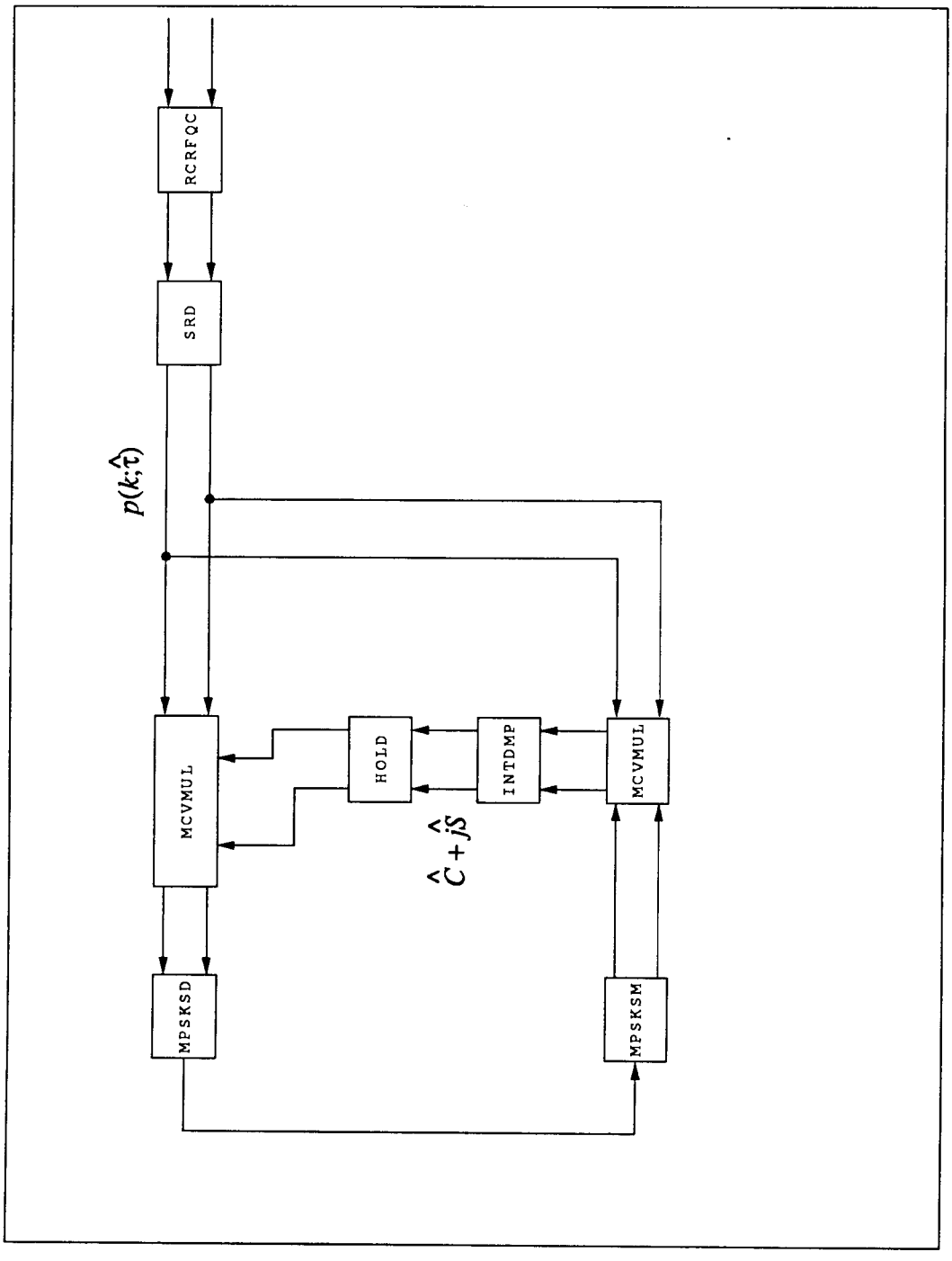


Fig. I.4-9a: DDMLFF, Decision Directed Maximum Likelihood FeedForward carrier synchronizer, C+jS (BPSK, QPSK)

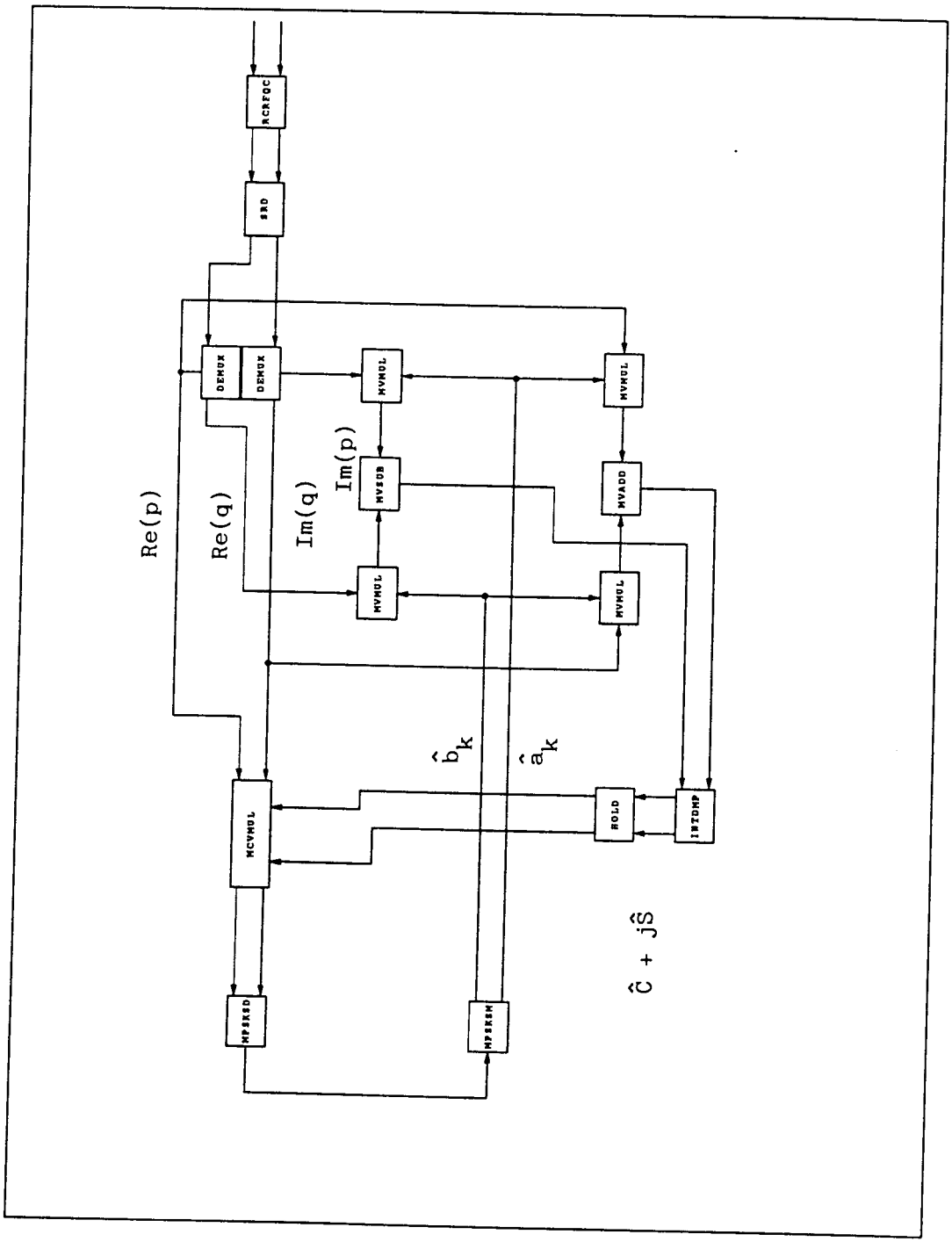


Fig. I.4-9b: DDMLFF, Decision Directed Maximum Likelihood FeedForward carrier synchronizer, C+jS (OQPSK)

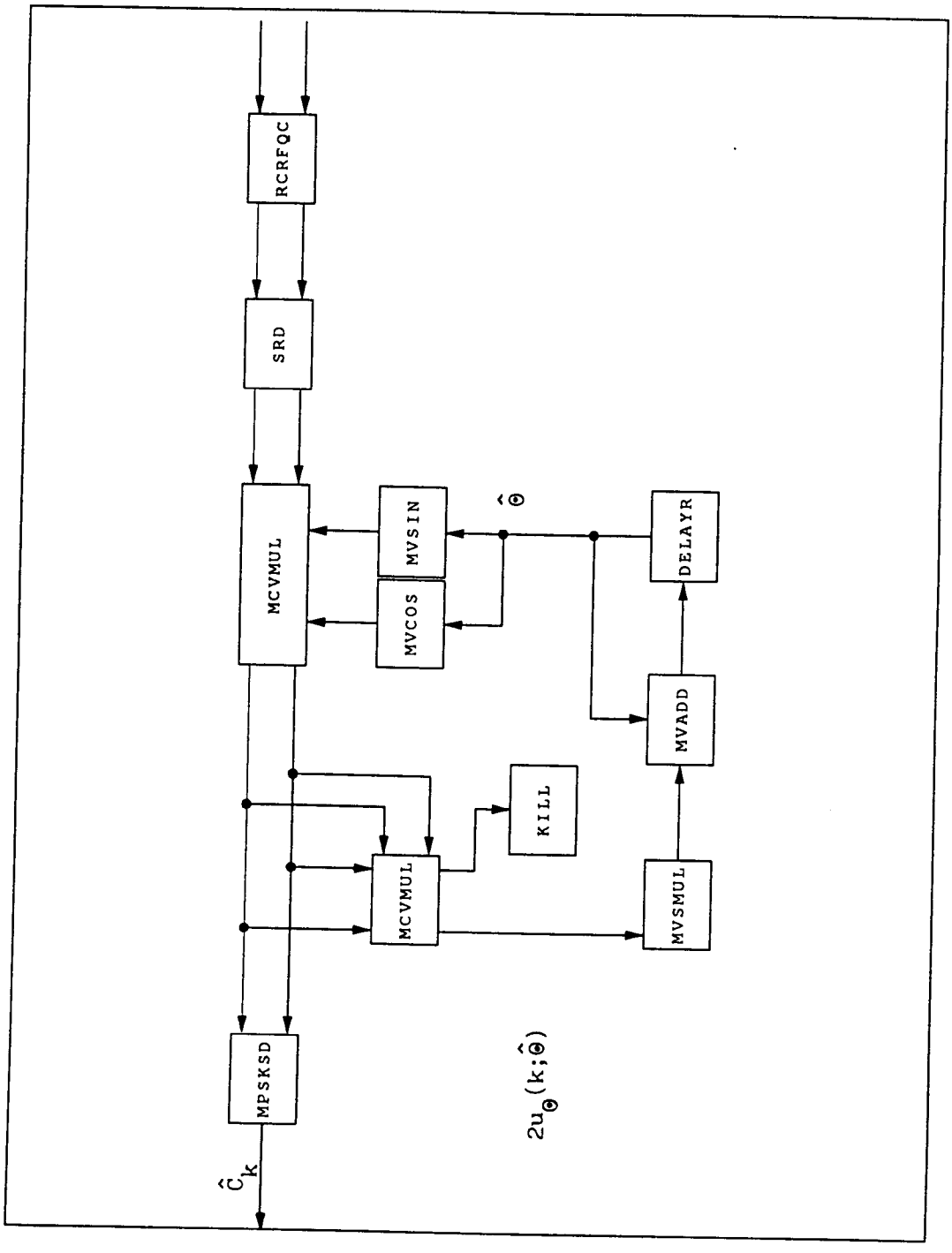


Fig. I.4-10a: NDAMLFB, Non Data-Aided Maximum Likelihood FeedBack carrier synchronizer (BPSK)

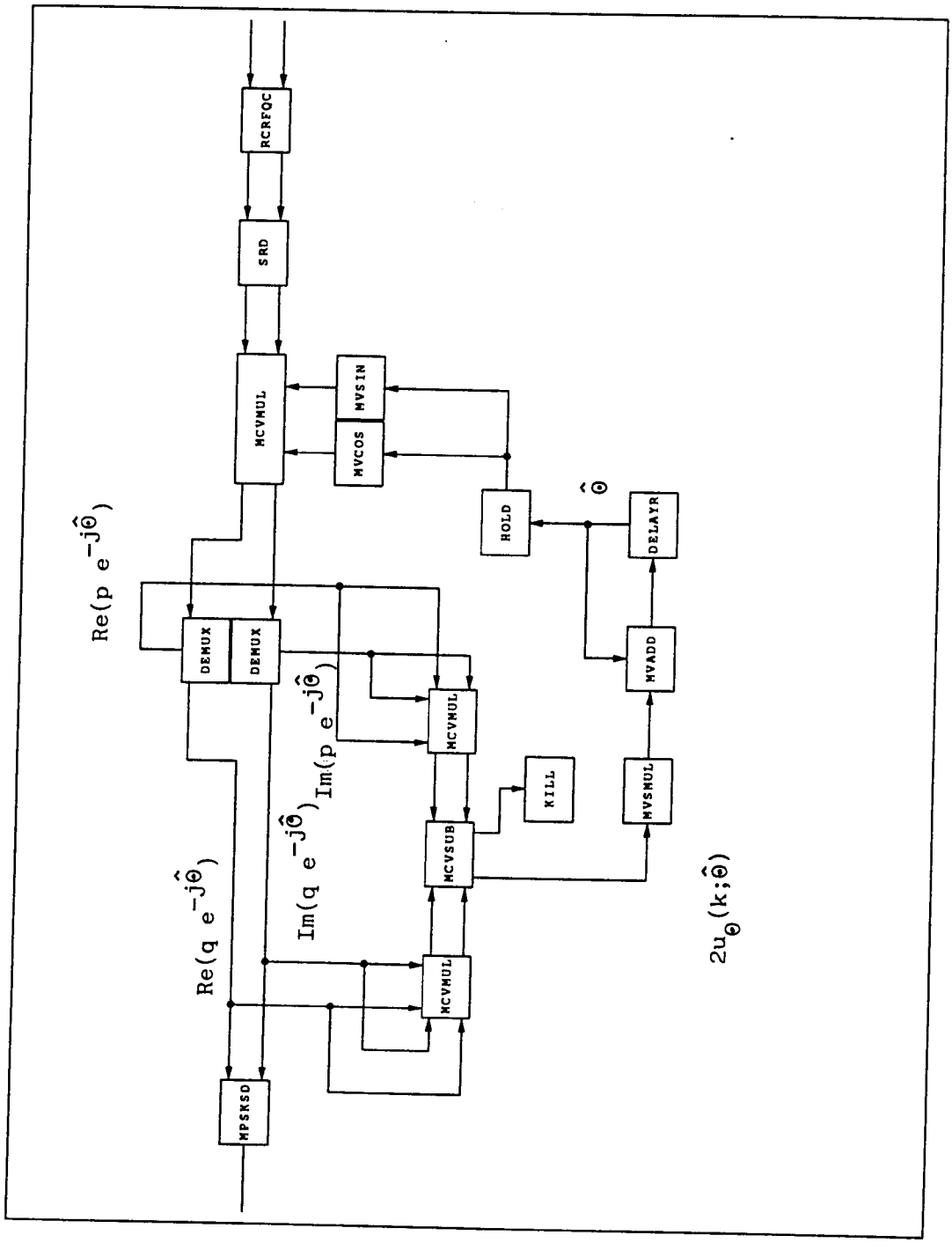


Fig. I.4-10b: NDAMLFB, Non Data-Aided Maximum Likelihood FeedBack carrier synchronizer (OQPSK)

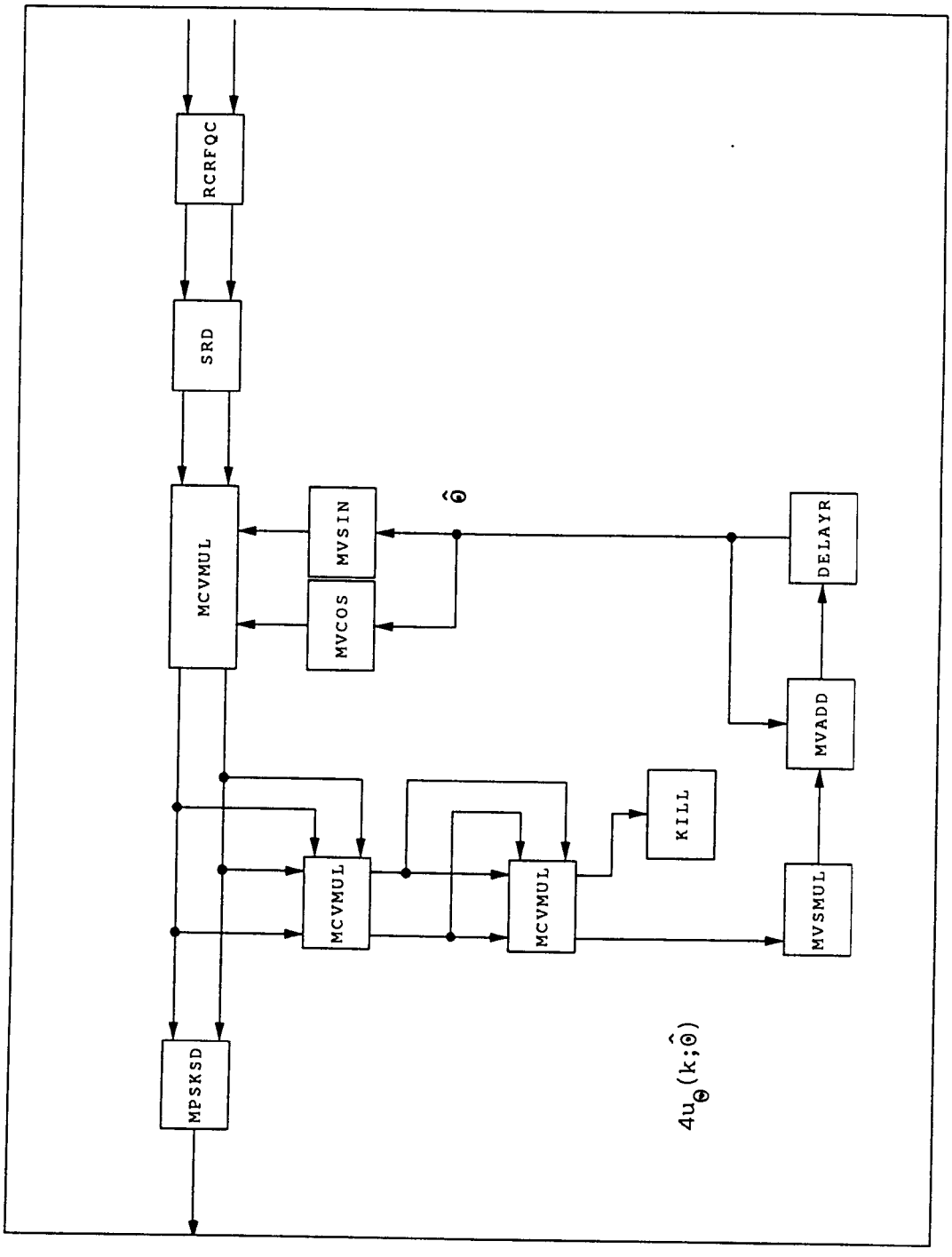


Fig. I.4-10c: NDAMLFB, Non Data-Aided Maximum Likelihood FeedBack carrier synchronizer (QPSK)

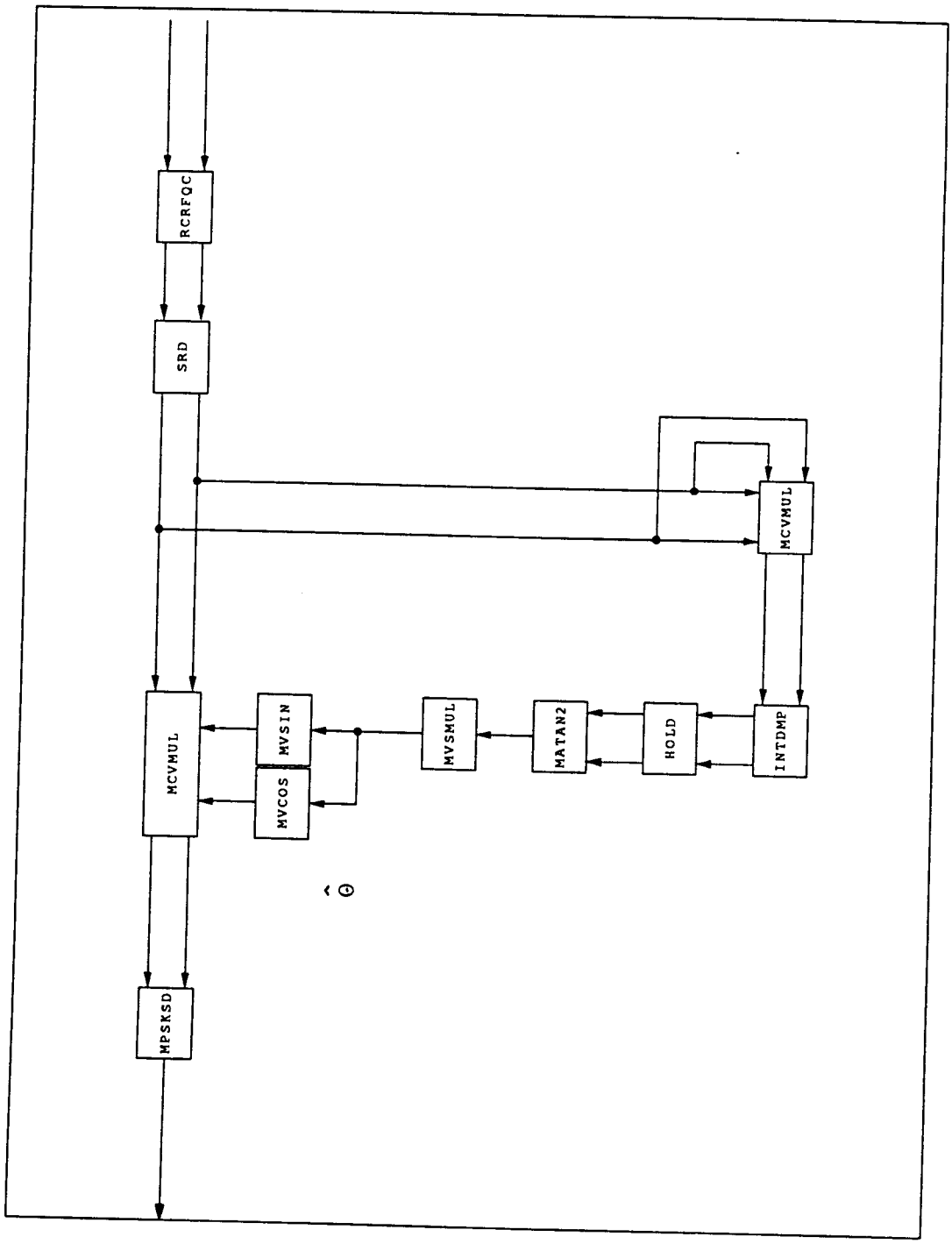


Fig. I.4-11a: NDAMLFF, Non Data-Aided Maximum Likelihood Feedforward carrier synchronizer (BPSK)

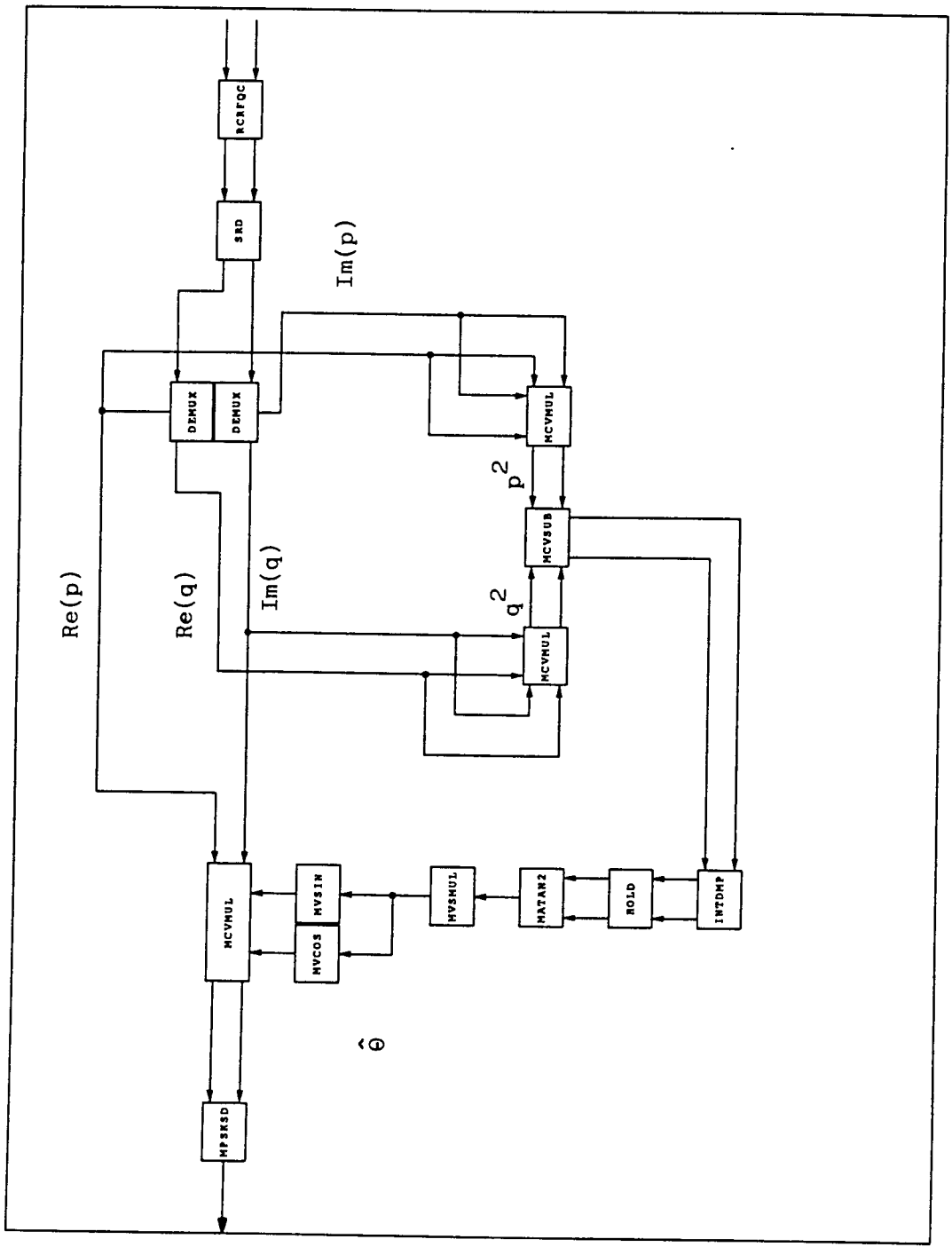


Fig. I.4-11b: NDAMLFF, Non Data-Aided Maximum Likelihood Feedforward carrier synchronizer (OQPSK)

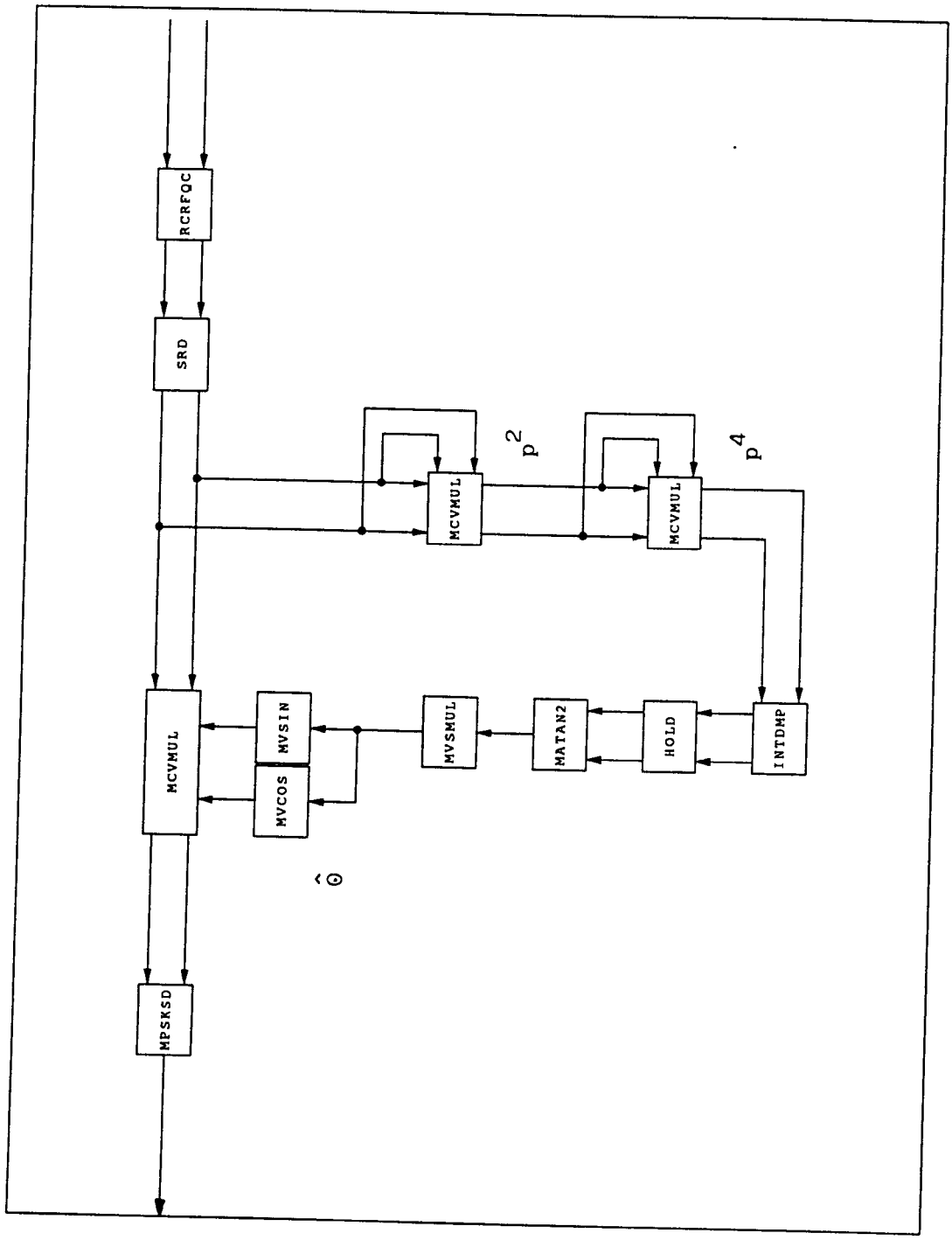


Fig. I.4-11c: NDAMLFF, Non Data-Aided Maximum Likelihood Feedforward carrier synchronizer (QPSK)

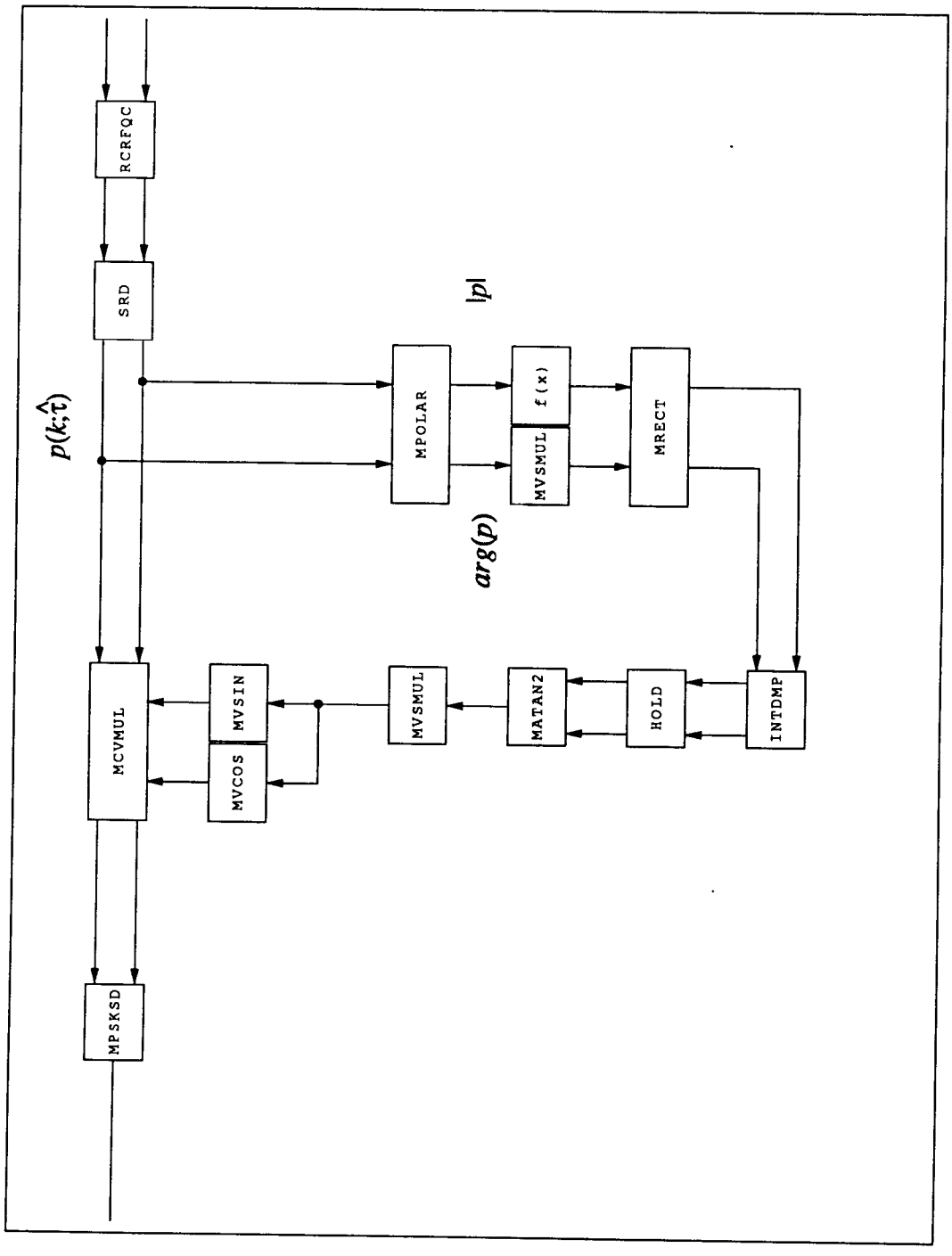


Fig. I.4-12a: V&VFF, Viterbi&Viterbi Feedforward carrier synchronizer for BPSK and QPSK

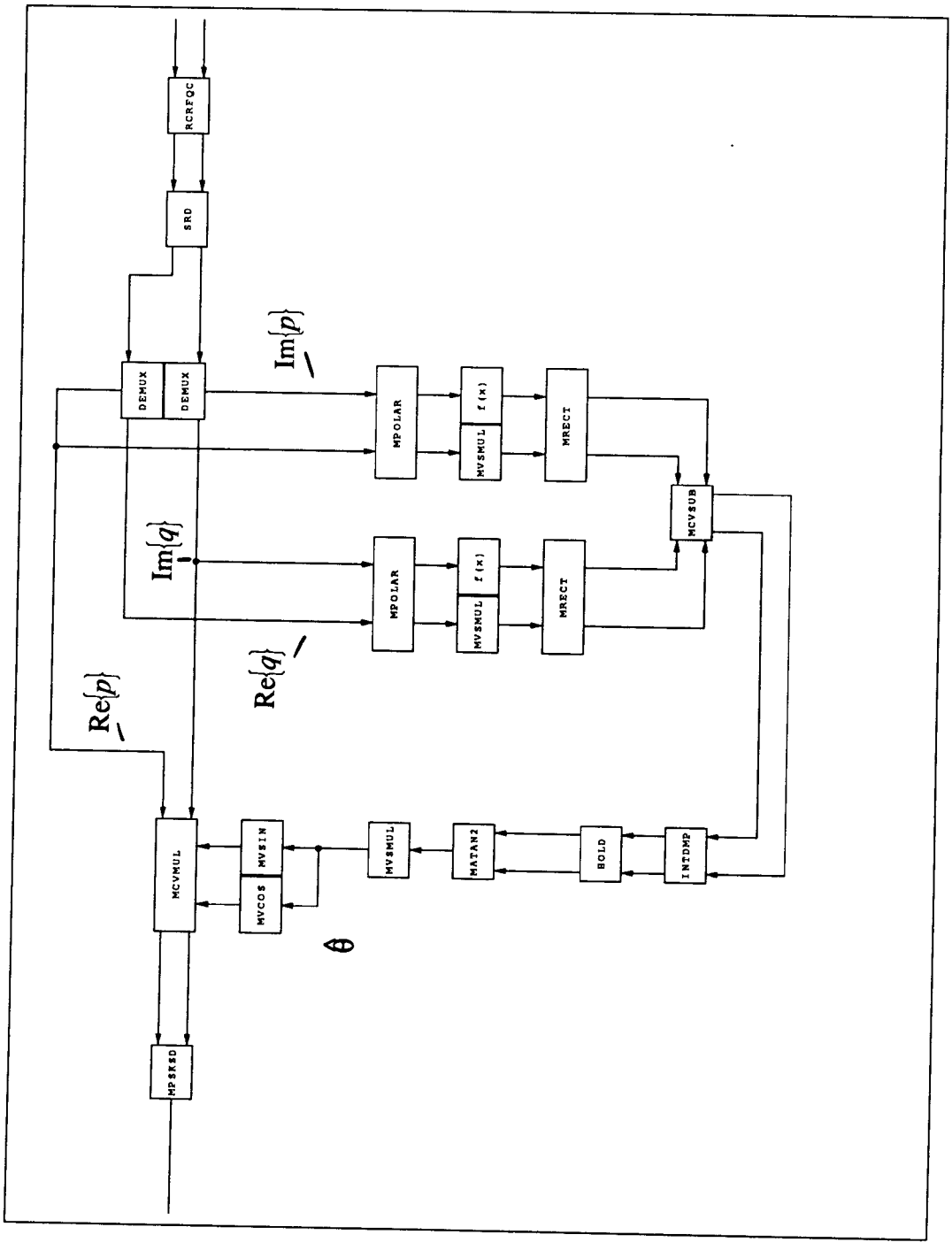


Fig. I.4-12b: V&VFF, Viterbi&Viterbi Feedforward carrier synchronizer for OQPSK

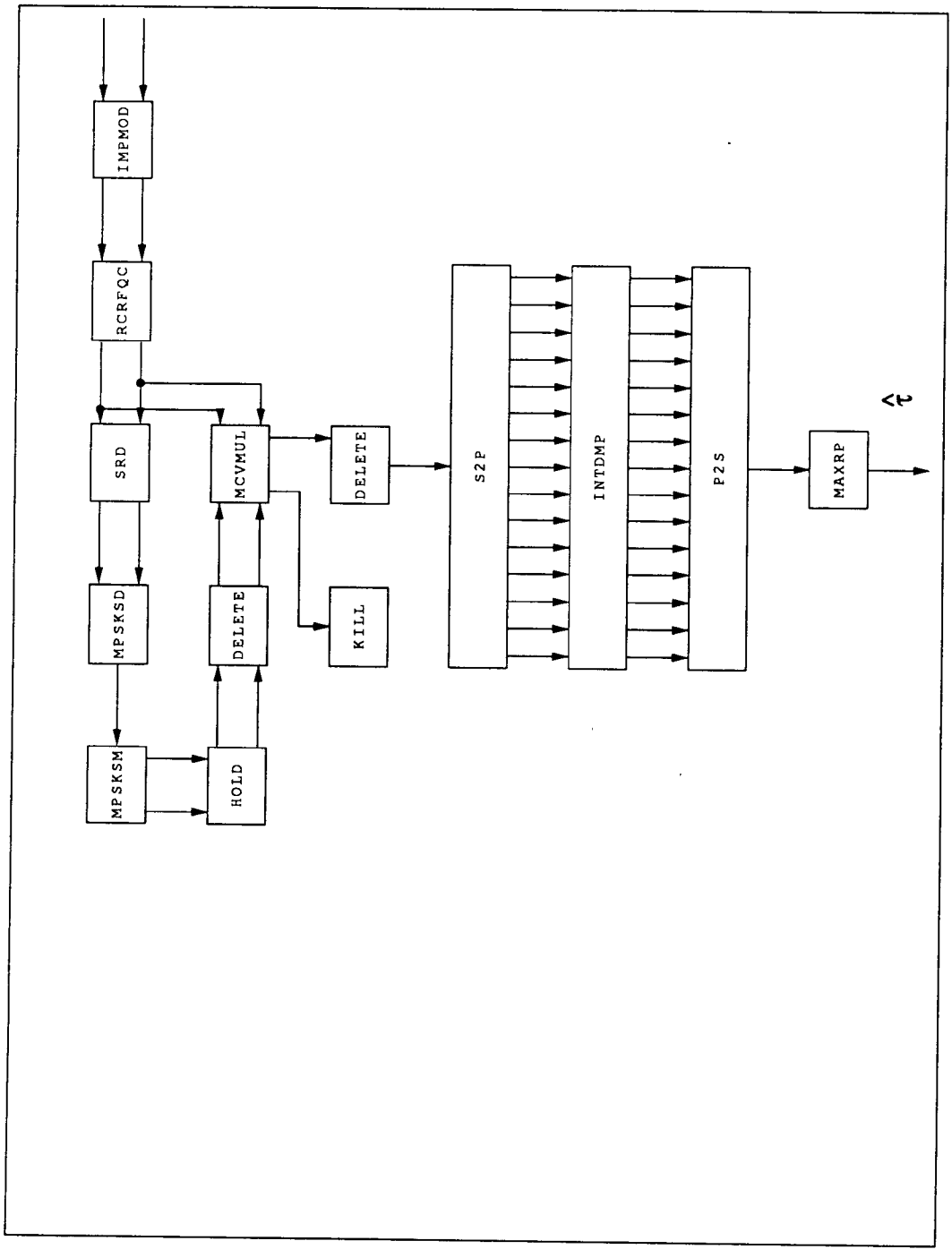


Fig. I.4-13: DDMLFF, Decision Directed Maximum Likelihood FeedForward symbol synchronizer

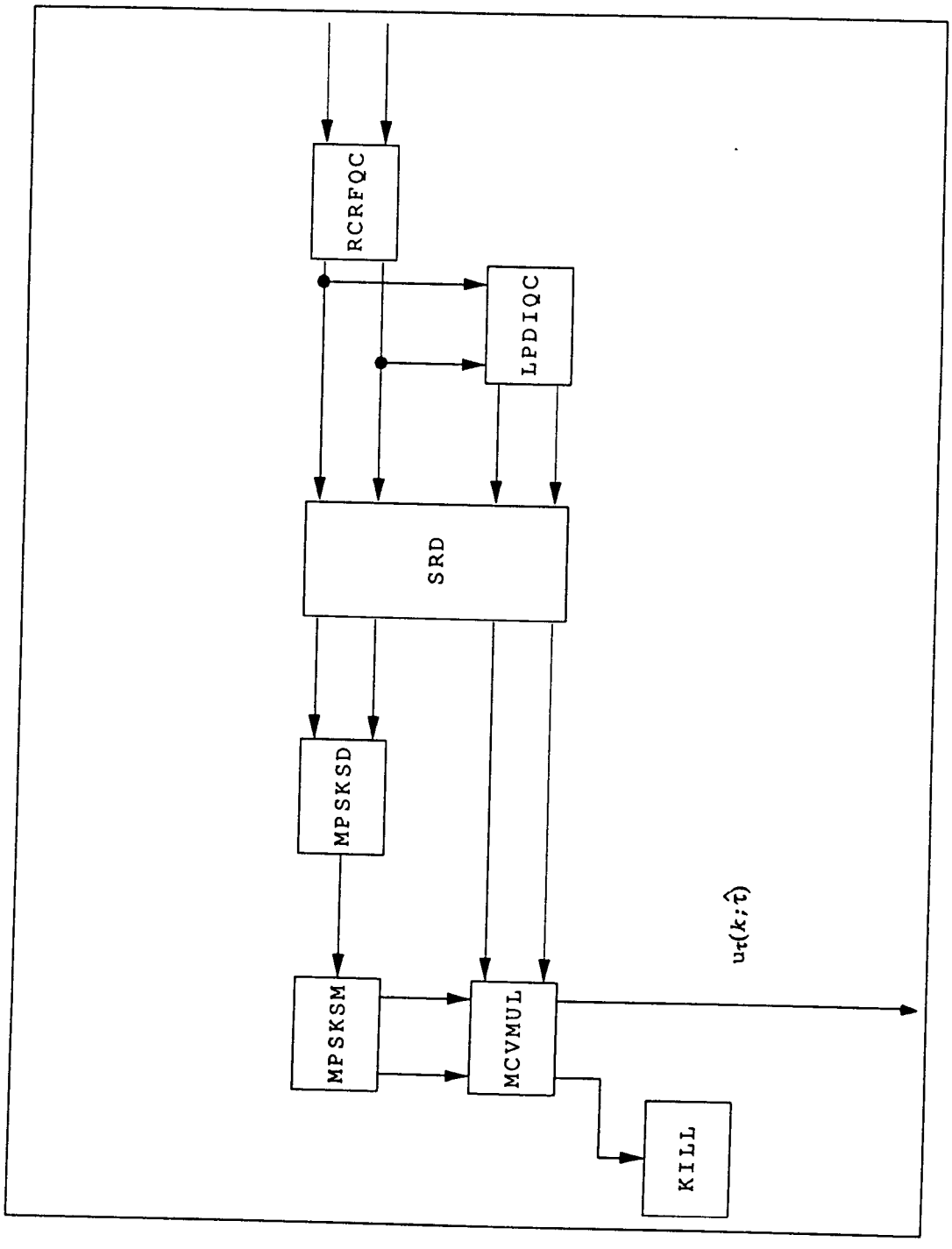


Fig. I.4-14: DDMLFB, Decision Directed Maximum Likelihood FeedBack symbol synchronizer

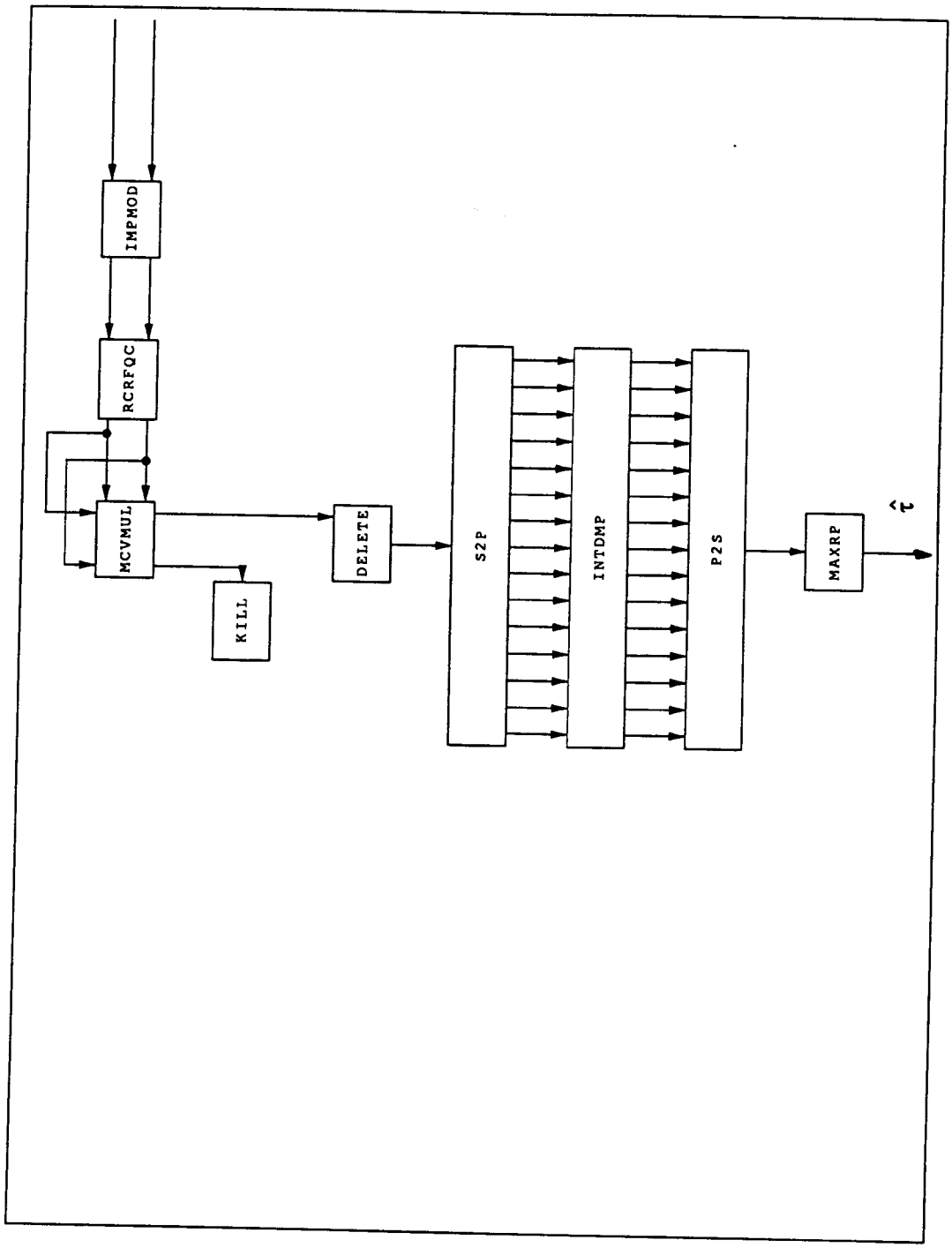


Fig. I.4-15: NDAMLFF, Non Data-Aided Maximum Likelihood FeedForward symbol synchronizer

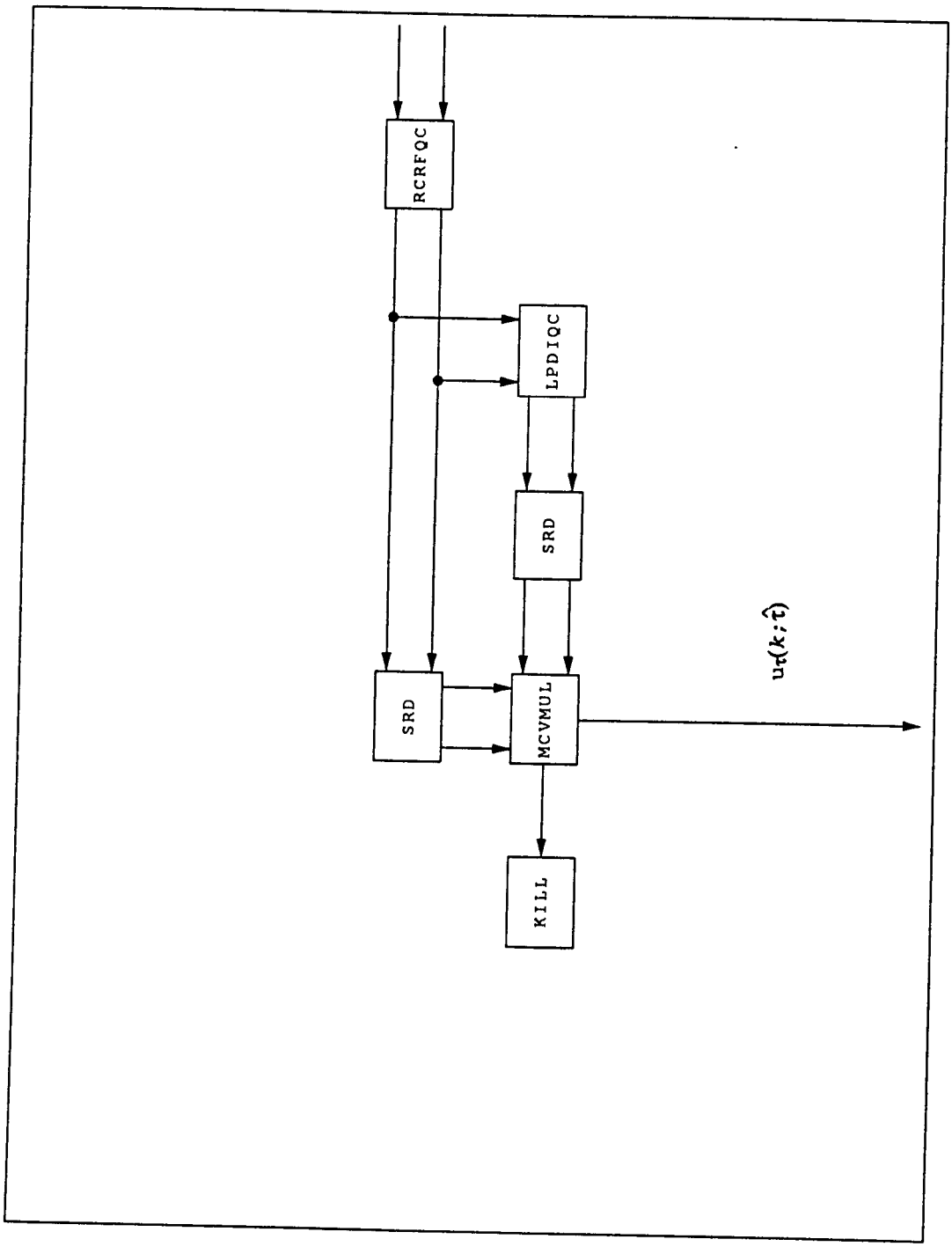


Fig. I.4-16: NDAMLFB, Non Data-Aided Maximum Likelihood FeedBack symbol synchronizer

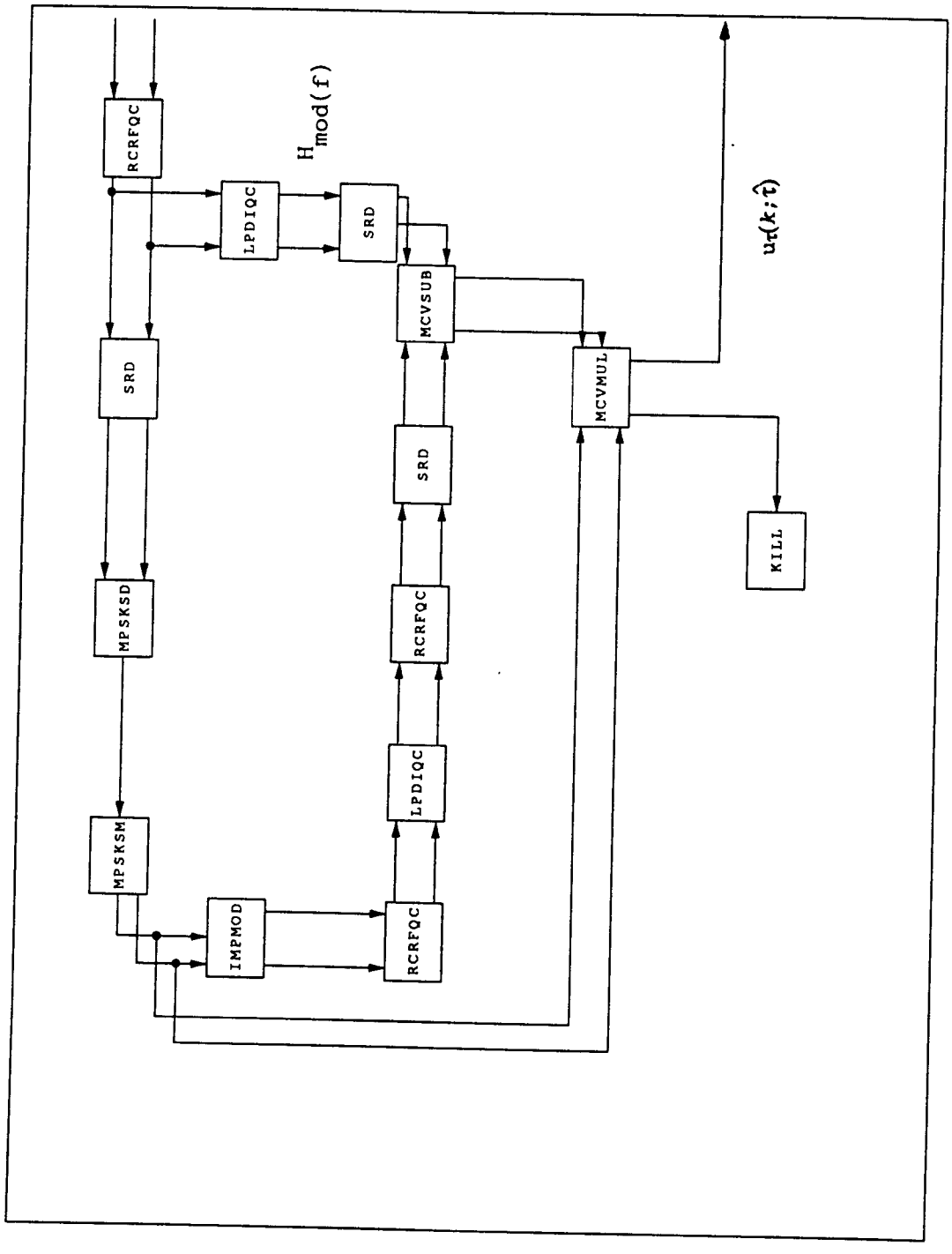


Fig. I.4-17: NDAMLFB$\overline{\text{SNFB}}$, Non Data-Aided Maximum Likelihood FeedBack symbol synchronizer with decision FeedBack SelfNoise cancellation

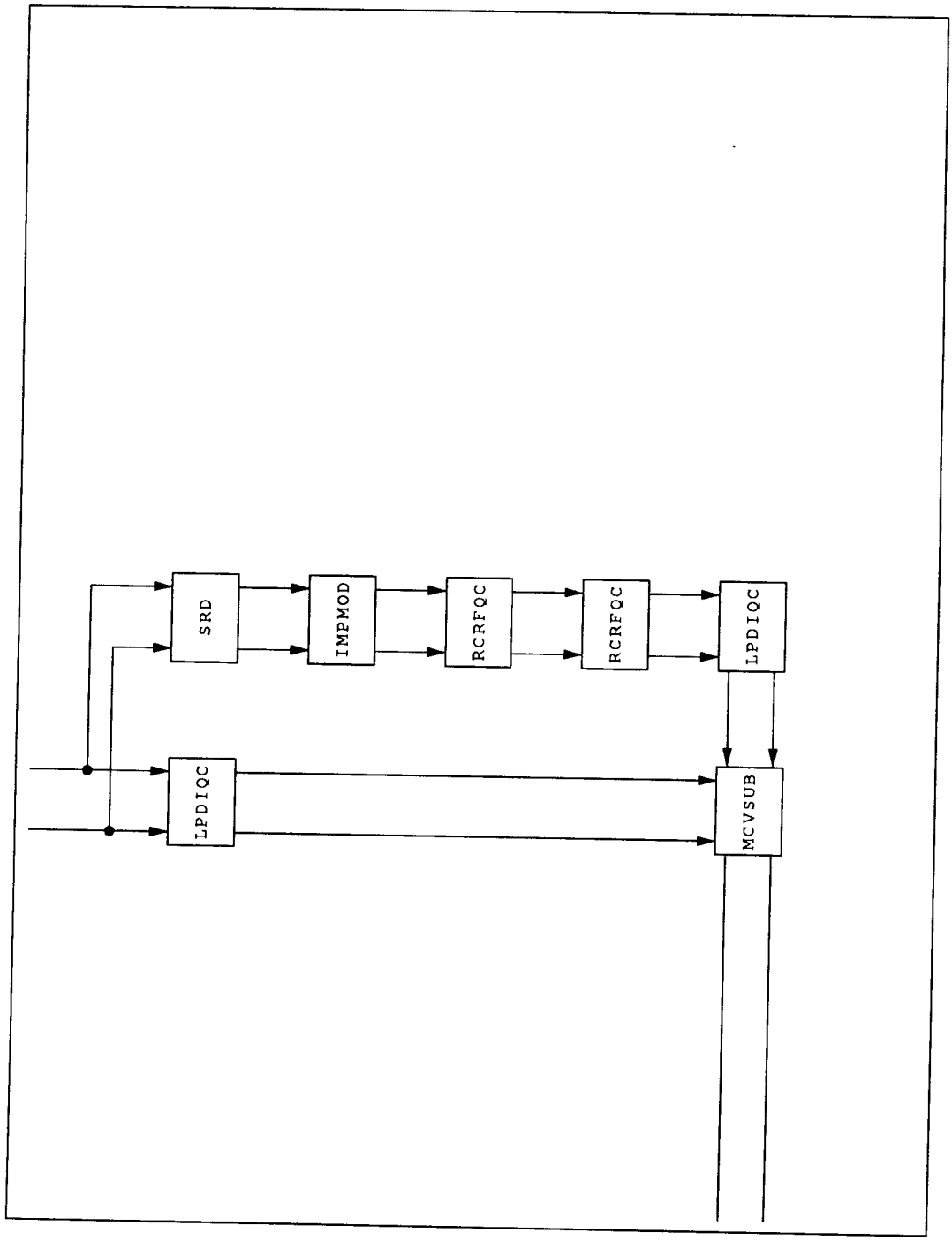


Fig. I.4-18: $\overline{\text{SNFF}}$, Modified derivative matched filter for FeedForward SelfNoise cancellation

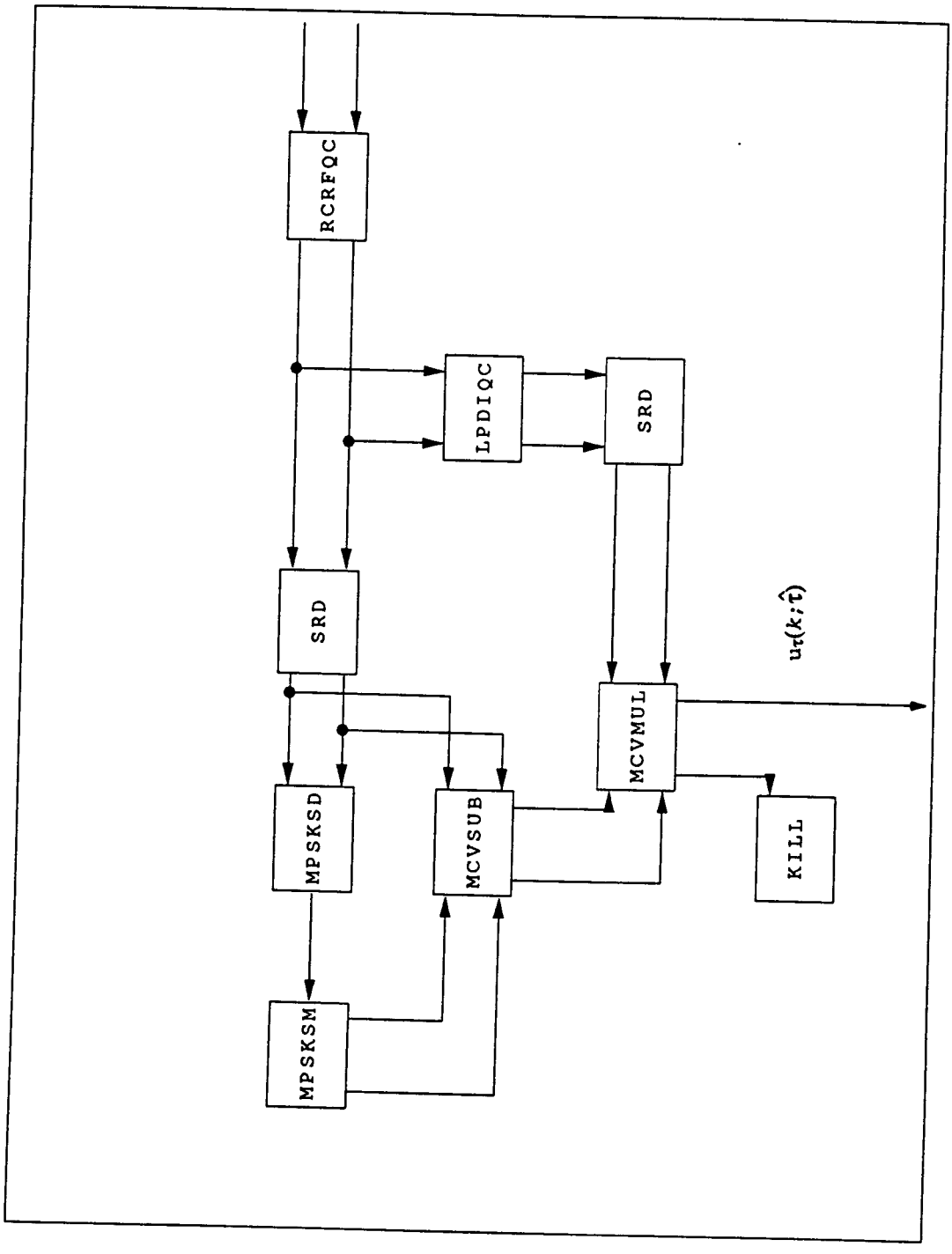


Fig. I.4-19: DDMMSEFB, Decision Directed MMSE FeedBack symbol synchronizer

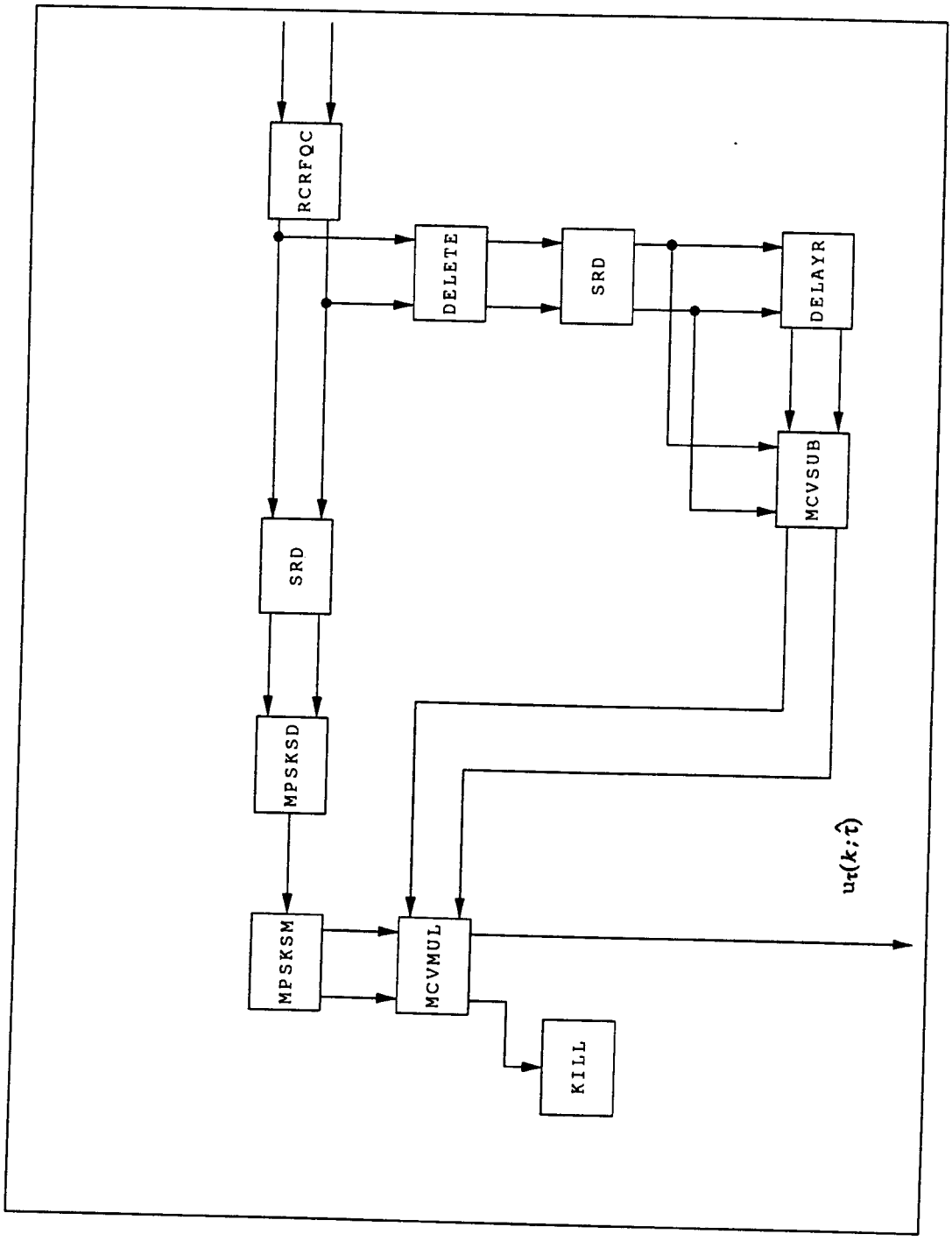


Fig. I.4-20: DDEL(FB), Decision Directed Early Late (FeedBack) symbol synchronizer

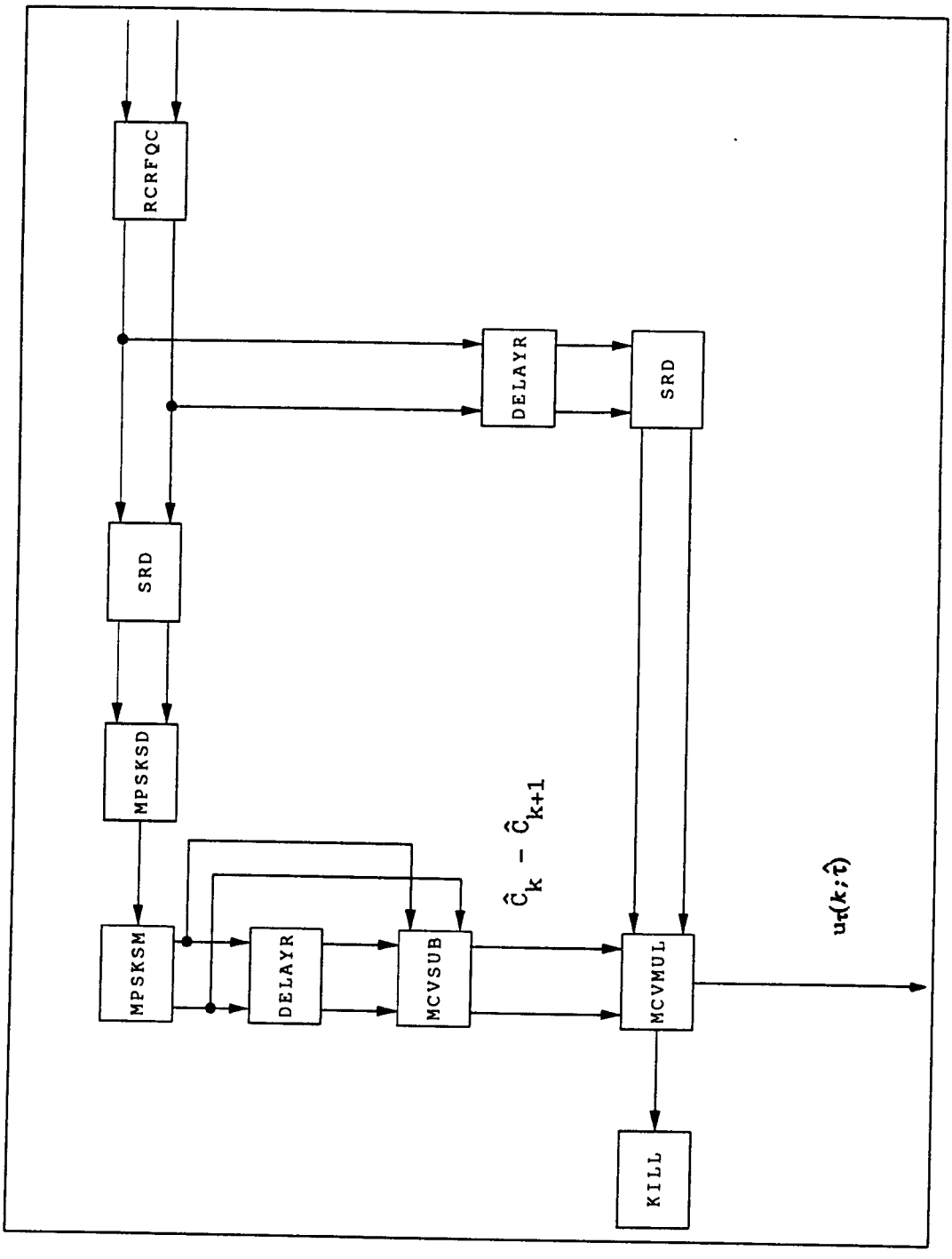


Fig. I.4-21: DTTL(FB), Data Transition Tracking Loop (FeedBack) symbol synchronizer

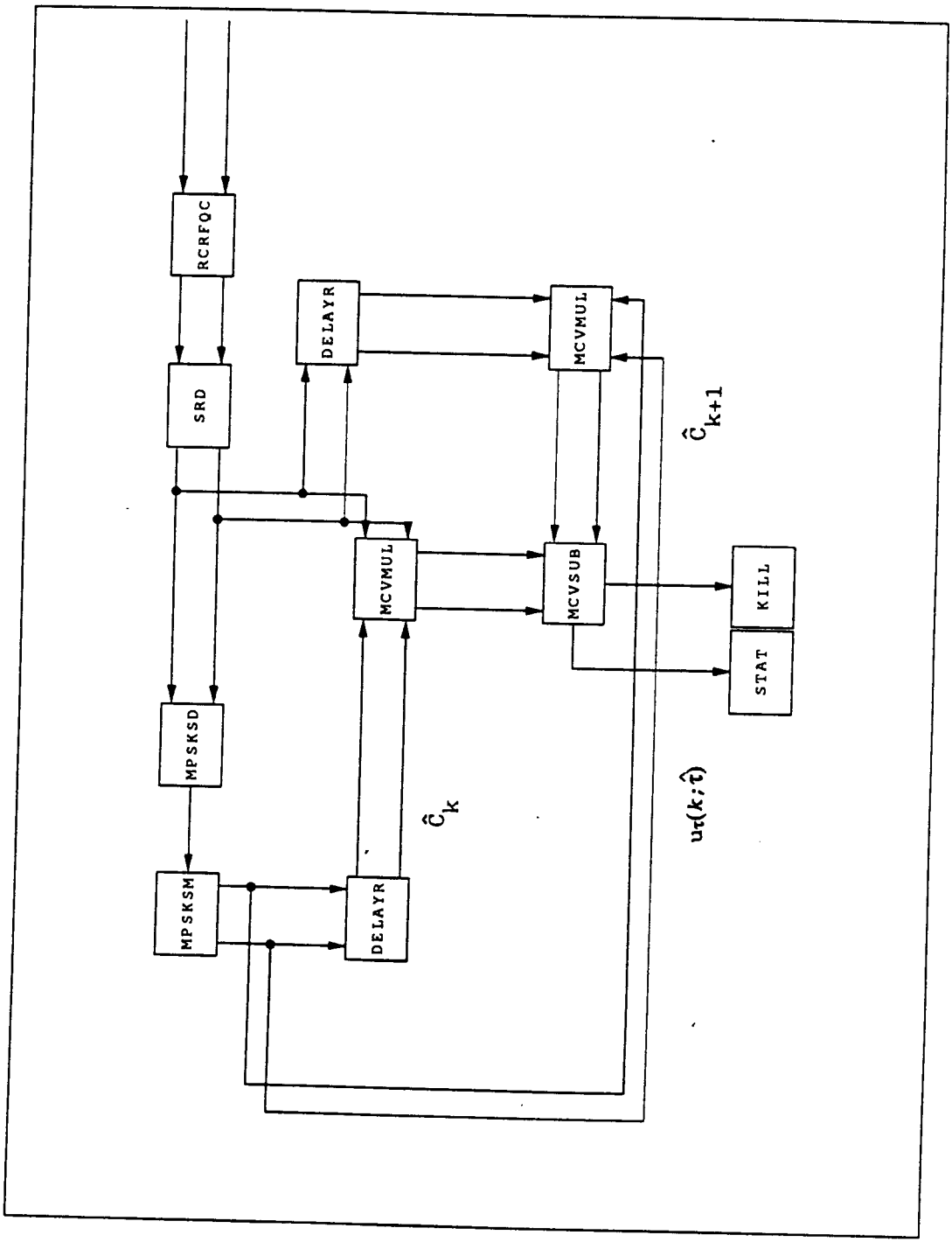


Fig. I.4-22: M&M(FB), Mueller&Müller (FeedBack) symbol synchronizer

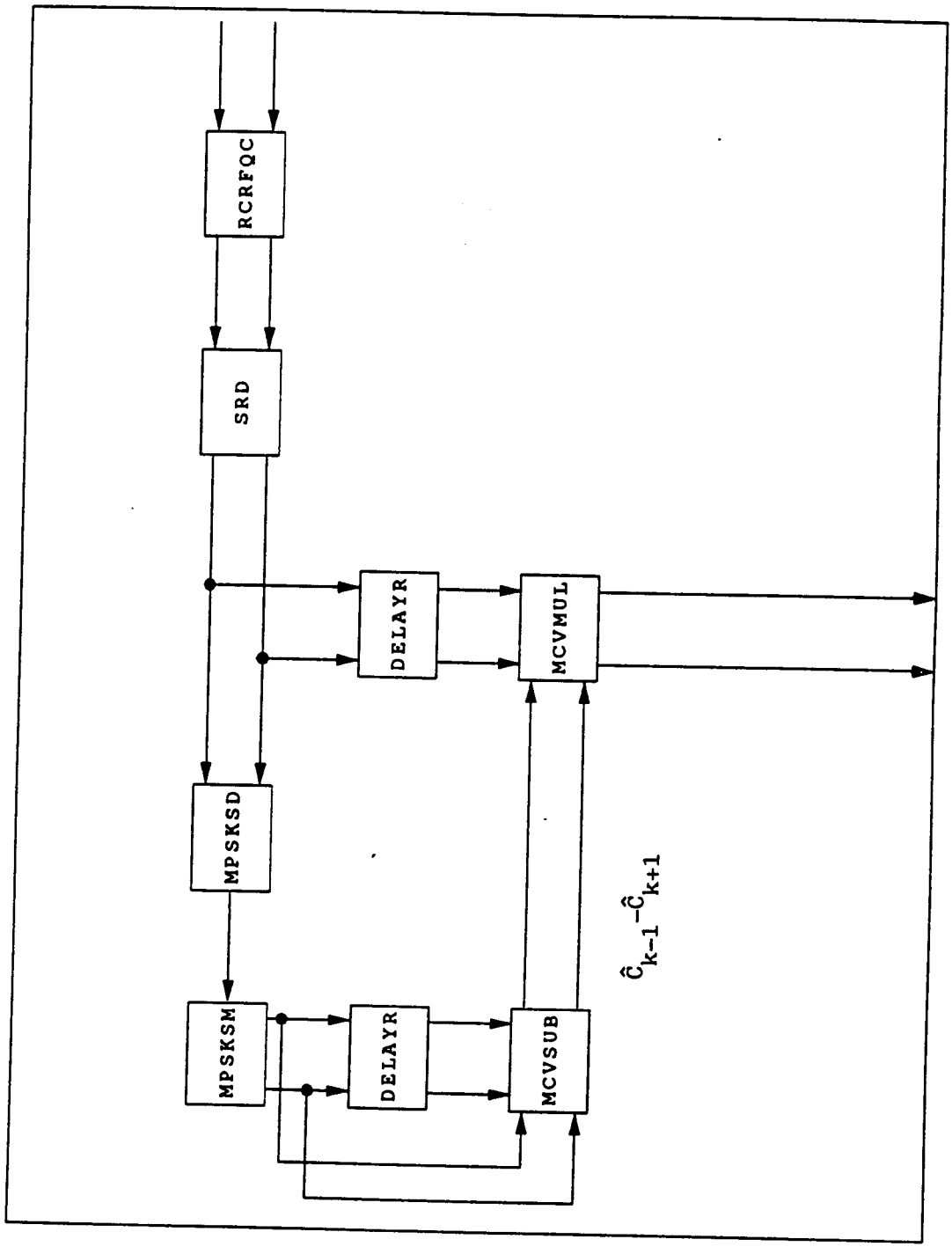


Fig. I.4-23: mM&M(FB), modified Mueller&Müller (FeedBack) symbol synchronizer

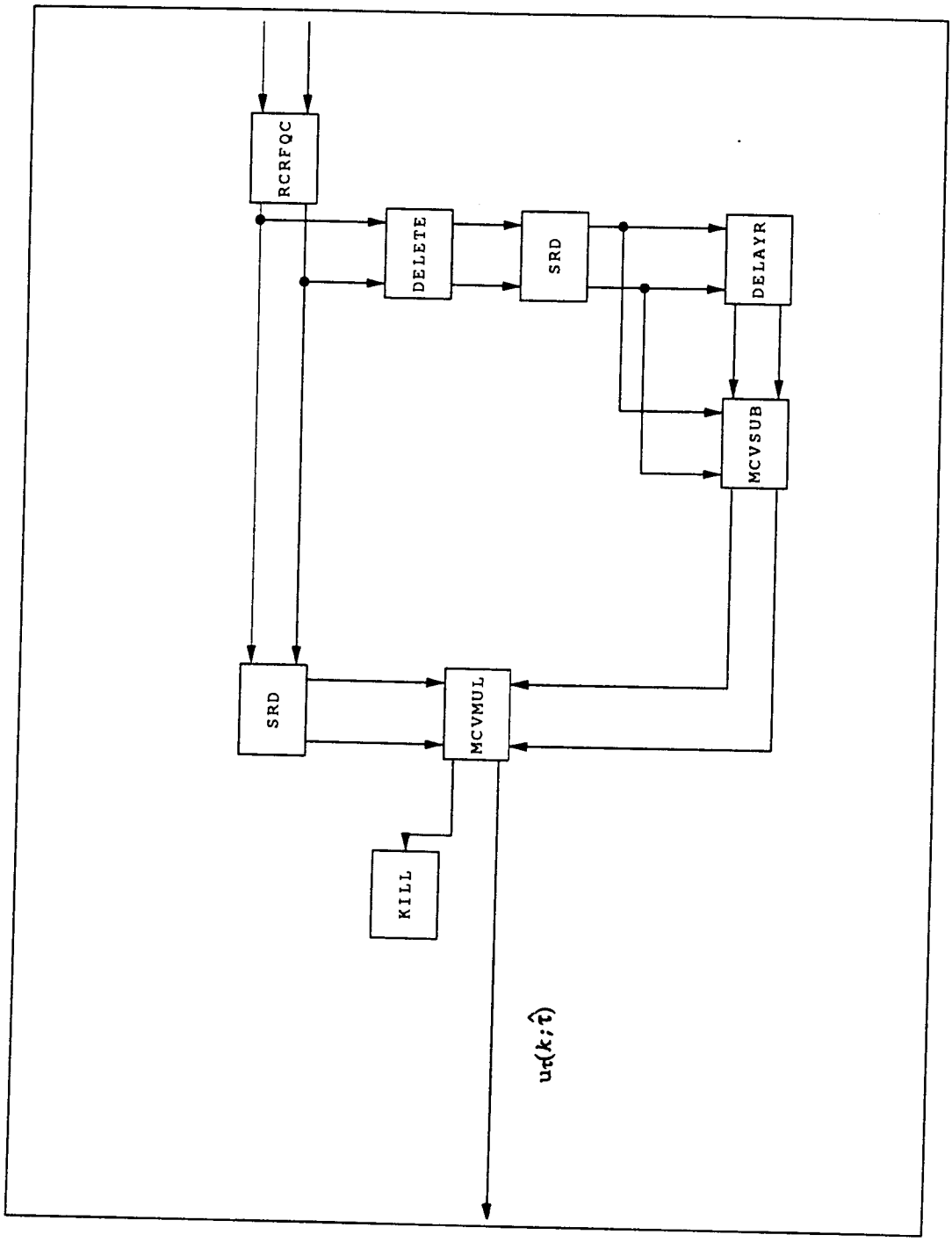


Fig. I.4-24: NDAEL(FB), Non Data-Aided Early Late (FeedBack) symbol synchronizer

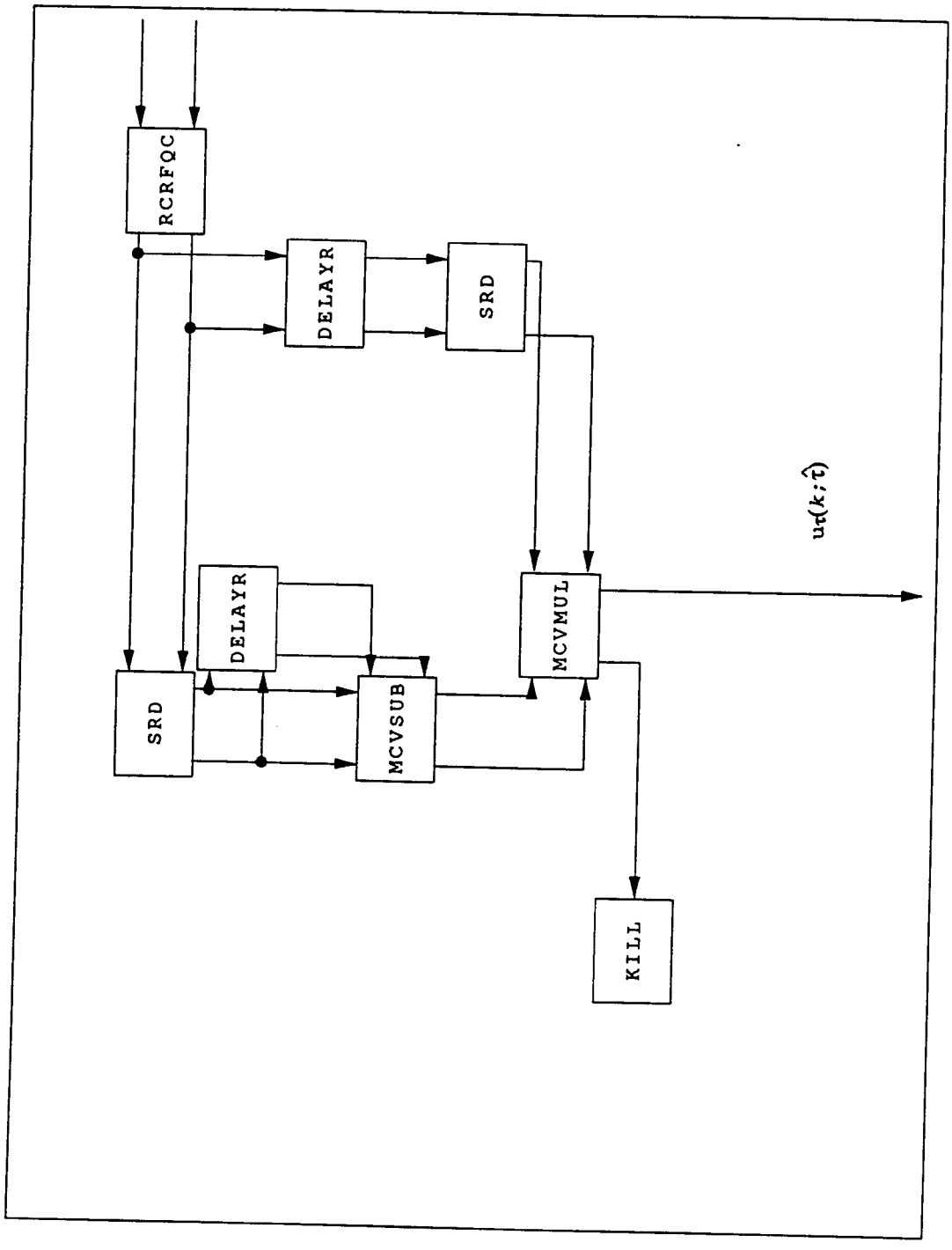


Fig. I.4-25: GA(FB), GArdner (FeedBack) symbol synchronizer

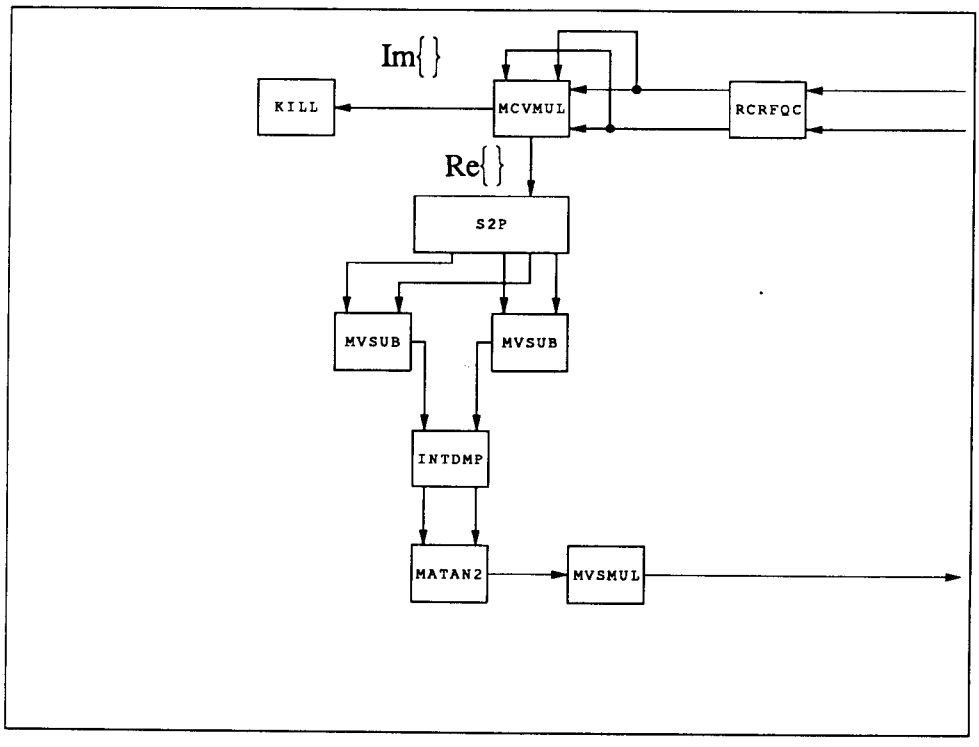


Fig. I.4-26: NDASQFF, Non Data-Aided Filter and Square FeedForward symbol synchronizer

Appendix I.A: List of Models in Alphabetical Order

This section contains a list of all models sorted in alphabetical order. For each model a short description is given.

Note that the following models have been renamed in COSSAP release 6.4:

Old Name	New Name
DELETE	DELR
DEMUX	DMUXR
HOLD	HOLDR
KILL	KILLR
P2S	P2SR
S2P	S2PR
SRD	SRDR

ACF1	Autocorrelation Function Estimation (Bartlett Periodogram Method)
ACF2	Autocorrelation Function Estimation (Welch Periodogram Method)
ADCONR	A/D Converter with Rounding Switching Levels
ADCONT	A/D Converter with Truncating Switching Level
ADIFI	Computes Absolute Difference between Two Integer Input Signals
AGCFF	Automatic Gain Control (feedforward)
APD	Avalanche Photodiode
APLC	Automatic Power Level Control
AWGN	Additive White Gaussian Noise Channel (Real)
AWGNQC	Additive White Gaussian Noise Channel (Complex)
BIAGVD	Computation of standard deviation factors (Variance Mod. Gaussian RV)
BIAPW	Piecewise Biasing of Input Signal Distribution
BIAPWD	Computation of interval bounds (Piecewise Const. Mod. Gaussian RV)
BSRC1	Pseudo Random Binary Source (1-32 Bit Shift Register)
BSRC2	Pseudo Random Binary Source (31 Bit Shift Register)
BUTF	Butterworth Filter (Real Input Signal)
BUTFQC	Butterworth Filter (Complex Input Signal)
CH2F	Chi-Square Fitting Test (Arbitrary Distribution)
CH2FG1	Chi Square Fitting Test of Gaussian RV (Known Mean and Variance)
CH2FG2	Chi Square Fitting Test of Gaussian RV (Unknown Mean and Variance)
CHBFQC	Chebychev Filter (Complex Input Signal)
CMPEQI	Compares Two Integer Signals (1 on Match)
CMPNEI	Compares Two Integer Signals (0 on Match)
CONSTI	Constant Integer Signal Source
CONSTR	Constant Real Signal Source
CONV	Convolution with Real Impulse Response (Real Input Signal)
CONVQC	Convolution with Complex Impulse Response (Complex Input Signal)
COUNT1	Counter (Error Counter)
COUNTP	Counter (Error Counter - Line Chart)
CPMM1	CPM Modulator (CPM: Continuous Phase Modulation)
CSYMIN	Complex Symbol Insert
CSYMRM	Complex Symbol Remove
DACONV	D/A Converter (Digital to Analog Converter)
DBGINI	Signal Input from Keyboard (INTEGER)
DBGINR	Signal Input from Keyboard (REAL)
DBGOUI	Signal Output to Screen (INTEGER)
DBGOUR	Signal Output to Screen (REAL)

DCOD1	Differential Encoder
DDCOD1	Differential Decoder
DELAYI	Delays Integer Signals (by multiples of the sampling time)
DELAYR	Delays Real Signals (by multiples of the sampling time)
DELCMP	Limiter with Data Compression
DELI	Deletes First Signal Elements (Integer Signals)
DELR	Deletes First Signal Elements (Real Signals)
DIGADD	Digital Adder
DIGMUL	Digital Multiplier
DIGSUB	Digital Subtractor
DIST1	Signal Power Distribution Output (Rayleigh paper)
DIST2	Peak Length Distribution Output
DMPC	Dump Complex Signal (graphics and file output)
DMPNI	Dump Integer Signals (graphics and file output)
DMPNR	Dump Real Signals (graphics and file output)
DMUXI	Demultiplexer (1 in 2, blockwise alternating, Integer Signals)
DMUXR	Demultiplexer (1 in 2, blockwise alternating, Real Signals)
DMUXSI	Demultiplexer (1 in N, blockwise, controlled by 2nd input signal, Integer Signals)
DMUXSR	Demultiplexer (1 in N, blockwise, controlled by 2nd input signal, Real Signals)
DPSKD1	Differential M-ary PSK Detector
DQ1	Computes Differential Quotient (Discrete Time Approximation)
DSC1	Hardlimiter
EKMSM0	Extended Kalman Smoother/Filter
EPSID	Maximum Likelihood Symbol Timing Estimation
EPSRE	Symbol Timing Estimation (by correlation)
ESTVA1	Variance Estimator for a Mean Value Estimator (known mean)
ESTVA2	Variance Estimator for a Mean Value Estimator
EYE	Eye Pattern (graphics and file output)
FDELSI	Time Varying (fractional) Delay - Synchronous to Input
FDELSO	Time Varying (fractional) Delay - Synchronous to Output
FDEVC	Rotates a Complex Signal by a Constantly Incrementing Angle (Constant Frequency Deviation)
FORKB	Forks a Bit_Vector Signal
FORKC	Forks a Complex Signal
FORKDI	Forks a Double Integer Signal
FORKDR	Forks a Double Real Signal
FORKI	Forks an Integer Signal
FORKR	Forks a Real Signal
GMSK	GMSK Modulator (Gaussian Minimum Shift Keying)
GSTAIL	Source for Random Data with Singlesided Gaussian Distribution
GSTLVT	Source for Random Data with Gaussian Tail Distr., Variable Threshold

GWSMR	Gliding Window Smoother (short term mean, N real input signals)
HIST1	Histogram of Input Signal Level (graphics and file output)
HLBP	HardLimiter (BandPass Signals, Constant Amplitude, n-th Harmonic)
HOLDI	Repeats Each Signal Element n Times (Integer Signals)
HOLDR	Repeats Each Signal Element n Times (Real Signals)
HYST	Hysteresis
IIRC	Infinite Impulse Response Filter (Real Inputs)
IIRCV	Infinite Impulse Response Filter with Variable Cutoff Frequency (Real Input Signals)
ILPF	Ideal Lowpass Filter (Real Input Signal)
ILPFQC	Ideal Lowpass Filter (Complex Input Signal)
IMPMOD	Impulse Modulator
INT1	Integrator (Discrete Time Approximation)
INTDMP	Integrate and Dump
INTRLI	Blockinterleaver / -deinterleaver (Integer Signals)
INTRLR	Blockinterleaver / -deinterleaver (Real Signals)
INVMOD	Inverse Modulo Operation
ISIGI	Input SIGnal Integer (Integer Input Signal Read From File)
ISIGR	Input SIGnal Real (Real Input Signal Read From File)
KFILS	Linear Kalman Filter
KILLI	Signal Sink (Integer Signals)
KILLR	Signal Sink (Real Signals)
LININT	Linear Interpolator (Real Input Signal)
LNR1	Lognormal Fading
LPDI	Differentiating Filter (Real Input Signal)
LPDIQC	Differentiating Filter (Complex Input Signal)
LTI	Linear Time Invariant Filter (Real Input Signal)
LTIQC	Linear Time Invariant Filter (Complex Input Signal)
LTV	Linear Time Variant Filter (Real Input Signal and Coefficients)
LTVQC	Linear Time Variant Filter (Complex Input Signal and Coefficients)
MATAN2	Computes Arcus Tangens of a Complex Input Signal
MAXI	Searches Maximum in an Integer Input Signal Block
MAXR	Searches Maximum in a Real Input Signal Block
MC1	Manchester Encoder
MCFFT	Computes the Complex (Inverse) DFT of a Complex Input Signal
MCVADD	Adds Two Complex Input Signals
MCVCSM	Multiplies a Complex Input Signal by a Complex Scalar
MCVDIV	Divides Two Complex Input Signals
MCVMGS	Computes the Squared Absolute Value of a Complex Input Signal
MCVMUL	Computes the Product of Two Complex Input Signals

MCVROT	Rotates Complex Input Signal by Constant Angle
MCVSUB	Subtracts Two Complex Input Signals
MFTAPC	Computation of Complex Matched Filter Coefficients (Time Variant)
MI2R	Integer Signal to Real Signal Conversion
MINI	Searches Minimum in an Integer Input Signal Block
MINR	Searches Minimum in a Real Input Signal Block
MLIND	M-ary Linear Demodulator (Programmable)
MLINM	M-ary Linear Modulator (Programmable)
MMSEQC	Minimum Mean Square Error Adaptive Equalizer (Complex Signals)
MPAMD	M-ary PAM Demodulator
MPAMM	M-ary PAM Modulator
MPOLAR	Converts Complex Signal from Cartesian to Polar Representaion
MPSKD	M-ary PSK Demodulator
MPSKM	M-ary PSK Modulator
MPSKSD	M-ary PSK Demodulator
MPSKSM	M-ary PSK Modulator
MR2I	Real Signal to Integer Signal Conversion (Rounding to Nearest Integer)
MRECT	Converts Complex Signal from Polar to Cartesian Representation
MSKD1	Minimum Shift Keying Detector
MUXAI	Multiplexer (N in 1, asynchronous, Integer Signals))
MUXAR	Multiplexer (N in 1, asynchronous, Real Signals))
MUXI	Multiplexer (2 in 1, blockwise alternating, Integer Signals)
MUXR	Multiplexer (2 in 1, blockwise alternating, Real Signals)
MVABS	Computes Absolute Value of a Real Input Signal
MVADD	Adds Two Real Input Signals
MVCLIP	Softlimiter (signal clipping)
MVCOS	Computes Cosine of a Real Input Signal
MVDIV	Divides Two Real Input Signals
MVEX	Exponentiate Real Input Signal (Base e)
MVEX10	Exponentiate Real Input Signal (Base 10)
MVIABS	Computes Absolute Value of an Integer Input Signal
MVIADD	Adds Two Integer Input Signals
MVIDIV	Divides Two Integer Input Signals
MVIMUL	Multiplies Two Integer Input Signals
MVISAD	Adds an Integer Scalar to an Integer Input Signal
MVISDI	Divides an Integer Input Signal by an Integer Scalar
MVISMU	Multiplies an Integer Input Signal by an Integer Scalar
MVISSU	Subtracts an Integer Scalar from an Integer Input Signal
MVISUB	Subtracts Two Integer Input Signals
MVMUL	Multiplies Two Real Input Signals

MVRG1	Pseudo Random Source (Uniform Distribution)
MVRG2	Pseudo Random Source (Uniform Distribution, Whichman/Hill Alg.)
MVSADD	Adds a Real Scalar to a Real Input Signal
MVSIN	Computes Sine of a Real Input Signal
MVSMUL	Multiplies a Real Input Signal by a Real Scalar
MVSQ	Computes Square of a Real Input Signal
MVSQRT	Computes the Square Root of a Real Input Signal
MVSSUB	Subtracts a Real Scalar from a Real Input Signal
MVSUB	Subtracts Two Real Input Signals
OHTRIR	Orthogonal Householder Transformation on Real Input Signals
P2SI	Parallel to Serial Conversion (N in 1, Sample by Sample Multiplexer, Integer Signals)
P2SR	Parallel to Serial Conversion (N in 1, Sample by Sample Multiplexer, Real Signals)
PEAKAA	Area Under Signal Peak (one output sample per peak)
PEAKAS	Area Under Signal Peak (output signal is running area)
PEAKLA	Length of Signal Peak (one output sample per peak)
PEAKLS	Length of Signal Peak (output signal is running length)
PPGENI	Periodic Integer Pattern Source
PPGENR	Periodic Real Pattern Source
PROD	Computes the Product of an Arbitrary Number of Real Input Signals
PSD1	Power Spectral Density Estimation (Bartlett Method, Real Signal)
PSD1QC	Power Spectral Density Estimation (Bartlett Method, Complex Signal)
PSD2	Power Spectral Density Estimation (Welch Method, Real Signal)
PSD2QC	Power Spectral Density Estimation (Welch Method, Complex Signal)
PULS	Periodic Puls Generator for Double-Trapezoidal Pulses
PULSLG	Periodic Puls Generator for Long Double-Trapezoidal Pulses
QUADCO	Balanced Quadricorrelator (frequency estimator)
QUANT	Quantizer
QUASI1	Quasi Analytical Estimation of Bit Error Probability (BPSK, AM)
QUASI2	Quasi Analytical Estimation of Bit Error Probability (QPSK)
RCIN	Interpolator with Raised Cosine Filter (Real Input Signal)
RCRF	Root Raised Cosine Filter (Real Input Signal)
RCRFQC	Root Raised Cosine Filter (Complex Input Signal)
REFGEN	Reference Signal Generation
RMEAN	Running Mean of Input Signal
RMIX1I	Random Multiplexer (2 in 1, no data lost, Integer Signals)
RMIX1R	Random Multiplexer (2 in 1, no data lost, Real Signals)
RMIX2I	Random Multiplexer (2 in 1, data partially lost, Integer Signals)

RMIX2R	Random Multiplexer (2 in 1, data partially lost, Real Signals)
RMIXI	Random Multiplexer (N in 1, no data lost, Integer Signals)
RMIXR	Random Multiplexer (N in 1, no data lost, Real Signals)
RSTI	Restore Dumped Integer Signals
RSTR	Restore Dumped Real Signals
RWALK1	Second Order Gauss-Markov Random Walk
S2PI	Serial to Parallel Conversion (1 in N, sample by sample Demultiplexer, Integer Signals)
S2PR	Serial to Parallel Conversion (1 in N, sample by sample Demultiplexer, Real Signals)
SAW	Sawtooth Nonlinearity
SCAT	Scatterdiagram (graphics and file output)
SCLBP	Symmetric Clipper (BandPass Signals, const. amplitude, n-th harmonic)
SFRAME	Symbol Framing (Insertion of Input Symbols into Frame Structure)
SIIN	Sin(x)/x - Interpolator (Real Input Signal)
SIINQC	Sin(x)/x - Interpolator (Complex Input Signal)
SRDI	Sampling Rate Decimation (Integer Signals)
SRDR	Sampling Rate Decimation (Real Signals)
SRXI	Sampling Rate Expansion (Integer Signals)
SRXR	Sampling Rate Expansion (Real Signals)
STAT	Signal Statistics - Mean, Power, Variance (Text Chart)
STATP	Signal Statistics - Mean, Power, Variance (Line Chart)
STOPI	Stops Simulation (by counting integer signal elements)
STOPR	Stops Simulation (by counting real signal elements)
SUMI	Computes the Sum of an Arbitrary Number of Integer Input Signals
SUMR	Computes the Sum of an Arbitrary Number of Real Input Signals
SYMDEC	M-ary to Binary Conversion
SYMEL1	Complex Nonlinearity (for M-ary PSK symbol elimination)
SYMGCD	M-ary to Binary Conversion, Gray Code
SYMGCG	Binary to M-ary Conversion, Gray Code
SYMGEN	Binary to M-ary Conversion
TAPFIL	Estimates Equiv. Channel Filter Coefficients (Time Variant)
TEBRS	Timing (Clock) Error Detector (Mueller&Mueller)
TERD	Timing (Clock) Error Detector for BPSK or QPSK (Gardner)

TIMEXT	Digital Filter&Square Timing (Clock) Error Detector for QAM or PSK
TRELL1	Trellis Encoder (programmable, 1 dimensional codes)
TWT1	Traveling Wave Tube (Scaled Amplitude Model by Thomas)
TWT2	Traveling Wave Tube (Quadrature Model by Hetrakul/Taylor)
TWT3	Traveling Wave Tube (General Amplitude/Phase Model, by Thomas et al.)
TWT4AP	Traveling Wave Tube (Amplitude-Phase Model by Saleh)
TWT4Q	Traveling Wave Tube (Quadrature Model by Saleh)
VITRB1	Viterbi Decoder (programmable, Euclidian Distance)
WGN	White Gaussian Noise Source (Real)
WGNQC	White Gaussian Noise Source (Complex)
WTSGAM	Computation of Unbiasing Weights (Abs. & Mean Mod. Gaussian RV)
WTSGBM	Computation of Unbiasing Weights (Bimod. Mean Mod. Gaussian RV)
WTSGM	Computation of Unbiasing Weights (Mean Mod. Gaussian RV)
WTSGMV	Computation of Unbiasing Weights (Mean Mod. Gaussian RV)
WTSGU	Computation of Unbiasing Weights (Unif. RV for Gaussian RV)
WTSGV	Computation of Unbiasing Weights (Variance Mod. Gaussian RV)
WTSGV	Computation of Unbiasing Weights (Variance Mod. Gaussian RV)
WTSGVM	Computation of Unbiasing Weights (Var. & Mean Mod. Gaussian RV)
WTSGVV	Computation of Unbiasing Weights (Variance Mod. Gaussian RV)
WTSPW	Computation of Unbias Weights (piecewise biased distribution)
WTTCM	Compute Total Weights from Single Weights

II. TRACKING PERFORMANCE UNDER IDEAL CONDITIONS

II.1 PRELIMINARIES

II.1.1 Introduction.

Part II of this study deals with the tracking performance of various carrier and symbol synchronizers, operating on a noisy carrier modulated (BPSK, QPSK or OQPSK) signal which conveys digital data. These synchronizers are analyzed under the assumption of "ideal" conditions, which involve the following :

– The transmission channel does not introduce any linear or nonlinear distortion; an additive white Gaussian noise channel is assumed.

– The carrier frequency is known to the receiver. This allows the received bandpass signal to be converted into a complex–valued baseband signal, which is centered at zero frequency (which is the centre frequency of the data filter). Also, the carrier phase and the time delay are constant.

– The data filter is matched to the received baseband pulse. The baseband pulse at the output of the matched filter satisfies the Nyquist–I criterion, which implies that there is no ISI at the optimum decision instants. Numerical results will be obtained under the assumption that the baseband pulse at the matched filter output is a cosine rolloff pulse with a rolloff parameter α, ranging from 0 to 1.

– Receiver operations (filtering, summation, multiplication, etc.) are ideal : no quantization effects, no truncation of impulse response duration, infinite precision, etc.

Tracking performance results are obtained both by theoretical and simulation analysis; a description of these analysis methods is given in sections II.1.2 and II.1.3. The tracking performance results for carrier and symbol synchronization are presented in sections II.2 to II.4; the nomenclature of various synchronizers is as in section I.2.

II.1.2 Theoretical analysis of tracking performance.

In the tracking mode, it is assumed that the estimates $\hat{\theta}$ and $\hat{\tau}$ of the carrier phase and the time delay exhibit small fluctuations about the correct values θ and τ. This allows us to linearize the system equations which describe the synchronizer operation, and to apply standard linear filter theory to evaluate the tracking error variances.

Feedback synchronization

In the case of feedback (FB) synchronization, linearization of the outputs of the phase error detector and the timing error detector about the stable equilibrium point (ϕ=0, ϵ=0) shows that the carrier and symbol synchronizers investigated in this study are not coupled in the tracking mode (see section I.3.1). Hence, the tracking performance can be obtained from the simplified model shown in Fig. I.3–6, which includes the following quantities :

$H_\theta(z)$: closed loop transfer function of carrier synchronizer (see I.3.1–4);

$N_\theta(k;0,0)$: phase error detector noise, evaluated at the stable equilibrium point [measured in rad];

$K_{\theta\theta}$: phase error detector slope, evaluated at the stable equilibrium point (see I.3.1–3) [dimensionless];

$H_\tau(z)$: closed loop transfer function of symbol synchronizer (see I.3.1–4);

$N_\tau(k;0,0)$: timing error detector noise, evaluated at the stable equilibrium point [dimensionless];

$K_{\tau\tau}$: timing error detector slope, evaluated at the stable equilibrium point (see I.3.1–3) [dimensionless].

When θ and τ are constant, the resulting variances of the estimation errors $\phi=\hat{\theta}-\theta$ and $\epsilon=(\tau-\hat{\tau})/T$ are given by (see I.3.1–5)

$$\mathrm{Var}[\phi] = \frac{1}{K_{\theta\theta}^2} \int_{-1/2}^{1/2} S_\theta(fT)\ |H_\theta(\exp(j2\pi fT))|^2\ d(fT) \qquad \text{(II.1.2–1a)}$$

$$\mathrm{Var}[\epsilon] = \frac{1}{K_{\tau\tau}^2} \int_{-1/2}^{1/2} S_\tau(fT)\ |H_\tau(\exp(j2\pi fT))|^2\ d(fT) \qquad \text{(II.1.2–1b)}$$

$\mathrm{Var}[\phi]$ is in rad^2, whereas $\mathrm{Var}[\epsilon]$ is dimensionless. In (II.1.2–1), $S_\theta(fT)$ and $S_\tau(fT)$ are the spectra of the loop noise sequences $\{N_\theta(k;0,0)\}$ and $\{N_\tau(k;0,0)\}$,

respectively :

$$S_\theta(fT) = \sum_{m=-\infty}^{+\infty} R_\theta(m) \exp(-j2\pi mfT) \qquad [rad^2]$$

$$S_\tau(fT) = \sum_{m=-\infty}^{+\infty} R_\tau(m) \exp(-j2\pi mfT) \qquad [dimensionless] \qquad \text{(II.1.2–2)}$$

where $\{R_\theta(m)\}$ and $\{R_\tau(m)\}$ are the loop noise autocorrelation sequences :

$$R_\theta(m) = E[N_\theta(k+m;0,0)N_\theta(k;0,0)] \qquad [rad^2]$$

$$R_\tau(m) = E[N_\tau(k+m;0,0)N_\tau(k;0,0)] \qquad [dimensionless] \qquad \text{(II.1.2–3)}$$

Feedforward synchronization

It has been pointed out in section I.3.2 that the tracking error variance resulting from a feedforward (FF) synchronizer is the same as that resulting from the corresponding feedback synchronizer, whose error signal equals the derivative, of the function to be maximized by the feedforward synchronizer. Therefore, the tracking error variance is given by (II.1.2–1), with $H_\theta(.)$ and $H_\tau(.)$ being the transfer functions of the averaging lowpass filter of the feedforward synchronizer.

Tracking error variance as a function of SNR

It follows from Fig. I.3–6 that, for constant θ and τ, the tracking errors can be interpreted as lowpass filtered versions of the loop noise. In the case of a quadratic synchronization algorithm the loop noise consists of three terms :

– A signal × signal term, which is caused by the random nature of the data symbols (self–noise). Its contribution to the tracking error variance does not depend on E_s/N_o.

– A signal × noise term, which is caused by the interaction between the signal and the additive noise at the input of the receiver. Its contribution to the tracking error variance is inversely proportional to E_s/N_o.

– A noise × noise term, which is caused solely by the additive noise at the input of the receiver. Its contribution to the tracking error variance is inversely proportional to the square of E_s/N_o.

Hence, three different regions of E_s/N_o can be distinguished. In the region of "high" E_s/N_o, the tracking error variance is limited by self–noise, and is essentially independent of E_s/N_o. In the region of "moderate" E_s/N_o, the

signal × noise term dominates, so that the tracking error variance is essentially inversely proportional to E_s/N_o; a reduction of E_s/N_o by some factor yields an increase of the tracking error variance by nearly the same factor. In the region of "low" E_s/N_o, the noise × noise term dominates, so that the tracking error variance increases essentially quadratically with the inverse of E_s/N_o.
For synchronization algorithms involving other nonlinearities (quadrupler, hard limiter for DD operation, etc.) a similar behaviour can be observed in the regions of moderate and high E_s/N_o. For low E_s/N_o, the tracking error variance increases more than linearly (and, for very low E_s/N_o, sometimes more than quadratically) with the inverse of E_s/N_o. The behaviour of the tracking error variance as a function of E_s/N_o is illustrated in Fig. II.1–1 (assuming a quadratic algorithm); the influence of the synchronizer bandwidth B_L is discussed below.

Variance expression for small synchronizer bandwidth

In the sequel, we will suppress the indices θ and τ in $S_\theta(fT)$, $S_\tau(fT)$, $H_\theta(.)$ and $H_\tau(.)$, and we will denote $K_{\theta\theta}$ and $K_{\tau\tau}$ by "slope". This allows us to make statements which are valid for both carrier and symbol synchronizers, or else it will be clear from the context whether we deal with carrier synchronizers or symbol synchronizers.

An important characteristic of a synchronizer is the one–sided noise bandwidth B_L of the transfer function H(.) of the closed loop (FB synchronizers) or the averaging lowpass filter (FF synchronizers). In all cases of practical interest, the synchronizer bandwidth B_L is much smaller than the symbol rate 1/T (at which the estimates are updated). Therefore, when considering the tracking error variance as a function of $2B_LT$, it is sufficient to keep in the Taylor series expansion of the variance expressions (II.1.2–1) about $2B_LT = 0$ only the linear and the quadratic terms in the variable $2B_LT$. This yields

$$\mathrm{Var} = (2B_LT)\,\frac{S(fT=0)}{(\mathrm{slope})^2} + K_F\,(2B_LT)^2 \sum_m |m|\,\frac{-R(m)}{(\mathrm{slope})^2} \qquad \text{(II.1.2–4)}$$

where Var denotes the phase error variance or the timing error variance, {R(m)} is the autocorrelation sequence of the synchronizer noise, given by (II.1.2–3), and K_F is a coefficient depending on the shape of the transfer function of the closed loop (FB synchronizer) or the averaging lowpass filter (FF synchronizer) :
– Accumulator filter (FF) : $K_F = 1$

– First–order loop (FB) or single pole filter (FF) : $K_F = 2$
– Second–order loop with perfect integrator and damping ζ :

$$K_F = 2\left[\frac{4\zeta^2}{1+4\zeta^2}\right]^2$$

It should be noted that the quadratic bandwidth term in (II.1.2–4) is correct only when for large m, R(m) decays faster than $1/m^2$. Otherwise, the summation in the quadratic bandwidth term of (II.1.2–4) does not converge; in this case (II.1.2–1) must be used for evaluating the tracking error variance. A sufficient condition for convergence of this summation is that the baseband pulse at the matched filter output decays faster than $1/t$ for large t; for a cosine rolloff pulse this sufficient condition is fulfilled when the excess bandwidth is nonzero.

The loop noise spectrum S(fT) consists of two terms, which are caused by additive noise and self–noise, respectively. The spectrum caused by additive noise is nonzero at fT=0, whereas for all considered synchronizers the self–noise spectrum vanishes at fT=0 (basically, this vanishing is due to the even symmetry of the baseband pulse g(t) at the output of the matched filter). This means that the linear bandwidth term in (II.1.2–4) is entirely due additive noise, whereas the quadratic bandwidth term in (II.1.2–4) contains a self–noise contribution plus a contribution due to additive noise. The additive noise contribution to the quadratic bandwidth term of (II.1.2–4) is negligible with respect to the linear bandwidth term of (II.1.2–4), because $2B_LT << 1$. Hence, the linear bandwidth term of (II.1.2–4) will be considered as the additive noise contribution to the tracking error variance, while the quadratic bandwidth term of (II.1.2–4) will be evaluated only for the self–noise component of the loop noise (the additive noise contribution to this quadratic term will be ignored). When the baseband pulse g(t) at the matched filter output has no longer even symmetry (e.g. because of channel distortions) the self–noise spectrum is nonzero at fT=0, so that the self–noise contribution to the tracking error variance is, like the additive noise contribution, essentially proportional to $2B_LT$.

As the additive noise contribution to the tracking error variance is proportional to $2B_LT$, and the self–noise contribution is proportional to $(2B_LT)^2$, the SNR at which self–noise limits the tracking performance decreases with increasing synchronizer bandwidth. This is illustrated in Fig. II.1–1.

In sections II.2 and II.3 we will present the coefficients of $2B_LT$ and $K_F(2B_LT)^2$ from (II.1.2–4), for various carrier and symbol synchronizers; the coefficient of $2B_LT$ will be considered as a function of E_s/N_o, where E_s denotes

the received energy per symbol ($E_s = E_b$ for BPSK and $E_s = 2E_b$ for (O)QPSK, where E_b denotes the received energy per bit) and N_o is the one–sided noise power spectral density at the receiver input. These coefficients allow one to calculate from (II.1.2–4) the linearized estimation error variance, when the synchronizer bandwidth is given.

Example

In order to illustrate the tracking analysis method we consider the example of the timing–dependent DDMLFB carrier synchronizer for BPSK (see section I.2.1.1). The corresponding phase error detector output is given by

$$u_\theta(k;\hat{\theta},\hat{\tau}) = \mathrm{Im}\,[\hat{a}_k\, p(k;\hat{\tau})\, e^{-j\hat{\theta}}] \qquad \text{(II.1.2–5)}$$

where $\hat{a}_k$ is the receiver's decision about the binary data symbol a_k, and $p(k;\hat{\tau})$ is the complex–valued matched filter output sample, taken at the instant $kT + \hat{\tau}$. In the case of BPSK, this matched filter output sample is given by :

$$p(k;\hat{\tau}) = \sum_m a_{k-m}\, g(mT + \hat{\tau} - \tau)\, e^{j\theta} + N_k(\hat{\tau}) \qquad \text{(II.1.2–6)}$$

where $g(t)$ is the baseband pulse at the matched filter output and $N_k(\hat{\tau})$ is the additive noise contribution; $\{N_k(\hat{\tau})\}$ is a sequence of independent complex–valued Gaussian random variables, with independent real and imaginary parts, each having a variance $N_o/(2E_s)$. Substituting (II.1.2–6) in (II.1.2–5) yields

$$u_\theta(k;\hat{\theta},\hat{\tau}) = \hat{a}_k \sum_m a_{k-m}\, g(mT + \hat{\tau} - \tau)\, \sin(\theta - \hat{\theta}) + \hat{a}_k \mathrm{Im}[N_k(\hat{\tau}) e^{-j\hat{\theta}}] \qquad \text{(II.1.2–7)}$$

For small estimation errors and large E_s/N_o, it can be assumed that the receiver's decisions are correct (i.e $\hat{a}_k = a_k$). Under this assumption, the synchronization surface $g_\theta(\phi,\epsilon)$ and loop noise $N_\theta(k;\phi,\epsilon)$ are given by

$$g_\theta(\phi,\epsilon) = E[a_k^2]\, g(\epsilon T)\, \sin(\phi) \qquad \text{(II.1.2–8)}$$

$$N_\theta(k;\phi,\epsilon) = a_k \sum_{m\neq 0} a_{k-m}\, g(mT - \epsilon T)\, \sin(\phi) + a_k \mathrm{Im}[N_k(\hat{\tau}) e^{-j\hat{\theta}}] \qquad \text{(II.1.2–9)}$$

The synchronization surface is zero for $\phi = 0$, even in the presence of a timing error ϵ : a timing error does not shift the location of the carrier phase tracking point, only the phase error detector slope is affected. Linearization of the synchronization surface about ($\phi = 0$, $\epsilon = 0$) yields :

$$g_\theta(\phi,\epsilon) \cong E[a_k^2]\, g(0)\, \phi = \phi \qquad \text{(II.1.2–10)}$$

where we have taken into account that $E[a_k^2] = 1$ and $g(0) = 1$. Hence, the phase error detector slope equals 1. The first term in the expression (II.1.2–9) of the loop noise is a self–noise term, which vanishes at the carrier phase tracking point $\phi = 0$; therefore, it does not enter the linearized tracking performance analysis. The second term in (II.1.2–9) is caused by additive noise; its power spectral density is constant, and equals $N_0/(2E_s)$, irrespective of $\hat{\tau}$ and $\hat{\theta}$. Hence the linearized tracking error variance is caused only by additive noise, and is given by

$$\mathrm{Var}[\phi] = (2B_L T)\,\frac{S(fT=0)}{(\mathrm{slope})^2} = (2B_L T)\,\frac{N_0}{2E_s} \qquad \text{(II.1.2–11)}$$

Until now, we have assumed that the receiver's decisions are correct. Taking decision errors into account, the phase error detector slope equals

$$E[\hat{a}_k a_k] = 1.\Pr[\hat{a}_k = a_k] + (-1).\Pr[\hat{a}_k = -a_k] = 1 - 2P_E \qquad \text{(II.1.2–12)}$$

where P_E is the receiver's bit error rate (BER). Decision errors give rise to a reduction of the phase error detection slope, and therefore, to an increase of the tracking error variance. The performance degradation due to decision errors becomes larger with decreasing E_s/N_0. This phenomenon is common to all DD synchronizers.

<u>The influence of timing errors on the tracking performance of carrier synchronizers</u>

Linearization of the synchronization surface about ($\phi = 0$, $\epsilon = 0$) does not give any indication about the influence of small timing errors on the tracking performance of carrier synchronizers, because the linearized synchronization surface contains only a term in the carrier phase error. The effect of timing errors on carrier synchronizers can be studied by assuming a fixed timing error ϵ, and linearizing the phase error detector output about $\phi = 0$. The resulting carrier

phase error variance, which depends on the timing error, is then averaged over the timing error.

This will now be illustrated by further considering the DDMLFB synchronizer for BPSK. Assuming that the receiver's decisions are correct, it follows from (II.1.2–8) that the phase error detector slope equals $g(\epsilon T)$, where ϵ denotes the fixed timing error. The corresponding carrier phase error variance, conditioned on the timing error, is given by

$$\begin{aligned}
\mathrm{Var}[\phi \mid \epsilon] &= (2B_L T)\frac{N_o}{2E_s}\frac{1}{g^2(\epsilon T)} \\
&\cong (2B_L T)\frac{N_o}{2E_s}\frac{1}{\left[1 - (-g''(0)T^2)\frac{\epsilon^2}{2}\right]^2} \\
&\cong (2B_L T)\frac{N_o}{2E_s}\left[1 + (-g''(0)T^2)\frac{\epsilon^2}{2}\right]^2 \\
&\cong (2B_L T)\frac{N_o}{2E_s}\left[1 + (-g''(0)T^2)\,\epsilon^2\right] \qquad \text{(II.1.2–13)}
\end{aligned}$$

where $g''(0)$ is the second derivative of $g(t)$, taken at $t = 0$. Averaging over the timing error yields

$$\mathrm{Var}[\phi] = (2B_L T)\frac{N_o}{2E_s}\left[1 + (-g''(0)T^2)\mathrm{Var}[\epsilon]\right] \qquad \text{(II.1.2–14)}$$

The quantity $(-g''(0)T^2)$ is an increasing function of the transmission bandwidth; for a cosine rolloff pulse, this quantity ranges from 3.29 (for $\alpha = 0$) to 5.16 (for $\alpha = 1$). Assuming that the r.m.s timing error equals 10% of the symbol period T, the effect of timing errors is an increase of the r.m.s carrier phase error by a factor of 1.026 at most; for instance, when the r.m.s carrier phase error in the absence of timing errors equals 10 degrees, the timing errors give rise to an r.m.s carrier phase error of 10.26 degrees only. This numerical example shows that timing errors have only a minor effect on the tracking performance of the DDMLFB carrier synchronizer for BPSK. Simulations reported in [Har'91] show that the phase error variance of the V&VFF carrier synchronizer for QPSK is only slightly increased when (static) timing errors are present. Using the same analytical method as for the DDMLFB carrier synchronizer for BPSK, it can be verified that most carrier synchronizers are quite insensitive to timing errors. Therefore, it is justified to analyze the carrier synchronizers under the assumption of perfect symbol timing.

The influence of phase errors on the tracking performance of symbol synchronizers

The influence of phase errors on the tracking performance of a symbol synchronizer can be investigated by considering a constant carrier phase error ϕ, linearizing the resulting timing error detector output about the tracking point $\epsilon=0$, evaluating the timing error variance conditioned on ϕ, and then averaging this conditional variance over ϕ. Using this approach, it can be verified that the timing error variance is only slightly increased when carrier phase errors are present. Hence, it is justified to evaluate the tracking error variance of a symbol synchronizer under the assumption of perfect carrier synchronization.

The influence of coding on the tracking performance.

The influence of convolutional and trellis coding on the tracking performance has been investigated in [Moe'90a]. The following results have been obtained, under very mild restrictions on the encoding and mapping rule :

– The tracking error variance of quadratic NDA synchronizers is the same as for uncoded transmission, for the same E_s/N_0 on the channel. Coding might have an influence on NDA synchronizers using higher order nonlinearities.

– At moderate and high E_s/N_0, the tracking error variance of DD synchronizers is the same as for uncoded transmission, for the same E_s/N_0 on the channel. At low E_s/N_0, the degradation due to decision errors is smaller than for uncoded transmission; indeed, the tentative Viterbi decoder decisions used in the case of coding are more reliable than the hard decisions used in the absence of coding.

II.1.3 Simulation analysis of tracking performance

For the simulation a similar approach as in the analysis has been taken. The various carrier phase and timing phase (error) detectors have been simulated to determine their performance. The phase detector parameters that have been determined are the expected value (the mean of the phase detector output), the total variance of the phase detector output (the loop noise variance), its power spectral density, and for more precision separately (with a narrow lowpass followed by a variance measurement) the power spectral density around f=0. From these figures the closed loop performance can be determined analytically in the same way as described in the previous section. So we have a mixed simulation/analysis ('hybrid') approach to performance evaluation rather than a pure simulation study in these cases.

In the ideal case considered in this part, synchronizer performance can be evaluated purely analytically in most cases. Where simulations have been used (i.e. for the hybrid approach) results are shown using markers (circles, triangles, etc.), except for the power spectra shown in Fig. II.2-1a,b and Fig. II.3-16. Analytical results are always shown using (solid, dashed, etc.) lines. Simulations have mainly been carried out for comparison and validation of simulation models, simulation methods, and analytical methods.

It is easily verified that the carrier phase detector performance is independent of the initial carrier phase value (but not independent of the carrier phase <u>error</u>). Therefore, a carrier phase shift has not been introduced in the simulation. Similar arguments apply for timing (considering an initial timing error).

Very important for the analysis is the slope of the phase error detector characteristics. Since the slope may depend on the signal to noise ratio it cannot just be determined for the noise-free case but must be determined as a function of the signal to noise ratio. Therefore, in the simulation a second phase detector has been used in parallel. For this phase detector a small phase error offset was introduced. The ratio of mean of the phase detector output to phase offset equals the slope. The offset must be selected appropriately. When the offset is too small, the simulation runs must be very long to get a sufficiently precise mean. When the offset is too large, the approximation of the slope tends to become smaller than the actual value due to the rounding of the detector S-curve.

Example

The above described method to determine the closed loop performance from the phase detector characteristic and loop noise power spectral density will be demonstrated for a NDAMLFB and a NDAMLFF synchronizer.

With linearization the variance of the carrier phase error σ_ϕ^2 is given by (II.1.2-1)

$$\sigma_\phi^2 = \mathrm{Var}[\phi] = \mathrm{Var}[\theta-\hat{\theta}] = \frac{1}{K_{\theta\theta}^2} \int_{-1/2}^{1/2} S(fT)\,|H(e^{j2\pi fT})|^2\, dfT$$

$$= \frac{S(0)}{K_{\theta\theta}^2}\,(2B_L T)$$

For a first order loop as shown in Fig. I.4-7a B_L is

$$B_L = \frac{K_F K_{\theta\theta}}{4T}$$

For an accumulator feedforward filter B_L is

$$B_L = \frac{1}{2TK}$$

Fig. II.1-2 shows curves determined analytically (see Fig. II.2-5 for the slope $K_{\theta\theta}$ and Fig. II.2-6 for the normalized loop noise power spectral density of the NDAML) and simulation results for the carrier phase error variance of a NDAML carrier synchronizer. Note that a closed feedback loop and the complete feedforward structure have been simulated.

Fig. II.1-2a compares results for a feedforward NDAML carrier synchronizer for QPSK, with an averaging filter noise bandwidth $B_L T = 0.05$ ($K = 10$) and a rolloff $\alpha = 0.5$. We find an excellent agreement between theory and simulation except for low E_s/N_o, where a deviation occurs. With the given filter noise bandwidth the noise level at the filter output is so strong at low E_s/N_o that linear analysis is no longer permissible. The argument function in the synchronizer yields a phase value $-\pi \leq \psi \leq -\pi$. After dividing by 4 the carrier phase estimate $\hat{\theta}$ is in the range $-\pi/4 \leq \hat{\theta} \leq -\pi/4$. For very low

E_S/N_0 the carrier phase estimate is uniformly distributed. Therefore, the variance can not exceed

$$\sigma_{\hat{\theta}}^2 = \frac{\Delta^2}{12} = (2\pi/4)^2/12 \text{ rad}^2 \approx 0.2 \text{ rad}^2$$

The simulation result for 0 dB is slightly below this bound (a synchronizer would never be operated with these parameters).

Fig. II.1-2b shows the results for the NDAML feedback carrier synchronizer for (O)QPSK with a loop noise bandwidth $B_L T = 0.001$ ($K_F K_{\theta\theta} = 0.004$). Again we find an excellent agreement. Since the loop noise bandwidth now is 50 times smaller than in the feedforward synchronizer example, theory and simulation also agree for low E_s/N_o.

II.2 TRACKING PERFORMANCE RESULTS FOR CARRIER SYNCHRONIZATION

II.2.1 Introduction

In this section the performance of the various carrier synchronizer algorithms will be studied. The steps to obtain the analytical results and the simulation results have been demonstrated in II.1.2 and II.1.3, respectively. For the DDML and NDAML results have been determined analytically. For the V&V and for synchronizers which are independent of timing, only special cases can be treated analytically. There, simulation has been used to determine the performance (hybrid approach). Again note that with one exception (the power spectra shown in Fig. II.2-1a,b) simulation results are shown as markers (stars, dots, squares, etc.) while analytical results are shown as (solid, dashed, dotted etc.) lines.

As discussed the performance analysis is separated in two steps. Here, the carrier phase (error) detector performance is determined, i.e. the phase detector characteristics and the properties of the detector output noise (e.g. power spectral density, variance). From these data for any given filter the error variance after correction can be calculated as shown in section II.1.2.

When the power spectral density of the loop noise is flat the carrier error variance is

$$\sigma_\phi^2 = \mathrm{Var}[\phi] = \mathrm{Var}[\theta-\hat{\theta}] = 2B_L T \frac{S(fT=0)}{K_{\theta\theta}^2} \qquad \text{(II.2.1-1)}$$

But (II.2.1-1) is also a good approximation when $H(e^{j2\pi fT})S(fT)$ contributes significantly to the integral in (II.1.2-1) only where S(f) is nearly flat. This is often the case when $H(z=e^{j2\pi fT})$ is narrowband. Therefore, in the following sections we will study S(fT=0). The effect of a non-flat power spectral density on loop performance will be analysed later.

The Cramer-Rao (CR) Bound

An important reference for the synchronizer performance is the Cramer Rao bound which is a lower bound on the error variance for any synchronizer. For an observation interval of length T (one symbol) the Cramer Rao bound is obtained as

$$\sigma_\phi^2 \geq \frac{N_0}{2E_S} \qquad ; \quad E_s\text{: energy per symbol}$$

Note that this bound applies for all three modulation schemes.

Similarly with an equivalent one-sided loop noise bandwidth B_L the Cramer Rao bound is

$$\sigma_\phi^2 \geq \frac{N_0}{2E_S}\,(2B_L T)$$

II.2.2 DDML, Decision Directed Maximum Likelihood carrier synchronizer. Timing assumed to be known.

FeedBack

For the analysis of the DDMLFB it is very instructive to observe the matched filter samples normalized to $\sqrt{E_s}$ after phase rotation. For BPSK and QPSK the output equals

$$\frac{p(k;\tau=\hat{\tau})}{\sqrt{E_S}}\, e^{-j\hat{\theta}} = c_k e^{j(\theta-\hat{\theta})} + N_k \qquad \text{(II.2.2-1)}$$

$$|c_k| = 1 \quad ; \quad a_k,\, b_k \in \{+1,-1\} \quad ; \quad E[N_k N_{k+\ell}^*] = \frac{N_0}{E_S}\,\delta_\ell$$

$$\delta_\ell = \begin{cases} 1 & \ell = 0 \\ 0 & \ell \neq 0 \end{cases}$$

BPSK: $c_k = a_k \quad ; \quad q() = 0$

QPSK: $c_k = (a_k + jb_k)/\sqrt{2} \quad ; \quad q() = p()$

and for OQPSK

$$\frac{p(k;\tau=\hat{\tau})}{\sqrt{E_S}} e^{-j\hat{\theta}} = \frac{1}{\sqrt{2}} [a_k + j \sum_m b_m g_{k-m-1/2})] e^{j(\theta-\hat{\theta})} + N_k$$

(II.2.2-2)

$$\frac{q(k;\tau=\hat{\tau})}{\sqrt{E_S}} e^{-j\hat{\theta}} = \frac{1}{\sqrt{2}} [\sum_m a_m g_{k-m+1/2} + jb_k] e^{j(\theta-\hat{\theta})} + N_{k+1/2}$$

$$g_k = g(kT)$$

In the DDMLFB synchronizers these results are multiplied by the decisions $\hat{a}_k - j\hat{b}_k$ to get the carrier phase error detector signal.

The power spectral density of the noise at the matched filter output equals the squared magnitude of the receive filter transfer function. Therefore, the autocorrelation function of the noise equals $g(\tau) = h(\tau) \odot h^*(-\tau)$ and the autocorrelation function of the noise samples is

$$E[N_k N^*_{k+m/2}] = \frac{N_0}{E_S} g_{m/2}$$

For BPSK and QPSK the loop noise power spectral density is flat because the loop noise samples N_k are uncorrelated (for the root Nyquist receive filter assumed here)

$$E[N_k N^*_{k+l}] = \frac{N_0}{E_S} \delta_l$$

Also, since $|c_k|=1$ there is no self-noise for BPSK and QPSK.

With OQPSK the noise samples of p() and q() are correlated. But in the phase detector output signal the imaginary part of N_k and the real part of $N_{k+1/2}$ occur. They again are uncorrelated. Thus, this part of the phase detector output noise spectrum is flat. But the quadrature disturbance in (II.2.2-2) causes self-noise which has not a flat spectrum.

In Fig. II.2-1 loop noise power spectral densities of DDMLFB are shown for BPSK, QPSK, and OQPSK. Fig. II.2-1a shows analytically determined loop noise power spectral densities and, for comparison, simulated curves, the spectra of Fig. II.2.1-b are simulation results only. For BPSK and QPSK the loop noise power spectral density is flat (Fig. II.2-1a) but for OQPSK the influence of the self-noise is clearly visible. We observe that the curves for E_S/N_0 = 6 dB have the same shape as the curves for E_S/N_0 = 16 dB (except for a shift towards larger variance). This further justifies the approach to neglect the additive noise contribution to the second term in (II.1.2-4). We also observe that the self-noise contribution increases with decreasing rolloff. This will be briefly explained without going into detail. The variance of the self-noise depends on $g_{k-1/2}$, i.e. on the raised cosine filter impulse response sampled in the middle between the correct sampling instants. Since the tails are stronger for small rolloff, the variance is larger for small rolloff.

Neglecting decision errors, the phase detector characteristics are proportional to $\sin(\theta-\hat{\theta})$ for all three modulation schemes. When the decision error probability is small, i.e. for high E_s/N_0, the slope approaches 1. With increasing decision errors the slope decreases (see Fig. II.2-2). How much the slope decreases depends on the decision error probability and on the way a particular decision error influences the mean of the phase detector output. From this we can expect that the slope is smaller for QPSK than for BPSK, since the decision error probability is much larger. This is indeed found in Fig. II.2-2. We also have the interesting fact that the OQPSK slope equals the QPSK slope for rolloff $\alpha = 0$ and is larger for $\alpha > 0$. It is difficult to give an intuitive reasoning for the ordering of the slope curves without carrying out a detailed analysis of the DDML synchronization algorithm performance. This analysis is tedious and lengthy, in particular for OQPSK. Since it would interrupt the outline of this chapter too much, we do not give it here.

For low E_S/N_0 decision errors also influence the loop noise variance (and the power spectral density) for QPSK and OQPSK since there is a correlation between phase detector noise and decision errors. For instance, in the case of QPSK the decision feedback will limit the phase of the detector signal to $\pm\pi/4$ (a larger phase will cause a decision on a symbol that is within $\pm\pi/4$ of the measured total phase). This explains why the QPSK

power spectral density is slightly below the BPSK power spectral density at E_S/N_0 = 6 dB (Fig. II.2-1).

In Fig. II.2-3a and Fig.II.2.3b the normalized power spectral density at fT=0 is shown (thus, self-noise is neglected for the moment but will be considered later). We find the same ordering as for the slopes: BPSK is best, QPSK is worst, OQPSK performance is between BPSK and QPSK. All three curves approach the Cramer Rao bound for moderate to high E_S/N_0 (approx. at 10 dB). Therefore, a good approximation for the error variance at moderate and high E_S/N_0 is

$$\sigma_\phi^2 \simeq \frac{N_0}{2E_S}(2B_LT) \qquad \text{(neglecting self-noise)}$$

<u>FeedForward</u>

As discussed in sections I.3.2 and II.1.2 the DDMLFF synchronizer has the same tracking performance as the DDMLFB feedback synchronizer.

II.2.3 NDAML, Non Data-Aided Maximum Likelihood synchronizer. Timing assumed to be known

<u>FeedBack</u>

Again, the matched filter samples after phase rotation given in II.2.3 are very helpful. For BPSK and QPSK the power spectral densities of the phase detector noise remain flat since the noise samples are uncorrelated (and remain uncorrelated after squaring or quadrupling). Also, there is no self-noise for BPSK and QPSK since $c_k^2 = 1$ (BPSK) and $c_k^4 = -1$ (QPSK), respectively. The loop noise power spectral density now is larger than $N_0/(2E_S)$ due to squaring loss (BPSK) and quadrupling loss (QPSK).

With OQPSK the power spectral density is not flat since the quadrature disturbance introduces self-noise and a correlation between the samples. In Fig. II.2-4 the analytically determined loop noise power spectral density for OQPSK is shown for a rolloff $\alpha = 1$. The curve, where self-noise is neglected clearly shows the effect of the correlation on the power spectral density.

From (II.2.2-1) it is easily derived that the phase detector characteristics for BPSK and QPSK are sinusoidal with unit slope, independent of E_S/N_0 and independent of the rolloff α. (see Fig. II.2-5). For OQPSK we also have a sinusoidal phase detector characteristic. The slope is a function of the rolloff α but still independent of E_S/N_0. This is readily verified when we determine the expected value of the phase detector output from (II.2.2-2)

$$E\left[\frac{1}{2} \frac{1}{E_S} \operatorname{Im}\left\{(p^2 - q^2)e^{-j2\hat{\theta}}\right\}\right] =$$

$$= \frac{1}{4} [2 - E[(\sum_m a_m g_{k-m+1/2})^2] - E[(\sum_m b_m g_{m-k-1/2})^2] \sin 2(\theta-\hat{\theta})$$

$$= \frac{\alpha}{4} \sin 2(\theta-\hat{\theta})$$

In Fig. II.2-6a and Fig. II.2-6b the power spectral density at $fT = 0$ normalized to $K^2_{\theta\theta}$ is shown. For high E_S/N_0 both BPSK and QPSK approach the Cramer Rao bound. For low E_S/N_0 a degradation due to squaring/quadrupling loss occurs. Since the quadrupling loss is larger than the squaring loss, the performance of the QPSK synchronizer is more degraded than the performance of the BPSK synchronizer. For OQPSK the performance curves are almost parallel to the BPSK curves when displayed in a logarithmic scale as in Fig. II.2-6a, i.e. they do not approach the Cramer Rao bound for high E_S/N_0. Instead a degradation proportional to $(2/\alpha)$ occurs. Thus, this NDA synchronizer does not work with zero rolloff. At low E_S/N_0 squaring loss occurs as for BPSK.

<u>FeedForward</u>

As discussed in sections I.3.2 and II.1.2 the NDAMLFF synchronizer has the same tracking performance as the NDAMLFB feedback synchronizer.

II.2.4 V&V, Non data-aided Viterbi&Viterbi carrier synchronizer. Timing assumed to be known.

BPSK, QPSK

In an NDAML carrier synchronizer for M-ary PSK the sampled matched filter output is raised to the M-th power to remove the data modulation (i.e. the symbol phase). Passing a complex signal x through a nonlinearity $f(x) = x^M$ corresponds to multiplying the phase by M and raising the amplitude to the M-th power. The V&V offers a new degree of freedom, the phase is still multiplied by M to remove the data modulation but for the amplitude an arbitrary nonlinearity may be used. So far mainly $f(u) = u^N$ has been investigated in any detail. Note that for N = M the NDAML algorithm is obtained.

Analytical tracking performance results for even powers N are given in [Vit'83] where the synchronizer algorithm is also proposed. From the performance figures given in this reference it can be concluded that the best tracking performance results for small values of N, i.e. $N \leq 2$. For moderate to high E_S/N_0 the performance becomes practically independent of the type of nonlinearity. The influence of the nonlinearity type is significant for low E_S/N_0 only.

OQPSK

In part I of this report an extension of the V&V for OQPSK is proposed. Analytical performance cannot be given. Therefore, the V&V performance for OQPSK was investigated by simulation. As above only the nonlinearities $f(u) = 1$ and $f(u) = u$ are considered here. Note that the case $f(u) = u^2$ corresponds to NDAML.

FeedForward/Feedback

Although the V&V algorithm is originally derived as a feedforward carrier synchronizer a feedback algorithm can be given by analogy with NDAML. For BPSK (M=2) and QPSK (M=4) the feedback synchronization algorithm is given by

$$u_\theta(k;\hat{\theta};\hat{\tau}) = \frac{1}{M} \operatorname{Im} \left\{ F(|p(k;\hat{\tau})|) \exp(jMp(k;\hat{\tau})) \, e^{-jM\hat{\theta}} \right\}$$

For the same reasons as discussed in sections I.3.2 and II.1.2 the V&VFF synchronizer has the same tracking performance as the V&VFB synchronizer. Therefore, the normalized power spectral densities apply for both types.

In Fig. II.2-7a,b the slopes of the V&VFB are shown. For BPSK and QPSK the phase detector slope is independent of the rolloff (Fig. II.2-7a). For OQPSK (Fig. II.2-7b) the slope depends on the rolloff with the largest value for rolloff $\alpha = 1$ (at a given E_S/N_0). In Fig. II.2.8a the normalized loop noise power spectral density at $fT = 0$ is shown for BPSK and QPSK for different nonlinearities f(u). For BPSK the performance curves are very close for the different nonlinearities with slightly increasing degradations in the ordering $f(u) = u, u^2, 1$. For QPSK larger differences than for BPSK occur, with best performance for $f(u) = u^2$ and u, small degradations for $f(u) = u^4$, and slightly larger degradations for $f(u) = 1$. But note that for high E_S/N_0 $f(u) = 1$ approaches optimum performance faster than $f(u) = u^4$. For OQPSK the performance curves are very close for the different nonlinearities with slightly increasing degradations in the ordering $f(u) = u, u^2, 1$, see Fig. II.2-8b. So the V&V performance for $f(u) = u$ is better than the NDAML (which corresponds to the V&V with $f(u) = u^2$). As with the NDAML performance degrades with decreasing rolloff. Note that the V&V for $f(u) = u$ and $f(u) = 1$ also operates with rolloff $\alpha = 0$ (not shown here) while the NDAML fails.

It is interesting to compare DDML, NDAML, and V&V for $f(u) = u^2$. The normalized power spectral density for these cases is shown in Fig. II.2-8c. We observe that for low E_S/N_0 the V&V performance is best while for high E_S/N_0 all three synchronizers approach the Cramer-Rao bound.

II.2.5 DDMLT̄, Decision Directed Maximum Likelihood carrier synchronization independent of symbol timing

As before the feedback version of the algorithm, i.e. DDMLFBT̄, is discussed but the tracking performance results also apply for the feedforward algorithm.

As prefilter a matched filter is assumed.

The performance of the DDMLT̄ can be analyzed for some special cases under idealized conditions (i.e. neglible decision error probability) only. Therefore, mainly simulation is used to obtain performance figures. We start with the approximate analysis and then compare it to the simulation results.

Analytical results

The following results apply for all three modulation schemes.

First we consider the case that the matched filter is sampled at one sample per symbol but with a deviation e ($|e| \leq T/2$) from the optimum sampling instant. An exact analysis is difficult but an approximation for high E_S/N_0 can be given easily. For high E_S/N_0 (when the influence of decision errors is negligible) the slope of the phase detector approximately equals g(e).

Thus, neglecting self-noise and decision errors the normalized power spectral density at fT=0 is

$$\frac{S(fT=0)}{K^2_{\theta\theta}} \simeq \frac{N_0}{2E_S}\frac{1}{g^2(e)} \leq \frac{N_0}{2E_S}\frac{1}{g^2(T/2)} \leq \frac{N_0}{2E_S}\,4$$

(the minimum of g(e) for $|e| \leq T/2$ is g(T/2) and the minimum of g(T/2) over all α is 1/2 which occurs for $\alpha = 1$).

Thus, a maximum degradation of 6 dB with respect to the Cramer Rao bound occurs for high E_S/N_0. When e is close to T/2, E_S/N_0 must be very high to have a negligible decision error probability. For E_S/N_0 values of interest, performance is degraded severely around $e \approx T/2$. Therefore, when

symbol timing is not (yet) available the matched filter output has to be sampled at a higher rate, M/T. For M even the sampling instants then are

$$kT + \frac{2i+1}{2M} T + e \quad ; \quad |e| < \frac{T}{2M}$$

$$i: -M/2, -M/2+1, \ldots , M/2-1$$

Again assuming correct decisions (i.e. for high E_S/N_0) the slope approximately equals

$$\frac{1}{M} \sum_{m=-M/2}^{M/2-1} g(\frac{2i+1}{2M} T + e)$$

With the same approximation and again neglecting self-noise, because of the correlation of the noise samples, for the normalized power spectral density at fT=0 is obtained

$$S(fT) = \frac{1}{M^2} \sum_{m=-M+1}^{M-1} (M-|m|)\ g(mT/M)\ \frac{N_0}{2E_S}$$

With M = 2 the slope approximately equals

$$\frac{1}{2} (g(T/4 + e) + g(-T/4 + e))$$

and the power spectral density is

$$S(fT=0) = \frac{1}{4} (2g(0) + 2g(T/2)) \frac{N_0}{2E_S}$$

For the normalized power spectral density is obtained.

$$\frac{S(fT=0)}{K_{\theta\theta}^2} = \frac{[2(g(0)+g(T/2)]\ \frac{N_0}{2E_S}}{[g(T/4+e)+g(-T/4+e)]^2} \leq \frac{2\ N_0/(2E_S)}{g(0)+g(T/2)} \leq \frac{4}{3} \frac{N_0}{2E_S}$$

Therefore, in the high E_S/N_0 limit a maximum degradation of about 1.25 dB with respect to the Cramer Rao bound occurs.

Finally, in the limit of continuous time operation the corresponding high E_S/N_0 approximation for the slope equals

$$\frac{1}{T}\int_{-T/2}^{T/2} g(t)dt$$

and the power spectral density at fT=0 is

$$S(fT) \simeq \frac{1}{T^2}\int_{-T}^{T}(T-|t|)g(t)dt\ \frac{N_0}{2E_S}$$

Simulation

In the simulations we have assumed that a clock frequency deviation is present. The timing error has been advanced in steps of T/16, starting with zero timing error. The performance figures given in Fig. II.2-9a,b,c and Fig. II.2-10a,b,c were obtained by averaging the slope and the loop noise power spectral density for all timing errors. Using this averaging over the timing error corresponds to the case of a clock frequency error. Another interesting case is a constant timing error which corresponds to the case of a very small or negligible clock frequency error. In this case we will have a timing error for which we have worst case performance and a timing error for which we have best case performance. Except for synchronizers with one sample per symbol it is difficult to determine the worst and best case timing errors. They may not only be different for the individual synchronizers, they also depend on E_s/N_0. Since this would cause an enormous amount of work we have used the averaging method here only in the simulations. The deviation of worst and best case from this average case is largest for two samples per symbol. It will decrease significantly for a larger number of samples per symbol since the actual timing error range equals approximately ±1/(2*number_of_samples_per_symbol).

In Fig. II.2-9a,b,c the phase detector slopes obtained by the simulations are given for BPSK, QPSK, and OQPSK, respectively. The slopes are shown for 2 samples per symbol (t_s=T/2) and for 4 samples per symbol (t_s=T/4). A significant difference is not found. Note that for OQPSK the number of

samples per symbols refers to the number of samples per symbol processed by the decision device and the subsequent phase detector components.

In Fig. II.2-10a,b,c the loop noise power spectral densities at $fT = 0$ obtained by the simulations are given for BPSK, QPSK, and OQPSK, respectively. The spectral densities are shown for 2 samples per symbol (t_s=T/2) together with the DDMLT curves. For BPSK the performance is close to DDMLT, for QPSK we find a degradation with respect to DDMLT which increases with E_S/N_0, and for OQPSK performance is significantly degraded at all E_S/N_0 (compare Fig. II.2-10c and Fig. II.2-3a,b). Note that again for OQPSK the number of samples per symbols refers to the number of samples per symbol processed by the decision device and the subsequent phase detector components or, equivalently, the number of (p,q) sample pairs per symbol. Thus, it is 1/2 of the sampling rate at the matched filter output.

At high E_S/N_0 the averaging method used here (rather than true integration) may give too pessimistic values because $e = 1/2$ where the decision error probability is 1/2 is always included. In practice, the range around $e = 1/2$ where the decision error probability is close to 1/2 becomes smaller with increasing E_S/N_0. Thus, the probability that e is in the range where the decision errors influence the performance decreases.

II.2.6 NDAMLT̄, Non data-aided maximum likelihood carrier recovery independent of symbol timing

As before the feedback version of the algorithm, i.e. NDAMLFBT̄, is discussed but the tracking performance results also apply for the feedforward algorithm.

As prefilter a matched filter is assumed.

The performance of the NDAMLT̄ can be analyzed for BPSK only. Therefore, simulation is used to complement the BPSK performance figures and to obtain performance figures for QPSK and OQPSK. In the simulations we have assumed that a clock frequency deviation is present. The timing error has been advanced in steps of T/16, starting with zero timing error.

The phase detector slopes and loop noise power spectral densities are averaged for all timing errors.

BPSK

The matched filter output signal is bandlimited to $\pm(1+\alpha)/2T$. After squaring a power spectral density as shown in Fig. II.2-11 is obtained. Spectral lines are present at f=0 and $f=\pm 1/T$ and the spectrum is spread to $\pm(1+\alpha)/T$.

For continuous time operation only (1) contributes to the spectrum at f=0. (see Fig. II.2-11). With non synchronized sampling at a rate 2/T again only (1) contributes to the spectrum of the sampled signal at f=0 since (2), the spectrum at f=2/T, equals zero. For non synchronized sampling at a rate 1/T (1) and (3) contribute to the spectrum of the sampled signal at f=0. (again, (2) equals zero). Thus, in this case a sampling time dependent aliasing occurs except for the case where the rolloff $\alpha=0$.

To summarize, for BPSK no aliasing occurs at f=0 for $\alpha\neq 0$ when the sampling rate is 2/T or higher, and for $\alpha=0$ when the sampling rate is 1/T or higher. Thus, in this case, sampled BPSK clockless carrier synchronization has the same performance as continuous time operation. For a sampling rate of 1/T and rollof $\alpha \neq 0$ performance depends on the timing error $e = \tau-\hat{\tau}$, extreme cases are NDAMLT (best case) and e=T/2 (worst case).

The phase detector characteristic is sinusoidal for all cases. For a sampling rate of 1/T the slope is a function of the timing error e as shown in Fig. II.2-12a. The slope is independent of the timing error for $\alpha=0$ (see Fig. II.2-12a) or sampling rates of 2/T and higher.

The normalized power spectral densities for continuous-time operation (and, thus, also for sampled operation without aliasing) and unsynchronized sampling at rate of 1/T are shown in Fig. II.2-13a in a logarithmic scale and in Fig. II.2-13b in a linear scale. For a rolloff $\alpha=0$ performance is identical (and best) independent of the type of operation. For $\alpha\neq 0$ performance of the continuous-time synchronizer is between best and worst case of unsynchronized sampling at a rate of 1/T but very close to the best case.

The averaged slopes obtained by simulation, see Fig. II.2-12b, are close to the curves for $t_0 = T/4$ in Fig. II.2-12a (The slope for $\alpha=0$ is slightly smaller than 1 because with the limited FFT length to be used in a simulation the actual rolloff differs from 0. The normalized power spectral densities determined by simulation, see Fig. II.2-13c, also show the predicted behaviour. For 2 samples per symbol the simulated performance is identical to the theoretical performance curves for continuous-time operation. For 1 sample per symbol and rolloff $\alpha=0$ we again find the same result as for continuous-time operation. For 1 sample per symbol and $\alpha\neq 0$ the curves are above the continuous-time operation curves.

For QPSK and OQPSK the analysis is more difficult. The performance at sampling errors between 0 and T/2 is mathematically not tractable. Therefore, only the special case e=T/2 is considered in the analysis.

<u>QPSK</u>

For $\alpha = 1$ and $e = T/2$ the slope equals 1/8 and the power spectral density at $fT = 0$ is

$$S(fT=0) = \frac{5}{16}\frac{N_0}{2E_S} + \text{higher order terms.}$$

Thus, in this worst case the normalized power spectral density is obtained as

$$\frac{S(fT=0)}{K^2_{\theta\theta}} = 20\,\frac{N_0}{2E_S} + \text{higher order terms}$$

which is about 13 dB above the Cramer Rao bound.

The (averaged) normalized power spectral densities determined by simulation are shown in Fig. II.2-13d. For 2 samples per symbol the simulated performance is better than for 1 sample per symbol. The degradation depends on the rolloff in both cases. Due to the averaging over the timing error the degradation is much less than 13 dB for 1 sample per symbol.

OQPSK

Using quadrupling for NDA OQPSK carrier recovery with sampling at kT (or kT + T/2) the slope equals -3/8 for rolloff $\alpha = 1$. With

$$S(fT=0) = \frac{9}{16}\frac{N_0}{2E_S} + \text{higher order terms}$$

for the normalized power spectral density is obtained

$$\frac{S(fT=0)}{K^2_{\theta\theta}} = 4\frac{N_0}{2E_S} + \text{higher order terms}$$

which is about 6 dB above the Cramer Rao bound.

The (averaged) normalized power spectral densities determined by simulation are shown in Fig. II.2-13e. As with QPSK the simulated performance is significantly better for 2 samples per symbol than for 1 sample per symbol. But performance is still far below NDAMLT for low E_S/N_0. Since the deviation is larger than 6 dB the worst case must appear for $e \neq 0$, T/2.

II.2.7 NDASQFFTO, MSK-type carrier recovery

The NDASQFFTO synchronizer for OQPSK and the NDAML have the same phase error variance (additive noise contribution and self-noise contribution). Indeed, it can be verified by linearizing the function arg(·) (around the mean) in the expressions of the carrier phase estimates that both synchronizers yield the same (linearized) estimates, and, therefore, the same carrier phase error variance (see Fig. II.2-6a,b). This verification is straightforward but lengthy and is not given.

II.2.8 Self-noise performance.

For the DDML and NDAML carrier synchronization with known timing self-noise occurrs only for OQPSK.

As explained in section II.1.2, the self-noise contribution to the tracking error variance is given by

$$\text{self-noise contribution} = K_F\ (2B_LT)^2\ (\sum_m |m|(-R(m)))/(\text{slope})^2$$

where $\{R(m)\}$ is the autocorrelation sequence of the self-noise component of the loop noise, and K_F is a constant of proportionality, depending on the loop filter (feedback carrier synchronization); the expression is also valid for feedforward carrier synchronizers.

Table II.2-1 shows the quantity $\Sigma|m|(-R(m))/(\text{slope})^2$, corresponding to DDML and NDAML carrier synchronizers, for rolloffs of 10%, 50% and 100%. Also shown is the quantity $(E_s/N_o)_c$, which is defined as the value of E_s/N_o at which additive noise and self-noise equally contribute to the tracking error variance; roughly speaking, self-noise is important only when E_s/N_o exceeds $(E_s/N_o)_c$. As the self-noise and additive noise contributions are proportional with $K_F(2B_LT)^2$ and $2B_LT$, respectively, $(E_s/N_o)_c$ is a decreasing function of the quantity $K_F(2B_LT)$. In most cases of practical interest, $2B_LT$ is so small that that self-noise becomes important only at large values of E_s/N_o, for which the corresponding additive noise contribution to the tracking error variance is essentially inversely proportional to E_s/N_o :

$$\text{additive noise contribution} = (2B_LT)\ A\ (N_0/2E_S)$$

with A=1 for DDML and $A=2/\alpha$ for NDAML. Under these assumptions, $(E_s/N_o)_c$ is inversely proportional with $K_F(2B_LT)$:

$$(E_s/N_o)_c = \frac{A\,(\text{slope})^2}{2\sum_m |m|(-R(m))}\ \frac{1}{K_F\ (2B_LT)}$$

Table II.2-1 gives the above $(E_s/N_o)_c$ for $K_F = 2$ and $2B_LT = 10^{-2}$. Values of $(E_s/N_o)_c$ corresponding to other values of K_F and/or $2B_LT$ are easily derived; for instance, when $K_F = 1$ and $2B_LT = 10^{-3}$, $(E_s/N_o)_c$ is obtained by adding 13 dB to the tabulated $(E_s/N_o)_c$ values.

Tab. II.2-1: Self-noise performance of carrier synchronizers (OQPSK), analytical results for DDML and NDAML, simulation results for V&V

	$\Sigma\|m\|(-R_m)/(\text{slope})^2$		$(E_s/N_o)_c$ @ $2B_LT=10^{-2}$, $K_F=2$
DD ML			
$\alpha = 0.1$	0.368	(-4.3 dB)	18.3 dB
$\alpha = 0.5$	0.203	(-6.9 dB)	20.9 dB
$\alpha = 1.0$	0.125	(-9.0 dB)	23.0 dB
NDA ML			
$\alpha = 0.1$	147	(21.7 dB)	5.3 dB
$\alpha = 0.5$	3.24	(5.1 dB)	14.9 dB
$\alpha = 1.0$	0.50	(-3.0 dB)	20.0 dB
V&V, f(u)=1 (simulation)			
$\alpha = 0.1$	10.9	(10.4 dB)	
$\alpha = 0.5$	2.36	(3.7 dB)	
$\alpha = 1.0$	0.46	(-3.4 dB)	
V&V, f(u)=u (simulation)			
$\alpha = 0.1$	23.0	(13.6 dB)	
$\alpha = 0.5$	1.79	(2.5 dB)	
$\alpha = 1.0$	0.94	(-0.3 dB)	

It is found that the self-noise of the NDAML and V&V strongly depends on the rolloff.

For low E_S/N_0 and stable carrier frequency the loop noise bandwidth will be small. For such applications, e.g. in satellite communications, the self-noise in general is negligible because of the small B_L.

II.3 TRACKING PERFORMANCE RESULTS FOR SYMBOL SYNCHRONIZATION

II.3.1 Introduction

As explained in section II.1.2, the timing error variance is obtained under the assumption of perfect carrier synchronization (ϕ=0). In sections II.3.2 and II.3.3 we evaluate the quantity $S(fT=0)/(\text{slope})^2$ for various DD and NDA symbol synchronizers; this quantity determines the additive noise contribution to the timing error variance. The self–noise contribution to the timing error variance is determined by the quantity $\Sigma(-|m|R(m))/(\text{slope})^2$, which is evaluated in section II.3.4. These quantities are to be used in (II.1.2–4) in order to compute the timing error variance for a given synchronizer bandwidth. All results are obtained analytically; a number of cases have been checked by computer simulations.

<u>Cramer–Rao bound</u>

The additive noise contribution to the timing error variance (i.e. the linear bandwidth term in (II.1.2–4)) resulting from any symbol synchronizer is lower bounded by the Cramer–Rao bound (CRB) :

$$\frac{S(fT=0)}{(\text{slope})^2} \geq \frac{N_0}{2E_s} \frac{1}{(-g''(0)T^2)} \qquad \text{(II.3.1–1)}$$

where $g''(t)$ denotes the second derivative, taken at t=0, of the baseband pulse $g(t)$ at the matched filter output. When $g(t)$ is a cosine rolloff pulse with rolloff α, the quantity $(-g''(0)T^2)$ ranges from 3.29 (for α=0) to 5.16 (for α=1). The CRB is inversely proportional with E_s/N_0. For all considered synchronizers, we will compare the actual $S(fT=0)/(\text{slope})^2$ with the CRB.

<u>Carrier–dependent symbol synchronization for BPSK and (O)QPSK</u>

In general, symbol synchronization of carrier modulated signals is performed by first converting the received bandpass signal to baseband, with use of the carrier phase estimate, and then applying a symbol synchronization algorithm for baseband signals to the resulting in–phase signal (for BPSK) or to both in–phase and quadrature signals (for (O)QPSK).

For a given E_b/N_o, the timing error variance resulting from a specific carrier–dependent symbol synchronization algorithm for BPSK can be represented as

$$\mathrm{Var}[\epsilon]\big|_{BPSK} = A + B\frac{N_o}{E_b} + F(N_o/E_b) \qquad \text{(II.3.1–2)}$$

where A is the contribution caused by self–noise, $B\,N_o/E_b$ is the dominating contribution at moderate E_b/N_o, and $F(N_o/E_b)$ is a more than linearly increasing function of N_o/E_b, which represents the dominating contribution at low E_b/N_o. In the case of (O)QPSK, symbol synchronization is obtained by combining the information from two identical algorithms for BPSK, operating on the in–phase and quadrature components after phase rotation. For given E_b/N_o, each of these components has the same SNR as the in–phase component for BPSK; therefore, the timing error variance for (O)QPSK is half the timing error variance for BPSK :

$$\mathrm{Var}[\epsilon]\big|_{(O)QPSK} = \frac{A}{2} + \frac{B}{2}\frac{N_o}{E_b} + \frac{1}{2}F(N_o/E_b) \qquad \text{(II.3.1–3)}$$

Taking into account that $E_s=E_b$ for BPSK and $E_s=2E_b$ for (O)QPSK, the timing error variances for a given E_s/N_o are given by

$$\mathrm{Var}[\epsilon]\big|_{BPSK} = A + B\frac{N_o}{E_s} + F(N_o/E_s) \qquad \text{(II.3.1–4a)}$$

$$\mathrm{Var}[\epsilon]\big|_{(O)QPSK} = \frac{A}{2} + B\frac{N_o}{E_s} + \frac{1}{2}F(2N_o/E_s) \qquad \text{(II.3.1–4b)}$$

Hence, for a given E_s/N_o, the comparison between $\mathrm{Var}[\epsilon]\big|_{BPSK}$ and $\mathrm{Var}[\epsilon]\big|_{(O)QPSK}$ is as follows :

– The self–noise contribution for (O)QPSK is half the self–noise contribution for BPSK.

– All three modulation formats yield the same dominating contribution (i.e. $B\,N_o/E_s$) at moderate E_s/N_o.

– At low E_s/N_o the timing error variance for (O)QPSK is larger than for BPSK. This is because $F(2x) > 2F(x)$ for a function F(.) which increases more than linearly with x. For instance, for quadratic symbol synchronizers we have $F(x) = C\,x^2$, so that the dominating contribution at low E_s/N_o for (O)QPSK is twice as large as for BPSK.

The above comparison is not valid for the symbol synchronizers which operate independently of the carrier phase estimate.

Closely related symbol synchronizers

In section I.2.2.4 we have identified three sets of "closely related" symbol synchronizers :

– DDEL(λ=1/2), DTTL

– DDEL(λ=1), M&M, mM&M

– NDAEL(λ=1/2), GA

Closely related synchronizers yield timing error detector outputs with the same long term arithmetical average, or, equivalently, with the same content at fT=0. It follows that these synchronizers have the same synchronization surface $g_{\tau}(\phi,\epsilon) = E[u_{\tau}(k;\hat{\theta},\hat{\tau})]$ and the same DC–value S(fT=0) of the loop noise spectrum. Hence, they yield the same $S(fT=0)/(\text{slope})^2$, and, therefore, the same additive noise contribution to the timing error variance. Closely related synchronizers usually have a different self–noise contribution to the timing error variance; this is because the self–noise contribution depends on the shape of the loop noise spectrum, which might be different for the various closely related synchronizers.

II.3.2 DD symbol synchronization

In this section we evaluate $S(fT=0)/(\text{slope})^2$ for various DD symbol synchronizers. First we give some general properties, then we consider specific algorithms.

General properties

Let us consider the DDMLFB symbol synchronizer for BPSK. Assuming perfect carrier synchronization ($\phi=0$), the corresponding timing error detector output is given by

$$u_\tau(k;\theta,\hat{\tau}) = \hat{a}_k \, \mathrm{Re}[p'(k;\hat{\tau}) \, e^{-j\theta}] \qquad \text{(II.3.2–1)}$$

where $\hat{a}_k = \mathrm{sgn}(\mathrm{Re}[p(k;\hat{\tau}) \; e^{-j\theta}]$ is the receiver's decision. Using the decomposition

$$\mathrm{Re}[p(k;\hat{\tau}) \, e^{-j\theta}] = A(k;\epsilon,\underline{a}) \; + \; N(k) \qquad \text{(II.3.2–2a)}$$

$$\mathrm{Re}[p'(k;\hat{\tau}) \, e^{-j\theta}] = A_d(k;\epsilon,\underline{a}) \; + \; N_d(k) \qquad \text{(II.3.2–2b)}$$

where $A(k;\epsilon,\underline{a})$ and $A_d(k;\epsilon,\underline{a})$ are signal components, $N(k)$ and $N_d(k)$ are additive noise components, and $\underline{a}$ represents the entire data sequence, (II.3.2–1) can be transformed into

$$u_\tau(k;\theta,\hat{\tau}) = \mathrm{sgn}(A(k;\epsilon,\underline{a}) + N(k)) \;\; (A_d(k;\epsilon,\underline{a}) + N_d(k)) \qquad \text{(II.3.2–3)}$$

In the absence of decision errors, the timing error detector output (II.3.2–1) reduces to

$$u_\tau(k;\theta,\hat{\tau}) = a_k \, A_d(k;\epsilon,\underline{a}) \; + \; a_k \, N_d(k)) \qquad \text{(II.3.2–4)}$$

The corresponding timing error detector characteristic is given by

$$g_\tau(0,\epsilon) = E[u_\tau(k;\theta,\hat{\tau})] = E_{\underline{a}}[a_k \, A_d(k;\epsilon,\underline{a})] \qquad \text{(II.3.2–5)}$$

where $E_{\underline{a}}[.]$ denotes statistical expectation with respect to the data sequence $\underline{a}$. Note that (II.3.2–5) does not depend on E_s/N_o. The DC–value of the loop noise spectrum, $S(fT=0)$, equals the DC–value of the spectrum of the additive noise term $N_d(k)$. We conclude that for moderate and high E_s/N_o (i.e. when decision errors can be ignored) $S(fT=0)/(\text{slope})^2$ is inversely proportional to E_s/N_o.

Now we consider the effect of decision errors. Let us represent by $P_E(\epsilon,\underline{a})$ the bit error probability for a given timing error ϵ and a given data sequence $\underline{a}$:

$$P_E(\epsilon,\underline{a}) = \Pr[\hat{a}_k = -a_k \mid \epsilon,\underline{a}] \qquad \text{(II.3.2–6)}$$

Then the timing error detector characteristic is given by

$$g_\tau(0,\epsilon) = E_{\underline{a}}[(1 - 2P_E(\epsilon,\underline{a}))\, a_k\, A_d(k;\epsilon,\underline{a})] \qquad \text{(II.3.2–7)}$$

Comparing (II.3.2–5) and (II.3.2–7) for a nonzero timing error ϵ, the following conclusions can be drawn :

– The presence of decision errors reduces the timing error detector characteristic and, as $g_\tau(0,0) = 0$, reduces the timing error detector slope at the tracking point $\epsilon=0$.

– As $P_E(\epsilon,\underline{a})$ is a decreasing function of E_s/N_o, the timing error detector slope decreases with decreasing E_s/N_o.

– As $P_E(\epsilon,\underline{a})$ is a decreasing function of the rolloff α, the timing error detector slope decreases with decreasing α.

It follows that at low E_s/N_o, where decision errors can no longer be ignored, a tracking performance degradation occurs because decision errors reduce the slope of the timing error detector characteristic; the corresponding $S(fT=0)/(\text{slope})^2$ increases more than linearly with N_o/E_s. Also, note that a reduction of the timing error detector slope affects the loop bandwidth.

For (O)QPSK, the timing error detector output is just the average of the outputs of two timing error detectors for BPSK, applied to the in–phase and quadrature components of the (O)QPSK signal. For a given E_b/N_o, the (O)QPSK timing error detector characteristic and slope are the same as for BPSK. In terms of E_s/N_o, the (O)QPSK timing error detector characteristics and slope are smaller than for BPSK by 3 dB in E_s/N_o, because $E_s=E_b$ for BPSK and $E_s=2E_b$ for (O)QPSK.

Similar conclusions can be drawn for other types of DD symbol synchronizers.

<u>DDMLFB, DDMLFF</u>

The DDMLFB and DDMLFF symbol synchronizers have identical tracking performance, because they both maximize the DD log–likelihood function.

For moderate and high E_s/N_o, the effect of decision errors can be ignored. In this case, $S(fT=0)/(\text{slope})^2$ is inversely proportional with E_s/N_o, and equals the CR bound :

$$\frac{S(fT=0)}{(\text{slope})^2} = \frac{N_o}{2E_s}\,\frac{1}{(-g''(0)T^2)} \qquad \text{(II.3.2–8)}$$

This indicates that the DDMLFB and DDMLFF synchronizer yield the smallest

possible additive noise contribution to the timing error variance, provided that the receiver's decisions can be considered as correct.

Curves for $S(fT=0)/(\text{slope})^2$, which take decision errors into account, are shown in Figs. II.3–1 to II.3–3. For increasing E_s/N_o, they converge to (II.3.2–8). The performance degradation at low E_s/N_o (i.e. the deviation from (II.3.2–8)) is caused by a reduction of the timing error detector slope (see Fig. II.3–4).

<u>DDEL(λ=1/2), DTTL</u>

The DDEL(λ=1/2) and DTTL symbol synchronizers yield the same $S(fT=0)/(\text{slope})^2$, because they are closely related.

The DDEL(λ=1/2) synchronizer approximates the derivative matched filter output sample at the decision instant $kT+\hat{\tau}$ by the difference of matched filter output samples at $(k+1/2)T+\hat{\tau}$ and $(k-1/2)T+\hat{\tau}$.

For moderate and high E_s/N_o, the effect of decision errors can be ignored. In this case, $S(fT=0)/(\text{slope})^2$ is inversely proportional with E_s/N_o, and is given by

$$\frac{S(fT=0)}{(\text{slope})^2} = \frac{N_o}{2E_s}\,\frac{1}{2(g'(-T/2)T)^2} \qquad \text{(II.3.2–9)}$$

Curves for $S(fT=0)/(\text{slope})^2$, which take decision errors into account, are shown in Figs. II.3–5 to II.3–7. For increasing E_s/N_o, they converge to (II.3.2–9). The performance degradation at low E_s/N_o (i.e. the deviation from (II.3.2–9)) is caused by a reduction of the timing error detector slope (see Fig. II.3–8).

Comparing Figs. II.3–5 to II.3–7 with Figs. II.3–1 to II.3–3, it follows that the additive noise contribution to the timing error variance of the DDEL(λ=1/2) symbol synchronizer exceeds that of the DDML synchronizer only by a fraction of a dB (0.6 dB at most). Hence, for λ=1/2, the approximation of the derivative matched filter output sample gives rise to a very small tracking performance degradation as compared to the DDML symbol synchronizer. Comparing Figs. II.3–8 and II.3–4, it can be seen that the slope of the DDEL(λ= 1/2) synchronizer is not a close approximation of the DDML synchronizer slope. Fortunately, using the derivative matched filter approximation with λ = 1/2 not only reduces the timing error detector slope, as compared to the DDML synchronizer, but also reduces the loop noise spectrum by such an amount that the slope reduction is nearly compensated for.

DDEL(λ=1), M&M, mM&M

The DDEL(λ=1), M&M and mM&M symbol synchronizers yield the same $S(fT=0)/(\text{slope})^2$, because they are closely related.

The DDEL(λ=1) synchronizer approximates the derivative matched filter output sample at the decision instant $kT+\hat{\tau}$ by the difference of matched filter output samples at $(k+1)T+\hat{\tau}$ and $(k-1)T+\hat{\tau}$.

For moderate and high E_s/N_o, the effect of decision errors can be ignored. In this case, $S(fT=0)/(\text{slope})^2$ is inversely proportional with E_s/N_o, and is given by

$$\frac{S(fT=0)}{(\text{slope})^2} = \frac{N_o}{2E_s}\,\frac{1}{2(g'(-T)T)^2} \qquad \text{(II.3.2–10)}$$

Curves for $S(fT=0)/(\text{slope})^2$, which take decision errors into account, are shown in Fig. II.3–9. For increasing E_s/N_o, they converge to (II.3.2–10). The performance degradation at low E_s/N_o (i.e. the deviation from (II.3.2–10)) is caused by a reduction of the timing error detector slope (see Fig. II.3–10).

The tracking performance of the DDEL(λ=1) symbol synchronizer is considerably worse than that of the DDML synchronizer; the degradation with respect to (II.3.2–8) is caused by the derivative matched filter approximation with λ=1, and increases with increasing rolloff. As can be seen by comparing Figs. II.3–10 and II.3–4, this is mainly due to the fact that the slope of the DDEL(λ= 1) synchronizer is substantially smaller than the DDML slope; note that the DDEL(λ=1) slope decreases with increasing rolloff, whereas the opposite behavior occurs for the DDML and DDEL (λ=1/2) synchronizers. Although the DDEL(λ=1) loop noise spectrum is smaller than that of the DDML synchronizer, this spectrum reduction cannot compensate for the small DDEL(λ=1) slope, so that the DD ML synchronizer outperforms the DDEL(λ=1) synchronizer.

II.3.3 NDA symbol synchronization.

In this section we evaluate $S(fT=0)/(\text{slope})^2$ for various NDA symbol synchronizers. First we give some general properties, then we consider specific algorithms.

General properties

Most NDA symbol synchronizers make use of a quadratic algorithm. In this case, the timing error detector output consists of three terms :

– A useful signal × signal term, whose statistical expectation equals the timing error detector characteristic. When the excess bandwidth of the received signal does not exceed 100%, the timing error detector characteristic is sinusoidal in ϵ with period one. The statistical fluctuations of the signal × signal term constitute self–noise, which dominates the timing error variance at high E_s/N_o.

– A zero–mean signal × noise term, having a spectrum which is inversely proportional to E_s/N_o; this term dominates the timing error variance at moderate E_s/N_o.

– A zero–mean noise × noise term, having a spectrum which is inversely proportional to the square of E_s/N_o; this term dominates the timing error variance at low E_s/N_o.

Being the statistical expectation of the signal × signal term, the timing error characteristic does not depend on E_s/N_o. The timing error characteristic, and its slope at the tracking point $\epsilon=0$, decrease with decreasing excess bandwidth, and become identically zero for zero excess bandwidth. Hence, the quadratic symbol synchronization algorithms have a poor performance for small excess bandwidth [Fra'74; Fra'80; Gar'88 (chap. 10)].

NDAMLFB, NDAMLFF, NDAMLFB$\bar{C}$Q, NDAMLFF$\bar{C}$Q

The NDAMLFB$\bar{C}$Q and NDAMLFF$\bar{C}$Q symbol synchronizers have the same tracking error variance as the NDAMLFB and NDAMLFF symbol synchronizers for QPSK, respectively, because they are mathematically equivalent.

The NDAMLFB and NDAMLFF symbol synchronizers have identical tracking performance, because they both maximize the low E_s/N_o limit (I.1.2–4) of the NDA log–likelihood function.

For moderate and high E_s/N_o, the effect of noise × noise terms can be ignored. In this case, $S(fT=0)/(\text{slope})^2$ is inversely proportional with E_s/N_o.

When the matched filter output pulse g(t) is a cosine rolloff pulse with rolloff α, ignoring the noise × noise terms yields :

$$\frac{S(fT=0)}{(slope)^2} = \frac{N_o}{2E_s}\frac{2}{\alpha\pi^2} \qquad \text{(II.3.3–1)}$$

This illustrates that the NDAML algorithm is not useful for very small rolloffs. Curves for $S(fT=0)/(slope)^2$, which take noise × noise terms into account, are shown in Figs. II.3–11 and II.3–12. For increasing E_s/N_o, they converge to (II.3.3–1). The timing error detector slope at the tracking point is shown in Fig. II.3–13; note that this slope becomes zero for zero rolloff.

NDAEL(λ=1/2), GA, NDAEL(λ=1/2)$\bar{C}$Q, GA$\bar{C}$Q

The NDAEL(λ=1/2)$\bar{C}$Q and GA$\bar{C}$Q symbol synchronizers have the same tracking error variance as the NDAEL(λ=1/2) and GA symbol synchronizers for QPSK, respectively, because they are mathematically equivalent.

The NDAEL(λ=1/2) and GA symbol synchronizers yield the same $S(fT=0)/(slope)^2$, because they are closely related.

The NDAEL(λ=1/2) synchronizer approximates the derivative matched filter output sample at the decision instant $kT+\hat{\tau}$ by the difference of matched filter output samples at $(k+1/2)T+\hat{\tau}$ and $(k-1/2)T+\hat{\tau}$. Curves for the resulting $S(fT=0)/(slope)^2$ are shown in Figs. II.3–14 and II.3–15. Comparing these curves with those from Figs. II.3–11 and II.3–12, it follows that the additive noise contribution to the timing error variance of the NDAEL(λ=1/2) symbol synchronizer exceeds that of the NDAML synchronizer only by a small fraction of a dB (0.2 dB at most). Hence, for λ=1/2, the approximation of the derivative matched filter output sample gives rise to a very small tracking performance degradation as compared to the NDAML synchronizer. From Fig. II.3–13, it can be seen that the slope of the NDAEL(λ= 1/2) synchronizer is not a close approximation of the NDAML synchronizer slope. Fortunately, using the derivative matched filter approximation with λ = 1/2 not only reduces the timing error detector slope, as compared to the NDAML synchronizer, but also reduces the loop noise spectrum by such an amount that the slope reduction is nearly compensated for.

NDASQFF, NDASQFF$\bar{C}$Q

The NDASQFF$\bar{C}$Q symbol synchronizer has the same tracking error variance as the NDASQFF symbol synchronizer for QPSK, because they are mathematically equivalent.

We consider the NDASQFF synchronizer with a matched filter as prefilter. It has been shown in [Fra'80] that the NDASQFF and NDAML synchronizers yield the same additive noise contribution to the timing error variance, provided that the excess bandwidth does not exceed 100%; hence, both synchronizers yield the same $S(fT=0)/(\text{slope})^2$. Their self–noise contributions might be different.

<u>NDAMLFB$\bar{C}$B, NDAMLFF$\bar{C}$B, NDAEL(λ=1/2)$\bar{C}$B, GA$\bar{C}$B, NDASQFF$\bar{C}$B</u>

These synchronizers use the NDAMLFB, NDAMLFF, NDAEL(λ=1/2), GA and NDASQFF symbol synchronization algorithms for QPSK, which turn out to be carrier–independent, as ad hoc carrier–independent algorithms for BPSK.

Let us consider the NDAMLFB$\bar{C}$B algorithm. Its timing error detector output is given by

$$\begin{aligned} u_\tau(k;\hat{\tau}) &= \mathrm{Re}[p^*(k;\hat{\tau})p'(k;\hat{\tau})] \\ &= \mathrm{Re}[p(k;\hat{\tau})e^{-j\theta}]\ \mathrm{Re}[p'(k;\hat{\tau})e^{-j\theta}] \\ &\qquad + \ \mathrm{Im}[p(k;\hat{\tau})e^{-j\theta}]\ \mathrm{Im}[p'(k;\hat{\tau})e^{-j\theta}] \end{aligned} \qquad \text{(II.3.3–2)}$$

The timing error detector output of the NDAMLFB algorithm for BPSK is given by the first term in the second line of (II.3.3–2). The second term is a noise × noise term, because the quadrature component of the BPSK signal with respect to the true carrier is additive noise only. This second term is statistically independent of and has the same statistics as the noise × noise term contained in the first term. Hence, using the NDAMLFB$\bar{C}$B algorithm instead of the NDAMLFB algorithm for BPSK doubles the noise × noise contribution to the timing error variance, whereas the contribution from the signal × noise and signal × signal terms remains unchanged. Noting that, for a given E_s/N_o, the NDAMLFB algorithm for (O)QPSK yields a noise × noise contribution which is twice as large as for BPSK, it follows that the NDAMLFB$\bar{C}$B algorithm has the same $S(fT=0)/(\text{slope})^2$ as the NDAMLFB algorithm for (O)QPSK.

The same conclusion holds for the other carrier–independent symbol synchronizers for BPSK ; the NDAMLFF$\bar{C}$B, NDAEL(λ=1/2)$\bar{C}$B, GA$\bar{C}$B and NDASQFF$\bar{C}$B symbol synchronizers for BPSK have the same $S(fT=0)/(\text{slope})^2$ as the NDAMLFF, NDAEL(λ=1/2), GA and NDASQFF symbol synchronizers for (O)QPSK

NDASQFFC̄O

The NDASQFFC̄O and the carrier–dependent NDASQFF symbol synchronizer for OQPSK have the same timing error variance (additive noise contribution and self–noise contribution). Indeed, it can be verified by linearizing the functions arg(.) in the expressions of the timing estimates that both synchronizers yield the same (linearized) estimates, and, therefore, the same timing error variance. This verification is straightforward but lengthy, and is not given.

NDANLFBC̄B/Q

It has been shown in [Moe'90b] that the NDANLFBC̄B/Q symbol synchronizer is free of self–noise, and that for moderate to high E_s/N_o, where the effect of nonlinear noise terms (such as noise × noise terms) can be ignored, $S(fT=0)/(\text{slope})^2$ is essentially inversely proportional to E_s/N_o :

$$\left.\frac{S(fT=0)}{(\text{slope})^2}\right|_{BPSK} = \frac{N_o}{2E_s}\,\frac{1}{2(g'(-T)T)^2} \qquad \text{(II.3.3–3)}$$

$$\left.\frac{S(fT=0)}{(\text{slope})^2}\right|_{QPSK} = \frac{N_o}{2E_s}\,\frac{1}{(g'(-T)T)^2} \qquad \text{(II.3.3–4)}$$

This is a striking result : at moderate and high E_s/N_o, the additive noise contribution to the timing error variance does not depend on the function F(.), used in the expression of the timing error detector output; the only restrictions are that the derivative F'(x) is nonzero at x=1, and $x^{1/2}F(x)$ is bounded when x approaches zero [Moe'90b]. This resembles the NDAVVFFT carrier synchronizer, whose additive noise contribution to the tracking error variance at moderate and high E_s/N_o does not depend on the function F(.), used in the expression of the carrier phase estimate. Note that (II.3.3–3) is the same as for the DDEL(λ=1) synchronizer (see (II.3.2–10)), whereas (II.3.3–4) is twice as large.

II.3.4 Self–noise performance.

As explained in section II.1.2, the self–noise contribution to the tracking error variance is given by

$$\text{self–noise contribution} = K_F\,(2B_LT)^2\,(\sum_m |m|(-R(m)))/(\text{slope})^2) \qquad \text{(II.3.4–1)}$$

where $\{R(m)\}$ is the autocorrelation sequence of the self–noise component of the loop noise, and K_F is a constant of proportionality, depending on the loop filter (feedback symbol synchronization); the expression is also valid for feedforward symbol synchronizers.

Table II.3–1 shows the quantity $\Sigma|m|(-R(m))/(\text{slope})^2$, corresponding to various DD and NDA symbol synchronizers, for rolloffs of 10%, 50% and 100%. Also shown is the quantity $(E_s/N_o)_c$, which is defined as the value of E_s/N_o at which additive noise and self–noise equally contribute to the tracking error variance; roughly speaking, self–noise is important only when E_s/N_o exceeds $(E_s/N_o)_c$. As the self–noise and additive noise contributions are proportional with $K_F(2B_LT)^2$ and $2B_LT$, respectively, $(E_s/N_o)_c$ is a decreasing function of the quantity $K_F(2B_LT)$. In most cases of practical interest, $2B_LT$ is so small that $(E_s/N_o)_c$ is in the region of E_s/N_o where the effect of decision errors and nonlinear noise terms (such as noise × noise terms) can be ignored; in that region of E_s/N_o, the additive noise contribution to the tracking error variance is essentially inversely proportional to E_s/N_o :

$$\text{additive noise contribution} = (2B_LT)\,\frac{S(fT=0)}{(\text{slope})^2}$$

$$= (2B_LT)\,A\,(N_o/2E_s) \qquad \text{(II.3.4–2)}$$

where A depends on the type of synchronizer and on the shape of the baseband pulse. Under these assumptions, $(E_s/N_o)_c$ is inversely proportional with $K_F(2B_LT)$:

$$(E_s/N_o)_c = \frac{A\ (\text{slope})^2}{2\ \sum_m |m|(-R(m))}\ \frac{1}{K_F\ (2B_LT)} \qquad \text{(II.3.4–3)}$$

Table II.3–1 gives the above $(E_s/N_o)_c$ for $K_F = 2$ and $2B_LT = 10^{-2}$. Values of $(E_s/N_o)_c$ corresponding to other values of K_F and/or $2B_LT$ are easily derived; for instance, when $K_F = 1$ and $2B_LT = 10^{-3}$, $(E_s/N_o)_c$ is obtained by adding 13 dB to the tabulated $(E_s/N_o)_c$ values.

The results from Table II.3–1 indicate that the self–noise contribution for BPSK is twice that of (O)QPSK. This is because the timing error detector output for (O)QPSK is the average of two terms (one from the in–phase channel and one from the quadrature channel), whereas the timing error detector output for BPSK consists only of the in–phase channel term. For (O)QPSK, these two terms are statistically independent and identically distributed. Hence, the timing error detector characteristic (and, consequently, also the timing error detector slope) for (O)QPSK is the same as for BPSK, whereas the loop noise power spectrum for (O)QPSK is only half that of BPSK (it is assumed that the (O)QPSK system and the BPSK system operate at the same E_b/N_o). Therefore, the timing error variance for BPSK is twice as large as for (O)QPSK.

For all considered symbol synchronizers, $(E_s/N_o)_c$ is a non–decreasing function of the rolloff α. Indeed, when the rolloff increases, the baseband pulse at the matched filter output decays more rapidly, so that the influence of self–noise is reduced.

The DD symbol synchronizers are less vulnerable to self–noise than their NDA counterparts; the difference in self–noise performance becomes more pronounced when the rolloff decreases. For small rolloffs, the slope of the quadratic NDA synchronizers is proportional with α (see Fig. II.3–13), whereas A from (II.3.4–2) is inversely proportional with α (see (II.3.3–1)). Therefore, assuming small α, $(E_s/N_o)_c$ from (II.3.4–3) is proportional with α; this explains why quadratic NDA symbol synchronizers exhibit a lot of self–noise when the rolloff is small.

Closely related synchronizers have the same DC–value S(fT=0) of the loop noise spectrum, but the shape of this spectrum (and, hence, the self–noise performance) might be different; the loop noise spectra S(fT) of the closely related DDEL(λ=1), M&M and mM&M symbol synchronizers are shown in Fig. II.3–16. Now let us compare the self–noise performance of the various closely related synchronizers.

– DD EL (λ=1/2)/DTTL : The DTTL symbol synchronizer is less sensitive to self–noise then the DD EL (λ=1/2) synchronizer.

– DD EL (λ=1)/M&M/modified M&M : The DD EL (λ=1) and modified M&M synchronizers yield the same self–noise performance, because their loop noise spectra are identical (see Fig. II.3–16); the M&M synchronizer is free of self–noise.

– NDA EL (λ=1/2)/GA : The GA symbol synchronizer is less sensitive to self–noise than the NDA EL (λ=1/2) symbol synchronizer.

For realistic values of $2B_LT$, the tracking error variance in a satellite communication environment (say, $E_s/N_o < 10$ dB) is dominated by the additive noise contribution (a possible exception are the quadratic NDA symbol synchronizers with small rolloff). Therefore, we have not considered the symbol synchronizers from section I.2.3.3 (i.e. DDMLFB$\bar{S}\bar{N}\bar{F}\bar{B}$, DDMLFB$\bar{S}\bar{N}\bar{F}\bar{F}$, NDAMLFB$\bar{S}\bar{N}\bar{F}\bar{F}$, DDMMSEFB), which eliminate self–noise at the expense of an increased additive noise contribution and a higher implementation complexity.

TABLE II.3–1 : Self–noise performance of symbol synchronizers

	$\Sigma\|m\|(-R(m))/(slope)^2$		$(E_s/N_o)_c$ @ $2B_LT=10^{-2}$, $K_F=2$	
	BPSK	(O)QPSK	BPSK	(O)QPSK
DDML				
$\alpha = 0.1$	0.410	0.205	12.65 dB	15.65 dB
$\alpha = 0.5$	0.0954	0.0477	18.43 dB	21.43 dB
$\alpha = 1.0$	0.00854	0.00427	27.53 dB	30.53 dB
DDEL(λ=1/2)				
$\alpha = 0.1$	0.388	0.194	12.97 dB	15.97 dB
$\alpha = 0.5$	0.154	0.0772	16.52 dB	19.52 dB
$\alpha = 1.0$	0.0556	0.0278	20.00 dB	23.00 dB
DTTL				
$\alpha = 0.1$	0.155	0.0776	16.94 dB	19.94 dB
$\alpha = 0.5$	0.0135	0.00677	27.09 dB	30.09 dB
$\alpha = 1.0$		Free of self–noise		
DDEL(λ=1)				
$\alpha = 0.1$	0.509	0.255	14.00 dB	17.00 dB
$\alpha = 0.5$	0.811	0.405	14.00 dB	17.00 dB
$\alpha = 1.0$	4.500	2.250	14.00 dB	17.00 dB
M&M		Free of self–noise		
mM&M		Same as DDEL(λ=1)		
NDAML, NDAML$\bar{C}$B/Q				
$\alpha = 0.1$	18.44	9.22	4.39 dB	7.39 dB
$\alpha = 0.5$	0.221	0.111	16.61 dB	19.61 dB
$\alpha = 1.0$	0.00934	0.00467	27.34 dB	30.34 dB
NDAEL(λ=1/2), NDAEL(λ=1/2)$\bar{C}$B/Q				
$\alpha = 0.1$	25.67	12.84	2.95 dB	5.95 dB
$\alpha = 0.5$	0.490	0.245	13.17 dB	16.17 dB
$\alpha = 1.0$	0.0703	0.0352	18.75 dB	21.75 dB
GA, GA$\bar{C}$B/Q				
$\alpha = 0.1$	10.28	5.14	6.93 dB	9.93 dB
$\alpha = 0.5$	0.0429	0.0215	23.74 dB	26.74 dB
$\alpha = 1.0$		Free of self–noise		
NDANLFB$\bar{C}$B/Q		Free of self–noise		

II.4 TRACKING PERFORMANCE RESULTS FOR JOINT CARRIER AND SYMBOL SYNCHRONIZATION

In section II.1.2 we have pointed out that the linearized dynamics of the carrier and symbol synchronizers are not coupled. This means that the carrier and symbol synchronizers do not interact during the tracking mode. As a consequence, the (linearized) carrier phase error variance is evaluated under the assumption of perfect symbol timing, and the (linearized) timing error variance is evaluated under the assumption of perfect carrier synchronization. Combining any carrier synchronization algorithm from section I.2.1 with any symbol synchronization algorithm from section I.2.2 gives rise to the same linearized tracking error variances as the carrier synchronizer (assuming perfect symbol timing) and the symbol synchronizer (assuming perfect carrier synchronization) considered individually; these individual tracking error variances have been dealt with in sections II.2 and II.3.

The JDDMLFF carrier and symbol synchronization algorithm from section I.2.3 gives rise to estimates which jointly maximize the DD log–likelihood function. The same estimates, and, hence, the same tracking performance, result from combining the DDMLFF carrier synchronizer with the DDMLFF symbol synchronizer, because the resulting estimates also jointly maximize the DD log–likelihood function. Hence, the JDDMLFF carrier and symbol synchronizer yields the same tracking error variances as the individual DDMLFF carrier and DDMLFF symbol synchronization algorithms. Similarly, the JNDAMLFF carrier and symbol synchronizer gives rise to the same tracking error variances as the individual NDAMLFF carrier and NDAMLFF symbol synchronization algorithms.

II.5 CONCLUSIONS

We have pointed out that, in a satellite communication environment, the tracking error variance of most carrier and symbol synchronizers is dominated by the additive noise contribution. Possible exceptions are the quadratic NDA synchronizers operating on a signal with a small excess bandwidth; for these synchronizers, the self–noise contribution increases faster than the additive noise contribution for decreasing excess bandwidth.

The additive noise contribution to the tracking error variance is proportional to the synchronizer bandwidth, and, for moderate to high E_s/N_o, inversely proportional to E_s/N_o. For low E_s/N_o, a tracking performance degradation occurs, because of decision errors (DD synchronizers) or the presence of higher–order (such as quadratic) noise terms (NDA synchronizers).

In a first approximation, the tracking performance of the various synchronizers can be compared by neglecting the degradation at low E_s/N_o and the self–noise at high E_s/N_o; this approximation yields

$$\mathrm{Var} \cong (2B_L T)\, A \frac{N_o}{2E_s} \qquad \text{(II.5–1)}$$

where the factor A depends only on the synchronizer type and on the received baseband pulse. For specific values of α, E_s/N_o and $2B_L T$, one should take into account also the tracking performance degradation at low E_s/N_o and the self–noise contribution in order to make a more accurate comparison. The following tables give the values of A for various carrier and symbol synchronizers, under the assumption that the received baseband pulse is a square–root cosine rolloff pulse with rolloff factor α.

TRACKING ERROR VARIANCE AT MODERATE E_s/N_o

$$\mathrm{Var} \cong (2B_L T)\, A \frac{N_o}{2E_s}$$

CARRIER SYNCHRONIZATION

		Degradation [dB] from CRB
DDML, NDAMLB/Q, V&VB/Q (same as CRB)		
$\alpha = 0.$	$A = 1.$	0.
$\alpha = 0.5$	$A = 1.$	0.
$\alpha = 1.$	$A = 1.$	0.
NDAMLO, NDASQFFTO		
$\alpha = 0.$	$A = \infty$	∞
$\alpha = 0.5$	$A = 4.$	6.
$\alpha = 1.$	$A = 2.$	3.
NDAMLTB (continuous–time operation)		
$\alpha = 0.$	$A = 1.$	0.
$\alpha = 0.5$	$A = 1.061$	0.26
$\alpha = 1.$	$A = 1.111$	0.45

TRACKING ERROR VARIANCE AT MODERATE E_s/N_o

$$\mathrm{Var} \cong (2B_L T)\, A \frac{N_o}{2E_s}$$

SYMBOL SYNCHRONIZATION

		Degradation [dB] from CRB
DDML (same as CRB)		
$\alpha = 0.$	$A = 0.304$	0.
$\alpha = 0.5$	$A = 0.266$	0.
$\alpha = 1.$	$A = 0.194$	0.
DDEL(λ=1/2), DTTL		
$\alpha = 0.$	$A = 0.308$	0.05
$\alpha = 0.5$	$A = 0.277$	0.17
$\alpha = 1.$	$A = 0.222$	0.58
NDAML, NDASQFF, NDAML$\bar{C}$B/Q, NDASQFF$\bar{C}$B/Q, NDASQFF$\bar{C}$O		
$\alpha = 0.$	$A = \infty$	∞
$\alpha = 0.5$	$A = 0.405$	1.84
$\alpha = 1.$	$A = 0.203$	0.20
NDAEL(λ=1/2), GA, NDAEL(λ=1/2)$\bar{C}$B/Q, GA$\bar{C}$B/Q		
$\alpha = 0.$	$A = \infty$	∞
$\alpha = 0.5$	$A = 0.406$	1.84
$\alpha = 1.$	$A = 0.211$	0.36
DDEL(λ=1), M&M, mM&M		
$\alpha = 0.$	$A = 0.5$	2.16
$\alpha = 0.5$	$A = 0.811$	4.84
$\alpha = 1.$	$A = 4.5$	13.66
NDANLFB$\bar{C}$B		
$\alpha = 0.$	$A = 0.5$	2.16
$\alpha = 0.5$	$A = 0.811$	4.84
$\alpha = 1.$	$A = 4.5$	13.66
NDANLFB$\bar{C}$Q		
$\alpha = 0.$	$A = 1.$	5.16
$\alpha = 0.5$	$A = 1.621$	7.84
$\alpha = 1.$	$A = 9.$	16.66

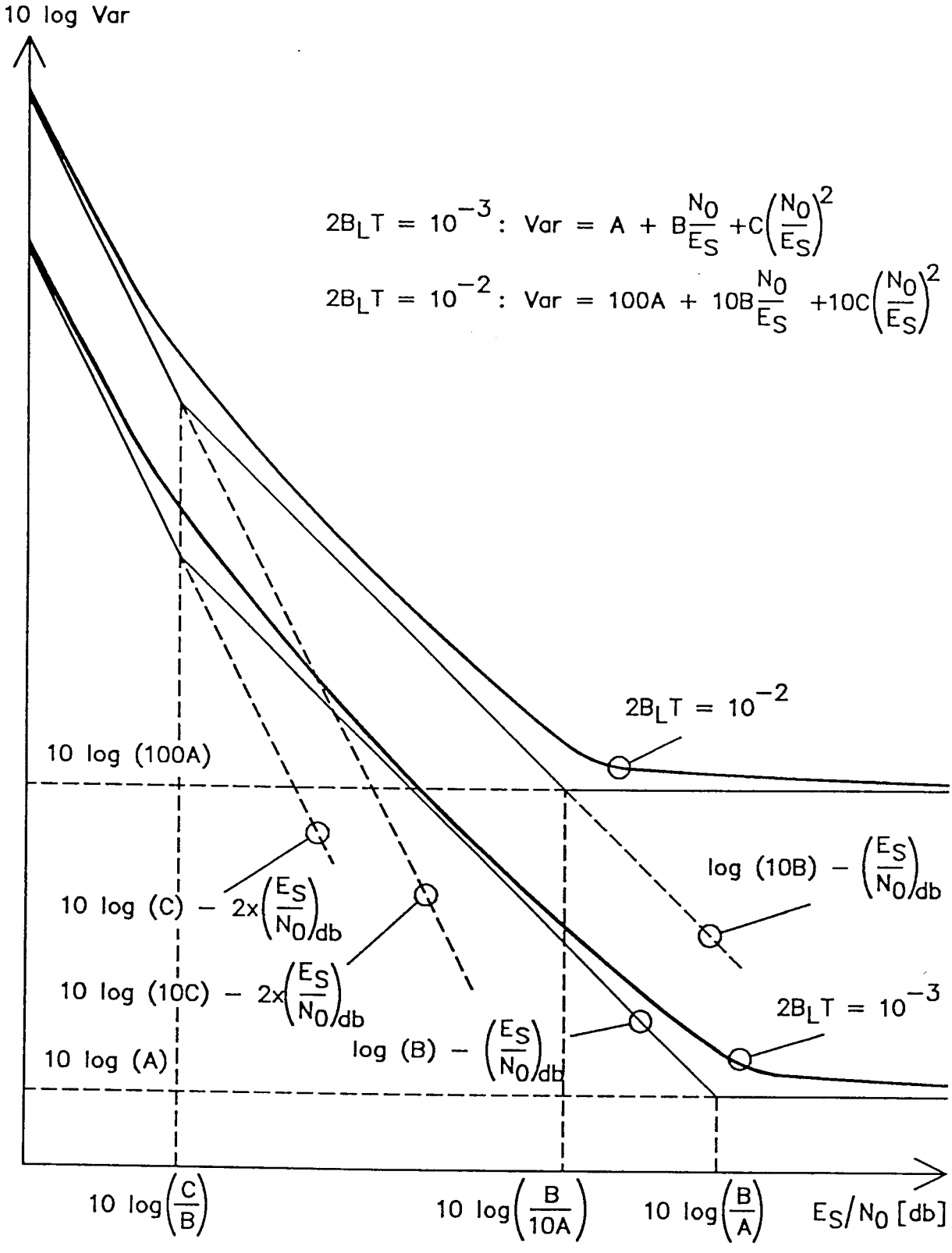


Fig. II. 1–1 : Typical behaviour of tracking error variance as a function of E_S/N_0

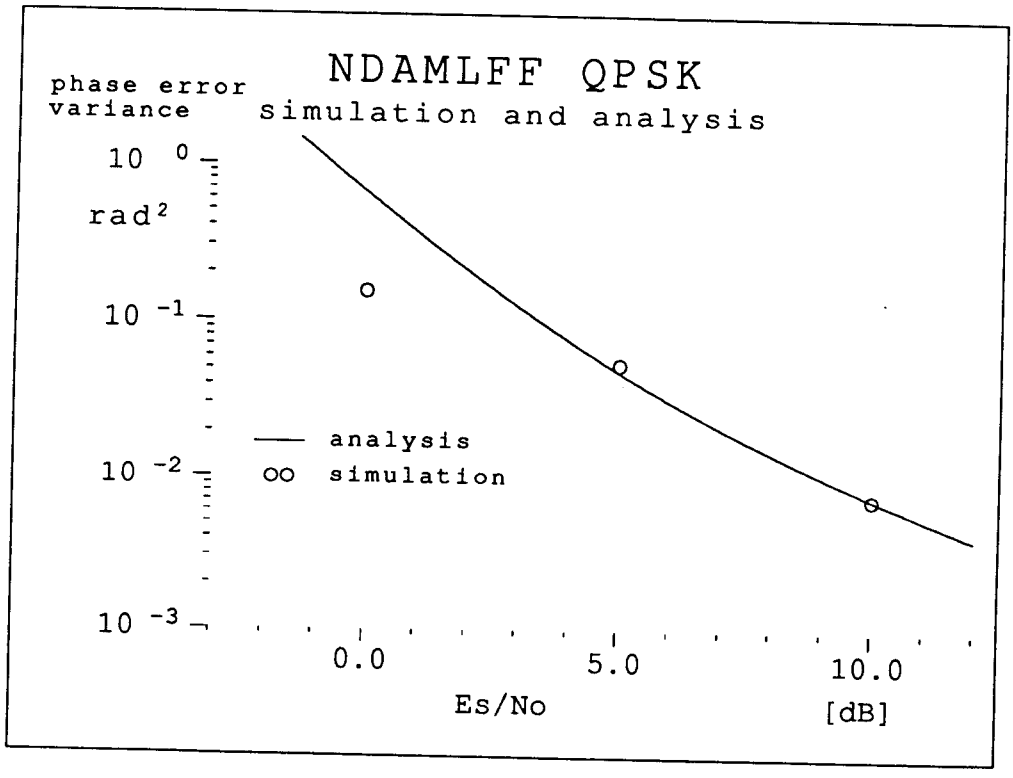


Fig. II.1-2a: NDAML **feedforward** carrier synchronizer, QPSK, $B_LT = 0.05$, simulation and analysis

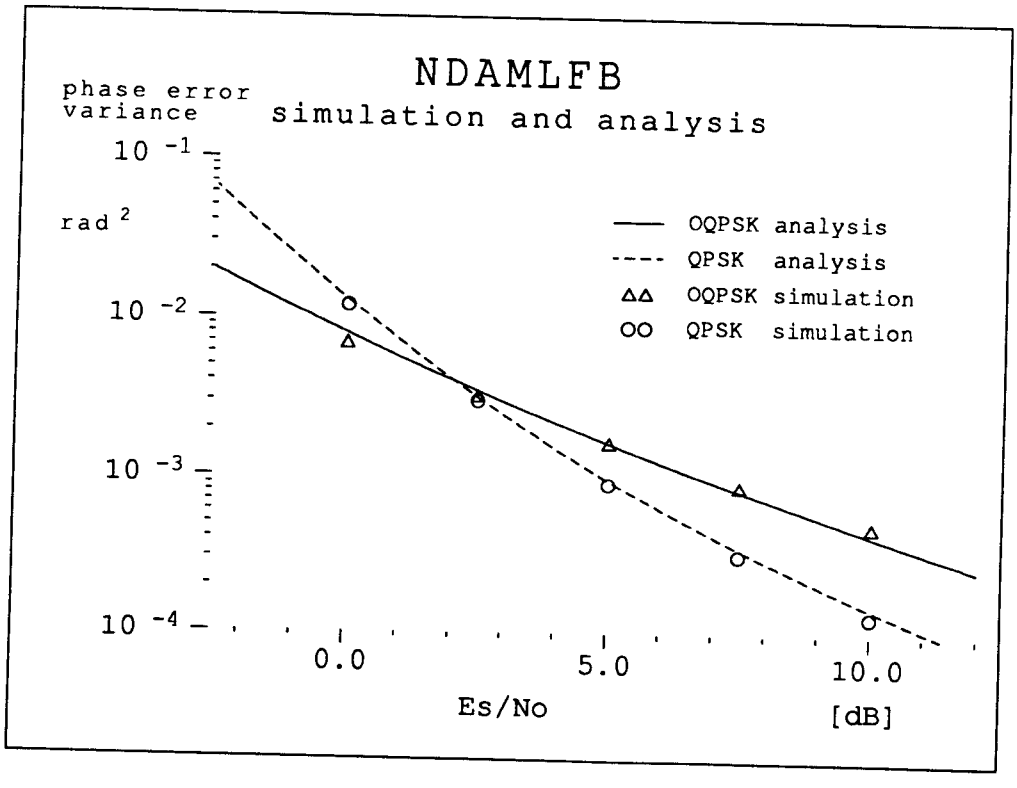


Fig. II.1-2b: NDAML **feedback** carrier synchronizer, (O)QPSK, $B_LT = 0.001$, rolloff α=0.5, simulation and analysis

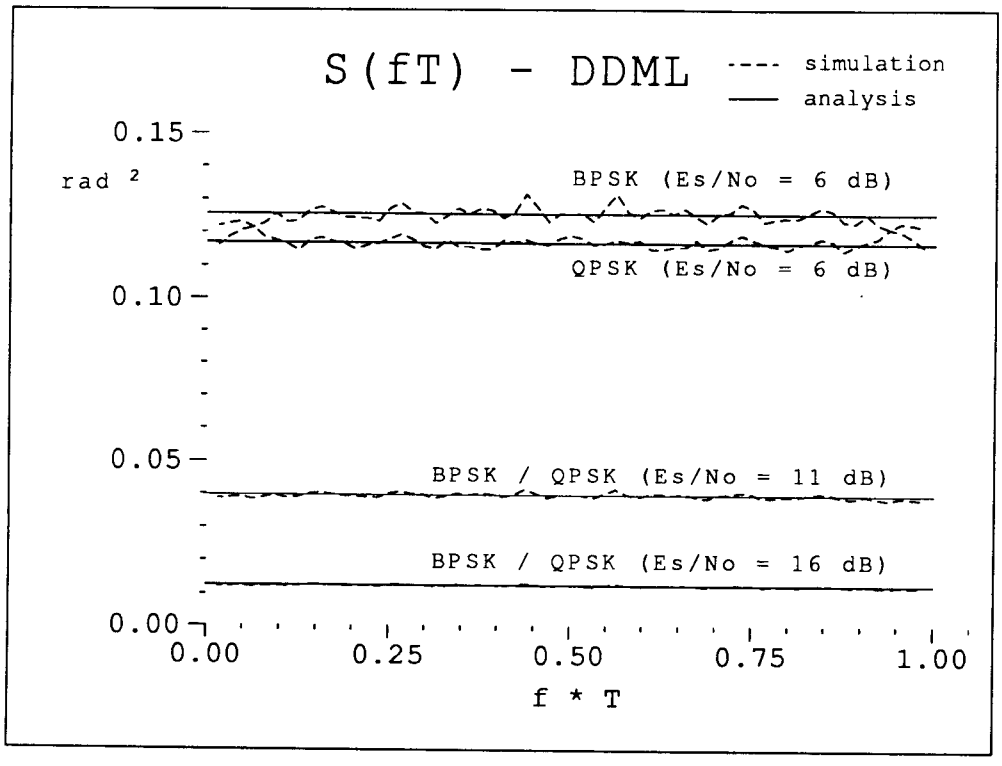


Fig. II.2-1a: Noise power spectral density of DDML for BPSK and QPSK

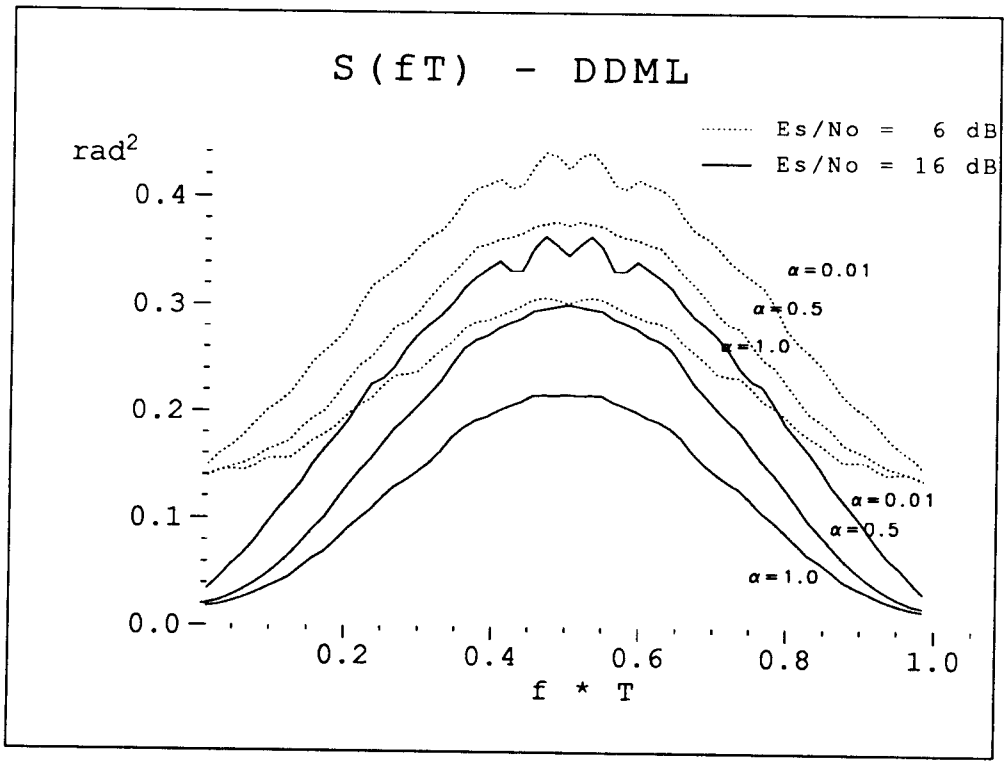


Fig. II.2-1b: Noise power spectral density of DDML for OQPSK, rolloff $\alpha = 0.01, 0.5, 1.0$

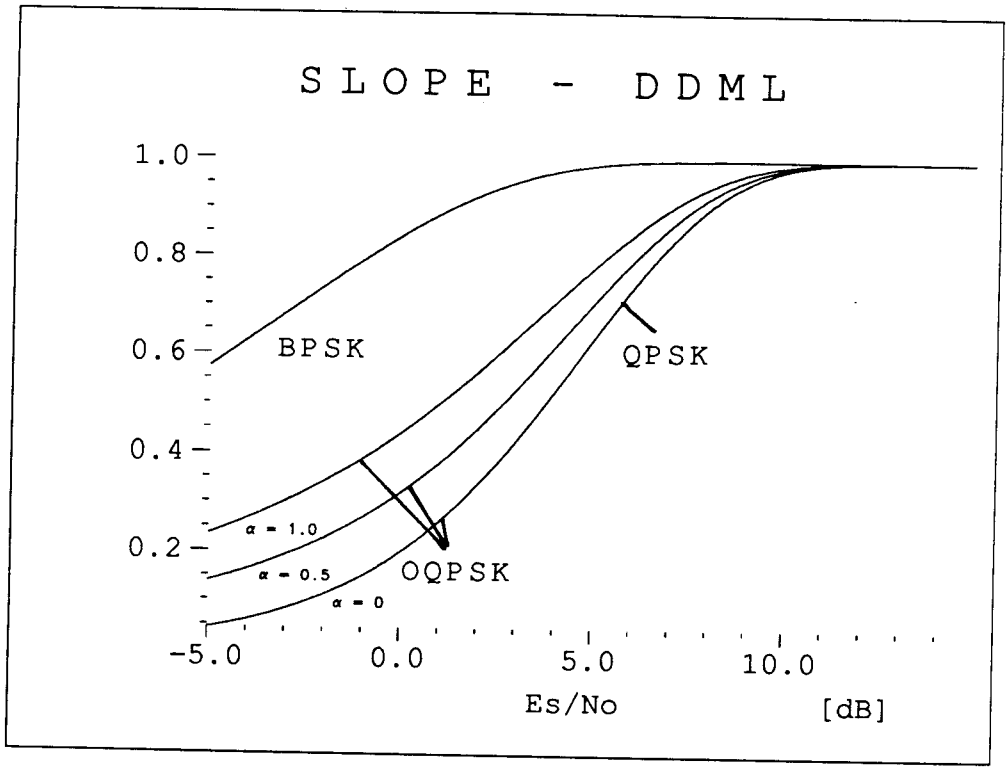


Fig. II.2-2: DDMLFB phase detector slopes $K_{\theta\theta}$

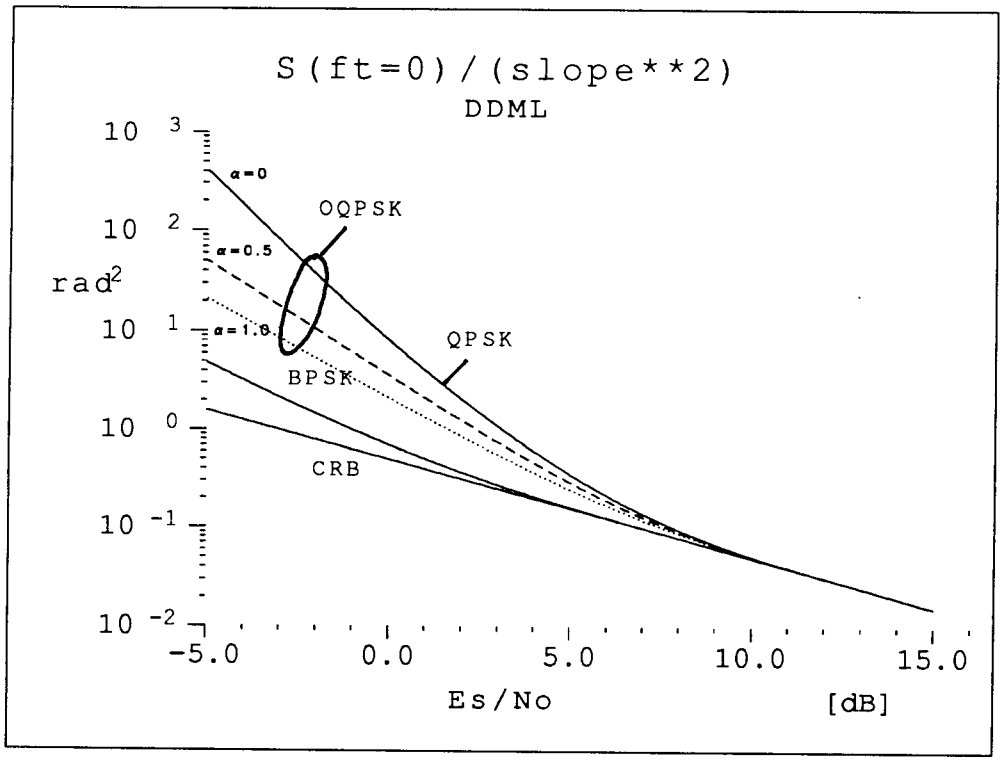


Fig. II.2-3a: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ for DDML, logarithmic scale

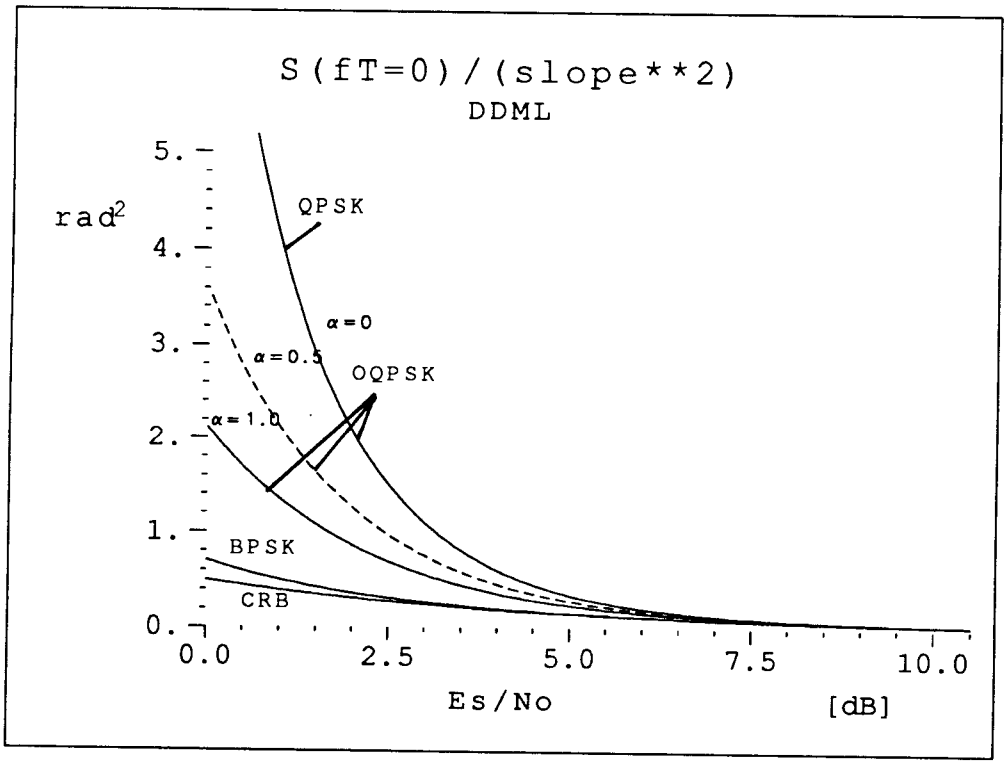


Fig. II.2-3b: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ for DDML, linear scale

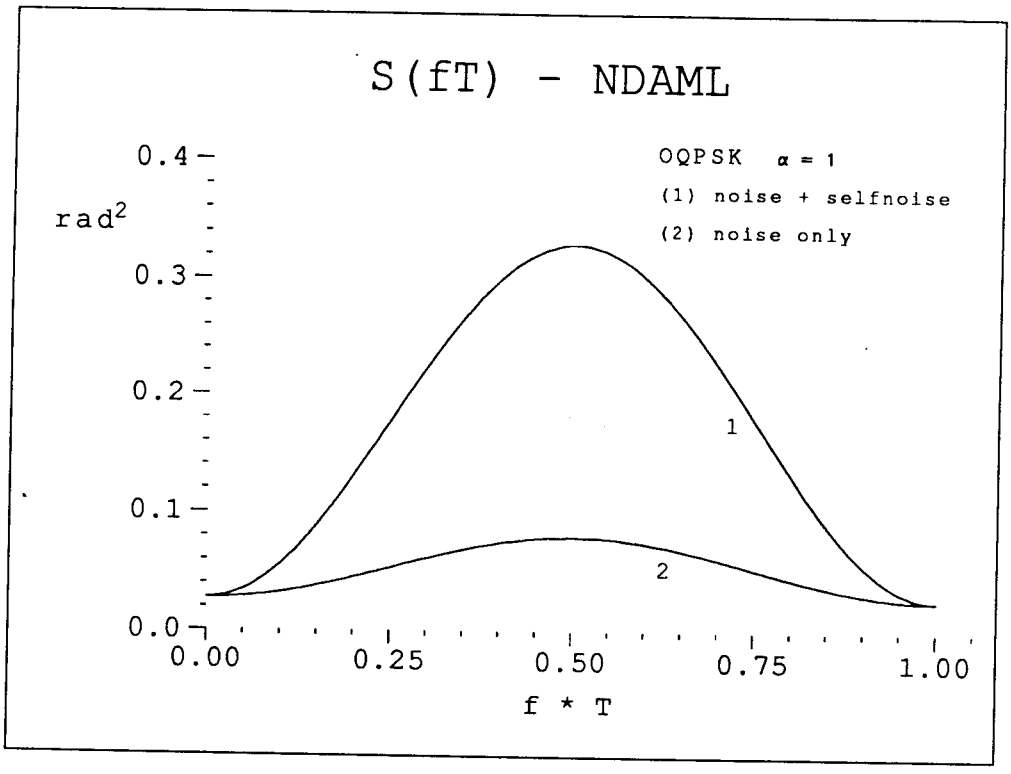


Fig. II.2-4: Noise power spectral density of NDAML for OQPSK, rolloff $\alpha = 1.0$

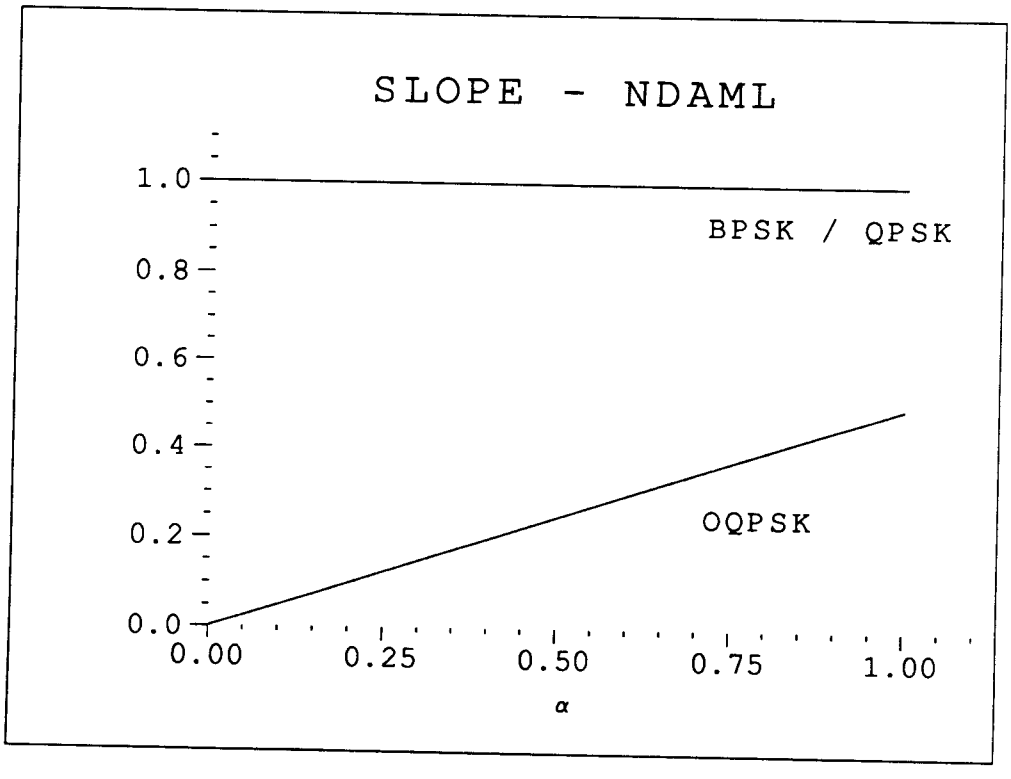


Fig. II.2-5: NDAMLFB phase detector slopes $K_{\theta\theta}$ as a function of rolloff α

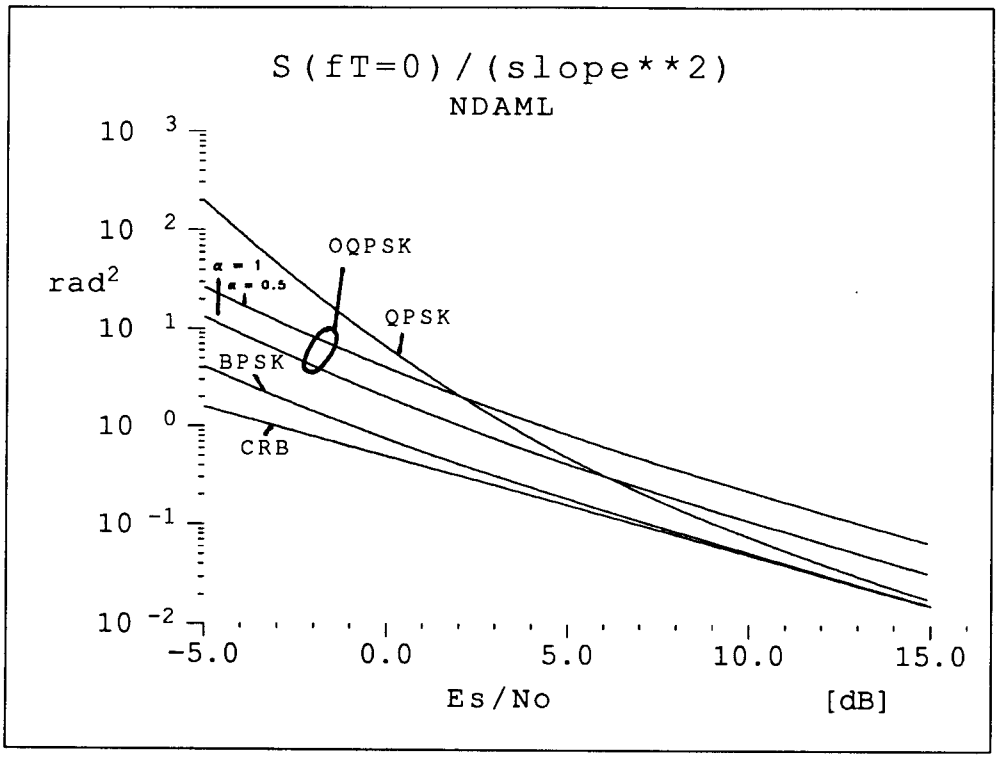


Fig. II.2-6a: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ for NDAML, logarithmic scale

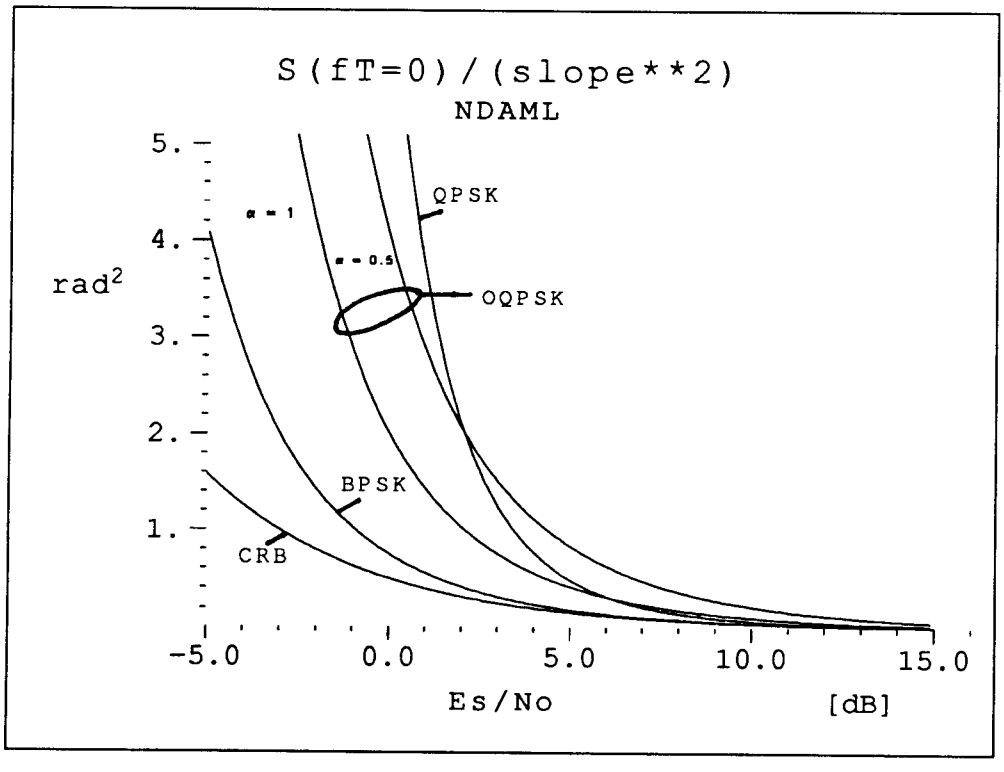


Fig. II.2-6b: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ for NDAML, linear scale

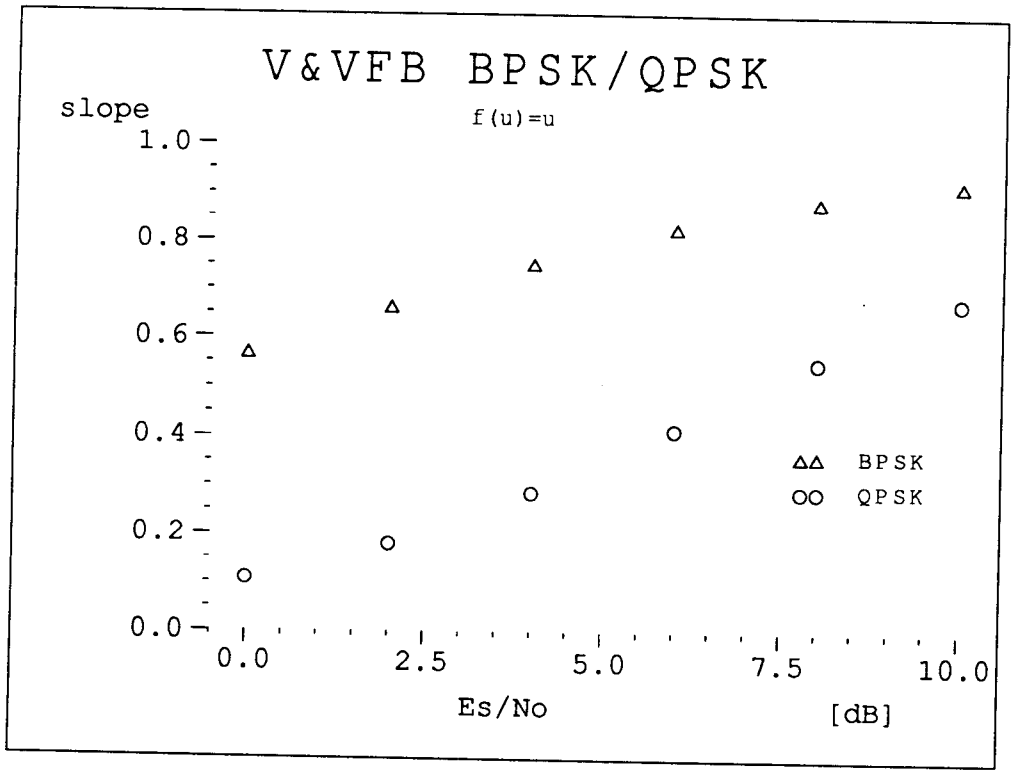


Fig. II.2-7a: V&VFB phase detector slopes $K_{\theta\theta}$ for BPSK and QPSK, nonlinearity f(u)=u

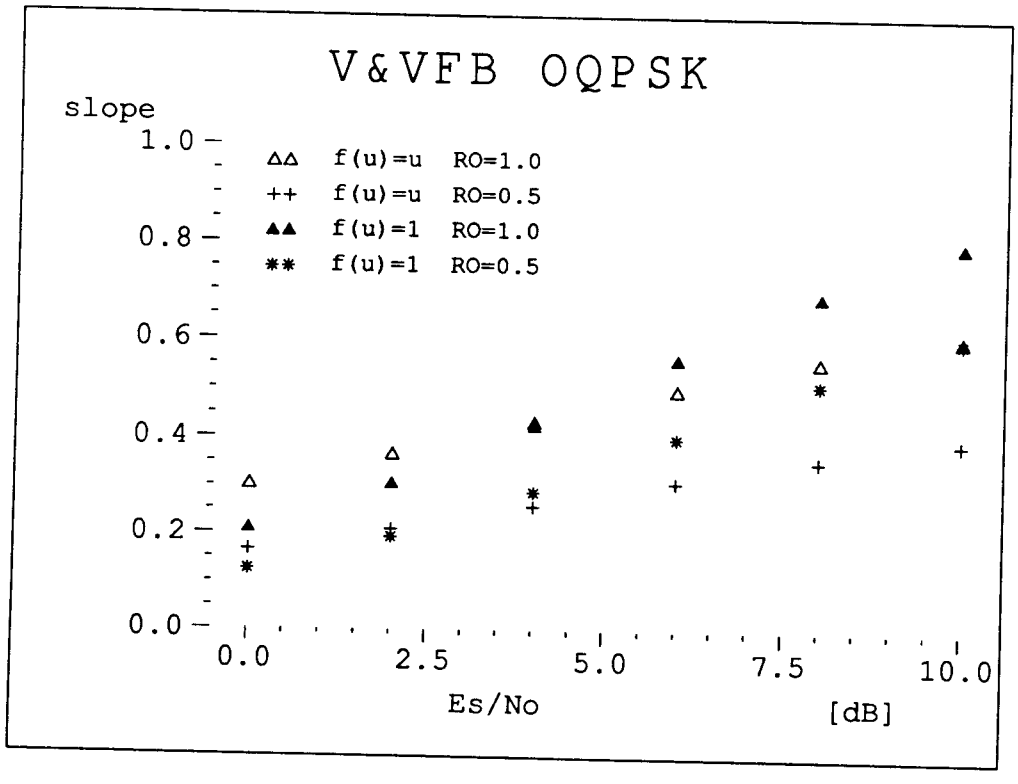


Fig. II.2-7b: V&VFB phase detector slopes $K_{\theta\theta}$ for OQPSK, nonlinearity f(u)=u and f(u)=1

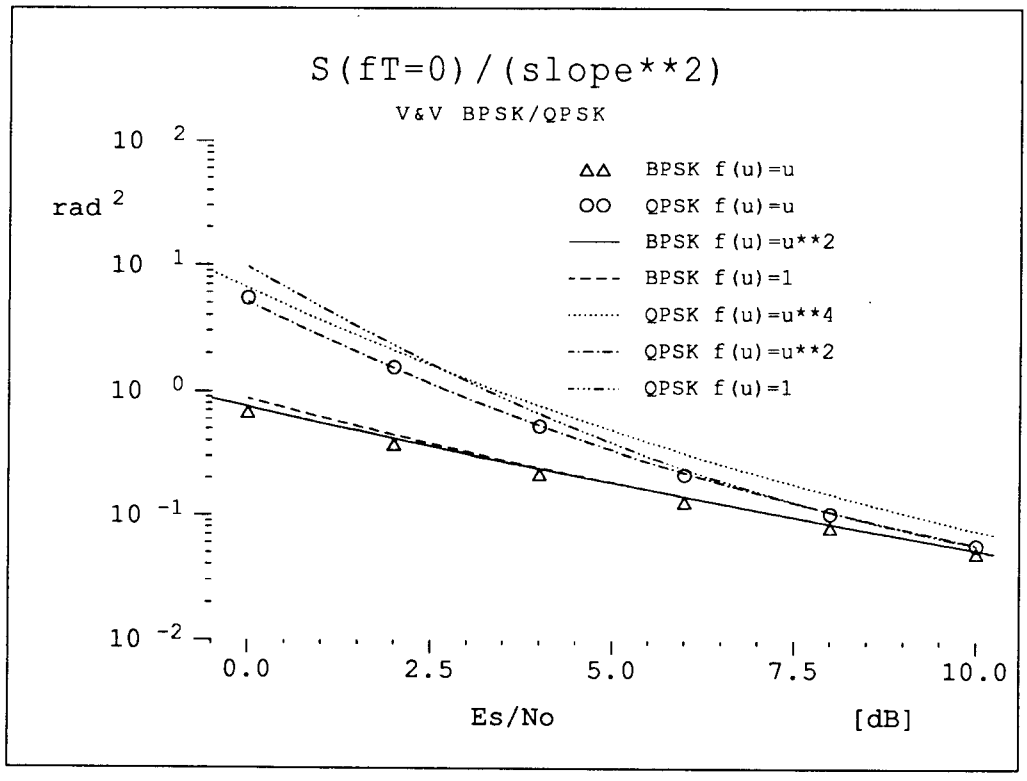


Fig. II.2-8a: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ of V&V for BPSK and QPSK, logarithmic scale

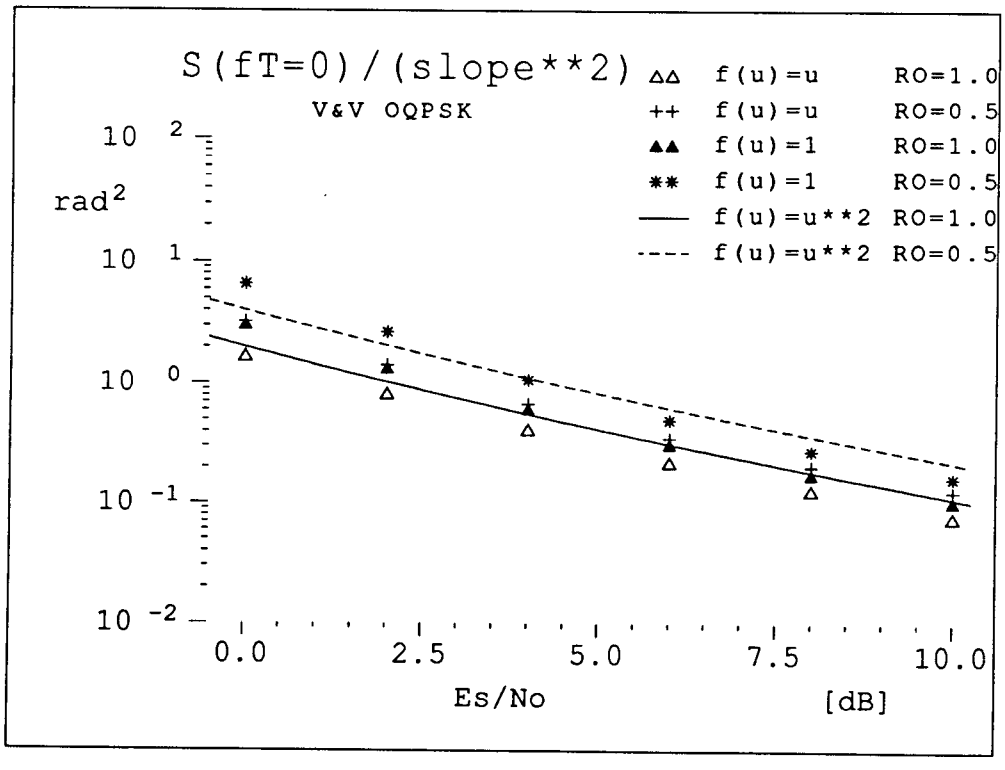


Fig. II.2-8b: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ of V&V for OQPSK, logarithmic scale

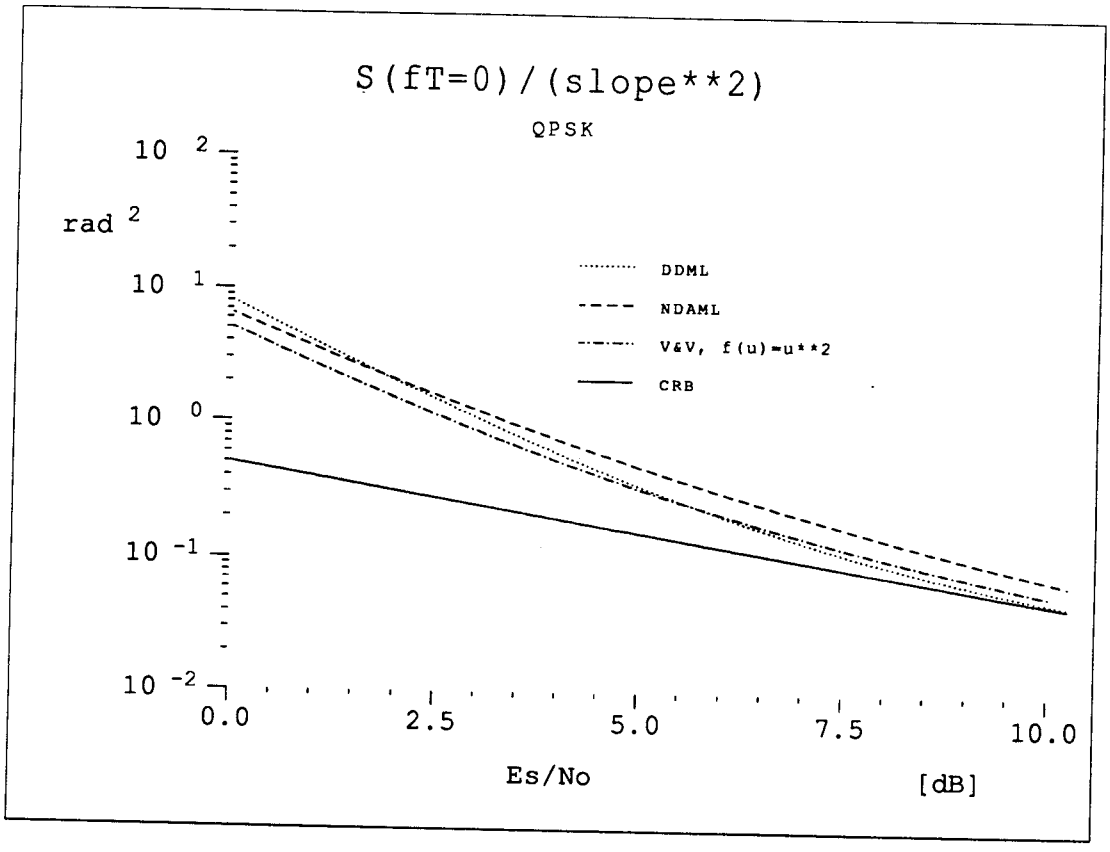


Fig. II.2-8c: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ of DDML, NDAML, and V&V with $f(u) = u^2$ for QPSK, logarithmic scale

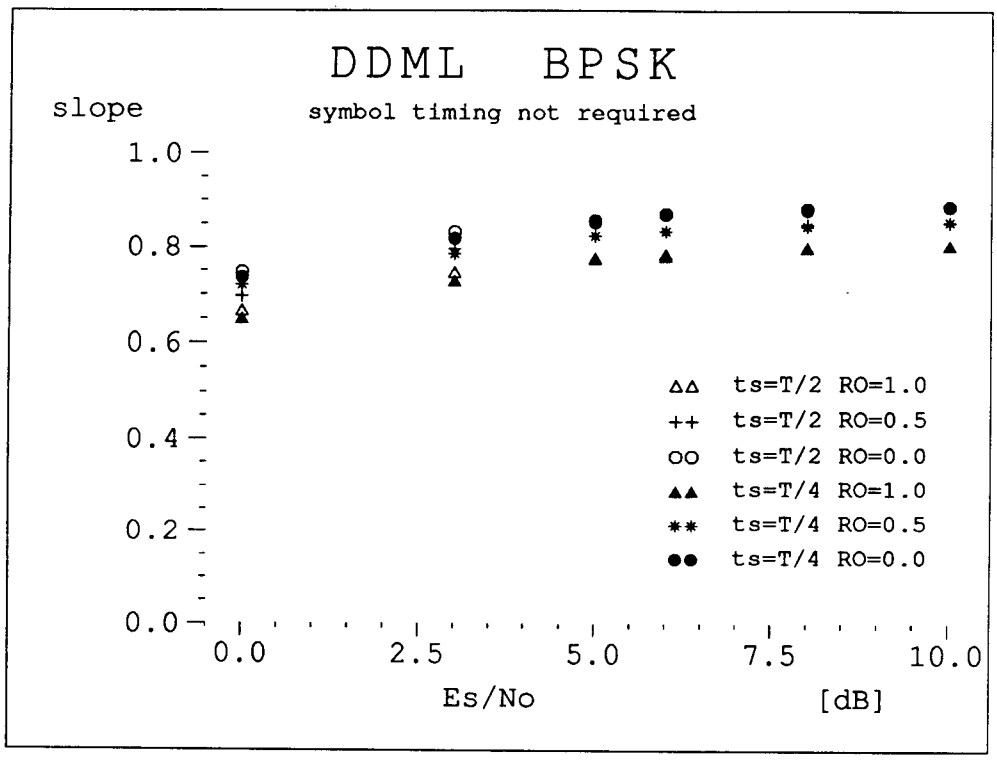


Fig. II.2-9a: DDML$\overline{T}$ phase detector slopes $K_{\theta\theta}$ for BPSK

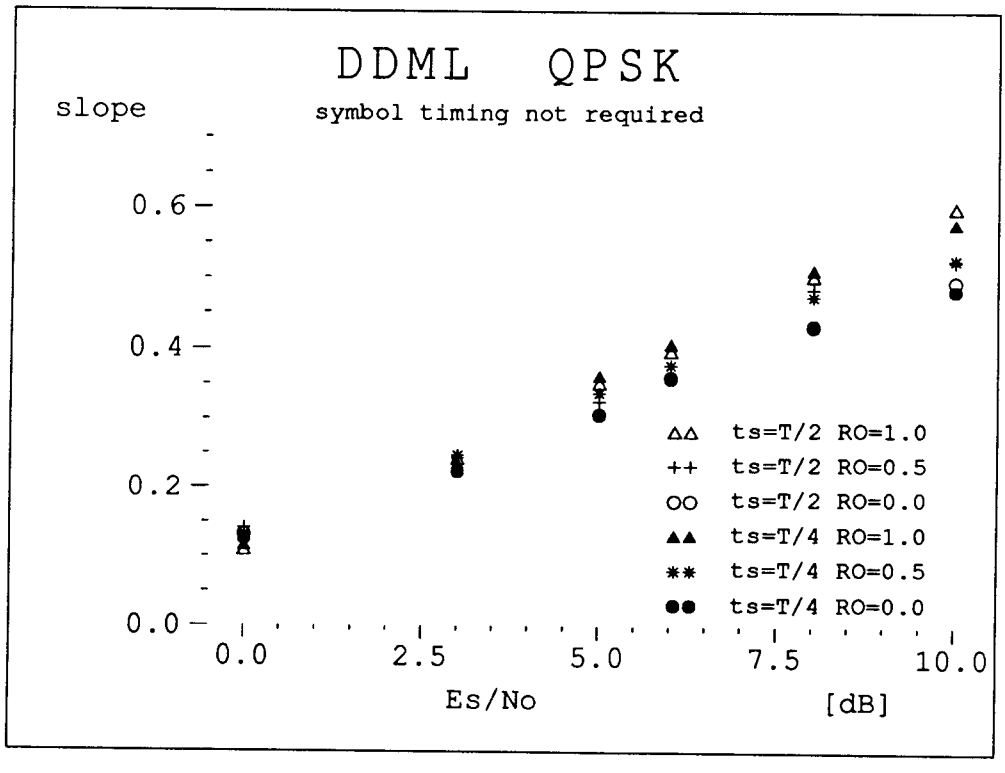


Fig. II.2-9b: DDML$\overline{T}$ phase detector slopes $K_{\theta\theta}$ for QPSK

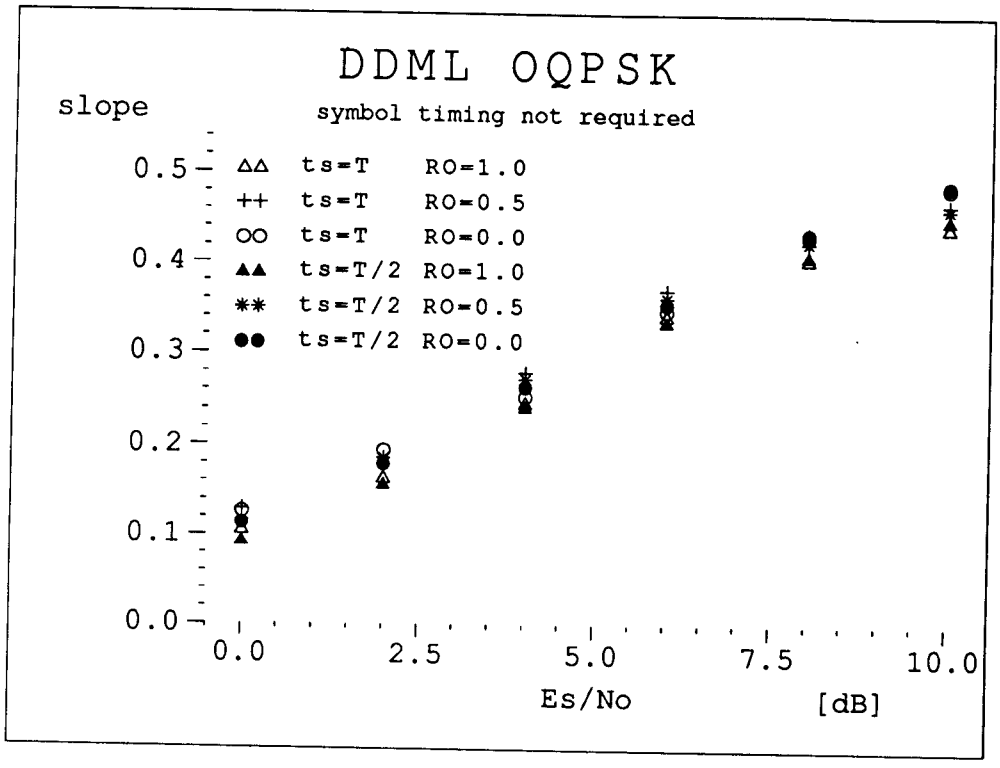


Fig. II.2-9c: DDML$\overline{T}$ phase detector slopes $K_{\theta\theta}$ for OQPSK

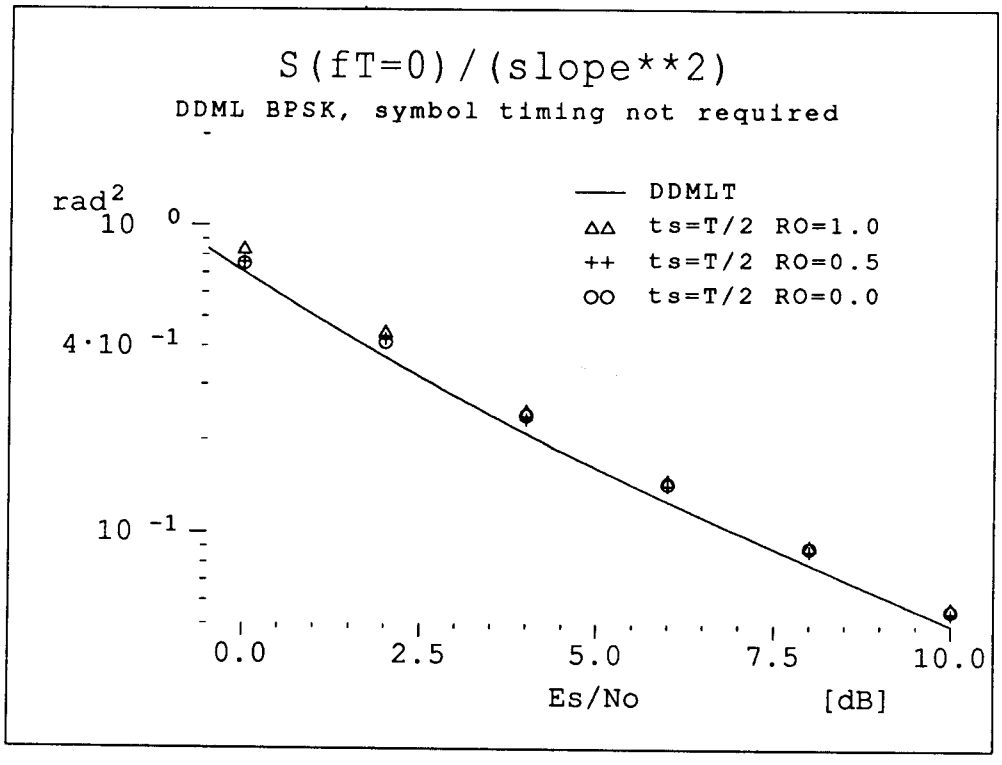


Fig. II.2-10a: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ of DDML$\overline{T}$ for BPSK, logarithmic scale

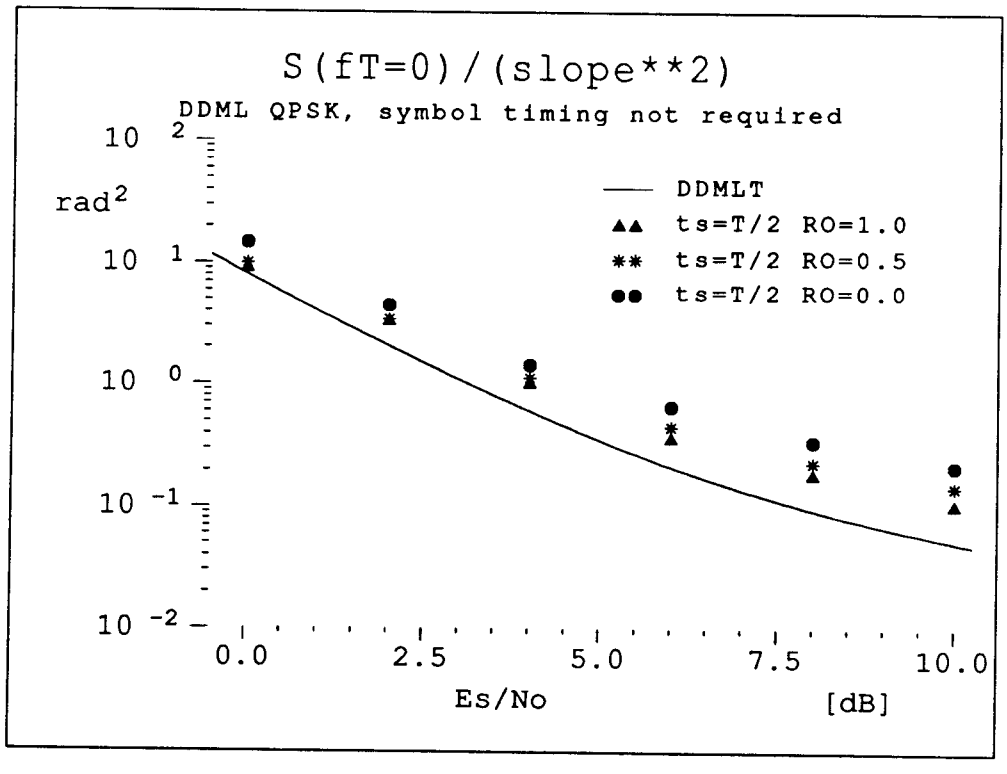


Fig. II.2-10b: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ of DDML$\overline{T}$ for QPSK, logarithmic scale

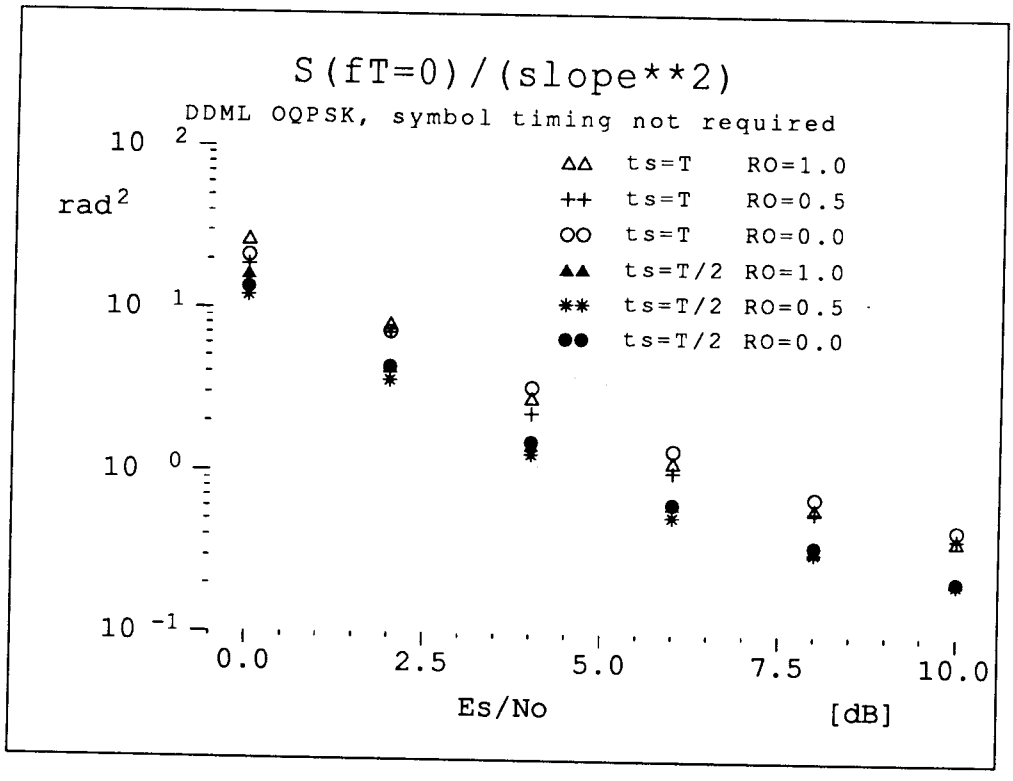


Fig. II.2-10c: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ of DDML$\overline{T}$ for OQPSK, logarithmic scale

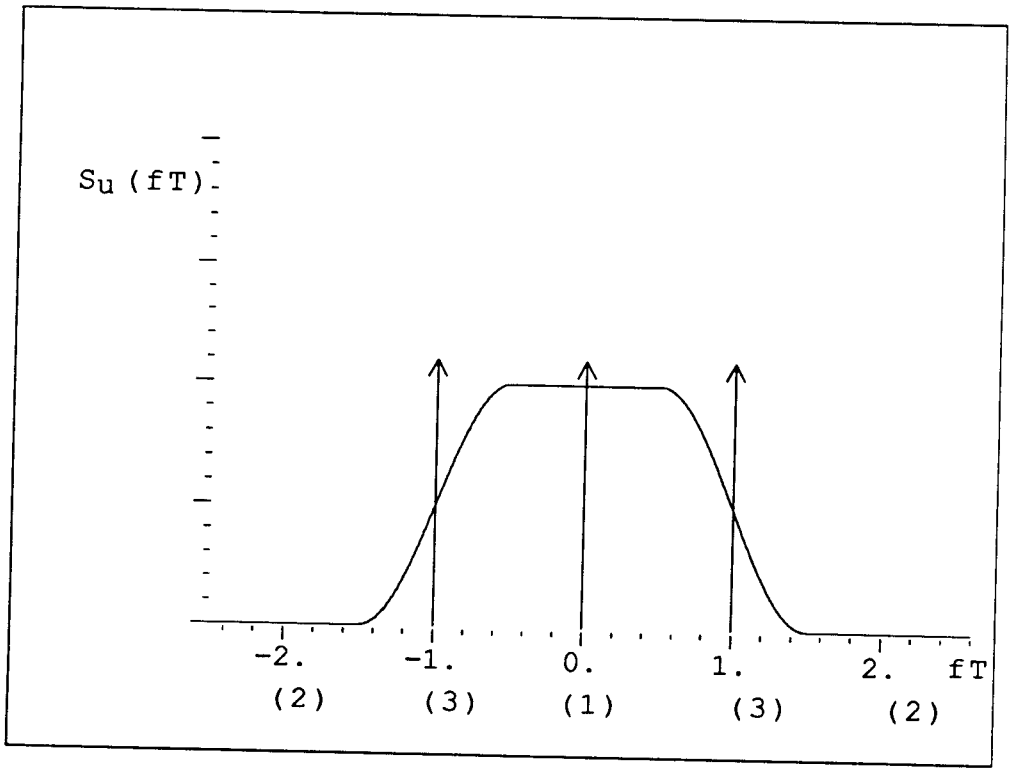


Fig. II.2-11: Shape of power spectral density of squared matched filter output

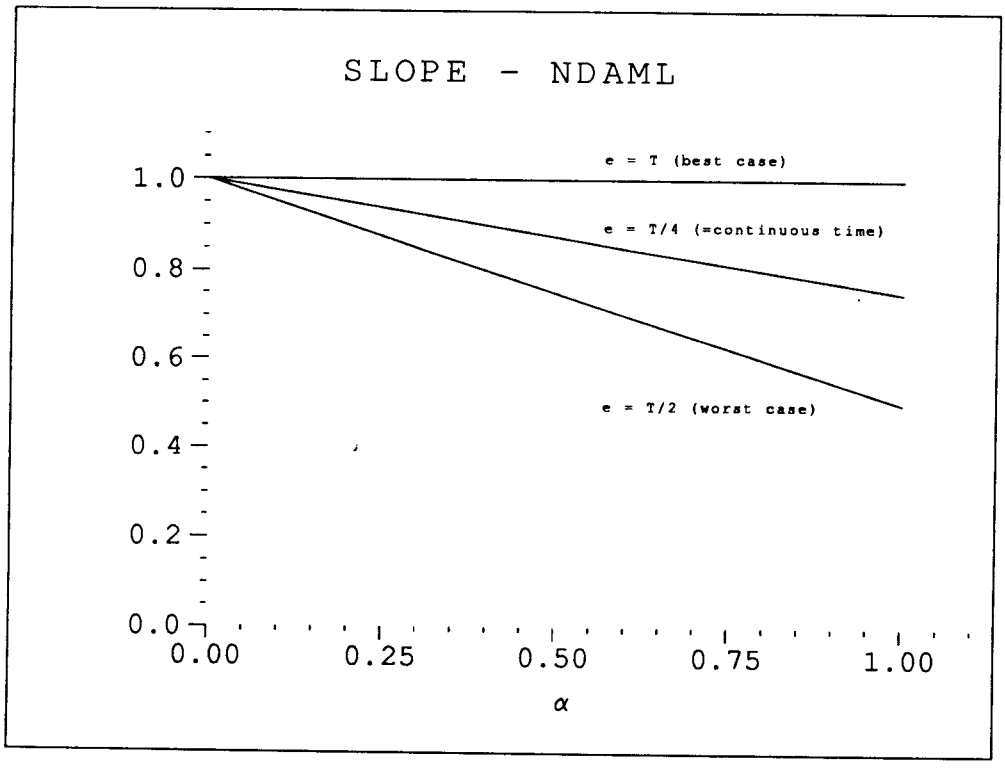


Fig. II.2-12a: NDAML$\overline{T}$ phase detector slope for BPSK as a function of rolloff for different timing phase offsets

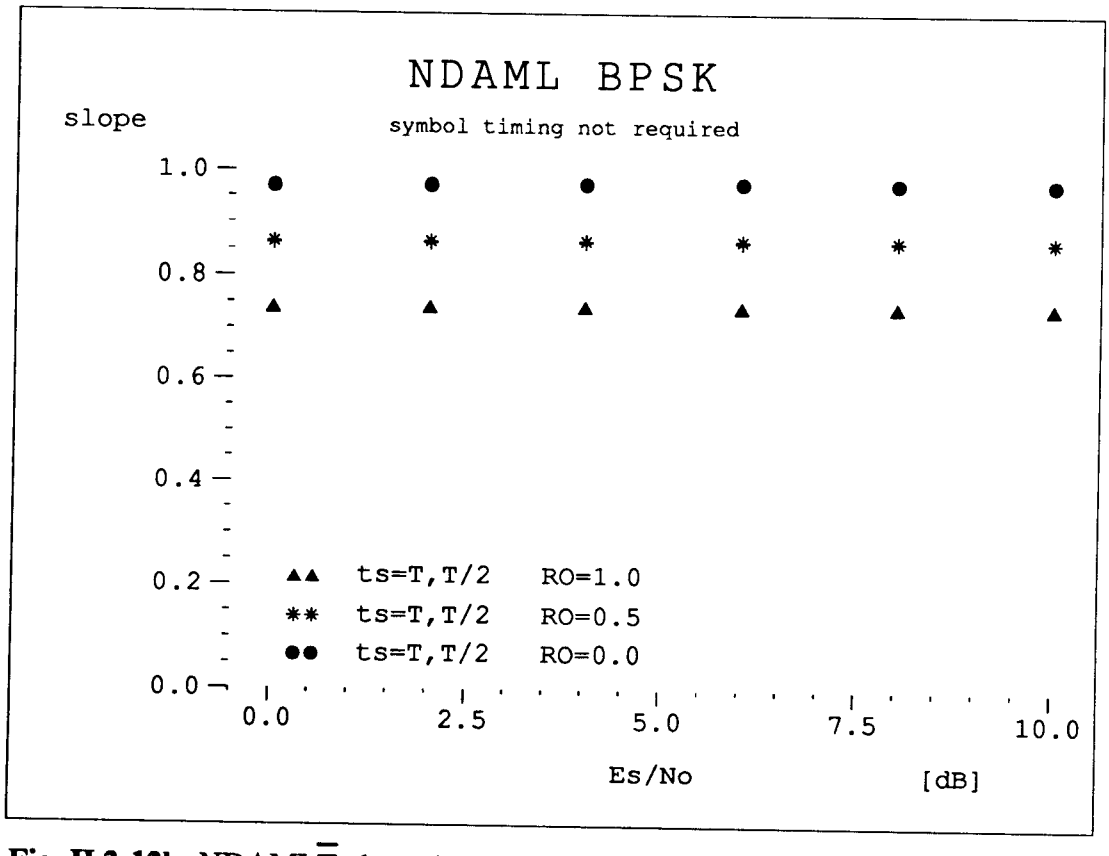


Fig. II.2-12b: NDAML$\overline{T}$ phase detector slope for BPSK as a function of E_S/N_0 for different sampling rates t_S

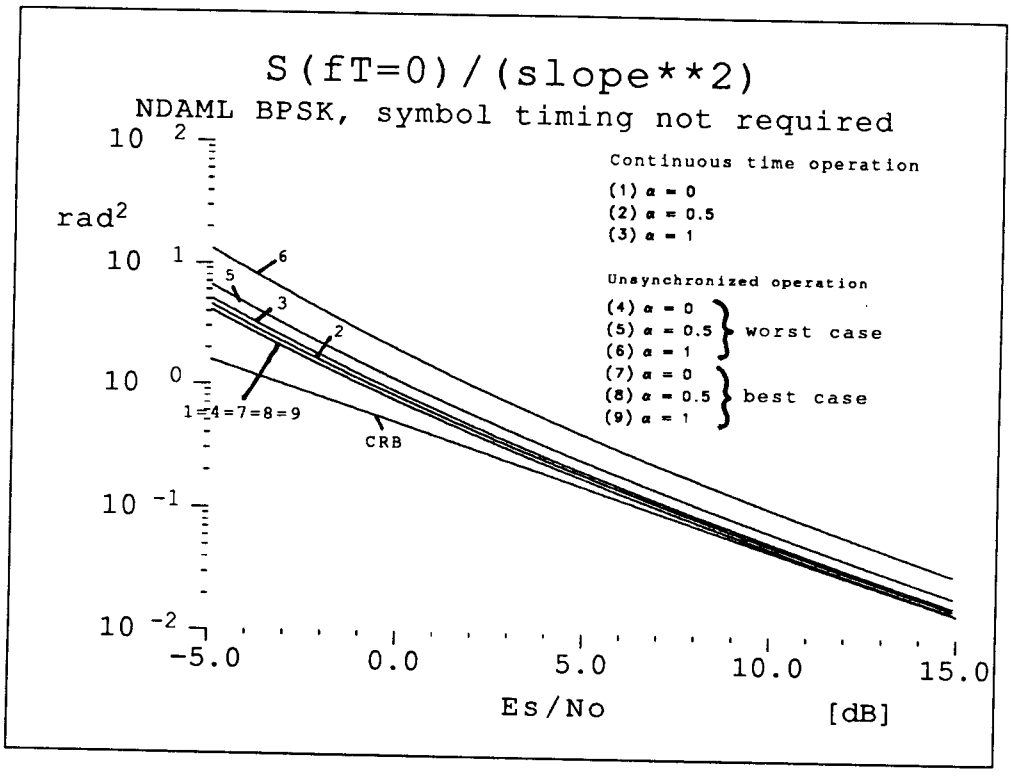


Fig. II.2-13a: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ for NDAML$\overline{T}$, BPSK, independent of timing, logarithmic scale

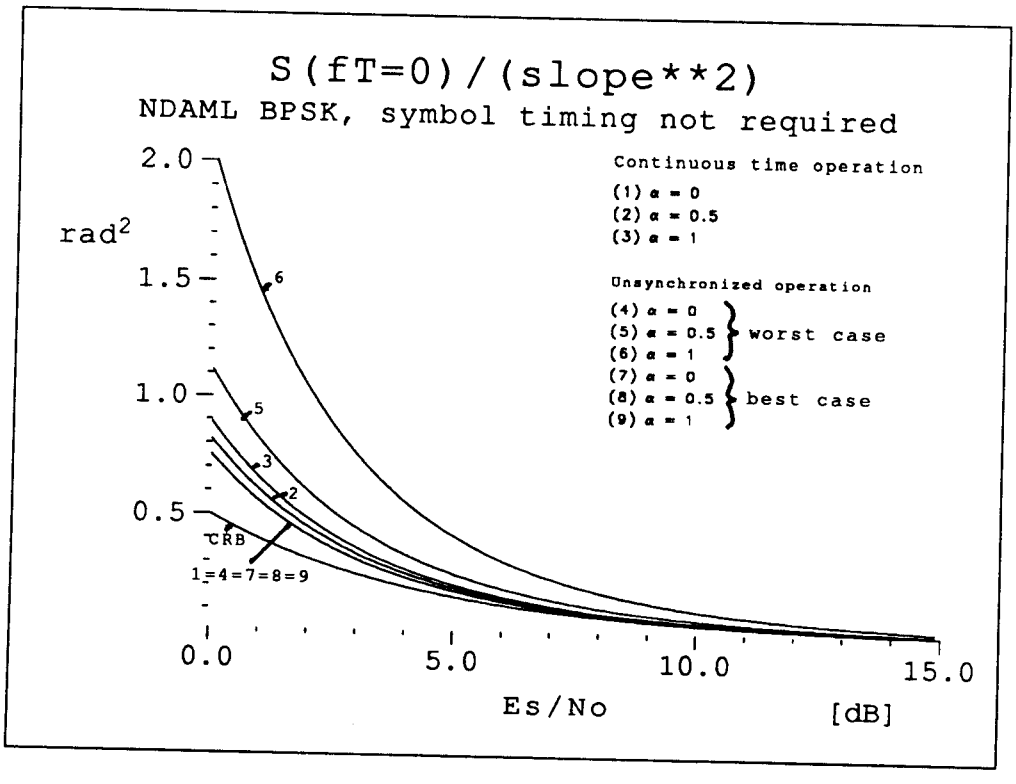


Fig. II.2-13b: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ for NDAML$\overline{T}$, BPSK, independent of timing, linear scale

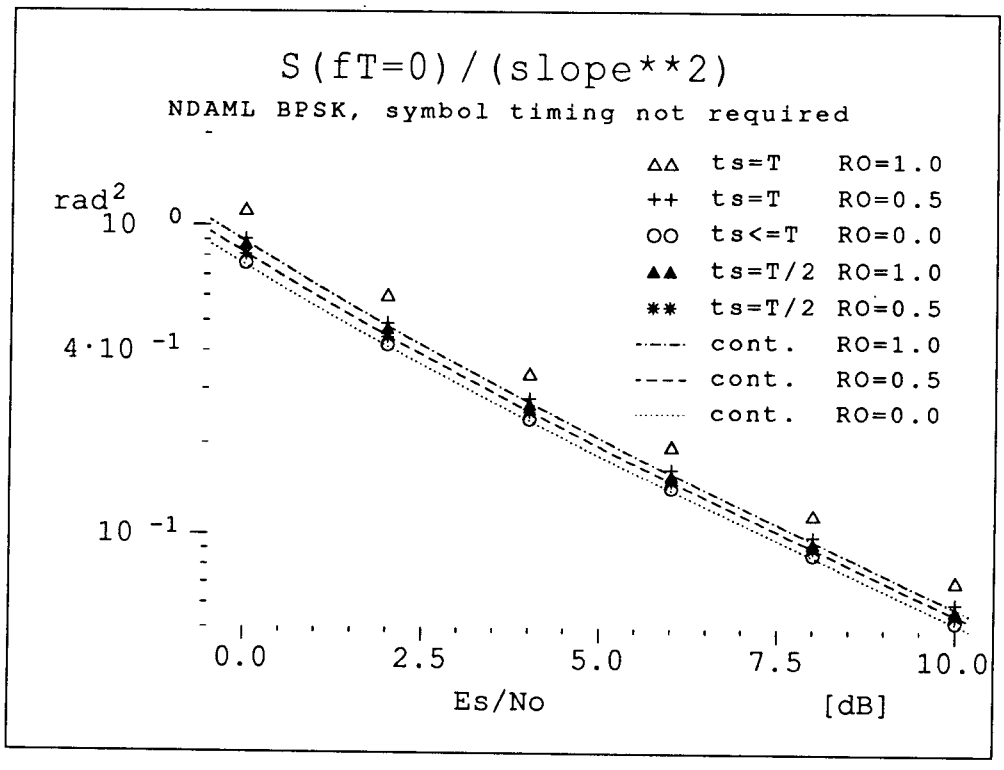


Fig. II.2-13c: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ for NDAML$\overline{T}$, BPSK, independent of timing, logarithmic scale

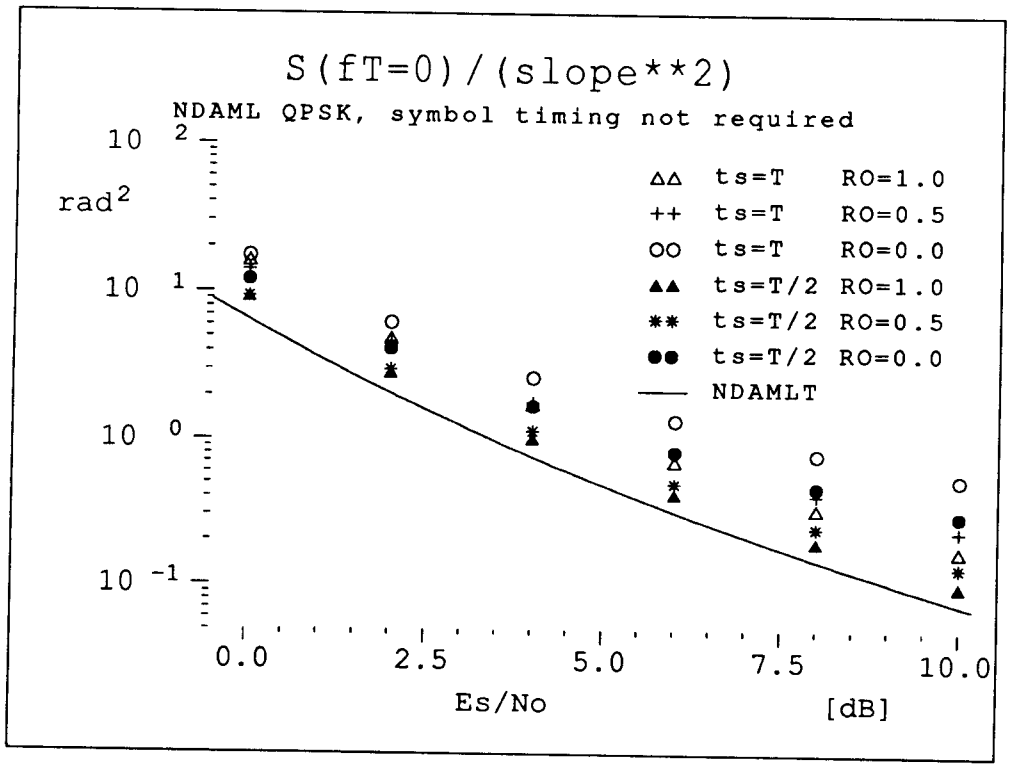


Fig. II.2-13d: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ for NDAML$\overline{T}$, QPSK, independent of timing, logarithmic scale

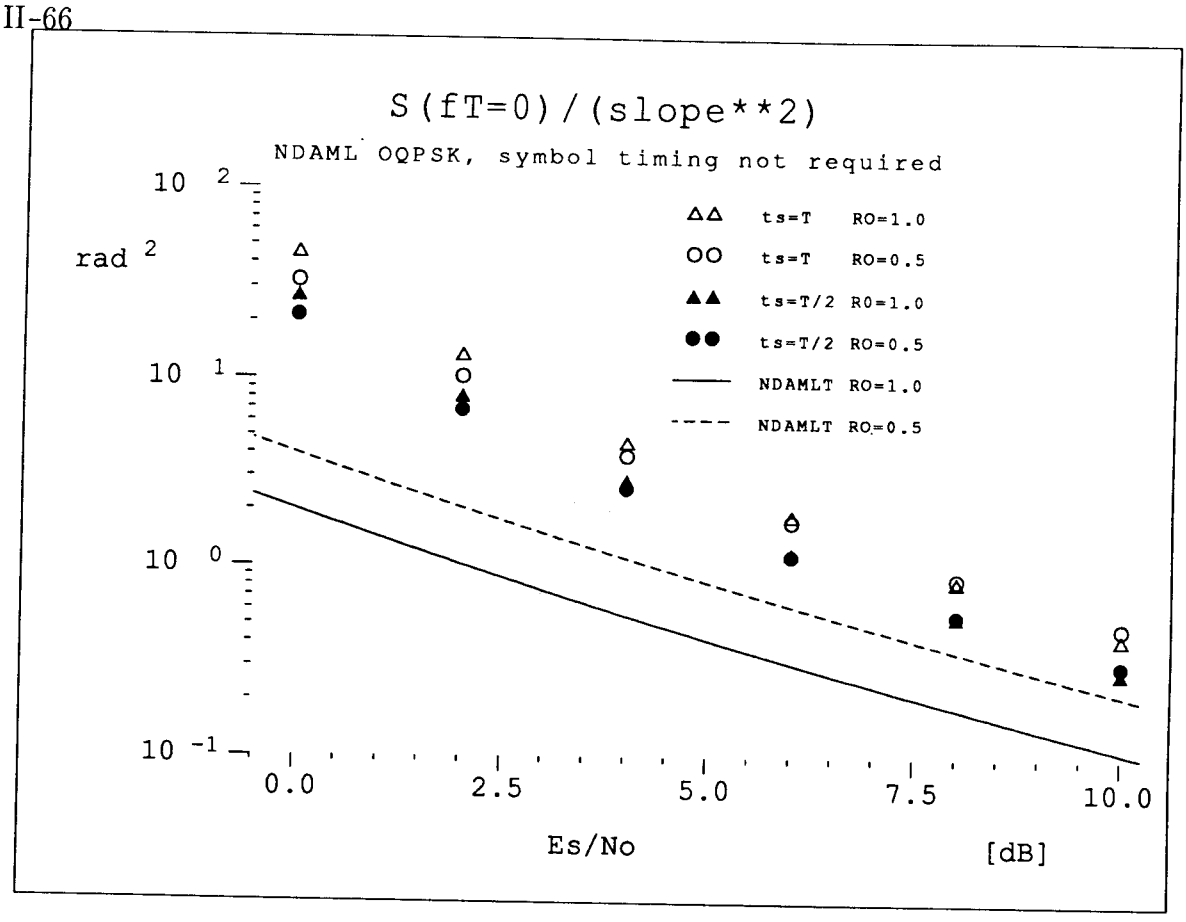


Fig. II.2-13e: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ for NDAML$\overline{T}$, OQPSK, independent of timing, logarithmic scale

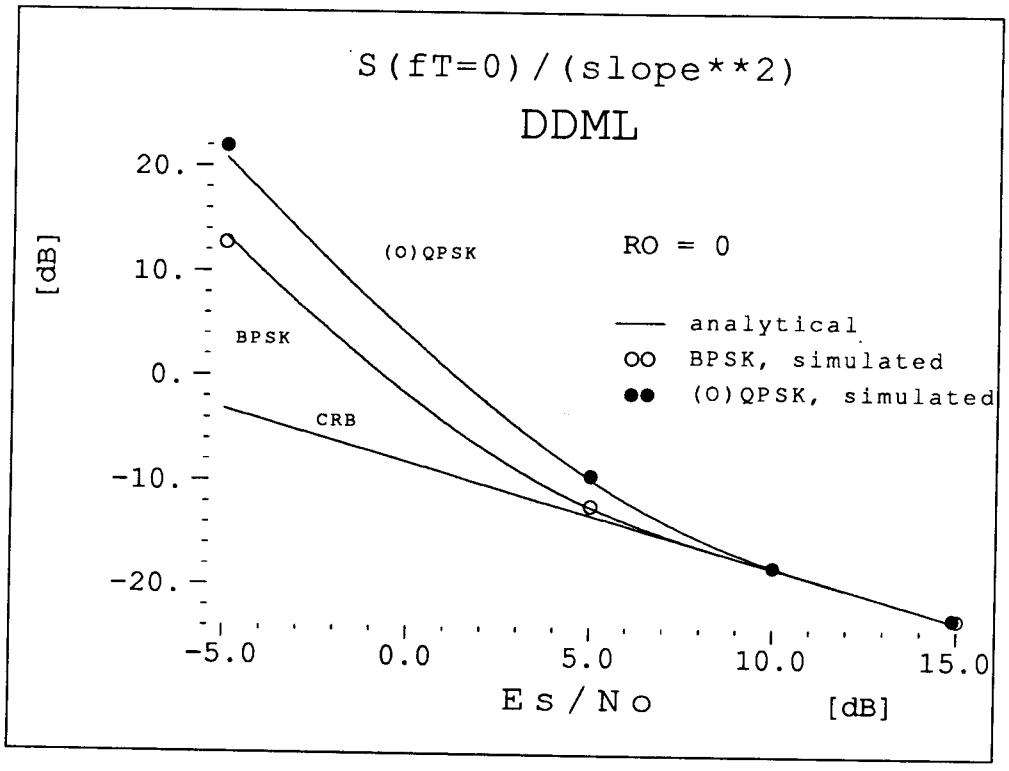


Fig. II.3-1: Timing: normalized power spectral density at fT=0 for DDML, rolloff α=0, logarithmic scale

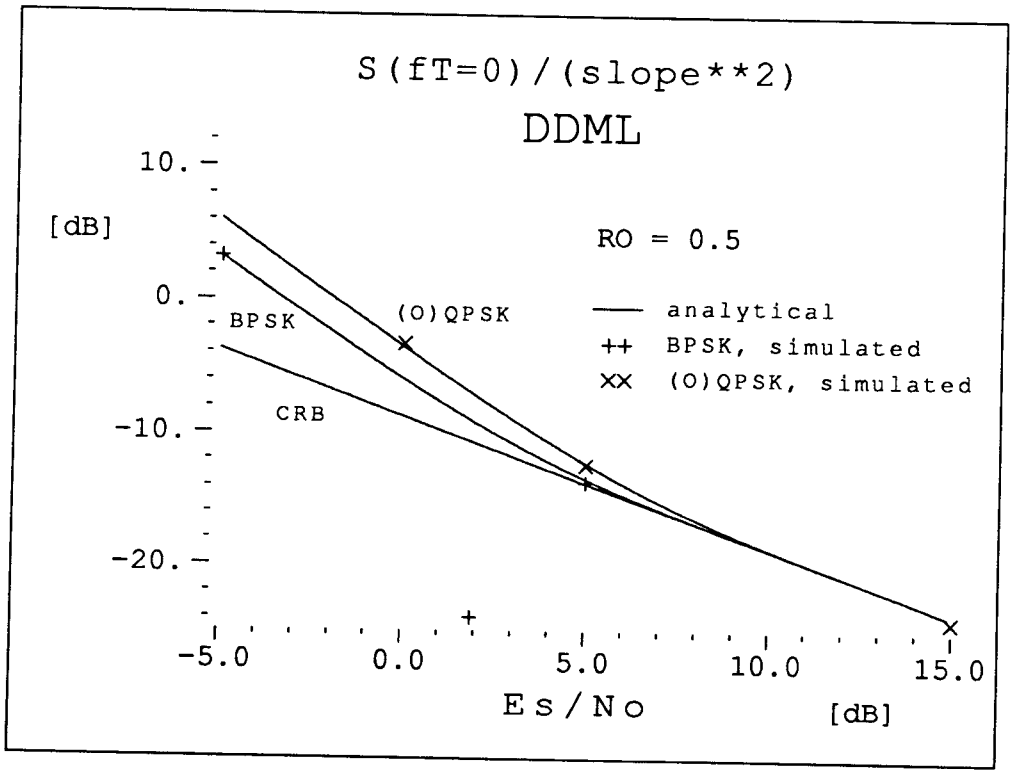


Fig. II.3-2: Timing: normalized power spectral density at fT=0 for DDML, rolloff α=0.5, logarithmic scale

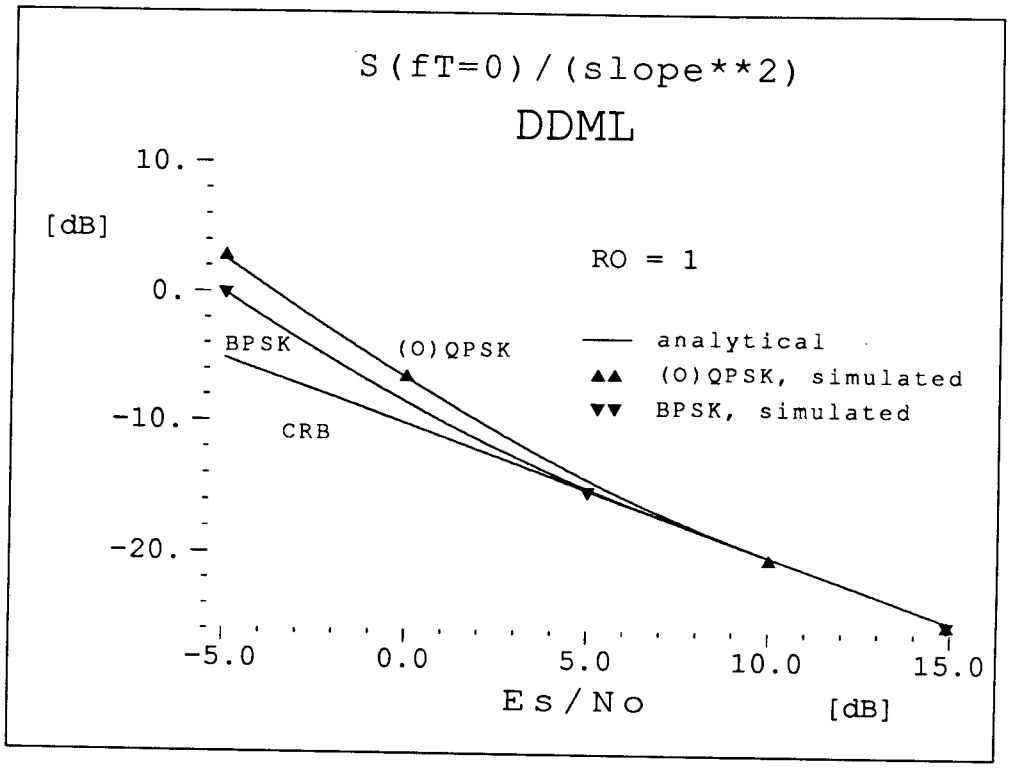


Fig. II.3-3: Timing: normalized power spectral density at fT=0 for DDML, rolloff α=1, logarithmic scale

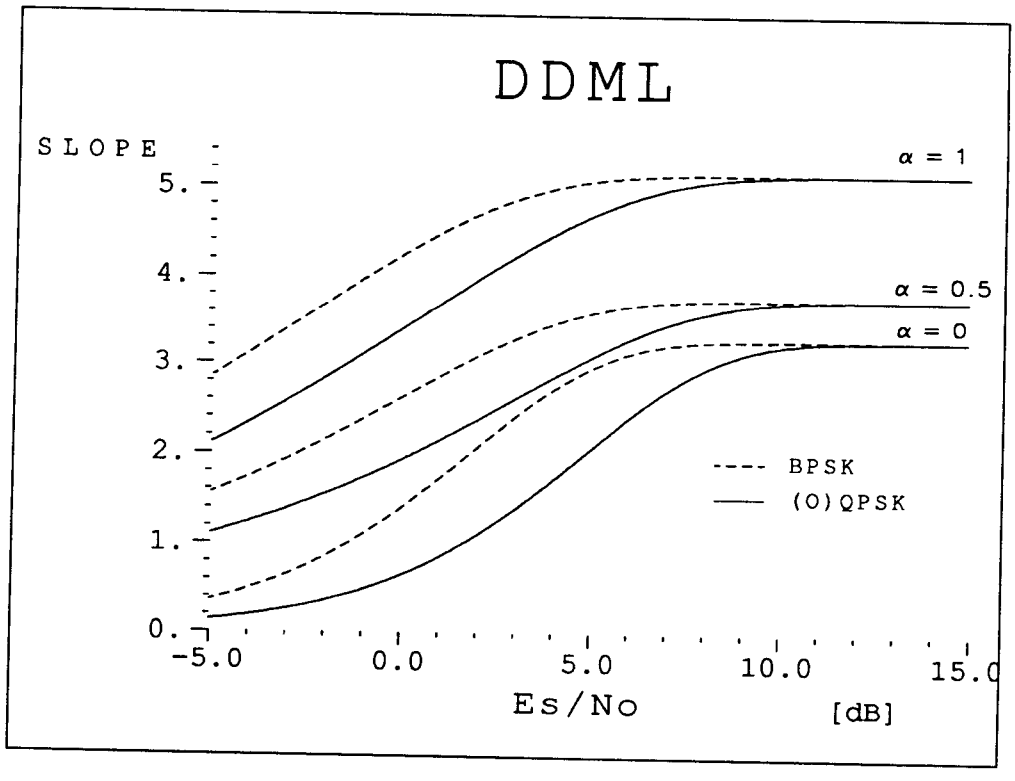


Fig. II.3-4: DDMLFB timing error detector slopes $K_{\tau\tau}$

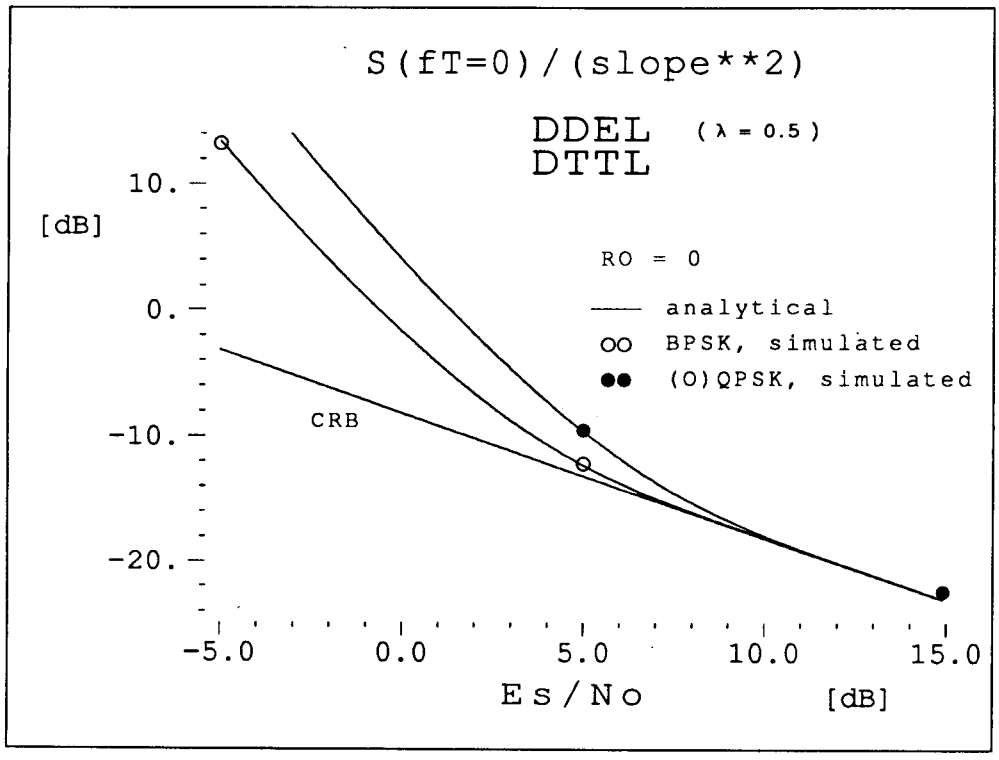


Fig. II.3-5: Timing: normalized power spectral density at fT=0 for DDEL (λ=1/2) and DTTL, rolloff α=0, logarithmic scale

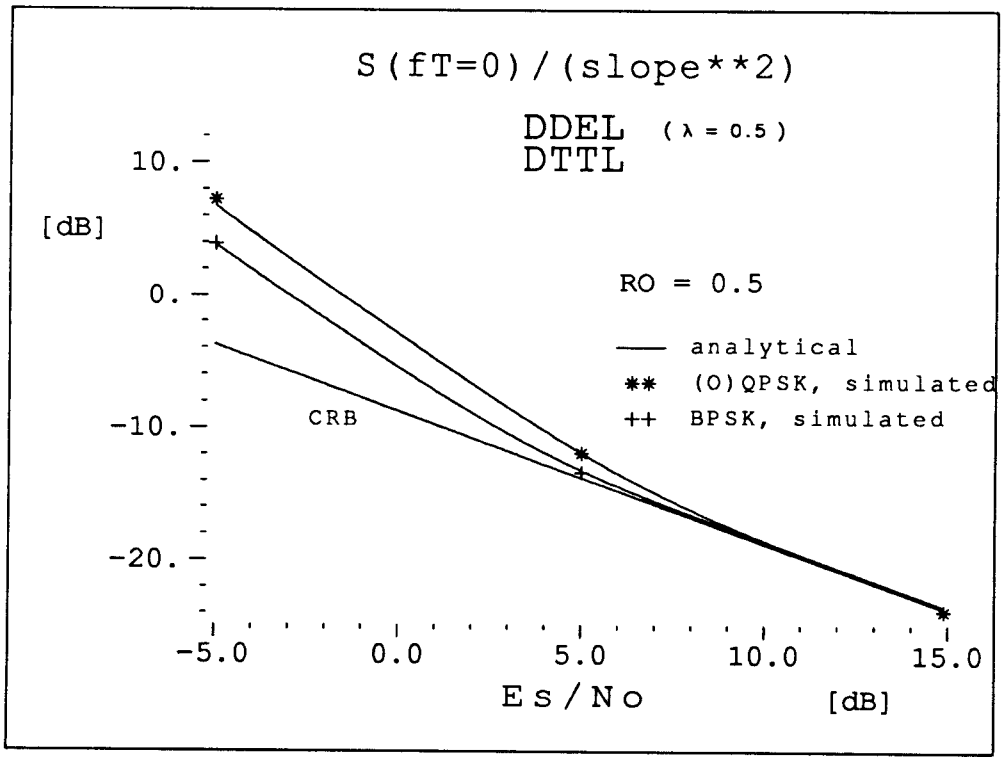


Fig. II.3-6: Timing: normalized power spectral density at fT=0 for DDEL (λ=1/2) and DTTL, rolloff α=0.5, log. scale

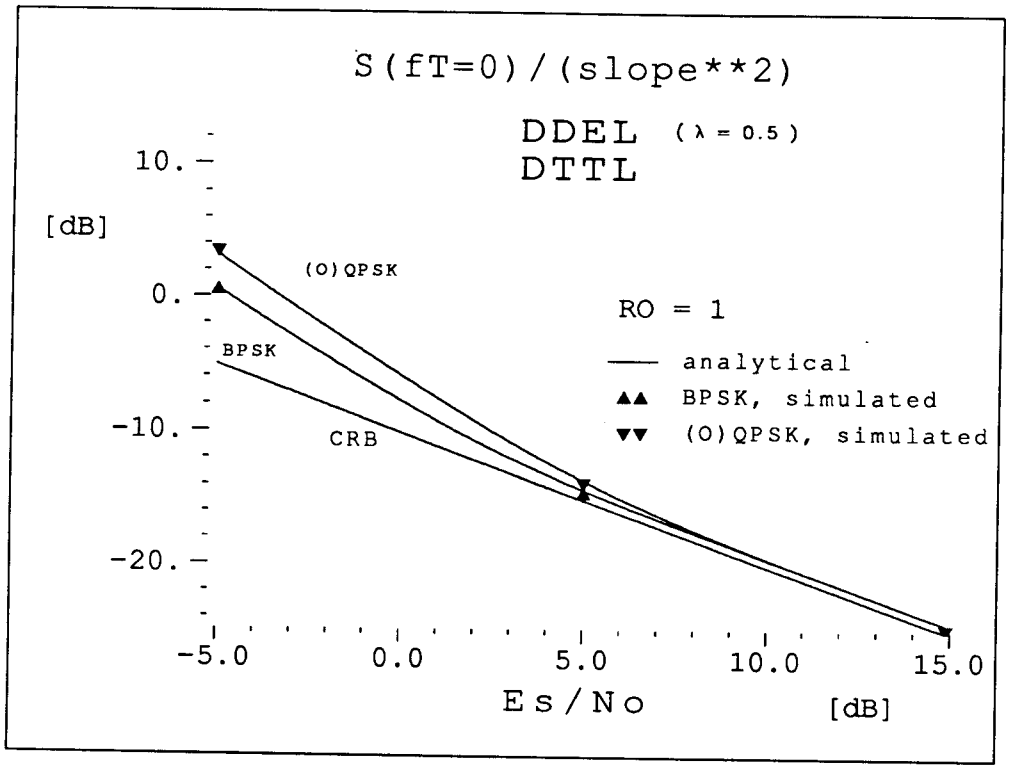


Fig. II.3-7: Timing: normalized power spectral density at fT=0 for DDEL (λ=1/2) and DTTL, rolloff α=1, log. scale

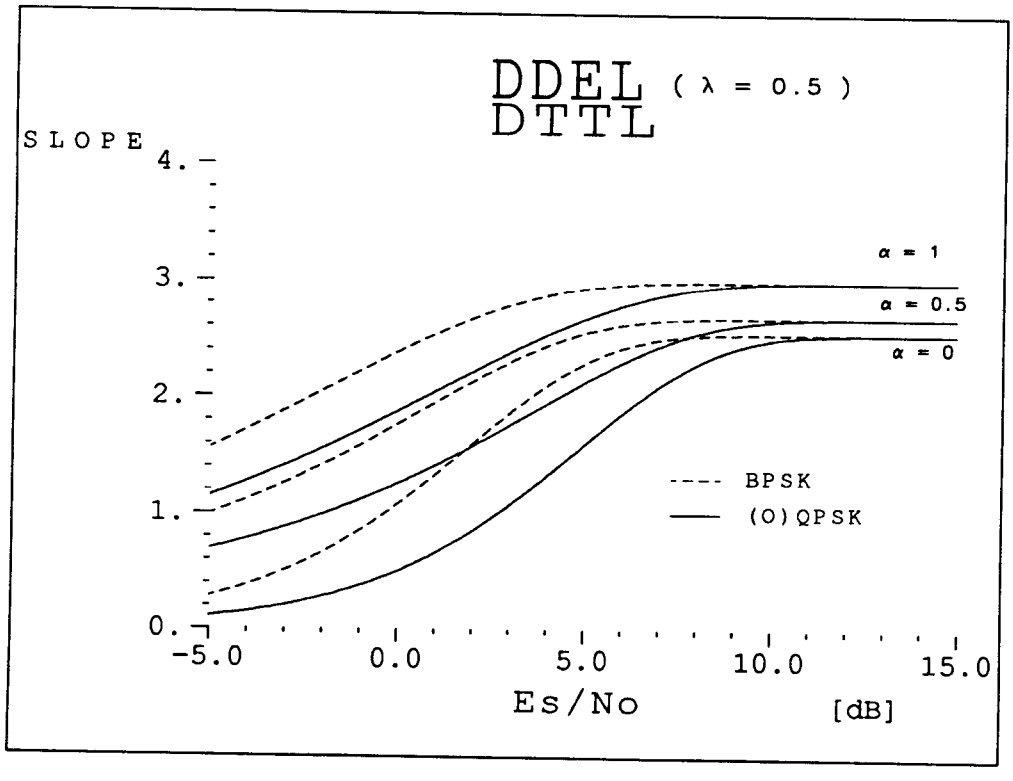


Fig. II.3-8: DDEL (λ=1/2) and DTTL timing error detector slopes $K_{\tau\tau}$

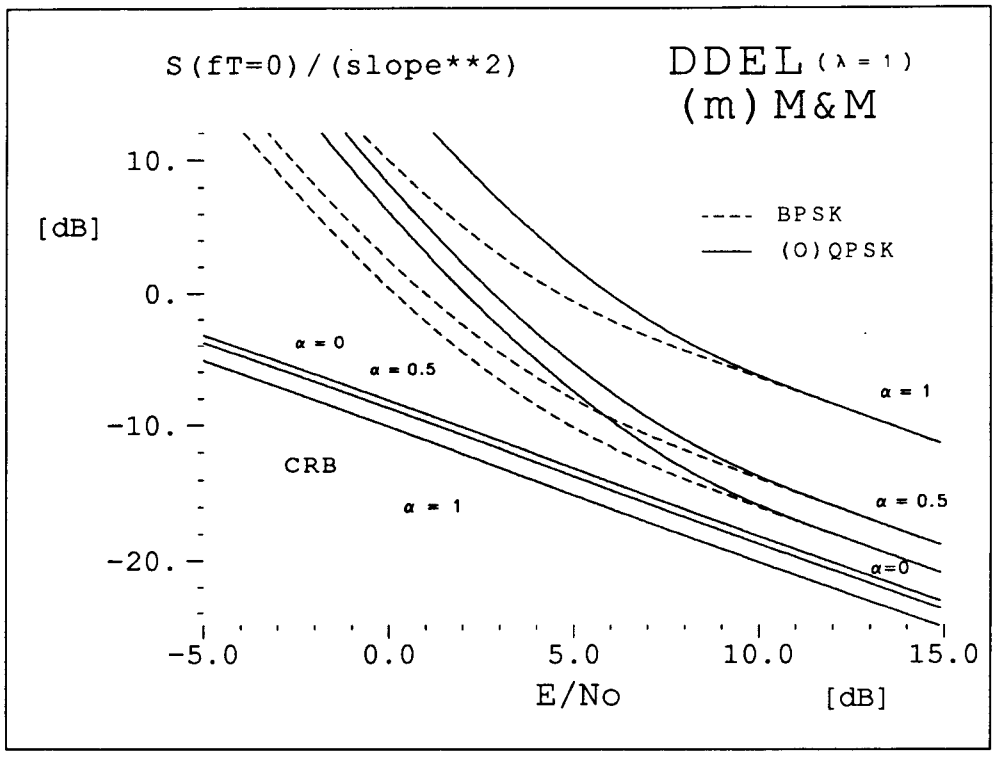


Fig. II.3-9: Timing: normalized power spectral density at fT=0 for DDEL (λ=1), M&M and mM&M, log. scale

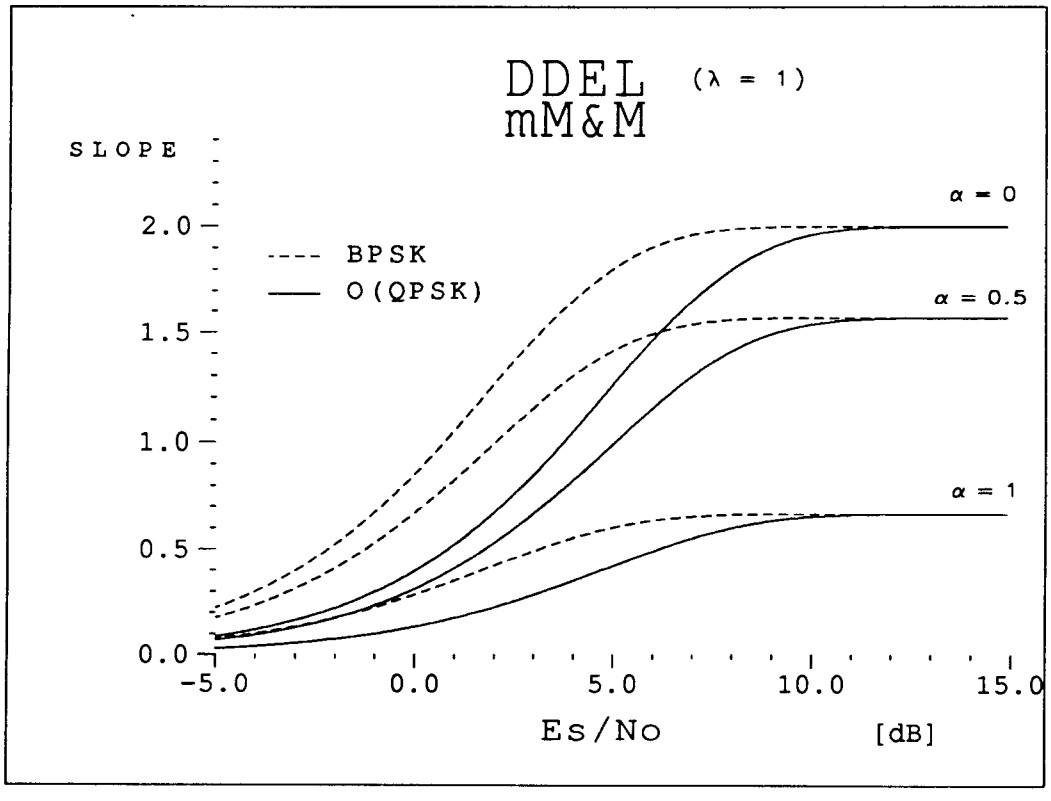


Fig. II.3-10: DDEL (λ=1), M&M, and mM&M timing error detector slopes $K_{\tau\tau}$

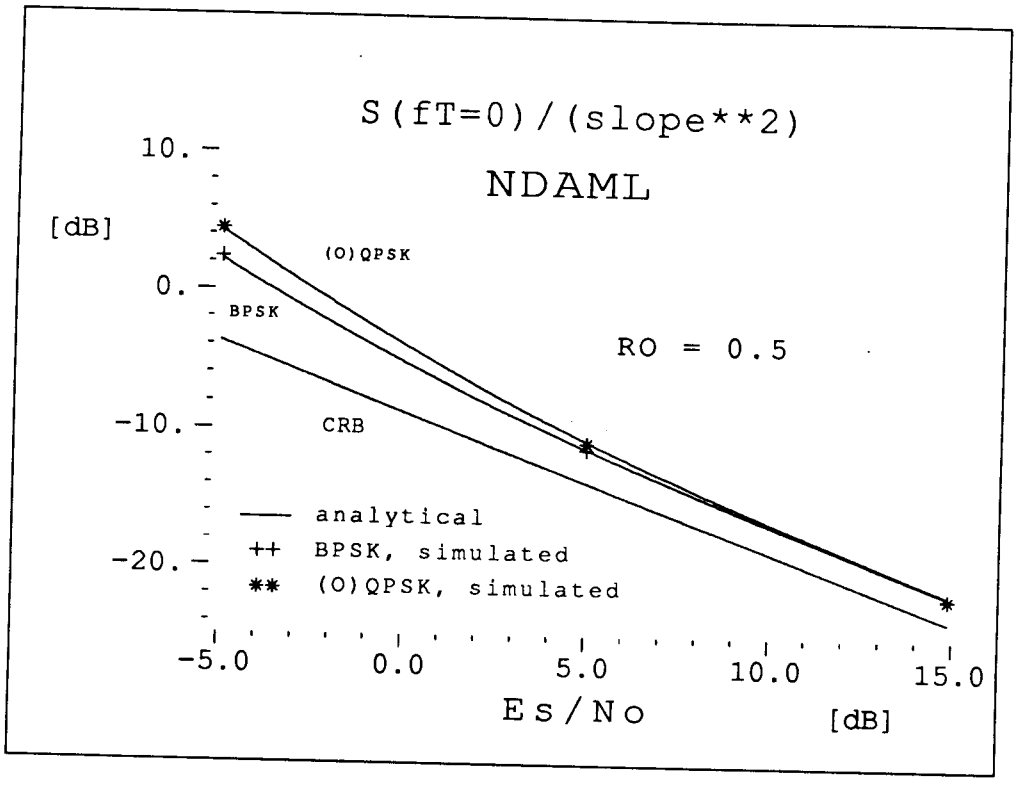


Fig. II.3-11: Timing: normalized power spectral density at fT=0 for NDAML, rolloff α=0.5, logarithmic scale

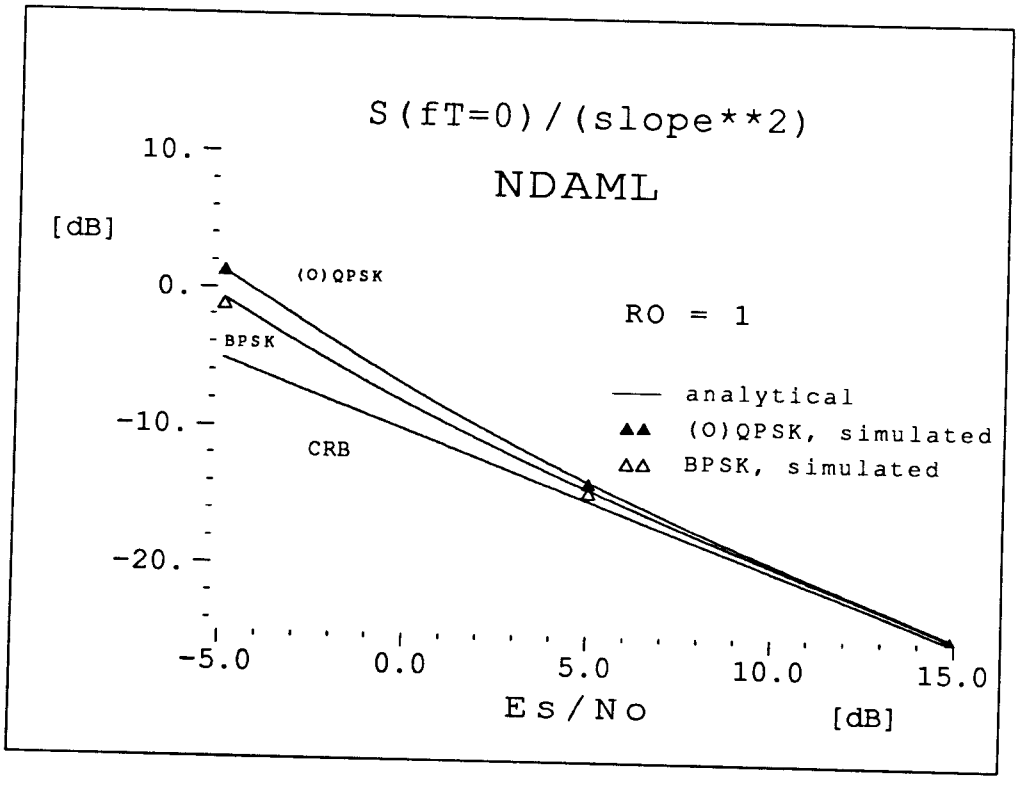


Fig. II.3-12: Timing: normalized power spectral density at fT=0 for NDAML, rolloff α=1, logarithmic scale

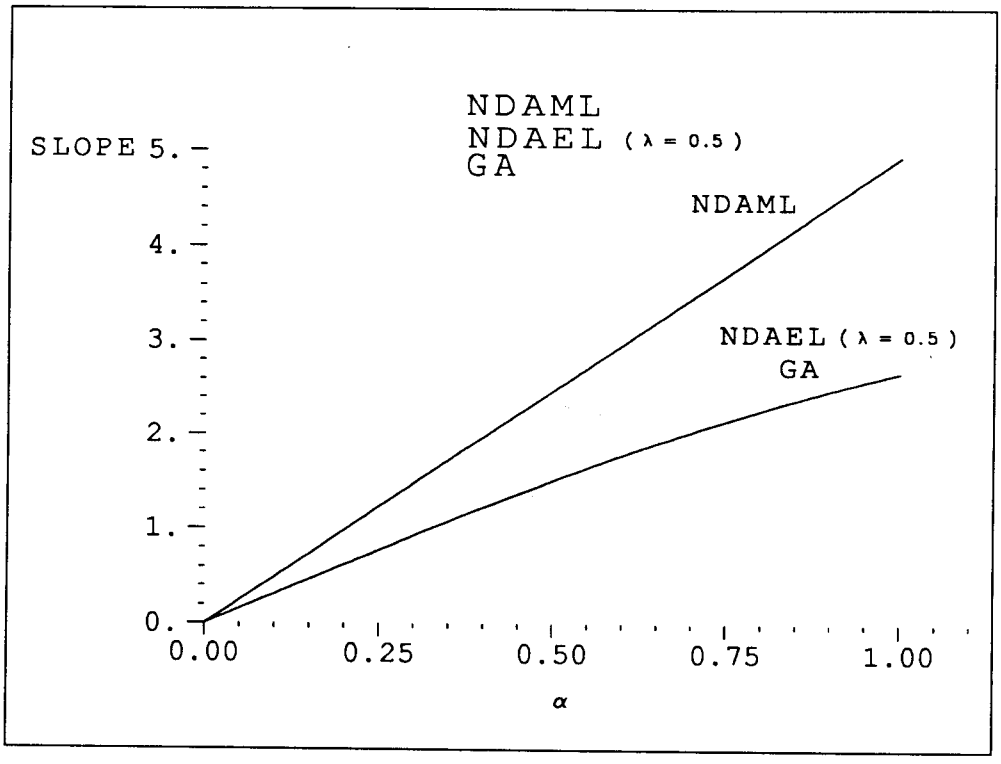


Fig. II.3-13: NDAML, NDA EL (λ=1/2), and GA timing error detector slopes $K_{\tau\tau}$

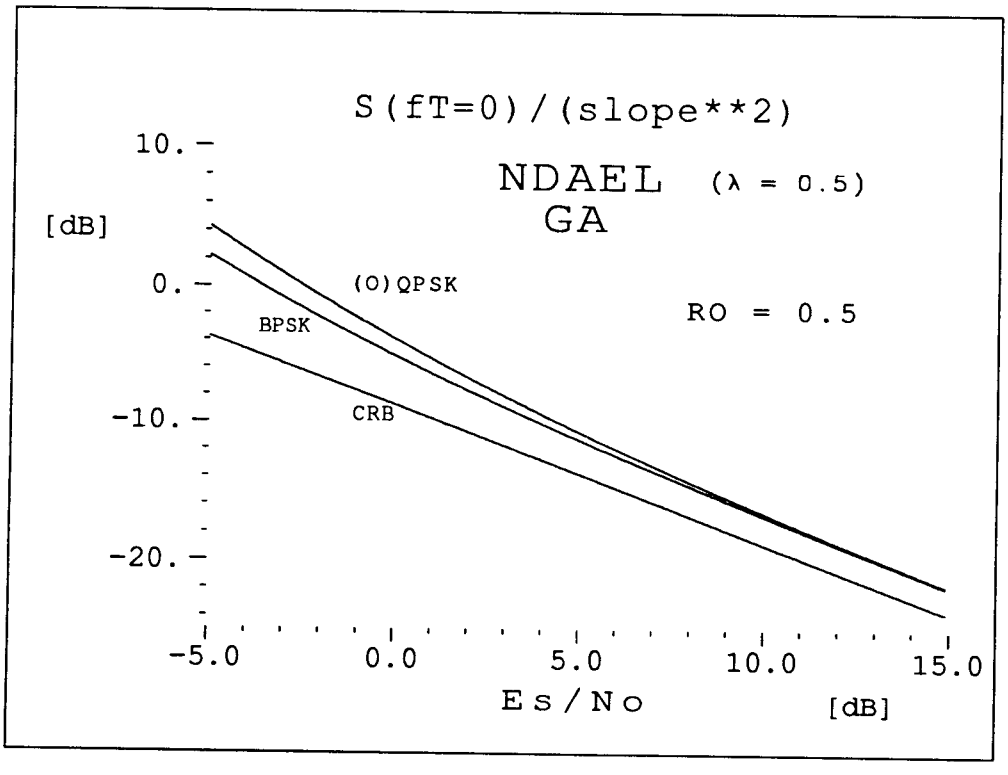


Fig. II.3-14: Timing: normalized power spectral density at fT=0 for NDA EL (λ=1/2) and GA, rolloff α=0.5, log. scale

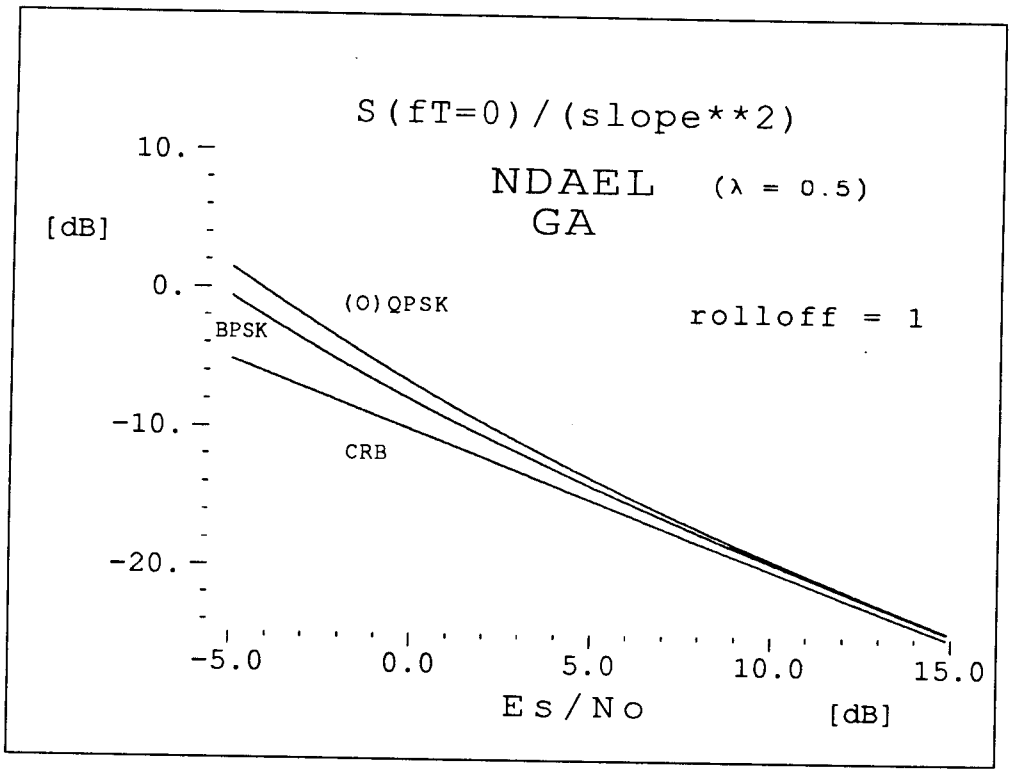


Fig. II.3-15: Timing: normalized power spectral density at fT=0 for NDA EL (λ=1/2) and GA, rolloff α=1, log. scale

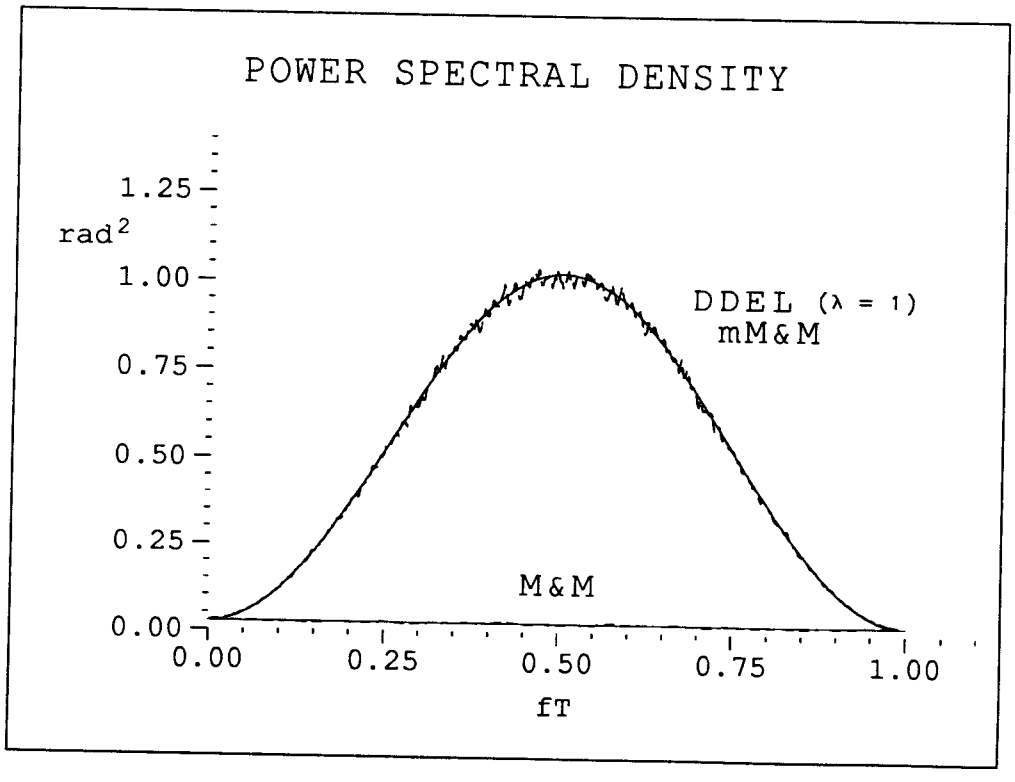


Fig. II.3-16: Noise power spectral density of DDEL (λ=1), modified M&M, and M&M, analytical and simulated curves

III. THE INFLUENCE OF TRACKING ERRORS ON THE BIT ERROR RATE

III.1 DERIVATION OF THE EXPRESSION FOR THE BER DEGRADATION

In Part III of this study we investigate the influence of tracking errors on the bit error rate (BER). As in Part II, "ideal" conditions are assumed : additive white Gaussian noise channel, zero carrier frequency error, Nyquist–I pulse at the matched filter output, and ideal receiver operations (no quantization, ...)

Usually, the BER degradation due to tracking errors is defined as the increase of E_b/N_o, measured in dB, necessary to maintain the same BER as without tracking errors. In this section, we derive an expression for the increase in E_b/N_o necessary to maintain the same signal–to–noise ratio at the input of the decision device as in the case of no tracking errors; "noise" is interpreted as the total disturbance, i.e. ISI and quadrature interference (both caused by tracking errors) adding to thermal noise. Strictly speaking, maintaining the same signal–to–noise ratio at the input of the decision device does not guarantee the same BER, because the distribution of the decision variable is no longer Gaussian when tracking errors are present. However, for small enough tracking error variances, the resulting ISI and quadrature interference are small as compared to the Gaussian additive noise term. Consequently, using central limit theorem arguments, the decision variable is essentially Gaussian, with a variance equal to that of the total disturbance; a more formal argumentation is given in Appendix III.A. In this case, the obtained expression for the degradation of the signal–to–noise ratio at the input of the decision device corresponds to the BER degradation.

III.1.1 Carrier phase errors only

The receiver bases its decision about the transmitted bit a_k upon the sign of the decision variable Re $[p(k;\hat{\tau})\ \exp(-j\hat{\theta})]$, where $p(k;\hat{\tau})$ is the complex–valued matched filter output sample at the estimated decision instant $kT + \hat{\tau}$, and $\exp(-j\hat{\theta})$ denotes the phase rotation by the carrier phase estimate $\hat{\theta}$. Assuming that there is a carrier phase error ϕ but no timing error ($\epsilon = 0$), we now investigate the decision variable for the various modulation formats.

– BPSK

In the case of BPSK, the decision variable for $a_k=1$ is given by

$$\mathrm{Re}[p(k;\hat{\tau})\, e^{-j\hat{\theta}}] = \cos\phi + N_{I,k} \qquad \text{(III.1.1–1)}$$

The first term in (III.1.1–1) is a useful term; its square mean is given by

$$(E[\cos\phi])^2 \cong 1 - \mathrm{Var}[\phi] \qquad \text{(III.1.1–2)}$$

The above approximation assumes that $\mathrm{Var}[\phi] \ll 1\ [\mathrm{rad}^2]$, which is fulfilled in all cases of practical interest. The second term in (III.1.1–1) is a zero–mean Gaussian noise term with variance equal to $N_0/(2E_b)$. Let us define SNR as the ratio of the square mean of the decision variable to the variance of the decision variable; denoting by SNR_0 the value of SNR which corresponds to $\phi = 0$, i.e. $\mathrm{SNR}_0 = 2E_b/N_0$, we obtain:

$$\frac{\mathrm{SNR}}{\mathrm{SNR}_0} = 1 - \mathrm{Var}[\phi] \qquad \text{(BPSK)} \qquad \text{(III.1.1–3)}$$

It is shown in Appendix III.A that a receiver with small tracking errors, operating at a given E_b/N_0, has essentially the same BER as a receiver without tracking errors, operating at $(\mathrm{SNR}/\mathrm{SNR}_0)\times(E_b/N_0)$. The corresponding BER degradation is given by:

$$\begin{aligned}\text{BER degrad. [dB]} &= -10 \log \left[\frac{\mathrm{SNR}}{\mathrm{SNR}_0}\right] \\ &\cong 4.3\ \mathrm{Var}[\phi] \qquad \text{(BPSK)} \qquad \text{(III.1.1–4)}\end{aligned}$$

The approximation in the second line of (III.1.1–4) involves the linearization $\log(1+x) \cong 0.43\ x$, which is valid for small BER degradations (for a BER degradation of 1 dB the approximation error is about 10%).

– QPSK

In the case of QPSK, the decision variable for $a_k=1$ is given by

$$\mathrm{Re}[p(k;\hat{\tau})\, e^{-j\hat{\theta}}] = \cos\phi - b_k \sin\phi + N_{I,k} \qquad \text{(III.1.1–5)}$$

where $b_k = \pm 1$ is the k–th quadrature bit. The first term in (III.1.1–5) is the same useful term as for BPSK. The second term in (III.1.1–5) is a zero–mean quadrature interference term, with a variance equal to:

$$E[\sin^2\phi] \cong \mathrm{Var}[\phi] \qquad \text{(III.1.1–6)}$$

The third term in (III.1.1–5) is a zero–mean noise term with variance equal to $N_o/(2E_b)$. The resulting ratio SNR/SNR_o is given by:

$$\frac{SNR}{SNR_o} = \frac{1 - \mathrm{Var}[\phi]}{1 + \frac{2E_b}{N_o}\mathrm{Var}[\phi]} \qquad \text{(QPSK)} \qquad \text{(III.1.1–7)}$$

The numerator in (III.1.1–7) is caused by the reduction of the mean of the decision variable, whereas the denominator is due to the quadrature interference. The corresponding BER degradation is given by:

$$\text{BER degrad. [dB]} = -10 \log \left[\frac{SNR}{SNR_o}\right]$$

$$\cong 4.3 \left(1 + \frac{2E_b}{N_o}\right) \mathrm{Var}[\phi] \qquad \text{(QPSK)} \qquad \text{(III.1.1–8)}$$

When $E_b/N_o > 1/2$ (or –3 dB) the effect of the quadrature interference is more important than the reduction of the mean of the decision variable.

– <u>OQPSK</u>

In the case of OQPSK, the decision variable for $a_k=1$ is given by

$$\mathrm{Re}\,[p(k;\hat{\tau})\, e^{-j\hat{\theta}}] = \cos\phi - \sum_m b_{k-m}\, g(mT - \tfrac{T}{2}) \sin\phi + N_{I,k} \qquad \text{(III.1.1–9)}$$

where g(t) is the baseband pulse at the matched filter output. The first and the third term in (III.1.1–9) are the same as for BPSK and QPSK. The second term in (III.1.1–9) is a zero–mean quadrature interference term, with a variance equal to:

$$\sum_m g^2(mT - \tfrac{T}{2})\, E[\sin^2\phi] \cong \sum_m g^2(mT - \tfrac{T}{2})\, \mathrm{Var}[\phi] \qquad \text{(III.1.1–10)}$$

For a given phase error variance, the variance of the quadrature interference term decreases with increasing excess bandwidth. The resulting ratio SNR/SNR_o is given by :

$$\frac{SNR}{SNR_o} = \frac{1 - Var[\phi]}{1 + \frac{2E_b}{N_o} \sum_m g^2(mT - \frac{T}{2}) Var[\phi]} \quad \text{(OQPSK)} \quad \text{(III.1.1–11)}$$

The numerator in (III.1.1–11) is caused by the reduction of the mean of the decision variable, whereas the denominator is due to the quadrature interference. The corresponding BER degradation is given by:

$$\text{BER degrad. [dB]} = -10 \log \left[\frac{SNR}{SNR_o}\right]$$

$$\cong 4.3 \left(1 + \frac{2E_b}{N_o} \sum_m g^2(mT - \frac{T}{2})\right) Var[\phi] \qquad \text{(OQPSK)}$$

(III.1.1–12)

When E_b/N_o satisfies the inequality

$$\frac{E_b}{N_0} > \frac{1}{2 \sum_m g^2(mT - \frac{T}{2})} \qquad \text{(III.1.1–13)}$$

the effect of the quadrature disturbance is more important than the reduction of the mean of the decision variable. The right–hand side of (III.1.1–13) is an increasing function of the excess bandwidth; when g(t) is a cosine rolloff pulse with rolloff factor α, this right–hand side ranges from 1/2 (or –3 dB) to 1 (or 0 dB) when α increases from 0 to 1. Note that for $\alpha = 0$, the BER degradation for OQPSK is the same as for QPSK.

Comparing (III.1.1–4), (III.1.1–8) and (III.1.1–12) for a given carrier phase error variance $Var[\phi]$, the following conclusions can be drawn :

– BPSK is least sensitive to carrier phase errors, because there is no quadrature interference when a carrier phase error is present. The BER degradation is caused solely by a reduction of the mean of the decision variable.

– QPSK and OQPSK are more sensitive to carrier phase errors than BPSK, as a carrier phase error gives rise not only to a reduction of the mean of the decision variable, but also to an increase of the variance of the decision variable (occurrence of quadrature interference). The effect of the quadrature interference increases with increasing E_b/N_o; for $E_b/N_o \gg 0$ dB, the BER degradation for (O)QPSK is much larger than for BPSK. The variance of the quadrature interference for OQPSK is not larger than for QPSK; its sensitivity to carrier phase errors is between the sensitivity for BPSK and that for QPSK, and decreases with increasing excess bandwith.

When evaluating the variance of the decision variable, we have taken into account only the additive noise and (if present) the quadrature disturbance. Strictly speaking, also the fluctuation of the useful term cos ϕ contributes to the variance. This contribution could be taken into account by adding to the second line of (III.1.1–4), (III.1.1–8) and (III.1.1–12) the term :

$$4.3 \frac{2E_b}{N_o} \mathrm{Var}[\cos \phi] \cong 4.3 \frac{2E_b}{N_o} \mathrm{Var}[\tfrac{1}{2} \phi^2]$$

As $\mathrm{Var}[\phi^2]$ is in the same order of magnitude as the square of $\mathrm{Var}[\phi]$, it is usually so small that it can be neglected with respect to the terms that are linear in $\mathrm{Var}[\phi]$.

III.1.2 Timing errors only

Now we investigate the decision variable Re $[p(k;\hat{\tau})\ \exp(-j\hat{\theta})]$ under the assumption that there is a timing error ϵT, but no carrier phase error ($\phi = 0$). The decision variable for $a_k = 1$ is given by

$$\mathrm{Re}[p(k;\hat{\tau})\ e^{-j\hat{\theta}}] = g(\epsilon T) + \sum_{m\neq 0} a_{k-m}\, g(mT + \epsilon T) + N_{I,k} \qquad \text{(III.1.2–1)}$$

The first term in (III.1.2–1) is a useful term, with a square mean given by

$$(E[g(\epsilon T)])^2 \cong 1 - (-g''(0)T^2)\ \mathrm{Var}[\epsilon] \qquad \text{(III.1.2–2)}$$

where $g''(t)$ denotes the second derivative, with respect to time, of $g(t)$. The above approximation assumes that $\mathrm{Var}[\epsilon] \ll 1$, which is fulfilled in all cases of practical interest. As $(-g''(0)T^2)$ increases with increasing excess bandwidth, the mean of the useful term in (III.1.2–1) decreases with increasing excess bandwidth for a given timing error variance. The second term in (III.1.2–1) is a zero–mean ISI term; its variance is given by :

$$\sum_{m\neq 0} E[g^2(mT + \epsilon T)] \cong \sum_m (g'(mT)T)^2\ \mathrm{Var}[\epsilon] \qquad \text{(III.1.2–3)}$$

For a given timing error variance, the variance of the ISI term decreases with increasing excess bandwidth. The third term in (III.1.2–1) is a zero–mean Gaussian noise term with variance $N_o/(2E_b)$. The resulting ratio $\mathrm{SNR}/\mathrm{SNR}_o$ is given by:

$$\frac{\mathrm{SNR}}{\mathrm{SNR}_o} = \frac{1 - (-g''(0)T^2)\ \mathrm{Var}[\epsilon]}{1 + \frac{2E_b}{N_o}\sum_m (g'(mT)T)^2\ \mathrm{Var}[\epsilon]} \qquad \text{(III.1.2–4)}$$

The numerator in (III.1.2–4) is caused by a reduction of the mean of the decision variable, whereas the denominator is due to the ISI. The corresponding BER degradation is given by:

$$\text{BER degrad. [dB]} = -10 \log \left[\frac{\mathrm{SNR}}{\mathrm{SNR}_o}\right]$$

$$\cong 4.3 \left[(-g''(0)T^2) + \frac{2E_b}{N_o}\sum_m (g'(mT)T)^2\right] \mathrm{Var}[\epsilon]$$

(III.1.2–5)

For a given value of the timing error variance Var[ϵ], the following conclusions can be drawn.

– The effect of ISI increases with increasing E_b/N_o.

– The value of E_b/N_o, above which the effect of ISI is more important than the reduction of the mean of the decision variable, increases with increasing rolloff, and is given by :

$$\frac{E_b}{N_o} = \frac{1}{2} \frac{-g''(0)T^2}{\sum_m (g'(mT)T)^2} \qquad \text{(III.1.2–6)}$$

When g(t) is a cosine roloff pulse with rolloff factor α, the right–hand side of (III.1.2–6) ranges from 0.5 (or –3 dB) to 11.48 (or 10.6 dB) when α increases from 0 to 1.

When evaluating the variance of the decision variable, we have taken into account only the additive noise and the ISI. Strictly speaking, also the fluctuation of the useful term g(ϵT) contributes to the variance. This contribution could be taken into account by adding to the second line of (III.1.2–4) the term :

$$4.3 \frac{2E_b}{N_o} \mathrm{Var}[g(\epsilon T)] \cong 4.3 \frac{2E_b}{N_o} \mathrm{Var}[-g''(0)\epsilon^2 T^2/2]$$

As Var[ϵ^2] is in the same order of magnitude as the square of Var[ϵ], it is usually so small that it can be neglected with respect to the terms that are linear in Var[ϵ].

III.1.3 Both carrier phase errors and timing errors

When both carrier phase errors and timing errors are present, a similar reasoning as in sections III.1.1 and III.1.2 yields

$$\frac{SNR}{SNR_o} = \frac{1 - Var[\phi] - (-g''(0)T^2)\,Var[\epsilon]}{1 + \frac{2E_b}{N_o} A_M\, Var[\phi] + \frac{2E_b}{N_o} \sum_m (g'(mT)T)^2\, Var[\epsilon]} \tag{III.1.3–1}$$

where A_M depends on the modulation format :

$$A_M = \begin{cases} 0 & \text{BPSK} \\ 1 & \text{QPSK} \\ \sum_m g^2(mT - \frac{T}{2}) & \text{OQPSK} \end{cases}$$

The corresponding BER degradation is given by

$$\begin{aligned} \text{BER degrad. [dB]} &= -10 \log \left[\frac{SNR}{SNR_o}\right] \\ &\cong 4.3 \left(1 + \frac{2E_b}{N_o} A_M\right) Var[\phi] \\ &\quad + 4.3 \left((-g''(0)T^2) + \frac{2E_b}{N_o} \sum_m (g'(mT)T)^2\right) Var[\epsilon] \end{aligned} \tag{III.1.3–2}$$

The approximation in the second line of (III.1.3–2) is valid when the BER degradation is small; in this case, the BER degradation is the sum of the degradations caused by the carrier phase errors and the timing errors individually.

III.2 VALIDITY OF THE GAUSSIAN APPROXIMATION

The degradations of the signal–to–noise ratio at the input of the decision device, that have been determined in section III.1, give rise to a BER degradation given by

$$\text{BER degrad. [dB]} = 10 \log \left[\frac{\text{SNR}_o}{\text{SNR}} \right] \qquad \text{(III.2–1)}$$

only when the distribution of the decision variable is Gaussian.

– In the absence of tracking errors, the decision variable is Gaussian with mean equal to 1 and variance equal to $N_o/(2E_b)$, which is caused by the additive noise.

– In the case of BPSK, carrier phase errors mainly reduce the mean of the decision variable, so that the decision variable remains essentially Gaussian.

– In all other cases, tracking errors not only reduce the mean of the decision variable, but also introduce a zero–mean disturbance (quadrature interference in the presence of carrier phase errors with (O)QPSK, ISI in the presence of timing errors with BPSK and (O)QPSK), which adds to the Gaussian noise at the input of the decision device. As this additional disturbance is non–Gaussian, the decision variable is also non–Gaussian. However, when the tracking error variance is sufficiently small with respect to the variance $N_o/(2E_b)$ of the Gaussian noise component at the input of the decision device, the quadrature interference or the ISI have only a minor effect on the shape of the distribution of the decision variable, so that (III.2–1) is still a good approximation.

In order to investigate the accuracy of (III.2–1) in the presence of tracking errors, we have considered the case of carrier phase errors for QPSK. The corresponding BER is given by (see III.1.1–5)

$$\begin{aligned} \text{BER} &= \tfrac{1}{2}\,\text{Prob}[\cos\phi + \sin\phi < N_{Ik}] + \tfrac{1}{2}\,\text{Prob}[\cos\phi - \sin\phi < N_{Ik}] \\ &= E\left[\tfrac{1}{2} Q\left[\frac{\cos\phi + \sin\phi}{\sigma_n} \right] + \tfrac{1}{2} Q\left[\frac{\cos\phi - \sin\phi}{\sigma_n} \right] \right] \end{aligned} \qquad \text{(III.2–2)}$$

where E[.] denotes averaging with respect to the carrier phase error ϕ, $\sigma_n^2 = N_o/(2E_b)$, and Q(.) is the area under the tail of the univariate Gaussian distribution :

$$Q(x) = \int_x^{\infty} \frac{1}{\sqrt{2\pi}} \exp\left[-\frac{u^2}{2}\right] du \qquad \text{(III.2–3)}$$

We have evaluated (III.2–2) by means of numerical integration, in the case of a carrier phase error with a Gaussian distribution (this is not the same as a Gaussian decision variable !) with zero mean and variance Var[ϕ], and a Tikhonov distribution, respectively. The Tikhonov distribution is given by

$$p(\phi) = C \exp\left[\frac{\cos(4\phi)}{16\ \mathrm{Var}[\phi]}\right] \qquad -\pi/4 \le \phi \le \pi/4 \qquad \text{(III.2–4)}$$

and corresponds to the phase error distribution in a PLL with a sinusoidal phase error detector characteristic (period of $\pi/2$) and white loop noise; the quantity Var[ϕ] in (III.2.4) is the phase error variance resulting from the <u>linearized</u> PLL equations, and C is the normalization constant [Lin'73]. In the case of a Gaussian carrier phase error, we have also performed some computer simulations. For all distributions, we select Var[ϕ] such that SNR/SNR_0 is constant, irrespective of the value of E_b/N_0; according to (III.1.1–7), the selected Var[ϕ] is given by

$$\mathrm{Var}[\phi] = \frac{1 - \dfrac{SNR}{SNR_0}}{1 + \dfrac{2E_b}{N_0}\dfrac{SNR}{SNR_0}} \qquad \text{(III.2–5)}$$

For a fixed value of SNR/SNR_0, the above phase error variance yields a constant BER degradation, irrespective of E_b/N_0, under the assumption of a Gaussian decision variable; the corresponding BER is given by

$$BER = Q\left[((2E_b/N_0)\times(SNR/SNR_0))^{1/2}\right] \qquad \text{(III.2–6)}$$

and is obtained by shifting to the right the BER curve in the absence of tracking errors by an amount equal to 10 log (SNR_0/SNR) [dB]. We have obtained the following results.

– In the case of SNR_0/SNR = 0.1 dB, the BER degradation for a Gaussian decision variable is 0.1 dB. For a Gaussian carrier phase error and a Tikhonov carrier phase error, the asymptotic (large E_b/N_0) BER degradations are about 0.102 dB. Hence, for BER degradations in the order of 0.1 dB or less (which is the case of practical interest in a satellite communication environment, as far as

degradations due to tracking errors are concerned), the actual BER degradation in dB is very closely approximated by the ratio SNR_0/SNR, expressed in dB. Also, the BER degradation is nearly independent of the specific shape of the phase error distribution.

– In the case of $SNR_0/SNR = 1$ dB, the BER degradation for a Gaussian decision variable is 1 dB. For a Gaussian carrier phase error and a Tikhonov carrier phase error, the asymptotic BER degradations are about 1.2 dB and 1.3 dB, respectively (see Fig. III.2–1). This indicates that for BER degradations in the order of 1 dB (which is much too large for satellite applications, as far as degradations due to tracking errors are concerned), the actual BER degradation in dB can still be estimated by taking SNR_0/SNR in dB, but the estimate is not very accurate. Also, the specific shape of the phase error distribution has some influence on the BER degradation.

– In the case of $SNR_0/SNR = 3$ dB, the BER degradation for a Gaussian decision variable is 3 dB. For a Gaussian carrier phase error and a Tikhonov carrier phase error, the asymptotic BER degradations are about 3.9 dB and 6.0 dB, respectively (see Fig. III.2–1). These results show that for BER degradations of several dB, the actual BER degradation is largely dependent on the specific shape of the phase error distribution; therefore, approximating the BER degradation in dB by the ratio SNR_0/SNR in dB can give rise to very large errors.

It should be noted that $SNR_0/SNR = 1$ dB and $SNR_0/SNR = 3$ dB have been introduced only for investigating to which extent the assumption of a Gaussian decision variable is still valid. For QPSK, the above values of 3 dB and even 1 dB are much too large for practical applications. Indeed, assuming that the tracking error variance is approximated by the Cramer–Rao bound, i.e.

$$\mathrm{Var}[\phi] \cong 2B_L T \frac{N_0}{4E_b} \qquad \text{(III.2–3)}$$

$SNR_0/SNR = 1$ dB and $SNR_0/SNR = 3$ dB correspond to $2B_L T \cong 0.5$ and $2B_L T \cong 2$, respectively, whereas more realistic values of $2B_L T$ are in the order of 0.01. For $E_b/N_0 = 6$ dB, $2B_L T \cong 0.5$ and $2B_L T \cong 2$ correspond to (linearized) r.m.s. phase errors of approximately 10 degrees and 20 degrees, respectively. Taking into account that for QPSK the spacing between stable and unstable equilibrium points equals 45 degrees, such large r.m.s. phase errors cannot be tolerated, because the synchronizer will exhibit strong nonlinear behaviour (cycle slips).

We expect that for carrier phase errors with OQPSK, and for timing errors with BPSK and (O)QPSK, the assumption of a Gaussian decision variable yields a similar accuracy as for QPSK; as mentioned before, for carrier phase errors with BPSK, the decision variable is essentially Gaussian, so that (III.2–1) is a very accurate approximation. Therefore, we conclude that BER degradations in the order of 0.5 dB or less are well approximated by the ratio SNR_0/SNR, expressed in dB; this is the range of BER degradations, due to synchronization errors, that is of interest in satellite applications.

III.3 DISCUSSION OF THE EXPRESSION FOR THE BER DEGRADATION

For small tracking error variances the BER degradation in dB, caused by carrier phase errors only or timing errors only, has been expressed as the product of two factors (see second line of (III.1.1–4), (III.1.1–8), (III.1.1–12) and (III.1.2–5)). The first factor takes the general form $C_1 + C_2E_b/N_o$. The second factor is the tracking (phase or timing) error variance; as explained in II.1.2, its general form is $C_3 + C_4N_o/E_b + F(N_o/E_b)$, where F(x) increases faster then linearly with x. The behaviour of the tracking error variance and the factor $C_1 + C_2E_b/N_o$ as a function of E_b/N_o is shown in Fig. III.3–1; all axes are scaled logarithmically, and the displayed functions are approximated by straight–line segments.

For the tracking error variance, two important values of E_b/N_o are $(E_b/N_o)_A$ and $(E_b/N_o)_B$, with $C_4 \times (N_o/E_b)_A = F((N_o/E_b)_A)$, and $(E_b/N_o)_B = C_4/C_3$. At $E_b/N_o = (E_b/N_o)_A$, the contributions of the terms $F(N_o/E_b)$ and C_4N_o/E_b are the same; at $E_b/N_o = (E_b/N_o)_B$, the contributions of the terms C_3 and C_4N_o/E_b are the same. The straight–line approximation in Fig. III.3–1 yields a maximum approximation error of about 0.3 decade, which occurs at $(E_b/N_o)_A$ and $(E_b/N_o)_B$.

– In the region $E_b/N_o < (E_b/N_o)_A$, the variance is dominated by the nonlinear term $F(N_o/E_b)$.

– In the region $(E_b/N_o)_A < E_b/N_o < (E_b/N_o)_B$, the variance is approximately inversely proportional with E_b/N_o; the term C_4N_o/E_b dominates.

– In the region $(E_b/N_o)_B < E_b/N_o$, the variance is approximately equal to the constant C_3; self–noise dominates.

For the factor $C_1 + C_2E_b/N_o$, an important value of E_b/N_o is $(E_b/N_o)_F = C_1/C_2$. At $E_b/N_o = (E_b/N_o)_F$, the contributions of the terms C_1 and C_2E_b/N_o are the same. The straight–line approximation in Fig. III.3–1 yields a maximum approximation error of 0.3 decade, which occurs at $(E_b/N_o)_F$.

– In the region $E_b/N_o < (E_b/N_o)_F$, the factor is approximately equal to the constant C_1; the BER degradation is caused mainly by the reduction of the mean of the decision variable.

– In the region $E_b/N_o > (E_b/N_o)_F$, the factor is dominated by the term C_2E_b/N_o, and approximately proportional to E_b/N_o; the BER degradation is caused mainly by the increase of the variance of the decision variable (quadrature interference, ISI).

Three cases can be distinguished with respect to the order of $(E_b/N_o)_F$, $(E_b/N_o)_A$ and $(E_b/N_o)_B$:

<u>Case 1</u> : $(E_b/N_o)_F < (E_b/N_o)_A$

In the region $E_b/N_o < (E_b/N_o)_A$ the BER degradation decreases with increasing E_b/N_o :

$$\text{BER degrad. [dB]} \cong F(N_o/E_b)\,(C_1 + C_2 E_b/N_o)$$

In the region $(E_b/N_o)_A < E_b/N_o < (E_b/N_o)_B$, the BER degradation is approximately constant :

$$\text{BER degrad. [dB]} \cong C_2 C_4$$

In the region $(E_b/N_o)_B < E_b/N_o$, the BER degradation increases approximately linearly with E_b/N_o :

$$\text{BER degrad. [dB]} \cong C_2 C_3 E_b/N_o$$

<u>Case 2</u> : $(E_b/N_o)_A < (E_b/N_o)_F < (E_b/N_o)_B$

In the region $E_b/N_o < (E_b/N_o)_F$, the BER degradation decreases with increasing E_b/N_o :

$$\text{BER degrad. [dB]} \cong C_1\,(F(N_o/E_b) + C_4 N_o/E_b)$$

In the region $(E_b/N_o)_F < E_b/N_o < (E_b/N_o)_B$, the BER degradation is approximately constant :

$$\text{BER degrad. [dB]} \cong C_2 C_4$$

In the region $(E_b/N_o)_B < E_b/N_o$, the BER degradation increases approximately linearly with E_b/N_o :

$$\text{BER degrad. [dB]} \cong C_2 C_3 E_b/N_o$$

<u>Case 3</u> : $(E_b/N_o)_B < (E_b/N_o)_F$

In the region $E_b/N_o < (E_b/N_o)_B$ the BER degradation decreases with increasing E_b/N_o :

$$\text{BER degrad. [dB]} \cong C_1 \left(F(N_o/E_b) + C_4 N_o/E_b\right)$$

In the region $(E_b/N_o)_B < E_b/N_o < (E_b/N_o)_F$, the BER degradation is approximately constant :

$$\text{BER degrad. [dB]} \cong C_1 C_3$$

In the region $(E_b/N_o)_F < E_b/N_o$, the BER degradation increases approximately linearly with E_b/N_o :

$$\text{BER degrad. [dB]} \cong C_2 C_3 E_b/N_o$$

For all three considered cases, a similar behaviour of the BER degradation is observed :

– For large E_b/N_o, the BER degradation is approximately linearly increasing with E_b/N_o. In this region, the tracking error variance is dominated by the self–noise and the BER degradation is mainly caused by quadrature interference or ISI (all three considered cases).

– For intermediate E_b/N_o, the BER degradation is approximately constant. In this region, either the tracking error variance is essentially inversely proportional to E_b/N_o and the BER degradation is mainly caused by quadrature interference or ISI (as in cases 1 and 2), or the tracking error variance is dominated by the self–noise and the BER degradation is mainly caused by the reduction of the mean of the decision variable (as in case 3).

– For small E_b/N_o, the BER degradation decreases with increasing E_b/N_o. In this region, either the tracking error variance is dominated by the nonlinear term $F(N_o/E_b)$ (all three cases), or the tracking error variance is essentially inversely proportional to E_b/N_o and the BER degradation is mainly caused by the reduction of the mean of the decision variable (as in cases 2 and 3).

The values $(E_b/N_o)_F$ are shown in Table III.3–1. In most cases, $(E_b/N_o)_F$ is far below the operating E_b/N_o for uncoded transmission, so that the BER degradation is determined mainly by the quadrature interference and the ISI. Exceptions occur for carrier phase errors with BPSK (no quadrature interference), and for timing errors when using a large excess bandwidth (very small ISI). The values $(E_b/N_o)_A$ for the various synchronizers can be estimated from the curves of $S(fT=0)/(\text{slope})^2$ given in Part II :

$$(E_b/N_o)_A\ [\mathrm{dB}] = \begin{cases} (E_s/N_o)_A\ [\mathrm{dB}] & \text{BPSK} \\ (E_s/N_o)_A\ [\mathrm{dB}] - 3\ \mathrm{dB} & \text{(O)QPSK} \end{cases}$$

where $(E_s/N_o)_A$ is the value of E_s/N_o for which $S(fT=0)/(\mathrm{slope})^2$ is a factor of two (or 3 dB) larger than the dominating term at large E_s/N_o, which is inversely proportional to E_s/N_o. We recall that $(E_s/N_o)_A$ does not depend on the synchronizer bandwidth. For all synchronizers, it turns out that $(E_b/N_o)_A < 3$ dB, which is far below the operating E_b/N_o for uncoded transmission. The values $(E_b/N_o)_B$ for the various synchronizers can be derived from Tables II.2–1 and II.3–1 :

$$(E_b/N_o)_B\ [\mathrm{dB}] = \begin{cases} (E_s/N_o)_c\ [\mathrm{dB}] & \text{BPSK} \\ (E_s/N_o)_c\ [\mathrm{dB}] - 3\ \mathrm{dB} & \text{(O)QPSK} \end{cases}$$

We recall that $(E_b/N_o)_B$ is inversely proportional to the synchronizer bandwidth; the self–noise contribution to the tracking error variance can be made much smaller than the additive noise contribution, by reducing the synchronizer bandwidth.

	$(E_b/N_o)_F$ [dB]
<u>Carrier phase errors</u>	
– BPSK	∞
– QPSK	–3.00
– OQPSK	
$\alpha = 0$	–3.00
$\alpha = 0.5$	–1.76
$\alpha = 1.0$	0.00
<u>Timing errors</u> (BPSK, (O)QPSK)	
$\alpha = 0$	–3.00
$\alpha = 0.5$	1.63
$\alpha = 1.0$	10.60

Table III.3–1 : $(E_b/N_o)_F$ for various rolloff factors α.

For a given rolloff α and a given synchronization algorithm, the BER degradation can be reduced by decreasing the synchronizer bandwidth; in practice, the bandwidth should not be too small in order to limit the influence of phase noise. In II.1.2, we have pointed out that the additive noise contribution to the tracking error variance is proportional to the synchronizer bandwidth, whereas the self–noise contribution is proportional to the square of the synchronizer bandwidth. Hence, when decreasing the synchronizer bandwidth by some factor, the BER degradation is reduced by approximately the same factor, in the range where the additive noise contribution to the tracking error variance dominates (i.e. $E_b/N_o < (E_b/N_o)_B$), and by approximately the square of this factor, in the range where self–noise dominates (i.e. $E_b/N_o > (E_b/N_o)_B$). Figure III.3–2 shows the two possible cases, corresponding to $(E_b/N_o)_B > (E_b/N_o)_F$ and $(E_b/N_o)_B < (E_b/N_o)_F$, respectively. We recall that the above results follow from the linearization $\log(1+x) \cong 0.43\ x$. For BER degradations larger than 0.5 dB, this linearization is not very accurate. In the case where the BER degradation is caused mainly by ISI or quadrature interference (i.e. $E_b/N_o > (E_b/N_o)_B$), the numerator in the expressions (III.1.1–7), (III.1.1–11) and (III.1.2–4) can be approximated by 1; this corresponds to neglecting the reduction of the useful signal at the input of the decision device. As $\log(1+x) \leq 0.43\ x$ for $x \geq 0$, the above linearization yields an overestimation of $10\ \log(SNR_o/SNR)$; the approximation error increases with increasing x. In the case where the BER degradation is caused mainly by the reduction of the useful signal at the input of the decision device (i.e. $E_b/N_o < (E_b/N_o)_B$), the denominator in the expressions (III.1.1–7), (III.1.1–11) and (III.1.2–4) can be approximated by 1; this corresponds to neglecting the effect of ISI and quadrature interference. As $-\log(1-x) \geq 0.43\ x$ for $0 \leq x < 1$, the above linearization yields an underestimation of $10\ \log(SNR_o/SNR)$; the approximation error increases with increasing $|x|$. Consequently, when decreasing the synchronizer bandwidth by some factor, the quantity $10\ \log(SNR_o/SNR)$ is reduced by a slightly larger factor when E_b/N_o is small enough (so that the tracking error variance is dominated by the additive noise contribution, and the BER degradation is caused mainly by the reduction of the useful signal at the input of the decision device), and by slightly less than the square of this factor when E_b/N_o is large enough (so that the tracking error variance is dominated by the self–noise contribution, and the BER degradation is caused mainly by the ISI or quadrature interference).

III.4 NUMERICAL RESULTS

For various carrier and symbol synchronizers, we have evaluated the degradation SNR_o/SNR [dB] of the signal–to–noise–ratio at the input of the decision device. For a few cases, computer simulations for the actual BER degradation are presented as well; these are discussed at the end of this section. Unless noted otherwise, the results are obtained for $2B_LT = 0.01$ and $K_F = 2$ (first–order loop (FB synchronizer) or single–pole averaging filter (FF synchronizer)).

Carrier synchronization

We have evaluated the degradation SNR_o/SNR [dB] for the DDML and NDAML synchronizers for BPSK and (O)QPSK. As indicated in Table T.7.1, many other carrier synchronizers yield a tracking performance which is similar to that of either the DDML or the NDAML synchronizer; consequently, their degradation SNR_o/SNR is similar to that of the DDML or NDAML synchronizer.

– BPSK (see Fig. III.4–1)

SNR_o/SNR [dB] is approximately inversely proportional to E_b/N_o, because there is no quadrature interference and the considered synchronizers are free of self–noise. The DDML and NDAML synchronizers yield a similar degradation, because they have a similar tracking performance (see Table T.7.1); the resulting degradation is very small. The degradation SNR_o/SNR for the V&V and NDAML$\bar{T}$ synchronizer (not shown) are very similar to the results for the DDML and NDAML synchronizers, because all four synchronizers have nearly the same tracking performance for moderate and large E_b/N_o.

– QPSK (see Fig. III.4–2)

SNR_o/SNR [dB] approaches a constant value when E_b/N_o increases. This is because the degradation is mainly caused by quadrature interference, and the synchronizers are free of self–noise. The DDML and NDAML synchronizers yield a similar degradation, because they have a similar tracking performance (see Table T.7.1); the resulting degradation is small. The degradation SNR_o/SNR for the V&V synchronizer (not shown) is very similar to the results for the DDML and NDAML synchronizers, because all three synchronizers have nearly the same tracking performance for moderate and large E_b/N_o.

– OQPSK (see Figs. III.4–3 to III.4–6)

The degradation is mainly caused by the quadrature interference; the synchronizers exhibit self–noise. When the self–noise contribution to the tracking error variance is much smaller than the additive noise contribution, SNR_0/SNR [dB] is approximately constant for $E_b/N_0 > 3$ dB; when self–noise dominates, SNR_0/SNR [dB] is approximately proportional to E_b/N_0. The NDAML synchronizer yields a larger degradation than the DDML synchronizer, because of its larger tracking error variance (see Table T.7.1); the difference between the tracking error variances resulting from both synchronizers increases with decreasing rolloff. We have not evaluated SNR_0/SNR for the NDASQFFTO and V&V synchronizers. The NDASQFFTO synchronizer yields the same additive noise contribution to the tracking error variance as the NDAML synchronizer; therefore, the NDASQFFTO and the NDAML synchronizer yield essentially the same SNR_0/SNR for small and moderate E_b/N_0, where self–noise has only a very small influence. The V&V synchronizer yields a slightly smaller tracking error variance as compared to the NDAML synchronizer for $\alpha = 0.5$ and $\alpha = 1$; for $\alpha = 0.1$, the tracking error variance resulting from the V&V synchronizer is about one order of magnitude smaller as compared to the NDAML synchronizer. Hence, for $\alpha = 1$ and $\alpha = 0.5$, the degradation $10 \log(SNR_0/SNR)$ for the V&V synchronizer is slightly smaller than for the NDAML synchronizer, whereas at $\alpha = 0.1$ the degradation $10 \log(SNR_0/SNR)$ for the V&V synchronizer is about one order of magnitude smaller than for the NDAML synchronizer.

For $\alpha = 1$, the effect of self–noise is small; the degradation is approximately constant, and small (see Fig. III.4–3).

For $\alpha = 0.5$, the degradation caused by the DDML synchronizer is small, whereas the NDAML synchronizer yields a much larger degradation (see Fig. III.4–4). Also, the effect of self–noise is larger than for $\alpha = 1$.

For $\alpha = 0.1$, the DDML synchronizer yields a small degradation, and the effect of self–noise is still modest (see Fig. III.4–5). The degradation caused by the NDAML synchronizer is very large, and mainly due to self–noise (see Fig. III.4–6); decreasing the synchronizer bandwidth reduces the degradation, especially in the range of E_b/N_0 where self–noise dominates the tracking error variance. Reducing the bandwidth of the NDAML synchronizer from $2B_LT = 0.01$ to $2B_LT = 0.003$ reduces $10 \log(SNR_0/SNR)$ at large E_b/N_0 by a factor of (somewhat smaller than) 11.11 (or about one decade), corresponding to the square of the bandwidth ratio.

Symbol synchronization

According to Table T.7.2, one can distinguish groups of synchronizers, with all synchronizers within one group having nearly the same additive noise contribution to the tracking error variance. The synchronizers, for which the degradation SNR_0/SNR [dB] has been evaluated, can be grouped as follows :

– Group 1 : DDML, DDEL(λ=1/2), DTTL

– Group 2 : NDAML, NDAEL(λ=1/2), GA

– Group 3 : DDEL(λ=1), M&M, mM&M

Group 1 contains DD symbol synchronizers using two samples per symbol, the symbol synchronizers from group 2 are of the NDA type, operating at two samples per symbol, and group 3 contains DD symbol synchronizers operating at one sample per symbol. In the region of E_b/N_0 where the additive noise contribution to the tracking error variance is much larger than the self–noise contribution, synchronizers belonging to the same group yield nearly the same degradation; on the other hand, for large E_b/N_0 where self–noise dominates, synchronizers belonging to the same group can give rise to substantially different degradations, because of different self–noise contributions.

We have evaluated the degradation SNR_0/SNR [dB] for the synchronizers mentioned above, assuming BPSK transmission. For (O)QPSK, the degradations are about half as large as for BPSK, because for a given E_b/N_0, the tracking error variances are half as large as for BPSK.

– $\alpha = 1$ (see Figs. III.4–7 and III.4–8)

The synchronizers from groups 1 and 2 have approximately the same additive noise contribution to the tracking error variance, and their self–noise contributions are small. Also, the BER degradation is caused mainly by the reduction of the mean of the decision variable, and not by ISI. As a consequence, SNR_0/SNR [dB] increases with E_b/N_0 only for large E_b/N_0 (see Fig. III.4–7); the resulting degradation is very small.

The synchronizers from group 3 have a much larger tracking error variance; also, the DDEL(λ=1) and the mM&M synchronizers yield a considerably larger self–noise contribution than the synchronizers from groups 1 and 2. As a consequence, the corresponding BER degradation is much larger (see Fig. III.4–8; note the change of scale as compared to Fig. III.4–7). Decreasing the synchronizer bandwidth reduces the degradation, especially in the range of E_b/N_0 where self–noise dominates the tracking error variance. For the M&M synchronizer, which is free of self–noise, reducing the synchronizer bandwidth

from $2B_LT = 0.01$ to $2B_LT = 0.003$ reduces $10 \log(SNR_o/SNR)$ by a factor of (slightly more than) 3.33 (or about half a decade), corresponding to the bandwidth ratio.

– $\alpha = 0.5$ (see Fig. III.4–9)

The degradation is caused mainly by the ISI. In the region where the self–noise contribution to the tracking error variance is much smaller than the additive noise contribution, and $E_b/N_o > 3$ dB, SNR_o/SNR [dB] is approximately constant. When self–noise dominates, SNR_o/SNR [dB] is approximately proportional to E_b/N_o. The resulting degradations are small.

– $\alpha = 0.1$ (see Figs. III.4–10 and III.4–11)

The degradation resulting from the synchronizers from groups 1 and 3 is small; the influence of self–noise is larger than for $\alpha = 0.5$. As for $\alpha = 0.5$, the degradation is caused mainly by the ISI. In the region where the self–noise contribution to the tracking error variance is much smaller than the additive noise contribution, and $E_b/N_o > 3$ dB, SNR_o/SNR [dB] is approximately constant. When self–noise dominates, SNR_o/SNR [dB] is approximately proportional to E_b/N_o.

The degradation resulting from the synchronizers from group 2 is very large, because of the very large tracking error variance, which is mainly caused by self–noise (see Fig. III.4–11). Decreasing the synchronizer bandwidth reduces the degradation, especially in the range of E_b/N_o where self–noise dominates the tracking error variance. Reducing the bandwidth of these synchronizers from $2B_LT = 0.01$ to $2B_LT = 0.003$ reduces $10 \log(SNR_o/SNR)$ at large E_b/N_o by a factor of (somewhat smaller than) 11.11 (or about one decade), corresponding to the square of the bandwidth ratio.

We have not evaluated SNR_o/SNR for the NDASQFF, NDAML$\bar{C}$B/Q, NDASQFF$\bar{C}$B/Q, NDASQFF$\bar{C}$O, NDAEL(λ=1/2)$\bar{C}$B/Q, GA$\bar{C}$B/Q and NDANLFB$\bar{C}$B/Q synchronizers.

– For moderate and large E_b/N_o, the NDANLFB$\bar{C}$B synchronizer has nearly the same tracking error variance as the M&M synchronizer for BPSK and, therefore, about the same degradation SNR_o/SNR [dB]. For moderate and large E_b/N_o, the NDANLFB$\bar{C}$Q synchronizer yields nearly twice the tracking error variance of the M&M synchronizer for QPSK and, therefore, about twice the degradation SNR_o/SNR [dB].

– The additive noise contribution to the tracking error variance of the NDASQFF synchronizer for BPSK and (O)QPSK is the same as for the corresponding NDAML synchronizer. Consequently, these synchronizers yield the same degradation SNR_o/SNR [dB], for small and moderate E_b/N_o, at which the influence of self–noise is small.

– The NDASQFF$\bar{C}$O synchronizer yields the same tracking error variance as the NDASQFF synchronizer for OQPSK. Hence, these two synchronizers yield the same degradation SNR_o/SNR [dB], which, for small and moderate E_b/N_o at which the influence of self–noise is small, is the same as for the NDAML synchronizer for OQPSK.

– The NDAML, NDASQFF, GA and NDAEL(λ=1/2) synchronizers for OQPSK are mathematically equivalent with the synchronizers having the corresponding name plus the suffix $\bar{C}$Q (e.g. NDAML and NDAML$\bar{C}$Q are equivalent); evidently, equivalent synchronizers yield the same ratio SNR_o/SNR.

– For moderate and large E_b/N_o, where the influence of noise x noise terms is small, synchronizers with a name containing the suffix $\bar{C}$B yield nearly the same tracking error variance as the synchronizers for BPSK with the corresponding name without that suffix (e.g NDAML$\bar{C}$B and NDAML); in this region of E_b/N_o, the corresponding synchronizers yield nearly the same SNR_o/SNR.

Computer simulations

Numerical values of the actual BER degradation are obtained from a Monte Carlo simulation of the entire receiver, consisting of the decision device and the (carrier or symbol) synchronizer under investigation. As the simulation of very small BER degradations requires excessive CPU time, we have restricted our attention to a few cases where SNR_0/SNR is between 0.1 dB and 1 dB. In addition, the analytical result (III.1.1–8) for the BER degradation, in the case of V&V (with $F(u) = u^2$) carrier synchronization for QPSK, has been compared with simulation results from [Har'91]; excellent agreement has been found, even for BER degradations as large as 0.8 dB.

Our simulation results can be summarized as follows.

– In the case of NDAML carrier synchronization for OQPSK, the simulated BER degradation closely matches the analytically obtained degradation SNR_0/SNR, both for $\alpha = 0.5$ and $\alpha = 0.1$ (see Figs. III.4–4 and III.4–6).

– In the case of M&M symbol synchronization with $\alpha = 1$ (see Fig. III.4–8), the simulated BER degradation at $E_b/N_0 = 0$ dB largely exceeds the analytically obtained degradation SNR_0/SNR; this is probably due to the large cycle slip rate (see Part VI, Fig. VI.3–10). At $E_b/N_0 = 5$ dB, the simulation result indicates a degradation which is about twice as large as the analytical result; a much better agreement is found at $E_b/N_0 = 10$ dB.

– In both cases of NDAEL(λ=1/2) and GA symbol synchronization with $\alpha = 0.1$, the simulated BER degradation exceeds the analytically obtained degradation SNR_0/SNR by a factor ranging from 2.5 to 4 (see Fig. III.4–11).

A possible explanation of the difference between analytical and simulation results is the following.

– The analytical result is based upon the linearized tracking error variance : the phase or timing error detector characteristic is approximated by a straight line $K_{\theta\theta}\phi$ or $K_{\tau\tau}\epsilon$, tangent to the characteristic at the tracking point, and the loop noise power spectral density $S(fT;\phi)$ or $S(fT;\epsilon)$, corresponding to a given tracking error ϕ or ϵ, is replaced by the spectral density $S(fT;0)$ at the tracking point. For small loop bandwidths, this results in a nearly Gaussian distribution of the tracking error, with a variance equal to the linearized variance. Note that BER degradations for a Gaussian tracking error have been reported in Fig. III.2–1; these BER degradations are slightly larger than SNR_0/SNR.

– For a sinusoidal phase or timing error detector characteristic, and a loop noise power spectral density which depends neither on frequency (white loop noise) nor on the tracking error, the tracking error has a Tikhonov distribution. For large tracking errors the actual (sinusoidal) restoring force is smaller than for the linearized characteristic. Consequently, for large tracking errors the actual (Tikhonov) distribution is larger than the Gaussian distribution. As a result, the Tikhonov distribution yields a larger BER degradation than the Gaussian distribution; this is confirmed by the results from Fig. III.2–1.
– For all investigated synchronizers, the loop noise power spectral density $S(fT;\phi)$ or $S(fT;\epsilon)$ is minimum for $fT=0$ and $\phi=\epsilon=0$. When the phase or timing error detector is sinusoidal, a Tikhonov distribution (which takes into account only $S(0;0)$) underestimates the tails of the actual distribution, because for large tracking errors, the actual loop noise power spectral density is larger than $S(0;0)$. Consequently, the actual BER degradation is larger than for a Tikhonov distribution.
The sharper the minimum of the loop noise power spectral density (self noise can give rise to sharp minima !) and the larger the synchronizer bandwidth, the more the actual BER degradation is expected to exceed SNR_0/SNR. Fortunately, the simulations show that in the range of 0.1 dB to 1 dB, the actual BER degradation has the same order of magnitude as SNR_0/SNR; for smaller degradations, a much better agreement is expected.

In order to illustrate that the linearized analysis may yield too optimistic results, we have considered the NDAML carrier synchronizer for OQPSK with very small rolloff ($\alpha = 0.1$). At $2B_LT = 0.01$ and $E_b/N_0 = 2$ dB, a closed–loop simulation of this carrier synchronizer gave rise to a tracking error variance which is about 20 times as large as the linearized variance; at $2B_LT = 0.001$ and $E_b/N_0 = 2$ dB, a good match between simulation and analysis was observed. This example clearly shows that $2B_LT = 0.01$ is too large for the considered synchronizer to operate in the linearized tracking mode. A similar behaviour is expected for the quadratic symbol synchronizers (such as NDAML, NDAEL and GA), because their timing error detector characteristic and loop noise power spectral density have the same shape as for the NDAML carrier synchronizer for OQPSK; this could explain the deviation between simulation and analysis that is found in Fig. III.4–11.

III.5 CONCLUSIONS

In a satellite communication environment, one wishes to limit the BER degradation caused by synchronization errors to an order of magnitude of 0.1 dB or less. Within this range of degradations, the statistics of the decision variable are nearly Gaussian, so that the BER degradation in dB is well approximated by the loss SNR_0/SNR, measured in dB, at the input of the decision device. This loss is essentially proportional to the tracking error variance, so that the specific shape of the distribution of the tracking error is not important. Consequently, synchronizers with a similar tracking error variance yield a similar BER degradation.

In the case of carrier phase errors, the BER degradation for (O)QPSK is mainly caused by the quadrature interference, which adds to the thermal noise. For BPSK, quadrature interference is absent; the BER degradation is caused solely by a reduction of the useful signal at the input of the decision device.

In the case of timing errors, the BER degradation for small and moderate excess bandwidths is mainly caused by the intersymbol interference, which adds to the thermal noise; for large excess bandwidths, the BER degradation is mainly caused by a reduction of the useful signal component at the input of the decision device. For the same E_b/N_0, the BER degradation for (O)QPSK is about half the degradation for BPSK, because the tracking error variance is half as large as for BPSK.

For most carrier and symbol synchronizers, it turns out that a synchronizer bandwidth of $2B_LT = 0.01$ gives rise to a BER degradation in the order of 0.1 dB at most. Exceptions are the quadratic NDAML carrier synchronizer for OQPSK and all quadratic symbol synchronizers, when operating at very small excess bandwidths : because of the large self–noise contribution to the tracking error variance, the resulting BER degradation is very large; in these cases it is advisable to use $2B_LT = 0.001$ for $\alpha = 0.1$.

III.A APPENDIX

When $a_k=1$, the sample at the input of the decision device can be written as $1 - \Delta + N$, where N is a Gaussian random variable with variance σ^2, equal to $N_0/(2E_b)$, and Δ is a random variable which is caused by the tracking errors. For instance, in the case of QPSK and assuming that there is a carrier phase error ϕ but no timing error, we have $\Delta = 1 - \cos\phi + b_k \sin\phi$. The corresponding BER is given by :

$$\begin{aligned} \mathrm{BER} &= E[\mathrm{Prob}[1 - \Delta + N < 0]] \\ &= E[Q\,(\frac{1 - \Delta}{\sigma})] \end{aligned} \qquad \text{(III.A–1)}$$

where E[.] denotes expectation over all possible Δ, and Q(.) is the area under the tail of the univariate Gaussian probability density function :

$$Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^{\infty} e^{-u^2/2}\, du \qquad \text{(III.A–2)}$$

Using for small Δ/σ a truncated Taylor series expansion about $\Delta = 0$ of Q(.) in (III.A–1), the BER is approximated by :

$$\begin{aligned} \mathrm{BER} &\cong E\left[Q\left[\frac{1}{\sigma}\right] - \frac{\Delta}{\sigma} Q'\left[\frac{1}{\sigma}\right] + \frac{1}{2}\left[\frac{\Delta}{\sigma}\right]^2 Q''\left[\frac{1}{\sigma}\right]\right] \\ &= Q\left[\frac{1}{\sigma}\right] - \frac{m}{\sigma} Q'\left[\frac{1}{\sigma}\right] + \frac{m^2 + s^2}{2\sigma^2} Q''\left[\frac{1}{\sigma}\right] \\ &= Q\left[\frac{1}{\sigma}\right] + \left[m + \frac{m^2 + s^2}{2\sigma^2}\right] Q''\left[\frac{1}{\sigma}\right] \end{aligned} \qquad \text{(III.A–3)}$$

where m and s^2 are the mean and the variance of the random variable Δ. The last line of (III.A–3) makes use of $Q''(x) + xQ'(x) = 0$, which follows from (III.A–2).

Now we consider a receiver without tracking errors and determine under which conditions it yields the same BER as a receiver with tracking errors. More specifially, we look for the constant ϵ, such that a decision device whose input equals $1 - \epsilon + N$ has the same BER as a decision device whose input equals $1 - \Delta + N$, where Δ is a random variable caused by tracking errors. The receiver without tracking errors yields :

$$\begin{aligned} \mathrm{BER} &= Q\left[\frac{1-\epsilon}{\sigma}\right] \\ &\cong Q\left[\frac{1}{\sigma}\right] - \frac{\epsilon}{\sigma} Q'\left[\frac{1}{\sigma}\right] + \frac{1}{2}\left[\frac{\epsilon}{\sigma}\right]^2 Q''\left[\frac{1}{\sigma}\right] \\ &= Q\left[\frac{1}{\sigma}\right] + \left[\epsilon + \frac{\epsilon^2}{2\sigma^2}\right] Q''\left[\frac{1}{\sigma}\right] \end{aligned} \tag{III.A–4}$$

Equating (III.A–3) and (III.A–4) yields a quadratic equation for ϵ; for small m/σ and s/σ the solution of this equation is approximated by :

$$\epsilon \cong m + \frac{s^2}{2\sigma^2} \tag{III.A–5}$$

The approximation (III.A–5) involves neglecting third–order and higher–order terms in m/σ and s/σ. Hence, the receiver with tracking errors, operating at a given E_b/N_o, yields the same BER as a receiver without tracking errors, operating at $(1-\epsilon)^2 E_b/N_o$; it follows from (III.A–5) that

$$(1-\epsilon)^2 \cong 1 - 2m + m^2 - \frac{s^2}{\sigma^2} \tag{III.A–6}$$

where we have neglected third–order and higher–order terms in m/σ and s/σ.

Now we consider for both receivers, which yield the same BER, the signal–to–noise ratio SNR at the input of the decision device; SNR is defined as the ratio of the square of the mean of the decision variable to the variance of the decision variable. For the receiver with tracking errors we obtain :

$$\begin{aligned} \mathrm{SNR} &= \frac{(1-m)^2}{\sigma^2 + s^2} \\ &\cong \frac{1 - 2m + m^2 - \frac{s^2}{\sigma^2}}{\sigma^2} \quad \text{(tracking errors)} \end{aligned} \tag{III.A–7}$$

where we have neglected third–order and higher–order terms in m/σ and s/σ. For the receiver without tracking errors we obtain :

$$\mathrm{SNR} = \frac{(1-\epsilon)^2}{\sigma^2} \quad \text{(no tracking errors)} \tag{III.A–8}$$

Taking into account (III.A–6), it follows that (III.A–7) and (III.A–8) are identical within third–order and higher–order terms of m/σ and s/σ. Hence, when both receivers yield the same BER, they have essentially the same SNR at the input of the decision device.

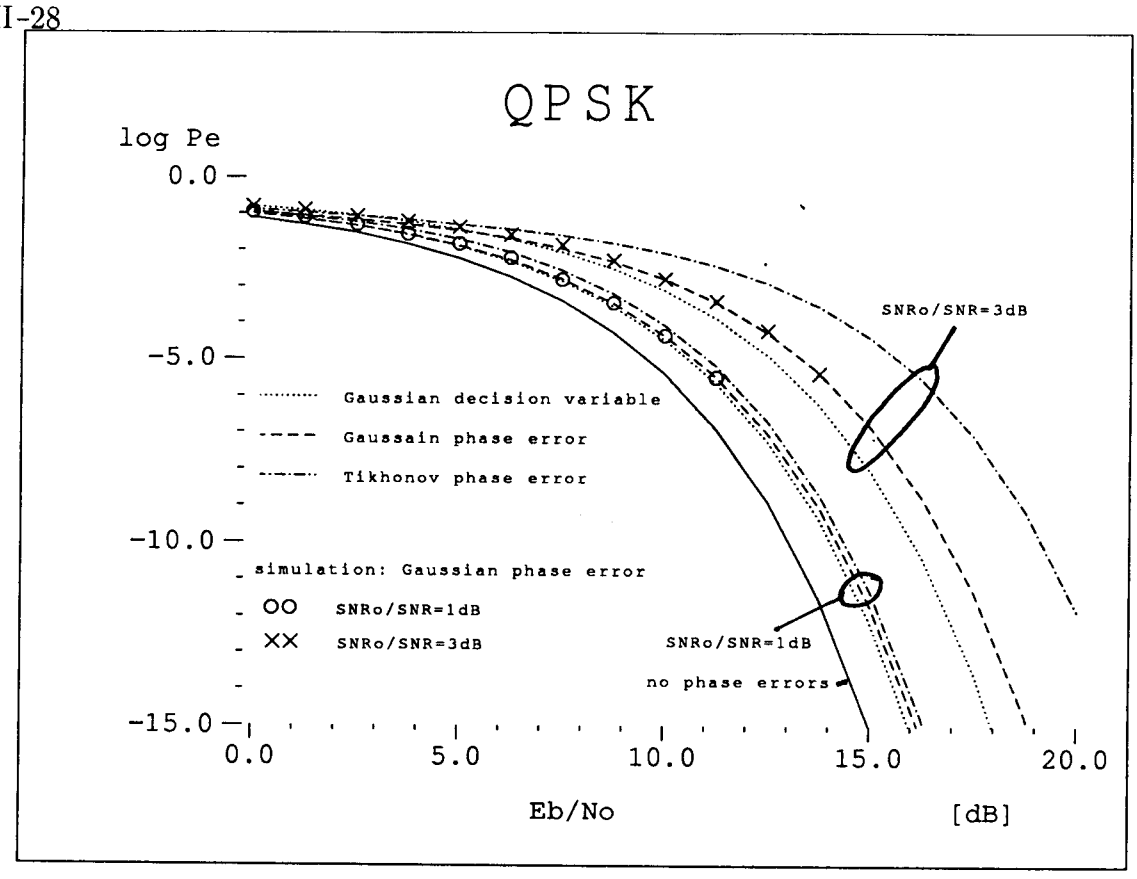


Fig. III.2-1: Influence of phase errors on BER

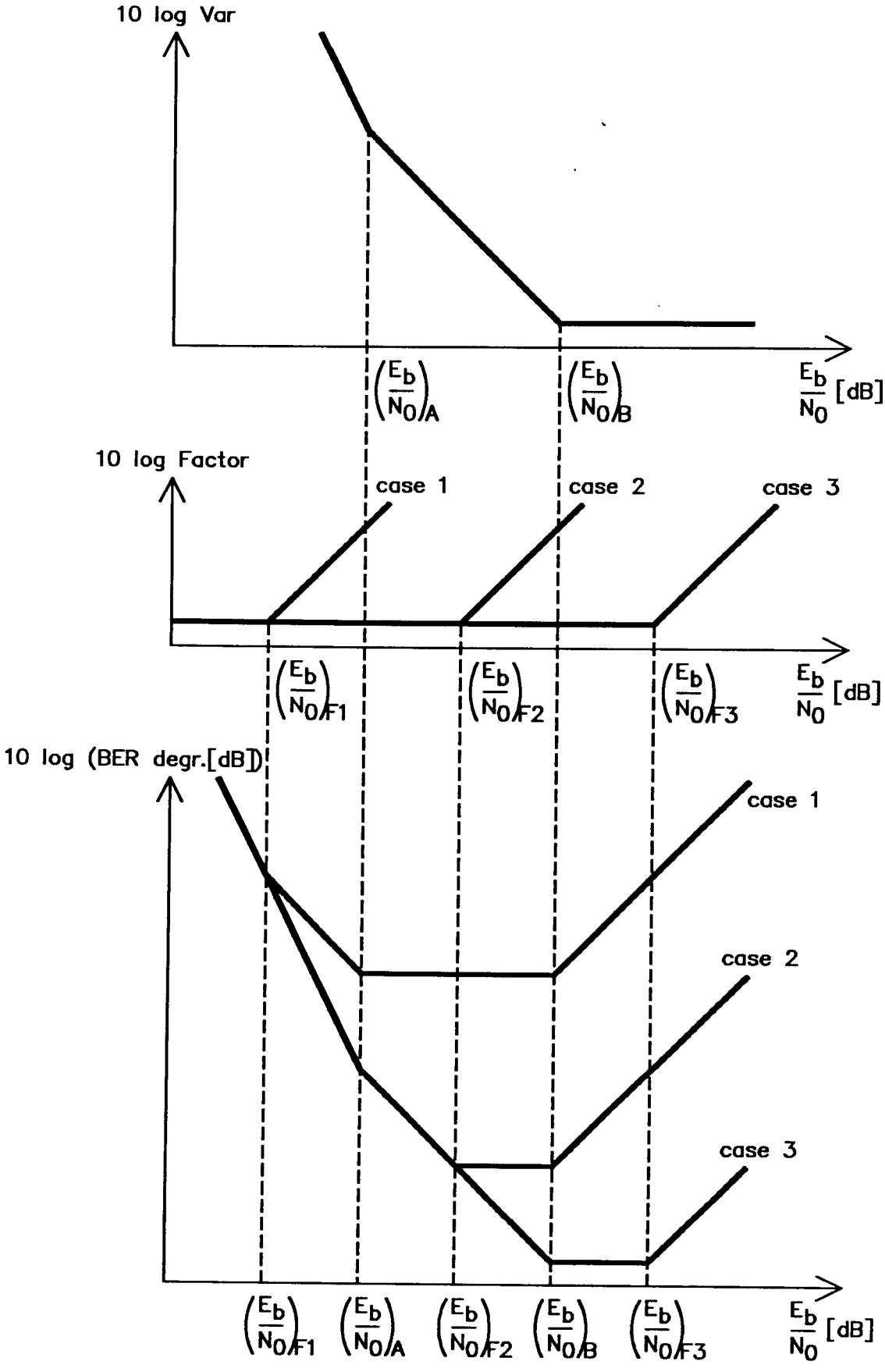
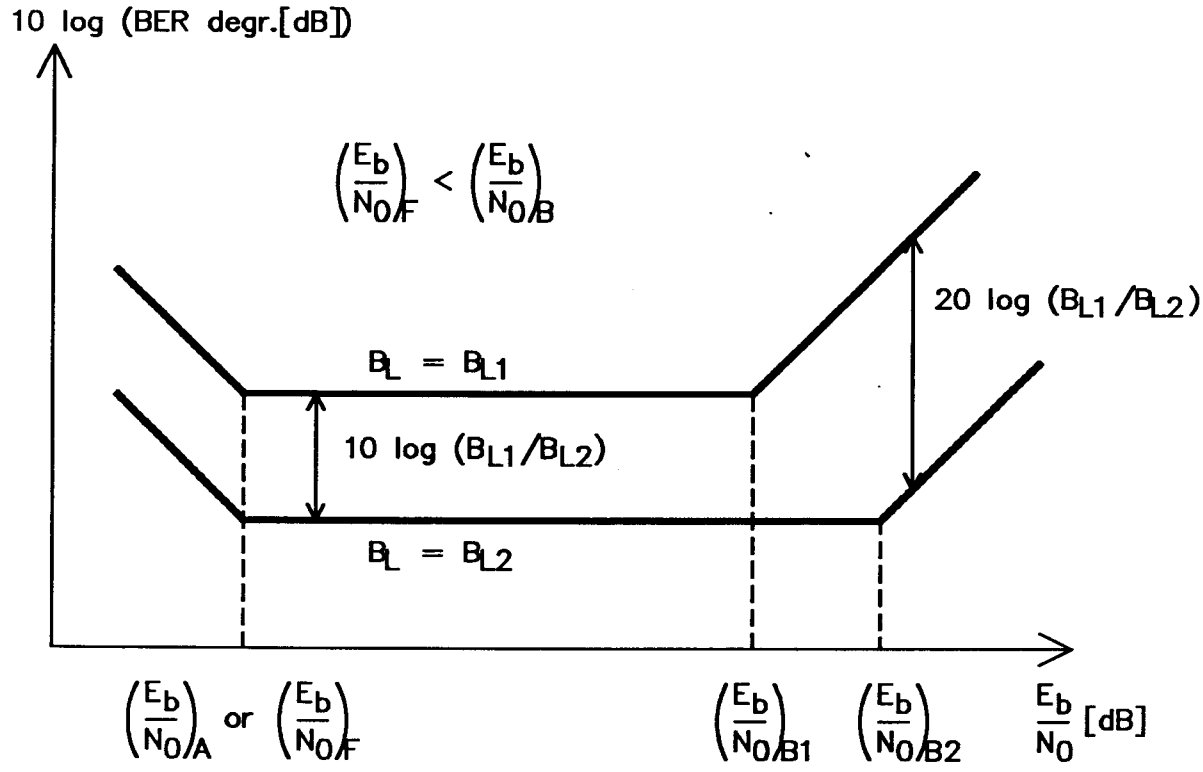


Fig.III.3–1 : Construction of the BER degr. [dB] curve



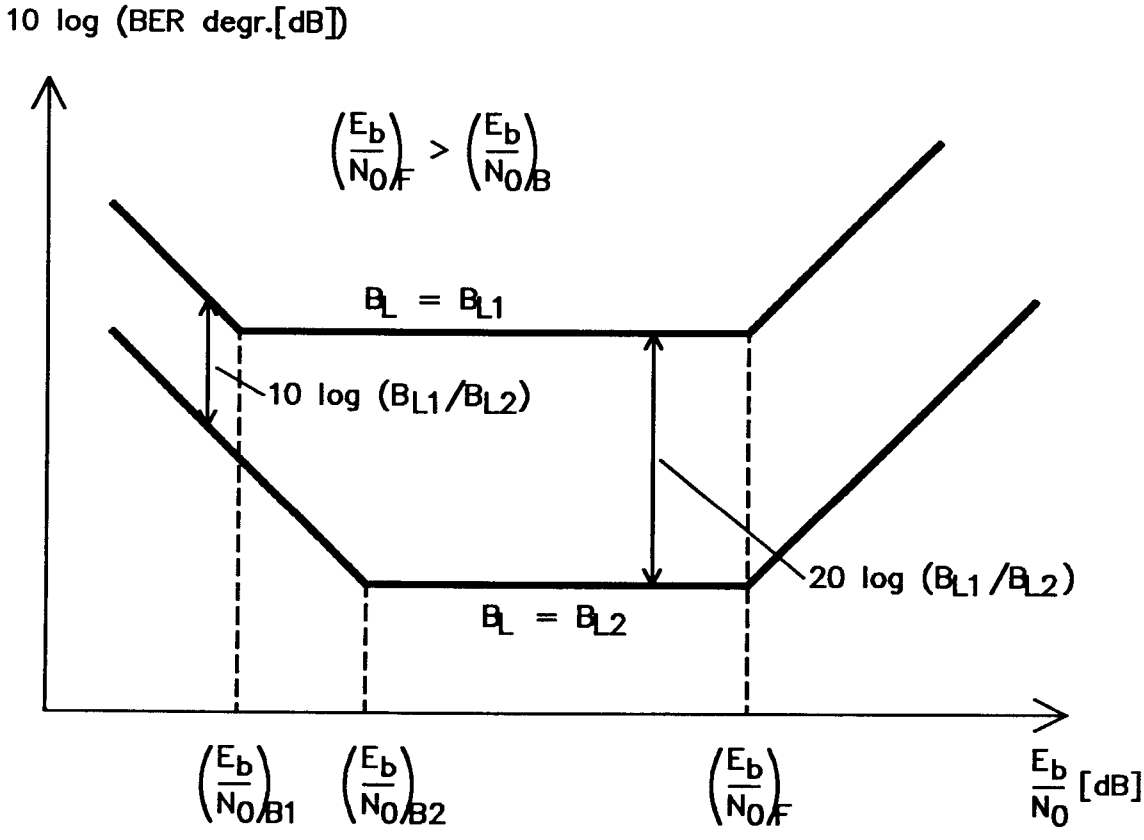


Fig.III.3–2 : Influence of synchronizer bandwidth on BER degradation ($B_{L1} > B_{L2}$)

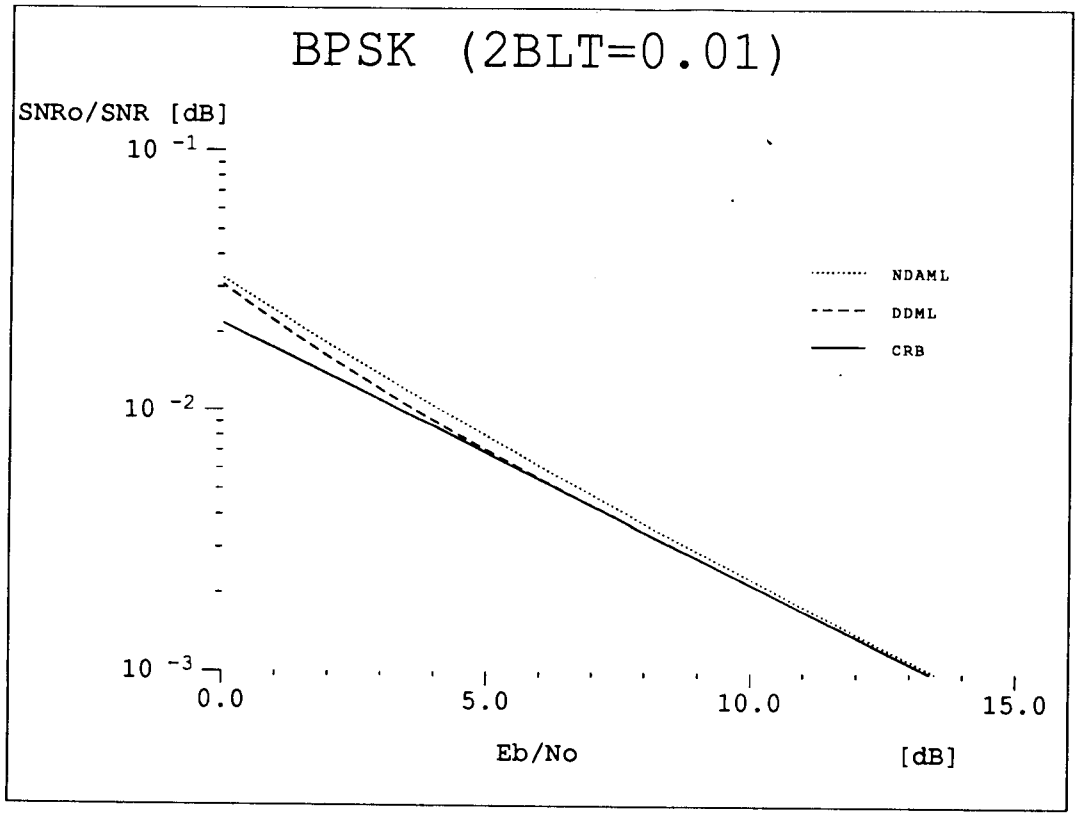


Fig. III.4-1: Carrier Synchronization: BPSK (2BLT = 0.01)

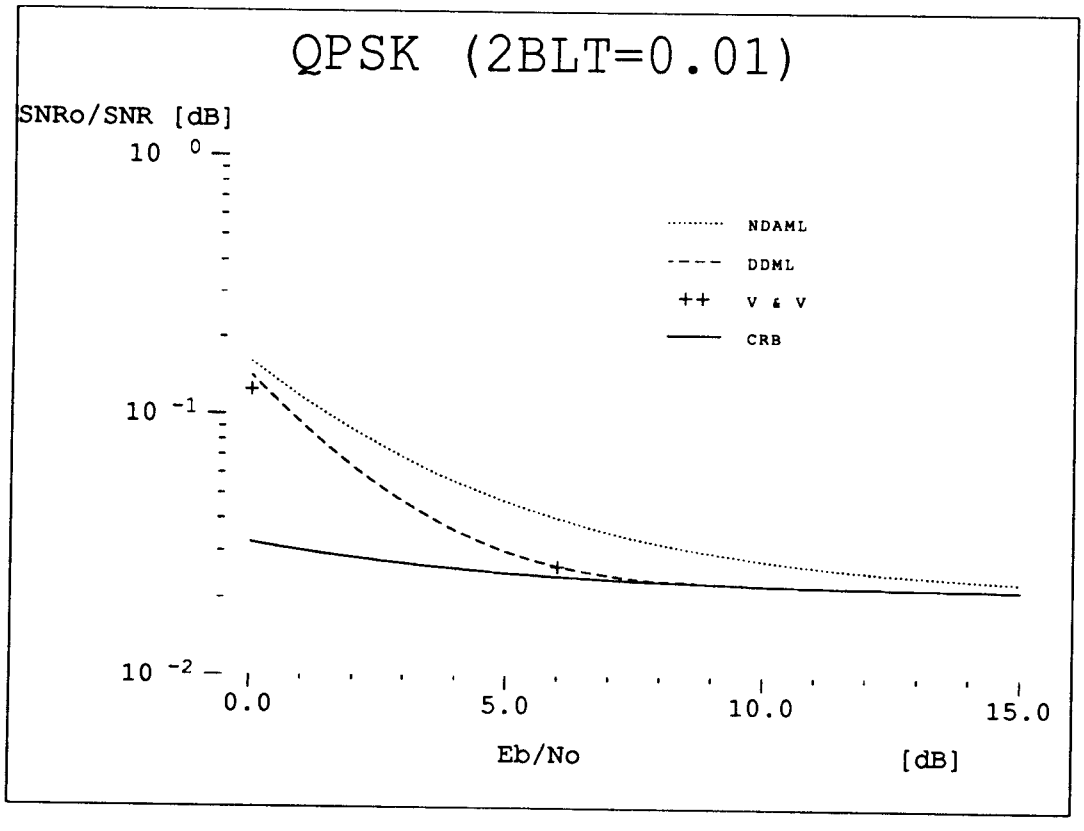


Fig. III.4-2: Carrier Synchronization: QPSK (2BLT = 0.01)

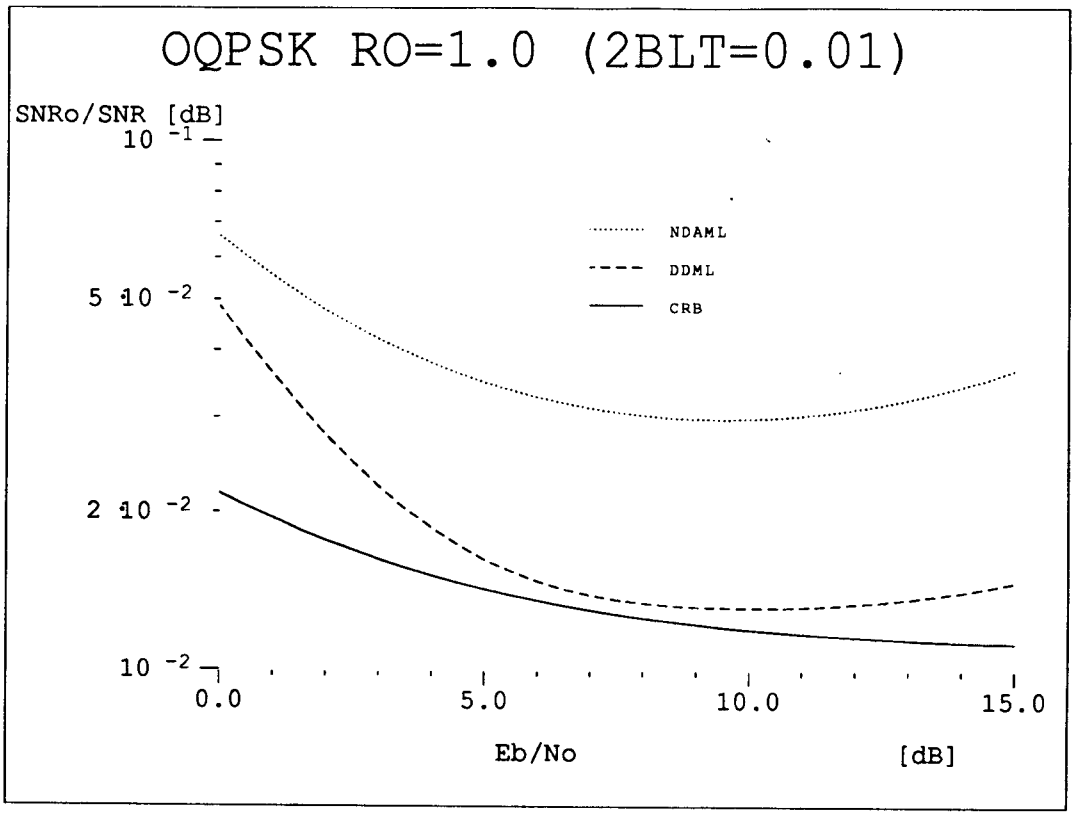


Fig. III.4-3: Carrier Synchronization: OQPSK, α=1 (2BLT = 0.01)

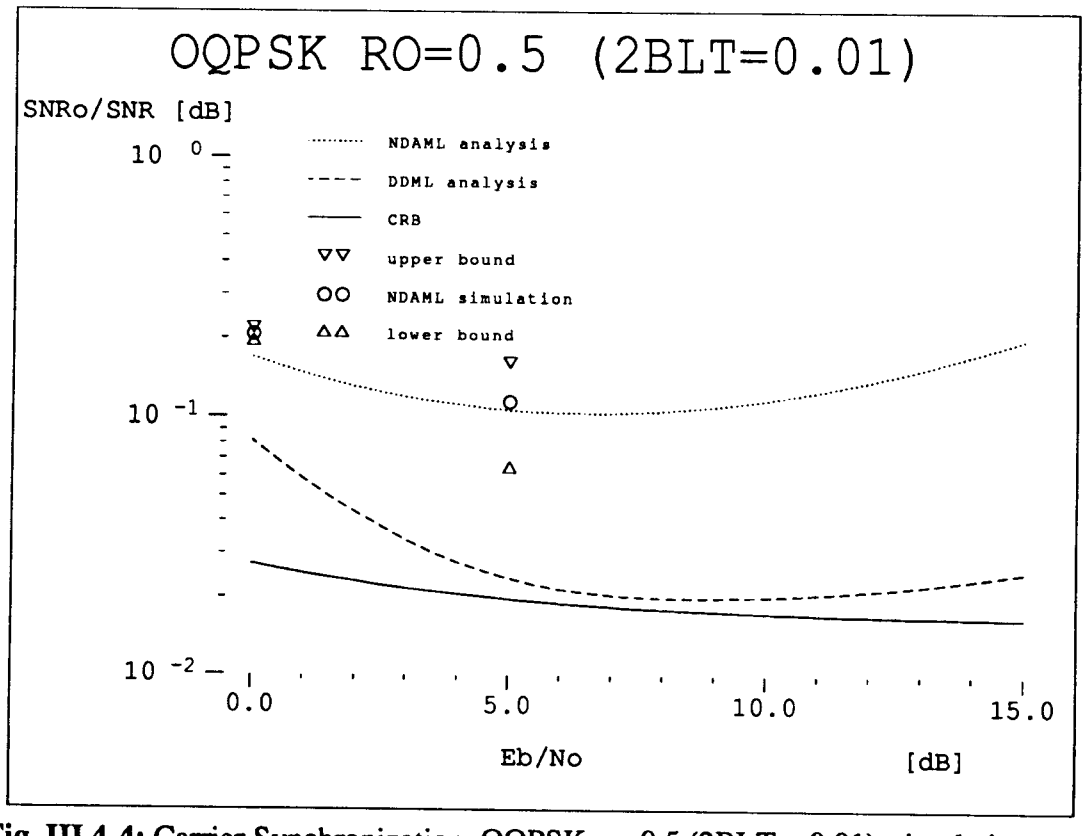


Fig. III.4-4: Carrier Synchronization: OQPSK, α=0.5 (2BLT = 0.01), simulation results shown with confidence interval ($\pm\sigma$)

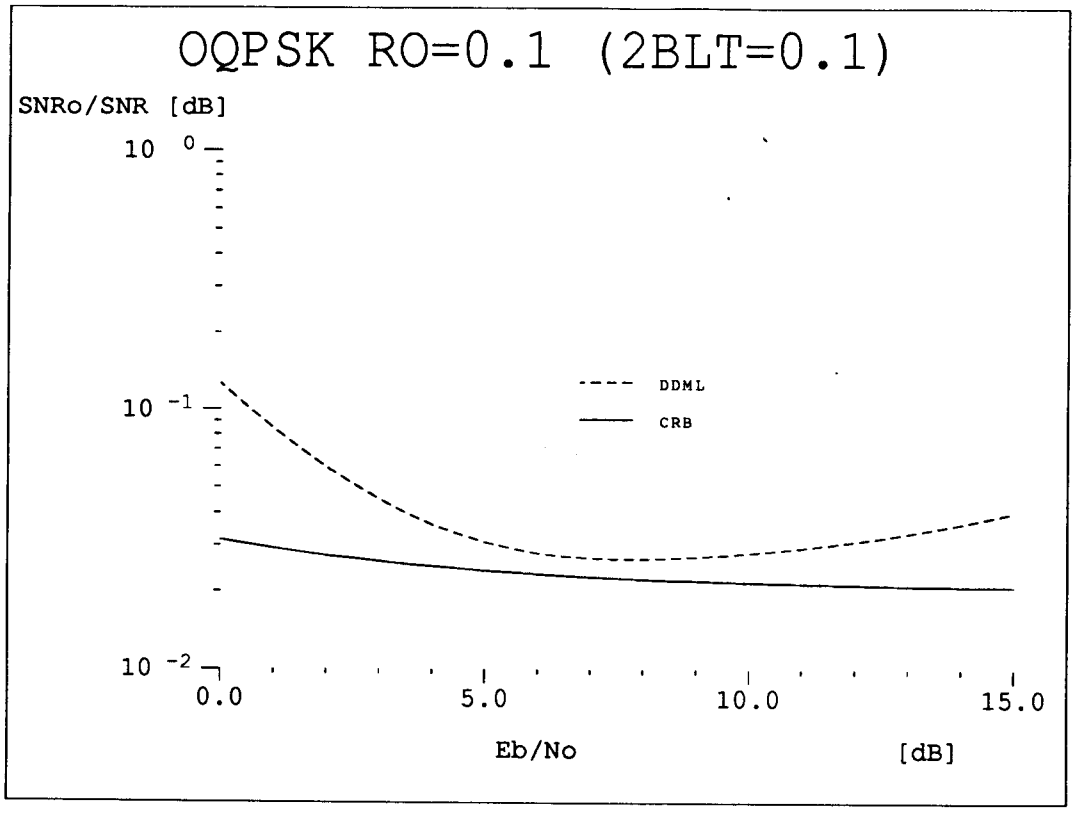


Fig. III.4-5: Carrier Synchronization: OQPSK, α=0.1 (2BLT = 0.01)

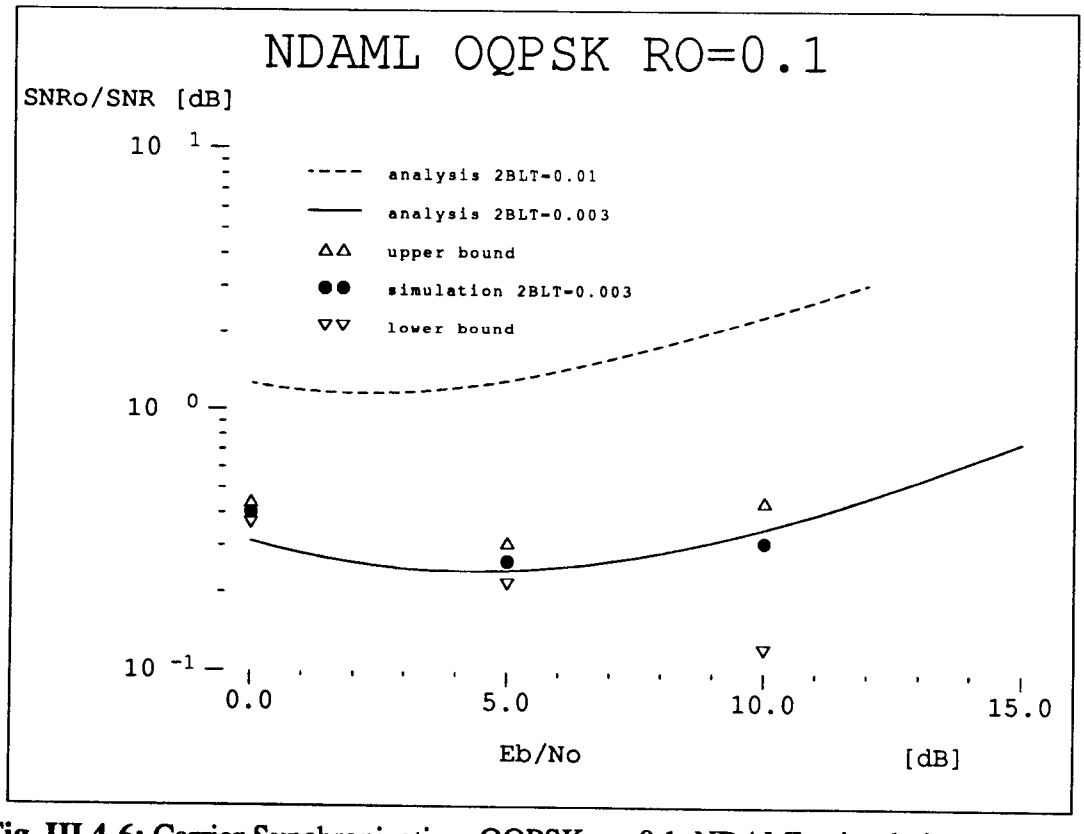


Fig. III.4-6: Carrier Synchronization: OQPSK, α=0.1, NDAML, simulation results shown with confidence interval ($\pm\sigma$)

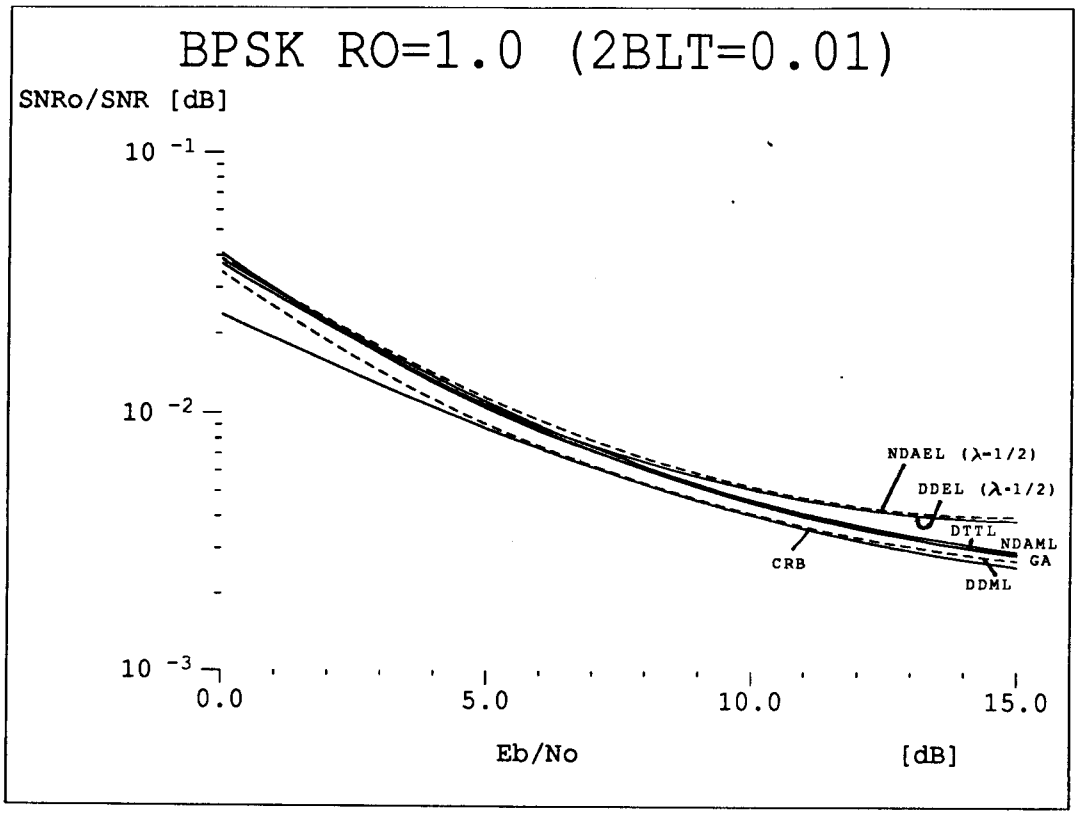


Fig. III.4-7: Symbol Synchronization: BPSK, α=1 (2BLT = 0.01)

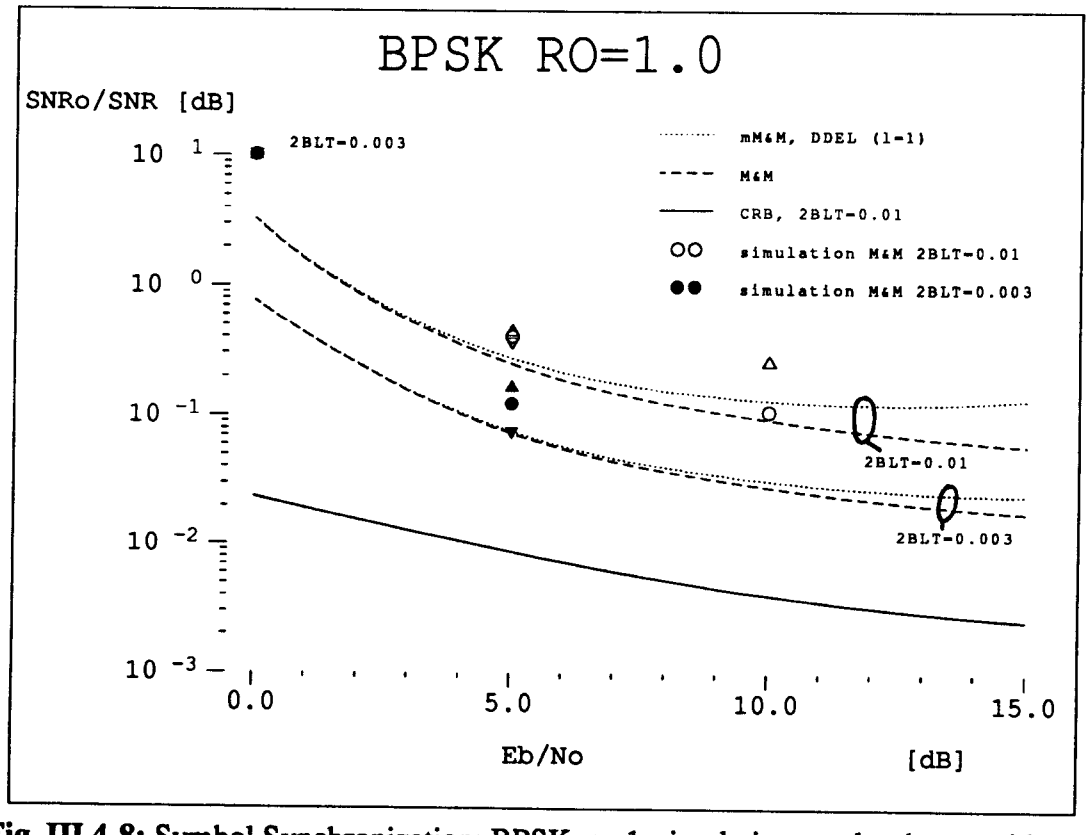


Fig. III.4-8: Symbol Synchronization: BPSK, α=1, simulation results shown with confidence interval ($\pm\sigma$)

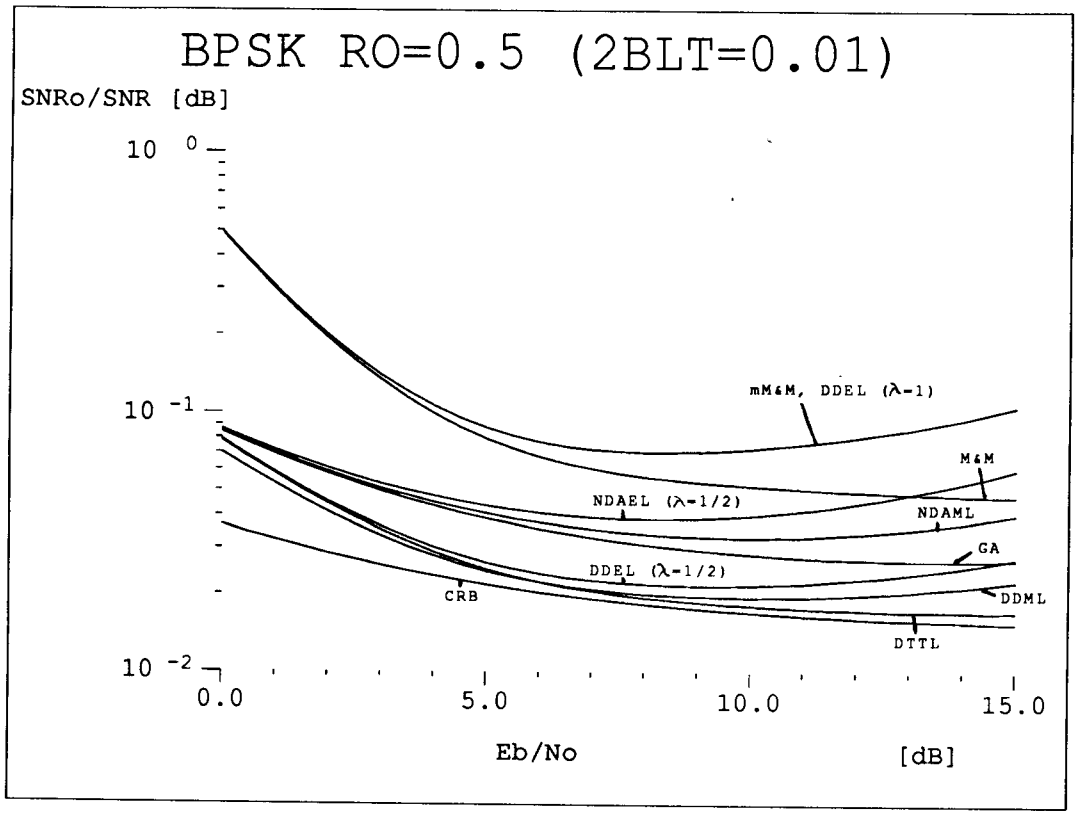


Fig. III.4-9: Symbol Synchronization: BPSK, α=0.5 (2BLT = 0.01)

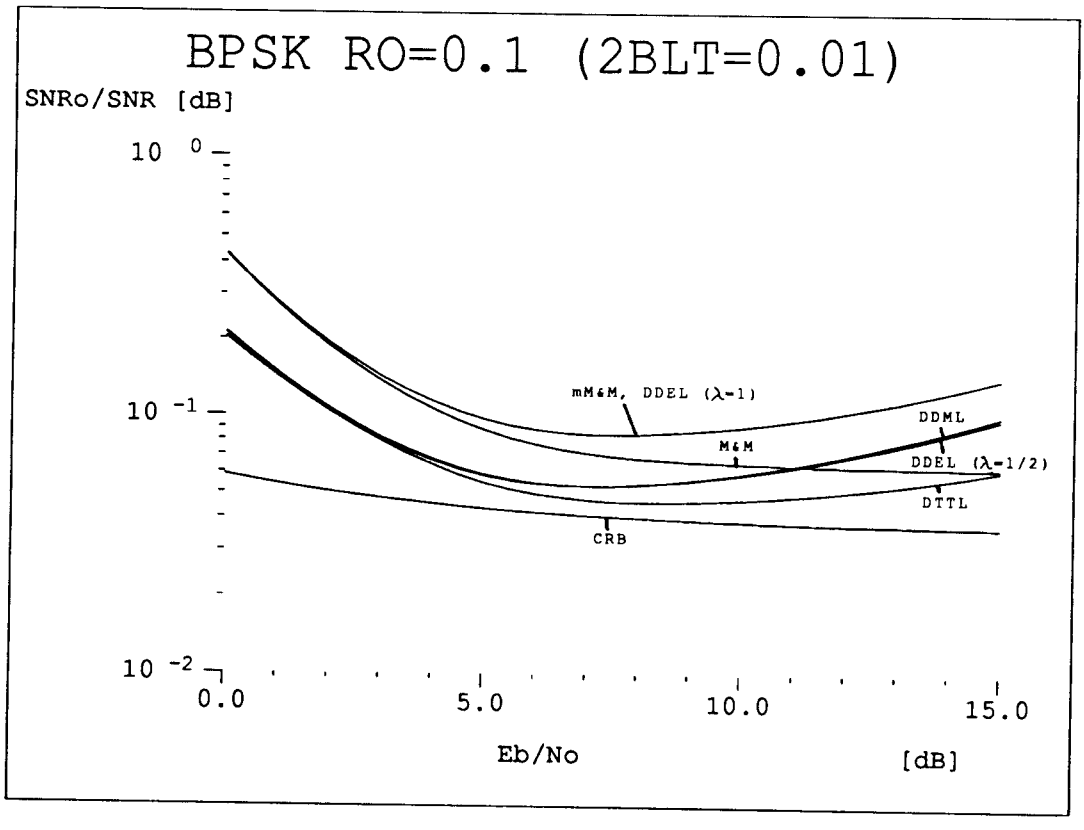


Fig. III.4-10: Symbol Synchronization: BPSK, α=0.1 (2BLT = 0.01)

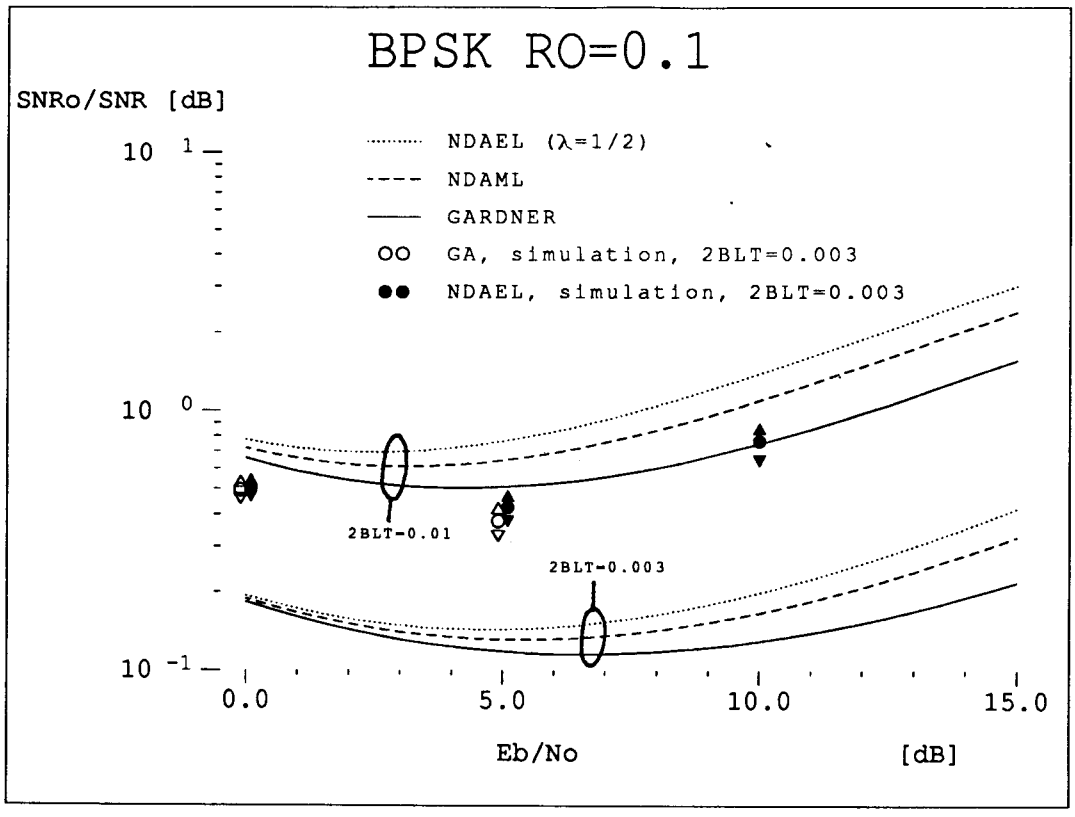


Fig. III.4-11: Symbol Synchronization: BPSK, α=0.1, simulation results shown with confidence interval ($\pm\sigma$)

IV. PHASE NOISE AND FREQUENCY ERRORS

IV.1 Introduction

When we study the influence of the additive Gaussian noise in the channel on synchronizer performance, the structure of the loop filter or the feedforward synchronizer filter is of minor importance. It is the loop noise bandwidth that determines the tracking error variance. Therefore, we cannot expect a better tracking performance by using more sophisticated filtering algorithms in a digital synchronizer. The various synchronizer types can be compared directly by their normalized loop noise power spectral densities as given in Part II (and in subsequent parts VII and VIII). From the results of these parts we can also conclude that it does not matter whether we have a feedback or a feedforward synchronizer, corresponding feedback/feedforward synchronizers have an identical tracking performance.

The situation is different when we consider distortions like frequency errors (e.g. due to Doppler shift) or phase noise. Without these distortions we may reduce the loop/feedforward filter noise bandwidth until we have the desired tracking error variance. But in general the synchronizer must also be capable of tracking frequency variations introduced by the channel (e.g. Doppler) and the oscillators (phase noise). This puts a lower bound on the feedforward filter or loop noise bandwidth. In many cases, the different distortions lead to conflicting requirements for the noise bandwidth which can only be resolved by introducing an appropriate (post-)filtering. For instance, to cope with large frequency errors an integrating loop filter should be used in a feedback synchronizer. We will also discuss in this part how a feedforward synchronizer can cope with large frequency errors. The link between the two topics of this section, phase noise and frequency error, is that their influence on tracking performance is strongly related with the filter structure. A third topic strongly related to filtering is acquisition. Since this is not part of the tracking performance we will discuss it in a subsequent part.

Of course we can implement similar filters as are known from analog synchronizers, e.g. use the discrete time equivalent of an integrating filter in a feedback synchronizer. But with digital signal processing we cannot only adapt the synchronizer to different operating conditions easily, we can also use more sophisticated algorithms for postprocessing. This makes a general

analysis very difficult, since postfilter structures may be tailored to particular applications. To give an example, assume a link from a geostationary satellite to a satellite in a low orbit. Then we will have a time varying Doppler shift with a periodic component. Now we could model the behaviour of the Doppler shift by a sinusoid with two unkown parameters, phase and amplitude. The synchronizer estimates these parameters and uses a Doppler frequency estimate for correction which is derived from this model. With the modelling we make maximum use of the a priori knowledge, thus, obtaining optimum estimates for the distortions.

As it is the link between the different topics of this section, we will first study (post-)filtering. Because of the large variety of filters we cannot treat them all. Here we will start with a postfilter algorithm which is based on the MMSE criterion, the Kalman filter approach. Assuming a particular model for the phase noise, the filter structure is obtained directly from the (MMSE) optimality criterion. The resulting filter is time-variant during acquisition but then approaches a steady state form which can be related to standard synchronizers.

We have a modulo 2π ambiguity with the carrier phase (error) and a modulo 1 ambiguity with the symbol timing (error). In addition, in a feedback synchronizer the mean of the detector output signal is a nonlinear function of the carrier phase/timing error. Therefore, we should have a nonlinear Kalman filter. Since nonlinear Kalman filters do not guarantee optimality it is not possible to provide the design rule for the best postfiltering algorithm.

Here we will use linearization, i.e. we neglect nonlinearities in the detector characteristic and derive the optimum filter for the linearized problem. Finally, a modulo 2π (modulo 1) reduction is introduced in the filter heuristically. Also there will be some simplifying assumptions in the modelling. Therefore, we cannot claim to have optimum performance, in particular, with respect to nonlinear phenomina (e.g. cycle slips). But even though the above approach is not an "ultimate" design rule it is a good starting point and also provides an interesting and systematical insight in the trade-off between additive channel noise and phase noise.

We do not give a derivation of the Kalman filter here since this is out of the scope of this report. Rather, appendix IV.A.1 summarizes the formulae

needed here. For details on Kalman filters the interested reader is referred to the available literature on Kalman filters, e.g. in [And'79] Kalman filters and nonlinear ('extended') Kalman filters are derived and their performance is studied.

Other basic filters discussed in this part are the gliding accumulator, which is a simple extension of the block processing accumulator for feedforward synchronizers, planar filtering, which is free of hang-up, and feedforward frequency estimation for feedforward synchronizers, which avoids feedback in the postfilter.

In subsequent sections the influence of the phase noise and of frequency errors on tracking performance is discussed. Since the principal results are identical, we will not treat carrier phase and clock synchronization separately except when the importance of the distortions and their magnitude are studied. A neutral variable is assumed to be confusing, therefore, the carrier phase will be used as the "vehicle" for derivation. Note that θ has to be replaced by $2\pi\epsilon$ when transferring results and formulae.

IV.2 POSTFILTERING

As mentioned before we cannot discuss all possible postfilters. We will rather discuss some basic structures. A very interesting approach which we will start with is the Kalman filter. It is a way to systematically derive a postfilter. Since we have to use approximations and linearization we cannot claim optimality. Nevertheless, the results give an interesting insight in the relations between distortions and filter structure. Also, it is a good starting point for a filter with improved acquisition performance. Finally, in the tracking mode we obtain a synchronizer structure which can be related to known analog feedback synchronizer structures.

Three other subsections discuss basic filter structures for feedforward synchronizers, the gliding accumulator, the planar filter, and feedforward frequency estimation.

IV.2.1 A (linearized) optimality approach

We base the derivation on a feedforward carrier synchronizer. The corresponding procedure for the feedback carrier synchronizer will come out as a side result of the feedforward case. For the applicability to timing synchronizers, see the last paragraph of the introduction.

From the accumulator filter we will obtain one carrier phase estimate each N symbols. Note that N = 1 corresponds to the case without accumulator filter. Since the carrier phase values in consecutive accumulation intervals are not statistically independent (i.e. the carrier phase does not make purely random jumps from interval to interval but varies more or less smoothly) we can reduce the error variance further by postfiltering the estimates. For a systematical approach we first must have a model of the carrier phase behaviour. The two most important phenomena to be modelled are the phase noise and frequency deviations, e.g. due to Doppler. The phase noise is a random fluctuation of phase occuring in any oscillator. The basic components of the power spectral density of the phase noise are shown in Fig. IV.2-1. Note that we have shown the phase noise power spectral density itself rather than the carrier power spectral density which usually is measured ('dBc'). Where the phase noise power spectral density is << 1 a linear relationship exists between both values. Since low frequency components (where the

phase noise is large) are tracked by the synchronizer the difference is not relevant for the analysis of synchronizer performance.

The individual contributions to the total phase noise in a transmission link are shown in Fig. IV.2-2. First, the transmit oscillator phase will fluctuate with respect to a (hypothetical) stable reference. Then, the channel may add random fluctuations. Finally, we have the contribution from the local oscillator(s). Since the received signal is down converted by the local oscillator, the resulting carrier phase is the difference between transmitted phase and local oscillator phase (note that this eliminates the influence of the (hypothetical) reference). Therefore, the carrier phase power spectral density will be the sum of transmit oscillator phase noise power spectral density and local oscillator phase noise power spectral density (assuming that the processes are independent and neglecting a possible influence of the channel). The total phase noise must be measured without feedback from the synchronizer (if there is any). With feedback we would already measure the influence of the synchronizer.

From Fig. IV.2-1 we see that the total phase noise variance, which is the integral over the power spectral density, is unlimited. First, this is due to the white noise floor. But this floor is cut off by the anti-aliasing filter before sampling and/or the receive matched filter. Second, the power spectral density goes to infinity for f -> 0. This corresponds to the fact that the phase difference between two free running oscillators approaches infinity with time (neglecting the modulo 2π reduction). As will be discussed in detail later this deviation is tracked by a feedback synchronizer and, therefore, the power spectral density of the phase error due to phase noise will be limited for f -> 0 with closed feedback. In a feedforward synchronizer these slow variations will not be suppressed by a finite length accumulator filter and so they will also be "tracked". This means, that the phase noise power spectral density at the phase rotator output will also be limited for f -> 0.

Now we can model the actual phase θ_0 and frequency $\Delta\omega_0$ by a discrete time model with sampling interval NT

$$\begin{aligned} \theta_0((k+1)NT) &= \theta_0(kNT) + \Omega_0(kNT) + w_\theta(kNT) \\ \Omega_0((k+1)NT) &= \Omega_0(kNT) + w_\Omega(kNT) \end{aligned} \qquad \text{(IV.2.1-1)}$$

θ_0 is the mean value of the phase in the interval for which the accumulator filtering is performed. $\Omega_0(kNT) = \Delta\omega_0(kNT)NT$ is the mean frequency deviation normalized to $1/(NT)$, i.e. the phase increment between two consecutive phase values due to a frequency error. Assuming that w_θ is a zero mean noise term with flat power spectral density, it causes a random phase walk ($S_{\Theta\Theta}(\omega) \sim 1/f^2$). Similarly, assuming that w_Ω is a zero mean noise term with flat power spectral density it represents a random frequency walk ($S_{\Theta\Theta}(\omega) \sim S_{\Omega\Omega}(\omega)/f^2 \sim 1/f^4$). The phase noise power spectral density corresponding to this approximation is shown by a dashed line in Fig. IV.2-1. With the above assumptions, equ. (IV.2.1-1) represents a two dimensional Gauss Markov model with state variables *phase* θ_0 and *frequency deviation* Ω_0. The noise terms w_θ and w_Ω are zero mean, white, Gaussian random variables with the variances

$$E[w_\theta^2] = \sigma_\theta^2$$
$$E[w_\Omega^2] = \sigma_\Omega^2 \qquad \text{(IV.2.1-2)}$$

The first equation in (IV.2.1-1) describes a random phase walk with an additional bias by a frequency deviation. The second equation corresponds to a random walk of this frequency deviation. The feedforward carrier synchronizer now outputs the phase estimate $\hat{\theta}$ which is the sum of the actual phase θ_0 and a noise term n_θ

$$\hat{\theta}(kNT) = \theta_0(kNT) + n_\theta(kNT) \qquad \text{(IV.2.1-3)}$$

Neglecting self noise terms, n_θ is a zero mean noise variable which can be approximated as being white, Gaussian noise. Its variance σ_M^2 is the variance of the accumulator output noise term. From equ. (IV.2.1-1) and (IV.2.1-3) we already get a first impression of the problem to be solved by the postfilter. By narrowband filtering, n_θ can be reduced significantly but then also θ_0 which is driven by the phase and frequency noise terms w_θ and w_Ω is also filtered and, therefore, the estimate no longer follows the actual phase fluctuations. The optimal filter for such a model is the Kalman filter. Its derivation is given in the appendix IV.A.1. The Kalman filter predicts

the state of the model (IV.2.1-1) at time kNT taking all measurements $\hat{\theta}(0)$, ..., $\hat{\theta}(k-1)$ up to time (k-1)NT into account. The predictions are determined according to the recursion

$$\tilde{\theta}(k+1|k) = \tilde{\theta}(k|k-1) + \tilde{\Omega}(k|k-1) + K_1(k)\,[\hat{\theta}(k) - \tilde{\theta}(k|k-1)]$$

$$\tilde{\Omega}(k+1|k) = \tilde{\Omega}(k|k-1) + K_2(k)\,[\hat{\theta}(k) - \tilde{\theta}(k|k-1)] \qquad \text{(IV.2.1-4)}$$

$\tilde{\theta}(k|k-1)$, $\tilde{\Omega}(k|k-1)$: prediction of state $\theta_0(k)$, $\Omega_0(k)$ based on the measurements up to time (k-1)NT

From the predictions we can also determine the best estimates at time kNT taking the measurements up to time kNT into account.

$$\tilde{\theta}(k|k) = \tilde{\theta}(k+1|k) - \tilde{\Omega}(k+1|k)$$

$$\tilde{\Omega}(k|k) = \tilde{\Omega}(k+1|k) \qquad \text{(IV.2.1-5)}$$

$\tilde{\theta}(k|k)$, $\tilde{\Omega}(k|k)$: optimally filtered estimate of state $\theta_0(k)$, $\Omega_0(k)$ based on the measurements up to time kNT

The coefficients $K_i(k)$ are given by

$$K_1(k) = \frac{\sigma_1^2(k) + \sigma_{12}^2(k)}{\sigma_1^2(k) + \sigma_M^2}$$

$$K_2(k) = \frac{\sigma_{12}^2(k)}{\sigma_1^2(k) + \sigma_M^2} \qquad \text{(IV.2.1-6)}$$

The variances $\sigma_i^2(k)$ are defined recursively

$$\sigma_1^2(k) = \left[\sigma_1^2(k-1)+2\sigma_{12}^2(k-1)+\sigma_2^2(k-1) - \frac{\left[\sigma_1^2(k-1)+\sigma_{12}^2(k-1)\right]^2}{\sigma_1^2(k-1)+\sigma_M^2} + \sigma_\theta^2\right]$$

$$\sigma_{12}^2(k) = \left[\sigma_{12}^2(k-1)+\sigma_2^2(k-1) - \frac{\sigma_{12}^2(k-1)\left[\sigma_1^2(k-1)+\sigma_{12}^2(k-1)\right]}{\sigma_1^2(k-1) + \sigma_M^2}\right]$$

$$\sigma_2^2(k) = \left[\sigma_2^2(k-1) - \frac{\sigma_{12}^4(k-1)}{\sigma_1^2(k-1)+\sigma_M^2} + \sigma_\Omega^2\right] \qquad \text{(IV.2.1-7)}$$

At first glance the above equations look very complicated and seem to be impractical. The (shocked) reader may ignore the details for the moment. They will become much clearer after the discussion of these equations.

We have defined two estimates, a prediction (eq. IV.2.1-4) and a filtered estimate (eq. IV.2.1-5). We will discuss the use of these two different estimates (and a third one which is not yet mentioned) later in this section. For the moment we note that the prediction makes only use of past measurements and the filtered estimate takes past and present time measurements into account. It can be shown that the filteres estimate has a lower error variance.

We now interpret eq. (IV.2.1-4). The prediction error $\hat{\theta}(k) - \tilde{\theta}(k|k-1)$ is weighted by $K_2(k)$ and added to the old normalized frequency prediction to determine the new normalized frequency prediction. The new phase prediction equals the sum of old prediction, phase increment due to (normalized) frequency prediction, and phase prediction error weighted by $K_1(k)$. From (IV.2.1-5) we see that the optimum frequency estimate at time kNT is also the optimum prediction for time (k+1)NT. The optimum filtered phase estimate at time kNT equals the phase prediction for time (k+1)NT minus frequency induced phase increment during the next time step. This is understood more clearly when we insert the second equation and solve the first equation for the prediction

$$\tilde{\theta}(k+1|k) = \tilde{\theta}(k|k) + \tilde{\Omega}(k|k)$$

i.e. the optimum prediction is the sum of optimally filtered estimates of phase and normalized frequency. Note that it is also possible to derive a recursion for the filtered estimates directly where the predictions do not occur explicitly. We have used the recursion for the predictions here because it gives better insight in the Kalman filter structure. The filtered estimates $\tilde{\theta}(k|k)$, $\tilde{\Omega}(k|k)$ are the optimum estimates based on the measurements up to time kNT and, therefore, should be used in NDA synchronizers (instead of the predictions). In DD synchronizers we must use the prediction of the phase $\tilde{\theta}(k|k-1)$ for correction of the samples of the k-th block prior to decision since these decisions are needed to determine the measurement $\hat{\theta}(k)$.

We note that the filter coefficients and variances are a function of time. This is due to the fact that the Kalman filter takes the acquisition procedure into account. To understand this we have to study the variances $\sigma_i(k)$ in more detail. The variances $\sigma_1^2(k)$ and $\sigma_2^2(k)$ are of particular importance. We have

$$E[(\theta_0(k) - \tilde{\theta}(k|k-1))^2] = \sigma_1^2(k)$$

$$E[(\Omega_0(k) - \tilde{\Omega}(k|k-1))^2] = \sigma_2^2(k)$$

$$E[(\theta_0(k) - \tilde{\theta}(k|k-1))(\Omega_0(k) - \tilde{\Omega}(k|k-1)] = \sigma_{12}^2(k)$$

i.e. the variances equal the prediction error variances and prediction error covariance. Consequently, the initial values of the variances correspond to the initial uncertainty. For instance, when we assume that before acquisition a frequency deviation is anywhere in the interval $[-\Delta\omega NT, +\Delta\omega NT]$ with equal probability, i.e. uniformly distributed, the variance σ_2 is the variance of a uniform distribution.

$$\sigma_2^2(0) = \frac{(\Delta\omega NT)^2}{3} \qquad \text{(IV.2.1-8)}$$

The initial estimate for the filter frequency is the expected value of the frequency deviation which is with the above assumptions the centre of the interval, i.e. $\tilde{\Omega}(1|0) = \tilde{\Omega}(0|0) = 0$. Since we do not know anything about the carrier phase the corresponding variance is $\sigma_1^2(0) \to \infty$. Note that this leads to $K_1(0) = 1$ and $K_2(0) = 0$ and, thus, the first filtered estimate $\tilde{\theta}(0|0)$ is the first accumulator output $\hat{\theta}(0)$, which is reasonable. Finally, we have

$$E[(\theta_0(k) - \tilde{\theta}(k|k-1))(\Omega_0(k) - \tilde{\Omega}(k|k-1))] = \sigma_{12}^2(k)$$

i.e. $\sigma_{12}^2(k)$ is the covariance of phase and frequency prediction error. Since phase error and frequency deviation initially are uncorrelated in our example, $\sigma_{12}(0) = 0$.

We observe in the recursions that the variances $\sigma_i^2(k)$ and the coefficients $K_1(k)$, $K_2(k)$ evolve with time as a function of the initial values, the accumulator output noise variance σ_M^2 and the phase and frequency random walk variances σ_θ^2 and σ_Ω^2. The filter coefficients (and the variances) approach steady state values which we will determine in the following. These values correspond to the tracking coefficients. Note that the variances and thus, coefficients are independent of the measured values $\hat{\theta}$. They only depend on the modelling assumptions. Therefore, they might be computed in advance (i.e. <u>off-line</u>) and stored, for instance in a PROM. This is in contrast to the extended Kalman filter, where the coefficients also depend on the values of the filtered estimates or predictions and, therefore, must be computed in real-time (i.e. on-line).

We still have to introduce (heuristically) a nonlinearity. As discussed above, in equ. (IV.2.1-4) the new prediction of the phase is the sum of the old phase prediction, old (normalized) frequency prediction, and weighted prediction error $\hat{\theta}(k) - \tilde{\theta}(k|k-1)$. Now assume that the prediction is approximately $+\pi$ but the measured phase value is close to $-\pi$. We get a wrong filter update because we assume a prediction error of 2π which is not present since a phase error of 2π corresponds to zero phase error. Therefore, we have to apply a modulo 2π operation to the error before calculating the

new filtered value. The resulting structure is shown in Fig. IV.2-3. The prediction error $\hat{\theta}(k) - \tilde{\theta}(k|k-1)$ is (modulo) reduced to the basic interval $[+\pi,-\pi)$ by a sawtooth nonlinearity, weighted by K(k) and fed as input to a copy of the phase noise generation model. At the input of the delay block we have the new prediction from which we can determine the filtered phase (and frequency) estimates according to (IV.2.1-5).

For k -> ∞ the variances σ_i^2 approach a steady state value which can be determined by letting $\sigma_i^2(k) = \sigma_i^2(k-1)$. The corresponding values are

$$\sigma_2^2 = \frac{\sigma_1^2 \sqrt{\sigma_\Omega^2}}{\sqrt{\sigma_1^2+\sigma_M^2}} + \sigma_\Omega^2$$

$$\sigma_{12}^2 = \sqrt{\sigma_\Omega^2}\,\sqrt{\sigma_1^2 + \sigma_M^2} \qquad \text{(IV.2.1-9)}$$

$$\sigma_1^4 = \sqrt{\sigma_1^2 + \sigma_M^2}\,(\sigma_1^2 + 2\sigma_M^2)\sqrt{\sigma_\Omega^2} + \sigma_\theta^2\,(\sigma_1^2 + \sigma_M^2)$$

From the third equation σ_1^2 cannot be calculated explicitly in the general case. When we assume that the accumulator output noise term is larger than the phase noise effect $\sigma_1^2 << \sigma_M^2$ then we have approximately from equation (IV.2.1-9)

$$\sigma_1^2 = \sqrt{\sigma_\theta^2\sigma_M^2 + \sqrt{4\sigma_M^6\sigma_\Omega^2}} = \sigma_\theta\sigma_M\sqrt{1 + 2\sigma_M\sigma_\Omega/\sigma_\theta^2} \qquad \text{(IV.2.1-10)}$$

What is the use of these results? When we know the variances of the phase random walk and the frequency random walk as well as the variance of the accumulator output noise term we can directly calculate the steady state values of the variances and also of the filter coefficients K_1, K_2.

Now let us take a closer look at the filter structure. We find that we have introduced feedback! This can be clearly seen from the filter structure shown in Fig. IV.2-3, i.e. even though the carrier phase is determined by a

feedforward synchronization algorithm we now have a feedback structure. But since we have a time-variant filter during acquisition we have a much better acquisition performance than with a common feedback synchronizer. As mentioned above, the first filtered estimate is $\tilde{\theta}(0|0) = \hat{\theta}(0)$, i.e. the first feedforward algorithm phase estimate. This significantly reduces the probability of hang-up as compared to a common feedback synchronizer.

In recursion (IV.2.1-4) we see directly that the dynamic behaviour of the filter is determined by $K_1(k)$ and $K_2(k)$. This is verified easily since the influence of a prediction error depends on the magnitude of these values. Therefore, these parameters are also closely related to the bandwidth of this filter. This makes possible a trade-off, i.e. by reducing the accumulator length samples are delivered to the Kalman filter at a higher rate. Since we get a better resolution of the phase process over time an improved filter performance can be expected. But we have to be aware that in the feedforward synchronizers we have a nonlinearity at the accumulator filter output (the argument function). Therefore, the accumulator output noise variance will increase more than proportionally with 1/N, in particular, for low signal to noise ratios.

An alternative for obtaining a higher prefilter output sampling rate without an increased nonlinearity loss is a gliding accumulator prefilter with blocklength N. Now we can output up to one output sample per input sample. But with such a prefilter consecutive estimation errors at the prefilter output are not statistically independent. Therefore, the phase noise measurement model assumed to derive the Kalman filter is not applicable anymore (there we have assumed statistically independent errors). We might still use the derived filter structure but the analytically derived recursions for the error weighting coefficients no longer are useful. Instead, a better set of coefficients could be determined for example by simulation.

A general remark should be given here. We have derived a filter structure and the corresponding coefficient recursions. Because of several simplifications the coefficients may not be optimum. In particular, during acquisition the filtered estimates may not be improved as predicted by the linear Kalman filter approach because we are in a nonlinear operating mode. But even when we do not use the coefficients obtained from the recursions the structure of the filter is useful. For instance, we can use the structure of

the Kalman filter but only two sets of coefficients $K(0)$, $K(\infty)$. One set is used during acquisition, the second after lock detection in the tracking mode. Finally, it should be mentioned that the matrices of a Kalman filter for a nonlinear problem ('extended Kalman filter') depend on the estimates and, therefore, have to be computed <u>on-line</u> which significantly increases the required processing power but also avoids the above discussed disadvantage of precomputed filter coefficients.

Two closely related modifications should also be discussed here. With the Kalman filter approach we can also derive a filter which is based on a prediction over more than one step NT ('prediction filter'). The optimum prediction of the phase over more than one NT step is obtained from the one step prediction by

$$\tilde{\theta}(k+N_L|k) = \tilde{\theta}(k|k) + N_L\tilde{\Omega}(k|k) = \tilde{\theta}(k+1|k) + (N_L-1)\tilde{\Omega}(k+1|k)$$

In Fig. IV.2-4 the influence of processing delay in a DD carrier synchronizer is shown. In particular at high data rates parallel processing (based on pipelining) may be necessary. For example, the prediction of the phase may be available after a delay of N_F symbols. Similarly, assume a processing delay of N_D symbols (including a potential decision delay). Then, the total delay in the decision feedback loop is $N_L = N_F + N_D$. This must be compensated by a prediction over N_L symbols. Of course, predicition over more than one step may result in worse filter dynamics, in particular, for large filter bandwidths.

With a digital implementation we can also go in the opposite direction, i.e. we can store the samples and use 'future' measurements for the filtered estimate $\tilde{\theta}(k)$ ('smoothing filter') and correct the samples prior to decision based on these smoothed estimates.

Finally, it is readily verified that all these filters can also be used with a feedback synchronizer. In this case we use the prediction $\tilde{\theta}(k|k-1)$ (eq. IV.2.1-7) as reference for the phase error detector. The phase detector output then equals (as long as linearization applies) the prediction error

required to update the filter estimates. Using the feedback synchronizer notation defined in part I of this report, the corresponding equations read

$$\tilde{\theta}(k+1|k) = \tilde{\theta}(k|k-1) + \tilde{\Omega}(k|k-1) + K_1(k)\, u_\theta(k, \tilde{\theta}(k|k-1))$$

$$\tilde{\Omega}(k+1|k) = \tilde{\Omega}(k|k-1) + K_2(k)\, u_\theta(k, \tilde{\theta}(k|k-1)) \qquad \text{(IV.2.1-11)}$$

Comparing (IV.2.1-11) and (IV.2.1-4) we see that in the tracking mode (i.e. when linearization applies) we have the same equations and performance for feedback and feedforward synchronizers (when the phase detector characteristic is normalized to have a unit slope).

Further it is interesting that (for the tracking mode) the following equivalence with the parameters of a classical phase-locked loop exists when we approximate its differential equation by a difference equation (Digital PLL)

$$K_1 = 2\zeta\omega_n NT \quad \text{and} \quad K_2 = (\omega_n NT)^2$$

This means from loop noise variance, phase random walk variance, and frequency random walk variance we can directly calculate the steady state values for K_1 and K_2 and, thus, the optimum natural loop frequency ω_n and damping factor ζ of the synchronizer. For the special case $\sigma_\theta^2 = 0$, i.e. when we only have a random frequency fluctuation driving the carrier phase, we obtain from (IV.2.1-9) and (IV.2.1-6)

$$(\omega_n NT)^2 = K_2 = \sigma_\Omega/\sigma_M \quad , \quad \zeta = \frac{K_1}{2\sqrt{K_2}} = 1/\sqrt{2}$$

Thus, we find that $\zeta = 0.707$ is the optimum damping of a feedback synchronizer when we only have a frequency random walk. Again remember that the major difference to a DPLL is in the controlled time variance during acquisition.

IV.2.2 Planar filtering

Instead of filtering the phase estimate, as discussed in the previous subsection, we can also filter the complex signal before applying the argument function. This is called planar filtering since we filter in the complex plane. A special case, of course, is the accumulator filter which is used in the definition for feedforward synchronization algorithms. Basically, this filter outputs one estimate of the quadrature components each NT symbols. For a given input symbol x the output signal y is given by

$$y(kNT) = \sum_{m=1}^{N} x[kNT+ (m-(N+1)/2)T] \quad ; \quad N \text{ odd} \qquad \text{(IV.2.2-1)}$$

Since we have one estimate per block of length N we also have to use this estimate for correction of the decision samples of one block. With a slight modification, the ability of the accumulator filter to cope with time varying signal can be enhanced. With the gliding accumulator we obtain one output value per symbol and, thus, one estimate per symbol. The corresponding equation is given by IV.2.2.-2.

$$y(kT) = \sum_{m=1}^{N} x[kT+ (m-(N+1)/2)T] \quad ; \quad N \text{ odd} \qquad \text{(IV.2.2-2)}$$

Note that for the accumulator (IV.2.2.-1) the estimation error of consecutive blocks is statistically independent when the noise terms in x(kT) are statistically independent. This is not true for the gliding accumulator since each noise term occurs in N consecutive values of y.

Finally, we are not restricted to the use of accumulator filters but we can also use all kinds of FIR or IIR filter.

IV.2.3 Frequency estimation

As we have seen in subsection IV.2.1 we introduce feedback when we want to have a frequency estimate to cope with initial frequency errors. To avoid or reduce the drawbacks of a feedback synchronizer estimation we might use frequency error detection or feedforward frequency estimation. A detailed discussion or performance analysis of frequency estimation is out of the scope of this study. Therefore, we will only give a very brief survey of potential ways. First, we can use frequency error detection methods as discussed for instance in [Gar'88]. Second, we can derive feedforward frequency estimation algorithms analogously to feedforward synchronization algorithms (in general these algorithms will require a significant amount of processing power). Finally, we can implement a postfilter as discussed in subsection IV.2.1 and use the frequency estimate to correct the local oscillator frequency, but neglect the corresponding filtered carrier phase estimate. The phase estimate may be taken from the accumulator filter, for instance.

IV.3 Influence of Phase Noise on Tracking Performance

The difference between the received phase and the local oscillator phase has systematic terms and phase noise. When modelling the influence of phase noise we can assume that the phase noise is added to the oscillator phase at the oscillator output (see Fig. IV.2-2). Therefore, it occurs as an additive phase at the digital synchronizer input. When we denote by $\psi(t)$ the phase noise that we have without feedback to the oscillator input, the received signal is given by

$$x_L(t) = \left[\sum_m a_m h_I(t-mT-\tau) + j \sum_m b_m h_Q(t-mT-\tau) \right] \exp(j(\theta(t)+\psi(t)) + n(t)$$

where $\psi(t)$ by definition is a zero mean random phase noise process and $\theta(t)$ absorbs a frequency difference and difference frequency drift terms. For optimum detection we must track the total phase $\theta(t)+\psi(t)$ so that it can be eliminated in the phase rotator.

When we have a feedback synchronizer with closed loop transfer function $H(\omega)$, the phase noise causes an additive component in the phase estimate with Fourier transform

$$H(\omega)\ \Psi(\omega)$$

where $\Psi(\omega)$ is the spectrum of $\psi(t)$. Since the phase estimate is used to correct the received signal this filtered phase noise is subtracted from the phase noise in the received signal. This is shown in Fig. IV.2-4. Thus, the untracked phase noise equals $[1-H(\omega)]\ \Psi(\omega)$ and, thus, has the power spectral density

$$|1-H(\omega)|^2\ S_{\psi\psi}(\omega)$$

with $S_{\psi\psi}(\omega)$: power spectral density of $\psi(t)$

The variance of the untracked phase noise which corrupts detection then is

$$\sigma^2_{\psi,\ FB} = \frac{1}{2\pi} \int_{-\pi/T}^{+\pi/T} |1-H(\omega)|^2\ S_{\psi\psi}(\omega)\ d\omega \qquad \text{(IV.3-1)}$$

In a feedforward synchronizer the phase is detected, filtered, and the filtered output is used to correct the signal phase prior to decision. This is shown in Fig. IV.2-4. When the (feedforward) filter of the feedforward synchronizer has the transfer function $H(\omega)$, the phase noise is also filtered by $H(\omega)$. Since this filtered phase noise occurs as an additive component in the filtered phase estimate which is used to correct the received signal prior to detection, the spectrum of the remaining phase noise on the samples for detection is as in the feedback synchronizer case $[1-H(\omega)]\ \Psi(\omega)$. Therefore, we have the same phase noise variance

$$\sigma^2_{\psi,\ FF} = \frac{1}{2\pi} \int_{-\pi/T}^{+\pi/T} |1-H(\omega)|^2\ S_{\psi\psi}(\omega)\ d\omega \qquad \text{(IV.3-2)}$$

From the above result it is obvious that we have the same phase noise performance for the feedback synchronizer as for the feedforward synchronizer, when we have a closed loop transfer function in the feedback synchronizer which equals the feedforward filter transfer function.

For instance, for a second order feedback synchronizer as well as for a feedforward synchronizer with accumulator filter $|1-H(\omega)|^2$ is proportional to ω^4 for $\omega \to 0$. For low frequencies the phase noise (see Fig. IV.2-1) is proportional to $1/\omega^4$. Therefore, the integrand (and, thus, the integral) in (IV.3-1) and (IV.3-2) is bounded, i.e. we have a finite contribution from the low frequency components of the phase noise to the total variance of the untracked phase noise $\sigma^2_{\psi,FF}$, $\sigma^2_{\psi,FB}$.

Consequently, in both cases, feedback and feedforward synchronizer, we have the situation that the phase noise is suppressed best when $H(\omega)$ is wideband, i.e. close to 1 also for large ω. On the other hand, thermal noise and self-noise contributions are suppressed best when $H(\omega)$ is narrowband, i.e.

decays for small ω already. A general optimization cannot be carried out, this depends on the actual phase noise spectrum. For a first order feedback synchronizer or a feedforward synchronizer with accumulator filter we only have one parameter for optimization. In this case optimization is obvious. With second order synchronizers the approach discussed in section IV.2.1 gives an insight in the trade-off.

When we compare the dashed approximation of the model used for this optimization approach with the actual phase noise power spectral density we find that the frequency flicker ($\sim 1/f^3$), phase flicker ($\sim 1/f$), and the white phase noise (constant) are not included in the model. While the difference between model and actual spectrum is limited (and can be minimized) for the first two terms, the lack of a constant term in the power spectrum of the model seems to be a severe disadvantage. But in the applications considered in this study the flat phase noise is specified (and designed) to be much smaller than the power spectral density of the additive noise component due to thermal and self-noise. Therefore, the influence of the phase noise on the tracking error power spectral density (and, thus, the tracking error variance) is negligible. Consequently, the actual shape of the phase noise power spectral density for low values has no influence on the optimization and, thus, it is not important to model it correctly. We should also consider that in many applications it is not the oscillator phase noise but the channel that dominates the random (and non-random) carrier phase and frequency fluctuations. This case may also be approximated by the model used in the optimization approach.

In both equations (IV.3-1) and (IV.3-2) we have assumed that the matched filter only cuts off the phase noise spectrum at frequencies $\pm 1/(2T)$. This approximation is only valid when the phase noise power spectral density (see Fig. IV.2-1) is much narrower than the data rate. A second assumption is in eq. (IV.3-2). In the feedforward synchronizers the accumulator filter is followed by a nonlinearity, the argument function. In the accumulator the inphase and the quadrature components are accumulated separately. When the phase noise variation over the accumulation period is small, linearization applies and the accumulator filter can be assumed to operate on the phase noise directly so that (IV.3-2) is valid. Assuming θ = constant and no noise, the accumulator/argument function is described by

$$\arg\left\{\sum_{m=1}^{N} e^{j(\theta+\psi(mT+kMT))}\right\} \tag{IV.3-3}$$

$$= \arg\left\{e^{j(\theta+\psi_k)} \sum_{m=1}^{N} e^{j\Delta\psi(mT+kMT))}\right\}$$

$$= \theta + \psi_k + \arg\left\{\sum_{m=1}^{N} e^{j\Delta\psi(mT+kMT))}\right\}$$

where ψ_k is the mean of the phase noise in the accumulation interval and $\Delta\psi$ is the deviation from the mean. When $\Delta\psi << 1$ (IV.3-3) approximately equals

$$\arg\left\{\sum_{m=1}^{N} e^{j(\theta+\psi(mT+kNT))}\right\} \approx \theta + \psi_k + \frac{1}{N}\sum_{m=1}^{N} \Delta\psi(mT+kNT)$$

$$= \theta + \frac{1}{N}\sum_{m=1}^{N} \psi(mT+kNT)$$

When this is not valid a degradation occurs. The estimate will deviate from the mean of the phase noise in the accumulation period. Since we have absorbed a frequency deviation (i.e. a systematic term, but not frequency fluctuations) in the phase θ, the deviation will have zero mean when averaged over all possible trajectories. The influence of a frequency error will be discussed in the next section.

When the untracked phase noise is not too strong, i.e. the detection error probability is not influenced significantly, we can use the normalized loop noise power spectral density derived without phase noise for calculating the performance of the data aided synchronizers (for the non data aided synchronizers the normalized loop noise power spectral density applies anyway). Then, the performance comparison of the different DD and NDA synchronizer can be based on the results given in the previous parts. Since a synchronizer with a lower loop noise power spectral density allows the use of a wider filter bandwidth, better phase noise tracking is possible.

With digital synchronizers we have the advantage that we may not need a VCO. Then, more stable oscillators may be used which also produce less phase noise. On the other hand we have an additional phase error variance

term due to quantization. As has been discussed in part IV, quantization noise can be kept low at low signal to noise ratios where synchronization is critical and may have a significant influence on detection performance. Phase noise mainly is critical at low signal to noise ratios when $H(\omega)$ must be narrowband for a sufficiently low carrier phase error variance.

So far we have talked about carrier phase noise only. The results which assume that linearization is applicable can be directly used when studying the influence of phase noise in clock synchronization. The nonlinear effects are different because we have a different nonlinearity. Although not purely proportional, (absolute) oscillator stability decreases and phase noise increases with the oscillator frequency. Since the carrier frequency in general is much higher than the data clock frequency, the phase noise effects are much stronger for the carrier phase. Therefore, we can assume that in the type of applications we discuss here, carrier phase noise is the critical point, and timing phase noise can be assumed to have a magnitude which allows linearization.

IV.4 Influence of Frequency Errors on Tracking Performance

IV.4.1 Introduction

As in the previous sections we start with the carrier synchronization which in general has to handle larger frequency deviation than the symbol synchronizer.

IV.4.2 Carrier Synchronization

Feedback Synchronizer

With an initial frequency error we have a steady state phase error in the tracking mode for a first order loop. Such a steady state phase error increases the cycle slip rate and the selfnoise. For large phase errors the selfnoise can be increased significantly. With large frequency errors a second order feedback synchronizer should be used instead which can compensate an initial frequency deviation without steady state phase error. Note that an accumulator (the digital equivalent of an ideal integrator) should be preferably used in the loop filter since then the steady state phase error is zero for any initial frequency deviation.

With a high rate of change of the frequency, we still may have a steady state phase error with a second order loop. But then we should consider the use of a third order loop which can be handled in a digital implementation. For instance, we may use a second order loop during acquisition and then switch to a third order loop during tracking.

Feedforward Synchronizer

The situation is different with feedforward synchronizers. The problem is in the accumulator filter. As discussed in section VII.2, we may want to have a long accumulation period to get a high signal-to-noise ratio at the accumulator output. This avoids degradation due to the argument function nonlinearity following the accumulator. To study the problem we assume in the first step that we do not have modulation. Then, we can directly feed the matched filter output to the accumulator. The phase estimate is given by

$$\hat{\theta} = \arg\left\{ \sum_{m=1}^{N} e^{j(\theta+\Delta\omega mT)} + n(mT) \right\} \tag{IV.4-1}$$

$$= \theta_c + \arg\{ \sum_{m=1}^{N} \exp[j\Delta\omega(m-(N+1)/2)T] + n(mT)\}$$

$$= \theta_c + \arg\{ \sum_{m=1}^{N} \cos[\Delta\omega(m-(N+1)/2)T] + n_I(mT) + j\, n_Q(mT)\}$$

where θ_c is the phase in the center of the accumulation interval (note that sine terms do not occur in the last equation because of pairwise cancellation). Since $E[\hat{\theta}] = \theta_c$, we have the correct phase estimate for the center sample of the accumulation period. Denote the argument term, which corresponds to the estimation error, by Θ_n. Letting

$$\mu = \sum_{m=1}^{N} \cos[\Delta\omega(m-(N+1)/2)T] \quad ; \quad \sigma^2 = N\, E[n_I^2]/\mu^2 \tag{IV.4-2}$$

and assuming that $n_I(mT)$ and $n_Q(mT)$ are statistically independent, zero mean, Gaussian random variables, the probability density funtion of θ_n is obtained by transformation as

$$p(\theta_n) = \frac{1}{2\pi} \exp\left[-\frac{1}{2\sigma^2}\right] + \frac{1}{\sqrt{2\pi\sigma^2}} \cos\theta_n \exp\left[-\frac{\sin\theta_n^2}{2\sigma^2}\right] Q(-\mu\cos\theta_n/\sigma)$$

(IV.4-3)

The left term corresponds to a uniformly distributed component in $[-\pi,\pi)$, the right term approximately equals a Gaussian distribution for $\sigma^2 << 1$, i.e. $\mu^2 >> N\, E[n_I^2]$. When this last condition is violated, the right term deviates from a Gaussian distribution and the uniformly distributed component increases. This corresponds to the degradation caused by the argument nonlinearity. Without frequency deviation $\mu = N$, and we have a minimum value of σ^2 for a given signal-to-noise ratio (see eq. (IV.4-2)).

With a frequency deviation $\mu < N$, σ^2 is increased and the signal-to-noise ratio at the accumulator output is decreased.

When we use an accumulator which outputs one estimate per block of N symbols, we in addition have the problem that this estimate is unbiased for the center of the block only. At the block ends we have a bias of $\pm\Delta\omega T(N-1)/2$ due to the frequency error. The bias degrades the decision error probabilities and so must be kept small. When the accumulator is followed by a postfilter as described in section IV.2.1 we can avoid the bias by calculating a phase estimate for each symbol based on the block phase and frequency error estimate. Alternatively the frequency estimate can be used separately to control the frequency error and prevent the bias. With a gliding accumulator the above problem does not occur since we get a block center estimate for each symbol. But we still need a postprocessing of the estimates to avoid excessive cycle slips (see section VI.4).

NDAFF

We now also consider the modulation. With BPSK the matched filter output is squared in the NDAML feedforward carrier synchronizer. Therefore, the frequency deviation is doubled at the accumulator input. The above analysis still can be used, when $\Delta\omega$ is replaced by $2\Delta\omega$. With QPSK, where the signal is raised to the fourth power before the accumulator filter, we have to replace $\Delta\omega$ by $4\Delta\omega$.

With OQPSK the situation is slightly different. We have to process two samples per symbol time T. With a frequency difference $\Delta\omega$ there is a phase difference $\Delta\omega T/2$ between the p and the corresponding q sample. The phase of the accumulator output is the phase in the center of the accumulator interval. With an odd number of symbols (where one symbol is defined as consisting of the inphase and the quadrature phase bit) this is the middle between the p and q sample of the center symbol of the accumulation period (with an even number of symbols this is the middle between the q sample of the last symbol of the first half and the p sample of the first symbol of the second half of the accumulation period). Because of the phase difference between the p and its corresponding q sample we have to replace each term in the sum of eq. (VII.4-2) by two terms

$$\mu = \sum_{m=1}^{N} \frac{1}{2} \{\cos[\Delta\omega(m- \frac{N+1}{2} + \frac{1}{4})T] + \cos[\Delta\omega(m- \frac{N+1}{2} - \frac{1}{4})T] \}$$

$$\mu \approx \cos\Delta\omega T/4 \sum_{m=1}^{N} \cos[\Delta\omega(m-(N+1)/2)T]$$

Thus, we have a reduction of the signal-to-noise ratio at the accumulator output as compared to the BPSK case by a factor of $\cos^2\Delta\omega T/4$ (which in general will be negligibly small as $\Delta\omega T << 1/N$ to prevent severe bias).

With the VV (Viterbi&Viterbi) carrier synchronizer the modulation is removed by multiplying the phase of the matched filter output appropriately, i.e. by 2 for BPSK and OQPSK and by 4 with QPSK, but allowing different nonlinearities for the amplitude. This makes it possible to increase the signal-to-noise ratio at the accumulator output by using the optimum nonlinearity. For amplitude nonlinearities of the type ρ^{2N} the influence of a frequency error on the phase error variance can be given analytically [Vit'83]. But the corresponding formula is difficult to evaluate.

DDFF

With decision feedback we face an additional difficulty. Now the modulation is removed based on decisions. As discussed previously, the mean of the phase estimate at the accumulator output is the value of the actual phase in the center of the accumulation period. If we do not have a postfilter which uses a frequency estimate to predict the phase (see section IV.2.1 and Fig. IV.2-4), the phase error correction (phase rotation) prior to decision can only be based on the estimate obtained from previous accumulation periods. For example, in the case of a gliding accumulator with accumulation length N, we then have a phase error, due to a frequency error $\Delta\omega$, of $\phi = \Delta\omega T(N-1)/2$ in the prediction. This will cause wrong decisions with a bias in the direction of the phase error. Feeding back these decisions, further increases the phase error.

How can we analyze this easily? Assume that we want to see the influence of a phase error ϕ in the decision samples. Since the actual value of the carrier phase has no influence on the synchronizer performance we can assume that the carrier phase is ϕ. Then, we have a phase error ϕ on the

decision samples when the predicted carrier phase $\tilde{\phi} = 0$, i.e. when we directly feed the matched filter output samples to the decision device (without rotation). To be unbiased, the mean of the DDFF carrier phase estimate should be ϕ, i.e. when we display the mean as a function of ϕ we should have a unity slope sawtooth characteristic. In Fig. IV.4-1 we have shown the mean of the DDMLFF carrier phase estimate as a function of phase ϕ when the decision samples are not corrected. We find that the mean of the estimate is smaller than the actual carrier phase (i.e. we have a bias) except for the noise free case.

Therefore, with decision directed synchronizers we need a frequency estimate on which a prediction of the phase error with respect to the phase in the center of the accumulation period can be based or which is used to correct the oscillator frequency so that the error is sufficiently small.

When we consider modulation and matched filtering, we must also take into account that for a large frequency deviation the received signal will be severely distorted when assuming root raised cosine matched filtering with a rolloff between 0 and 1. The signal no longer is in the right position with respect to the filter centre frequency, the decision error probability is increased strongly. This could only be avoided by having a separate demodulation path where the received signal is corrected before matched filtering. We will not consider such an approach here.

IV.4.3 Symbol synchronization

Feedback Synchronizer

With feedback symbol synchronizers we have the same situation as with the feedback carrier synchronizers. By appropriate system design the influence of a symbol clock frequency deviation can be avoided in the tracking mode. Also, the influence of e.g. Doppler is much smaller for symbol synchronization than for carrier synchronization.

Feedforward Synchronizer

Because of smaller Doppler frequency we can make the accumulation period in feedforward synchronizers in general long enough to avoid a performance degradation due to the maximum selection nonlinearity ($\hat{\tau}$: $\max\{\cdot\}$) without major degradation by clock frequency deviation. In principle, we have the same effects as discussed with the feedforward carrier synchronizers except that the frequency doubling/quadrupling does not occur.

IV.4.4 Burst transmission

When we talk about tracking performance we usually assume that we have a continuous transmission. In burst transmission we have the situation that the frequency error can be kept small from burst to burst but the carrier phase at the beginning of each burst cannot be derived from the previous burst, i.e. it is unknown. For the timing we assume that the symbol timing is known at the beginning of the burst (i.e. it can be derived from the previous bursts) which is the case in most burst transmission systems. When this is not the case the carrier phase estimation below must be replaced a joint carrier/symbol timing estimation.

For very long burst lengths it might be reasonable to update the frequency estimate during the burst, i.e. use intra-burst postfiltering. Then we have acquisition at the beginning of the burst followed by a tracking mode as in the continuous transmission. Here we will restrict our attention to the case where we can assume a constant but unknown frequency error during each burst. In this case we can use two types of filter, the accumulator (AC) or the gliding accumulator (GL). With the accumulator filter we obtain one phase estimate per block which is unbiased for the centre of the burst and has a bias of $\pm\Delta fT(N-1)/2$ at burst beginning and end. To avoid demodulation performance loss the maximum bias must be small, i.e. the residual frequency error must not be too large $\Delta fTN << 1$. When the frequency variations from burst to burst are too strong to provide a sufficiently precise frequency prediction, a better alternative is a gliding accumulator. Then we have an unbiased phase estimate for each symbol at the cost of a reduced accumulator length and, thus, higher variance at burst beginning and end. Acquisition is discussed in section V.3. Note that with a gliding accumulator we introduce a relation between consecutive phase estimates. In this case we have to extend the basic range of the phase estimate to avoid phase reference jumps of $\pm 2\pi/M$ (which causes symbol errors with differential decoding or loss of ambiguity resolution), see section VI.4.

APPENDIX IV.A.1 KALMAN FILTER APPROACH

Starting with the model (IV.2.1-1) discussed in section IV.2.1

$$\theta_0((k+1)NT) = \Omega_0(kNT) + \theta_0(kNT) + w_\theta(kNT)$$
$$\Omega_0((k+1)NT) = \Omega_0(kNT) + w_\Omega(kNT) \qquad \text{(IV.A.1-1)}$$

the noise terms w_θ and w_Ω are assumed to be zero mean independent Gaussian variables with the variances

$$E[w_\theta^2] = \sigma_\theta^2$$
$$E[w_\Omega^2] = \sigma_\Omega^2 \qquad \text{(IV.A.1-2)}$$

The output of the accumulator filter can be described by

$$\hat{\theta}(kNT) = \theta_0(kNT) + n_\theta(kNT) \qquad \text{(IV.A.1-3)}$$

where the loop noise term n_θ has the variance

$$E[n_\theta^2] = \sigma_M^2 \qquad \text{(IV.A.1-4)}$$

Model equation (IV.A.1-1) and measurement equation (IV.A.1-3) can be described by the following vector/matrix equation

$$x_{k+1} = F\, x_k + G\, w_k$$
$$z_k = H^T x_k + v_k \qquad \text{(IV.A.1-5)}$$

where

$$x_k = \begin{bmatrix} \theta_0(kNT) \\ \Omega_0(kNT) \end{bmatrix} \quad ; \quad w_k = \begin{bmatrix} w_\theta(kNT) \\ w_\Omega(kNT) \end{bmatrix}$$
$$v_k = n_\theta(kNT) \quad ; \quad z_k = \hat{\theta}(kNT) \qquad \text{(IV.A.1-6)}$$

The system matrices F, G, H^T are

$$F = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \quad ; \quad G = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$H^T = [\ 1 \quad 0\ ] \tag{IV.A.1-7}$$

Further defining

$$Q = E[w_i w_i^T] = \begin{bmatrix} \sigma_\theta^2 & 0 \\ 0 & \sigma_\Omega^2 \end{bmatrix} \quad ; \quad R = \sigma_N^2 \tag{IV.A.1-8}$$

the recursion for the prediction $\tilde{x}_{k+1|k}$ is

$$\tilde{x}_{k+1|k} = \tilde{x}_{k|k-1} + K_k\ (z_k - H^T \tilde{x}_{k|k-1}) \tag{IV.A.1-9}$$

The filtered estimates can be determined from

$$\tilde{x}_{k|k} = \tilde{x}_{k|k-1} + \Sigma_{k|k-1}\ H\ (H^T\ \Sigma_{k|k-1}\ H + R\ )^{-1} \tag{IV.A.1-10}$$

with the initial condition

$$\tilde{x}_{0|-1} = E[x_0] = E\left[\begin{bmatrix} \theta_0(0) \\ \Omega_0(0) \end{bmatrix} \right] \tag{IV.A.1-11}$$

The variable $\tilde{x}_{k|k-1}$ is the predicted state

$$\tilde{x}_{k|k-1} = \begin{bmatrix} \tilde{\theta}(k|k-1) \\ \tilde{\Omega}(k|k-1) \end{bmatrix} \tag{IV.A.1-12}$$

With this definition (IV.A.1-9) is the vector notation of the two scalar equations (IV.2.1-4).

Note that when F^{-1} exists (IV.A.1-10) can be written

$$\tilde{x}_{k|k} = \tilde{x}_{k|k-1} + F^{-1} K_k (z_k - H^T F \tilde{x}_{k-1|k-1}) \qquad \text{(IV.A.1-13)}$$
$$= F^{-1} \tilde{x}_{k+1|k}$$

The matrix K_k is obtained from

$$K_K = F \Sigma_{k|k-1} H (H^T \Sigma_{k|k-1} H + R)^{-1} \qquad \text{(IV.A.1-14)}$$

where the matrix $\Sigma_{k|k-1}$ is defined recursively

$$\Sigma_{k+1|k} = F \left[\Sigma_{k|k-1} - \Sigma_{k|k-1} H (H^T \Sigma_{k|k-1} H + R)^{-1} H^T \Sigma_{k|k-1}\right] F^T +$$
$$+ GQG^T \qquad \text{(IV.A.1-15)}$$

with the initial condition

$$\Sigma_{0|-1} = E[(x_0 - E(x_0))(x_0 - E(x_0))^T] \qquad \text{(IV.A.1-16)}$$

From the above equations we obtain the equations (IV.2.1-6) and the recursions (IV.2.1-7) when we again calculate the corresponding scalar equations from the above vector equations.

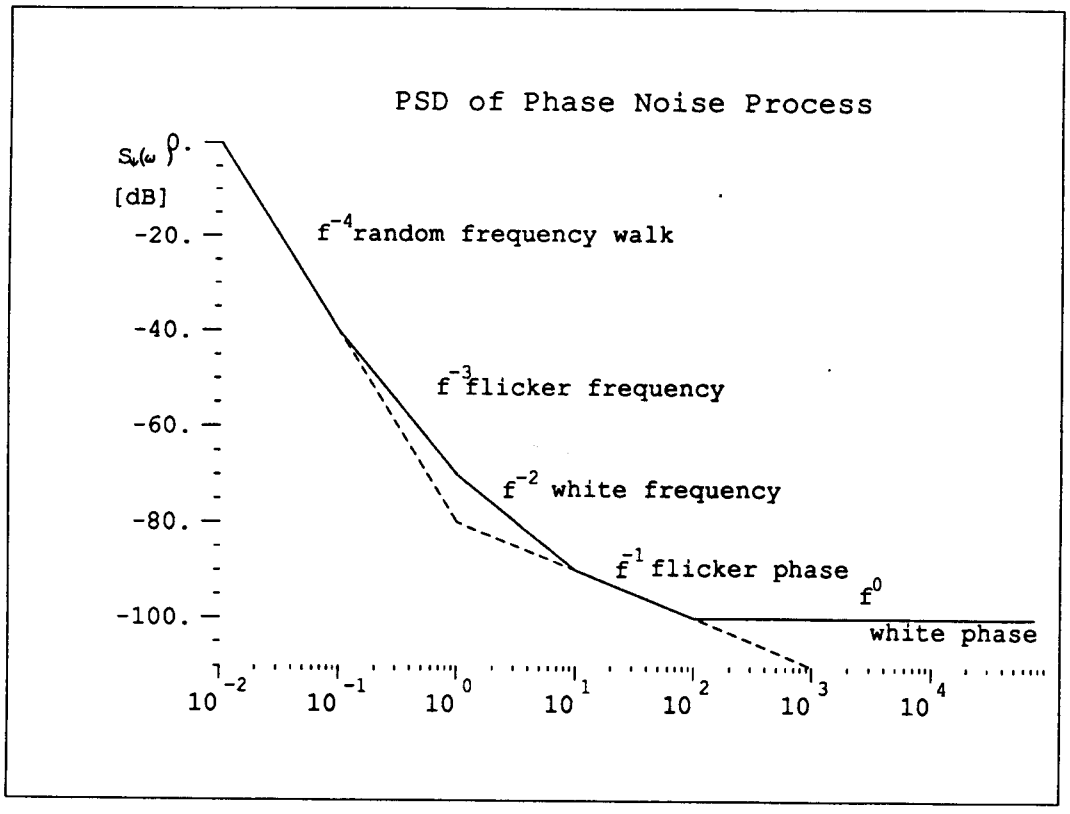


Fig. IV.2-1: Basic components of oscillator phase noise power spectral density (dashed line: approximation by second order random walk model)

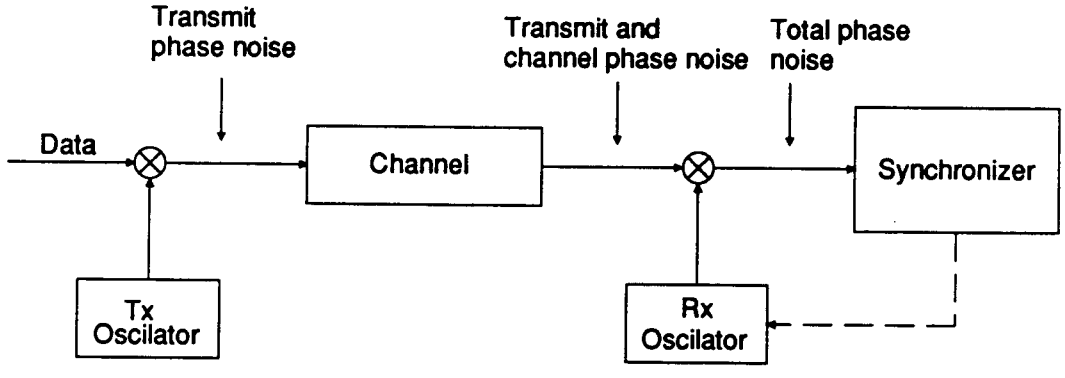


Fig. IV.2-2: Phase noise contributions in a transmission link

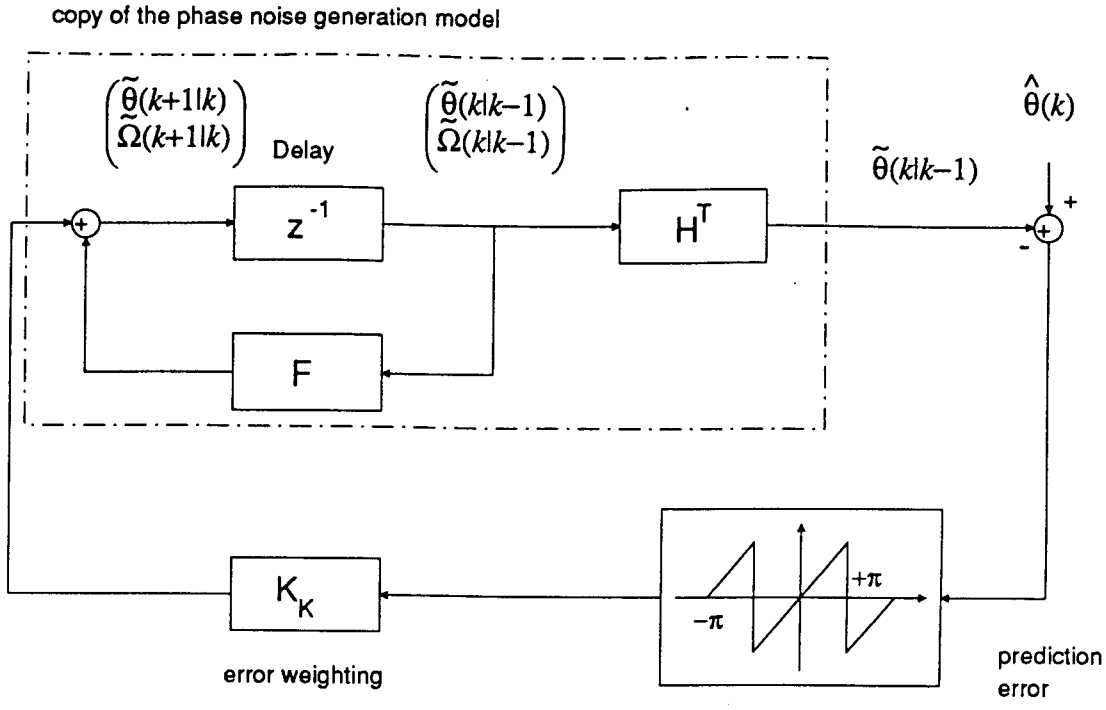


Fig. IV.2-3: Basic structure of the postfilter (z^{-1}: delay), system matrices:

$$F = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}, H^T \; [\,1\;0\,]$$

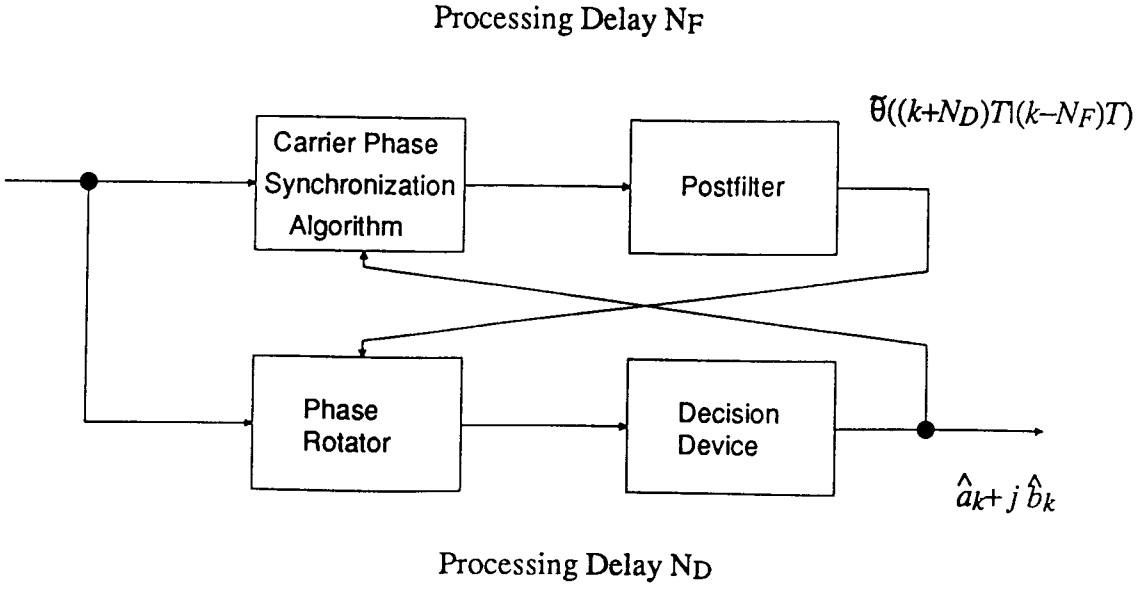


Fig. IV.2-4: Influence of processing delay in a decision directed synchronizer

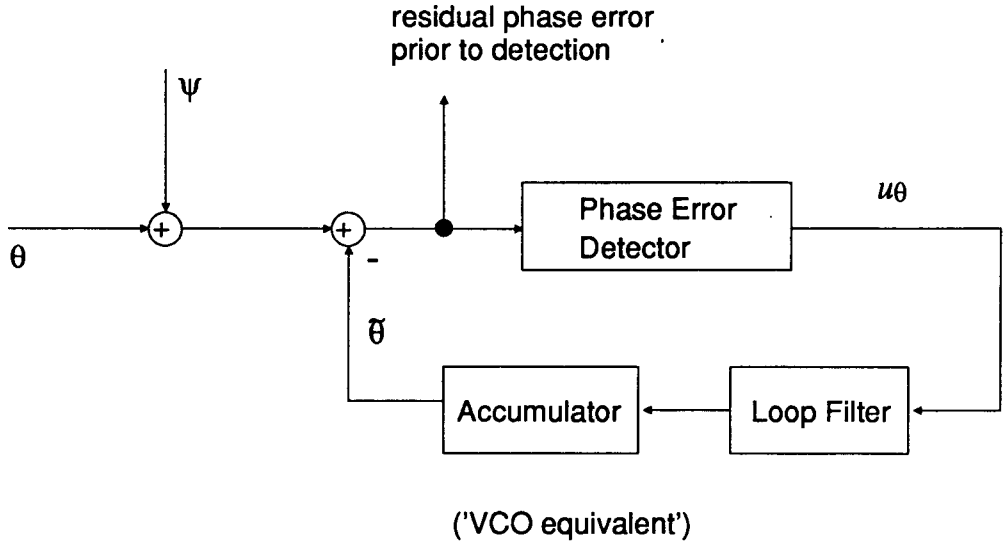


Fig. IV.3-1: Equivalent blockdiagram for the feedback carrier phase synchronizer

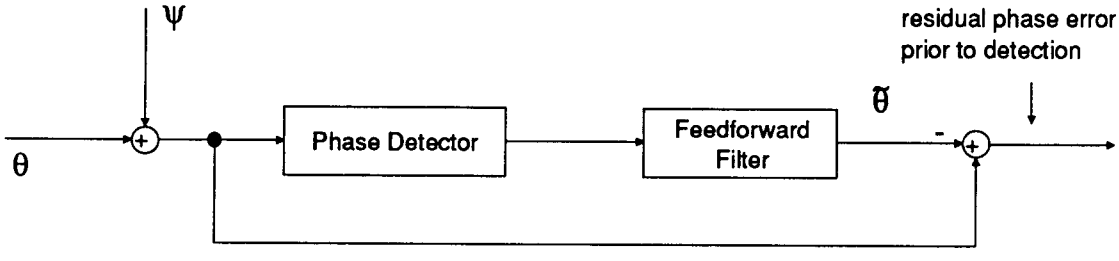


Fig. IV.3-2: Equivalent blockdiagram for the feedforward carrier phase synchronizer

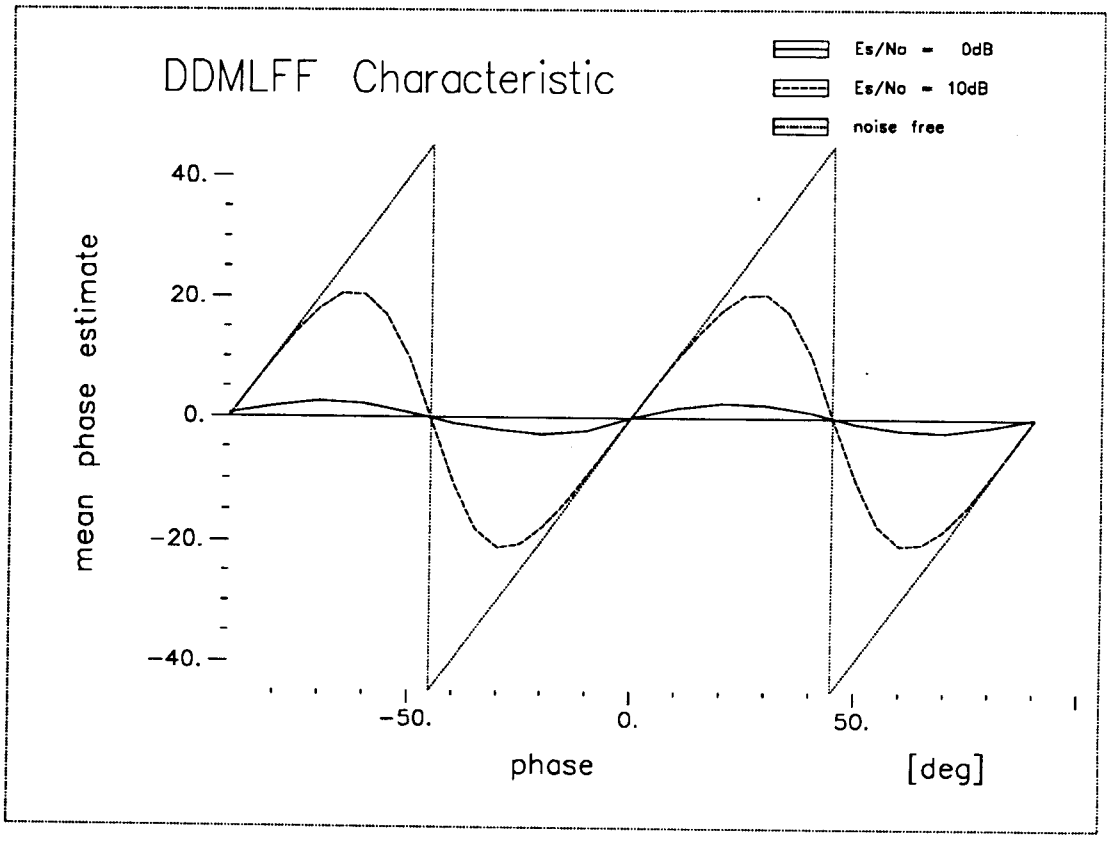


Fig. IV.4-1: Mean of phase estimate as a function of phase for QPSK DDMLFF carrier synchronization algorithm (decision samples not corrected), simulation

V. ACQUISITION

V.1 Introduction

In this part we will qualitatively and, as far as possible, quantitatively compare the acquisition performance of digital carrier and clock synchronizers.

Acquisition can be defined as the transition from some initial state to steady state performance (i.e. the tracking mode in a feedback synchronizer). Thus, acquisition can only occur in systems which have memory (or, equivalently a state). This memory is introduced by loop filter and VCO in feedback synchronizers and by the postfilter in feedforward synchronizers. Since we need a carrier phase estimate and a timing error estimate for correction prior to decision in DD synchronizers, decision feedback inherently also introduces memory.

Feedforward type synchronizers are in general used to overcome some of the acquisition problems occuring in feedback synchronizers. Therefore, we start with the discussion of the acquisition parameters and effects of feedback synchronizers so that we can refer to this part when discussing the feedforward synchronizer acquisition effects and parameters.

V.1.1 Feedback Synchronizer

We will restrict our attention to first and second order loops here. Third order loops are known to have critical acquisition performance [Mey'90], [Gar'79] and, therefore, are rarely used during acquisition. When we restrict our attention to second order loops with integrating loop filter (or, to be more precise, filters having a discrete time equivalent of an integrator) the state of the loop is described by the phase error and the integrator output level, which corresponds to the frequency estimate. Note that the VCO frequency does not correspond to a state in the strict sense. On the other hand, the frequency error is related more closely to the specification of analog front end performance. Therefore, in material on synchronizer acquisition performance we often find trajectories for acquisition displayed in a phase error versus frequency error plane ('phase plane'). Note that when

we are talking about VCOs this not only refers to the analog VCO but also to its digital equivalent.

In first order loops we only have one state variable, the phase error (since we do not have an integrator in the loop filter in this case). This also means that we have fewer parameters and effects influencing acquisition.

Phase (or Timing Error) Acquisition

Let us first assume that the initial frequency error is zero. In continuous mode data transmission this is not a very likely situation. With burst mode transmission this may be different. The frequency of the received signal and of the down converting oscillators in the receiver will not drift rapidly. Therefore, with a second order loop the frequency error at the beginning of a burst will be close to zero when the frequency error estimate at the end of the previous burst is close to the correct value. In such a case, we only have phase acquisition. When the initial phase error is not too large, linearization applies and we can use the closed loop transfer function to determe acquisition performance. When the phase error is large, we have to take the phase detector characteristic into account. The resulting non-linear difference (differential) equation can only be solved in some special (noise free) cases. When we also take noise into account, Fokker-Planck techniques have to be used and analysis become tedious [Mey'80]. On the other hand, when the loop signal-to-noise ratio is high, the influence of the noise on acquisition time becomes negligible. We can study acquisition performance based on the variables determined for the noise free case (except that we have to take the influence of the phase detector characteristic into account). In such a case it is the shape of the detector characteristic which determines how fast the phase acquisition proceeds.

A stable operating point corresponds to a zero-crossing of the phase detector characteristic with positive slope. Since any synchronizer characteristic must be periodic at least with 2π, we must also have a zero-crossing with negative slope between two stable operating points. This is an unstable tracking point. Now, under the assumption of zero frequency error we have a critical situation when the initial phase error is close to this unstable tracking point. In the vicinity of this unstable tracking point the phase detector output is close to zero and, thus, we only have a small or even no signal driving the phase error towards a stable equilibrium point. It can take

a very long time before the system leaves the unstable tracking point. This phenomenon is called **'hang-up'**. It is an unacceptable situation for burst transmission where rapid acquisition is required. From the above discussion it is clear that any feedback synchronizer suffers from hang-up.

Frequency Acquisition

In the more general case we initially have a frequency error, i.e. a difference between carrier frequency and the rest frequency of the VCO (or of its digital equivalent). Most of the feedback carrier and symbol synchronization algorithms operate on a 'one estimate per symbol' basis. This limits the maximum frequency error to $1/(2\Delta T)$. Since we only consider synchronizers preceeded by a (narrowband) matched filter, the range is even narrower.

First order loop: With a first order loop the deviation of the VCO frequency from its rest frequency is proportional to the phase detector output. In case of an initial frequency error the phase will increase (or decrease) permanently with time. When the phase error corresponds to the value for which the phase detector characteristic takes on its maximum value, the VCO frequency will also take on its maximum value. This maximum frequency value and the corresponding minimum frequency value determine the input frequency range in which acquisition is possible with a first order synchronizer. Since there is no further state involved, frequency acquisition will occur without cycle slipping. This frequency range is referred to as the lock-in range. Note that the frequency range for which acquisition is possible is also the frequency range for which the system remains in lock. The latter range is usually referred to as the hold-in range. Thus, for a first order loop hold-in range and lock-in range are identical.

Second order loop: For a second order loop the situation is different. When we have a frequency error, the phase error is an increasing or decreasing function with time. Thus, we will have a time varying phase detector output which modulates the VCO. Without feedback to the VCO this signal would have the shape of the phase detector characteristic and be periodic. Due to the modulation of the VCO the signal no longer is periodic and contains a dc component (non zero mean) which drives the VCO in the correct direction. Assume that the VCO frequency is too small, i.e. we have a phase error which is an increasing function with time. When the phase error is close to the value where the phase detector characteristic takes on its

maximum value we have the maximum VCO frequency and, thus, the smallest frequency error. This means that the rate of change of the phase takes on its lowest value. When the phase error is close to the value where the phase detector characteristic takes on its minimum value, we have the lowest VCO frequency and, thus, the largest frequency error. This means the phase detector output remains longer in the vicinity of the maximum than in the vicinity of the minimum, over each complete cycle we have a dc component (or non zero mean). This effect is shown in Fig. V.1-1 for a phase locked loop. The dc component is integrated in the loop filter integrator so that after each cycle the VCO has come closer to the correct frequency. It is easily seen, that this dc term is strongest for small initial frequency errors and decreases with an increasing initial frequency error but is never zero. That means, from a theoretical point of view we have an infinite initial frequency error range where the system can go into lock. Practically, this range is limited by unavoidable dc offsets and quantization effects. This range is usually referred to as the pull-in range. The hold-in range of a second order loop is not limited by the synchronization algorithm used but only by the numerical range of the integrator.

Another interesting parameter which is related to acquisition is the pull-out frequency. This is the maximum frequency step that can be applied to the synchronizer without leading to cycle slips.

Acquisition Time - Phase Acquisition

When we linearize the phase detector characteristic arround the stable tracking point we have a phase error range where the deviation between linear approximation and actual characteristic is small. Within this range we can use linear approximation to determine phase acquisition performance. The acquisition time depends on loop bandwidth and loop damping. Precalculated figures are available also in publications [Bla'76]. For the DD synchronizer we have in the noise free case a sawtooth characteristic, therefore, the linear approximation results are correct for all phase errors except $\phi = 180^{o}$. In all other cases we have a phase error range where, after a maximum, we have a negative slope. Note that for low E_S/N_o all phase detector characteristics tend to become sinusoidal.

Although acquisition time depends on the initial phase error, loop damping, and loop noise bandwidth, the order of magnitude is given by the inverse of the loop noise bandwidth ($1/B_L$). The only exception is a small range around the zero crossing at the unstable tracking point. Here we have a very small restoring force so that we may get an extremly extended acquisition time, i.e. we may have hang-up in the direct vicinity of the zero crossing.

Frequency Acquisition without Cycle Slips

For a first order loop we only can have frequency acquisition without cycle slips. In the second order loop there is a frequency error range where we have acquisition without cycle slips (note that this range has no strict boundary because it depends on the initial phase error). Again, the acquisition times are in the order of magnitude of $1/B_L$.

Frequency Acquisition - Second Order Loops

When we want to study the frequency acquisition for large frequency errors it is helpful to observe the behaviour of a first order phase locked loop, when a frequency error is too large for the synchronizer to lock. As discussed above, in this case we observe a periodic oscillation at the phase error detector output (beat-note), see Fig. V.1-1. This oscillation has a dc term. In a second order loop we also observe oscillations at the phase detector output, but now the dc term is integrated in the loop filter. Thus, the frequency error is reduced until we finally have lock-in. Based on this observation a good approximation of the frequency error reduction rate can be derived. While the integrator in the loop filter is responsible for the long term frequency error reduction we can neglect its influence on the short term behaviour, i.e. during one period of the phase error detector output oscillations. This means we approximate the short term behaviour of the second order loop by a first order loop. For the first order loop we can calculate mean and duration of one period of the beat-note. From mean and duration we can calculate the loop filter integrator dc increment per beat note which is proportional to the frequency error reduction per beat note [Mey'90]. When we denote the frequency error by Δf we obtain for a phase locked loop with sinusoidal phase detector characteristic (period 2π)

$$\frac{d\Delta f}{dt} = - \frac{2}{\pi^2\zeta^2\left[1 + \frac{1}{4\zeta^2}\right]^3} \frac{B_L^2}{\Delta f/B_L} \tag{V.1-1}$$

or, equivalently

$$\frac{d(\Delta f/B_L)}{d(t/T)} = - \frac{2}{\pi^2\zeta^2\left[1 + \frac{1}{4\zeta^2}\right]^3} \frac{B_L T}{\Delta f/B_L} \tag{V.1-2}$$

From (V.1-1) the acquisition time T_{acq} is readily obtained by integration

$$\frac{T_{acq}}{T} = \frac{\pi^2\zeta^2\left[1 + \frac{1}{4\zeta^2}\right]^3}{4B_L T} \left[\frac{\Delta f}{B_L}\right]^2 \tag{V.1-3}$$

All phase (and timing error detector) characteristics considered in this study have the maximum of their derivative for zero error. Therefore, they are "upper bounded" by a sawtooth characteristic having the same slope for zero error and the same period.

With the same approach as for the sinusoidal characteristic given above we obtain for a sawtooth characteristic with period 2π

$$\frac{d(\Delta f/B_L)}{d(t/T)} = - \frac{4}{3\zeta^2\left[1 + \frac{1}{4\zeta^2}\right]^2} \frac{B_L T}{\Delta f/B_L} \tag{V.1-4}$$

By integration we again obtain the acquisition time T_{acq}

$$\frac{T_{acq}}{T} = \frac{3\zeta^2\left[1 + \frac{1}{4\zeta^2}\right]^2}{8B_L T} \left[\frac{\Delta f}{B_L}\right]^2 \tag{V.1-5}$$

We observe that the frequency error reduction rate decreases with increasing frequency error. The acquisition time increases proportional to the square of

the frequency error. Therefore, when large frequency errors may occur, aided acquisition must be used. For comparison we may briefly discuss sweep acquisition. For loop signal-to-noise ratios above 10 dB the maximum sweep rate is limited by

$$\left|\frac{d(\Delta f/B_L)}{d(t/T)}\right| \leq \frac{1}{1.25\pi\zeta^2\left[1 + \frac{1}{4\zeta^2}\right]^2} B_L T \qquad \text{(V.1-6)}$$

to ensure acquisition. Now we have a constant frequency error reduction rate independent of the maximum frequency error. The acquisition time is given by

$$\frac{T_{acq}}{T} = \frac{5\pi\zeta^2\left[1 + \frac{1}{4\zeta^2}\right]^2}{4B_L T} \frac{\Delta f}{B_L} \qquad \text{(V.1-7)}$$

General Remarks on Acquisition Parameters

For the first order loop the extrema of the phase detector characteristic determine the hold-in range and lock-in range. For the second order loop the comparison is more difficult. In this case not only the extrema determine the performance, but the complete characteristic. When the absolute value of a characteristic is larger than the absolute value of another characteristic for all phase errors, we have stronger pull-in effects, i.e. shorter acquisition time and, for given DC offset levels, a larger pull-in range. The hold-in range is infinite independent of the characteristic.

A 'fair' comparison of feedback synchronization algorithms is difficult, though. For a given filter and VCO we must multiply the phase detector output of different synchronization algorithms such that we have the same slope at $\phi = 0$ to obtain the same closed loop tracking performance. This means, when we want to compare different synchronization algorithms we have to normalize the different phase detector characteristics so that they have the same slope at $\phi = 0$. On the other hand, the loop signal-to-noise ratio also depends on the power spectral density of the loop noise of the different algorithms. For identical loop bandwidth we may have a lower loop signal-to-noise ratio with one synchronizer than with another. Therefore, we could for a given loop signal-to-noise ratio select a larger bandwidth. In this

case comparison with normalized slope would not be correct. Further difficulties in comparison arise from the fact that the phase detector characteristics of most of the synchronizers depend on E_s/N_0. Finally, we have assumed an infinite loop signal-to-noise ratio. For low loop signal-to-noise ratios we have a probability that the loop is in lock at each moment in time rather than a dedicated acquisition time.

V.1.2 Feedforward Synchronizer

With a feedforward synchronizer we do not necessarily have acquisition effects. As discussed with the feedback synchronizers acquisition effects occur only when we have memory in the system. Since we inherently have memory with decision directed synchronization algorithms, in this case we will also have acquisition effects. With NDA synchronization algorithms and an accumulator filter which outputs one sample per block we do not have any acquisition. With a different postfilter, e.g. the Kalman filter discussed in section VII.2.1 or a gliding accumulator, we have acquisition effects.

Acquisition performance therefore mainly depends on the type of synchronization algorithm, NDA or DD, and on the postfilter. In the case of NDA synchronization algorithms the postfilter has an influence on the feedforward synchronization algorithm performance only when we feedback a frequency estimate to a local mixer oscillator (or the sampling clock generator for timing synchronization). In the case of DD synchronization algorithms we always have an interaction (i.e. an influence in both directions) between algorithm and postfilter since the postfilter has to provide a prediction of phase or timing for correction of the samples used for decision. This decisions then are used in the synchronization algorithm to determine further estimates of phase or timing.

Note that when we talk about local mixing oscillators or sampling clock generators this also includes their digital equivalents.

We will introduce abbrevations in the subsequent sections to ease the discussion of the different scenarious. The first four letters refer to the acquisition scenario (PAF0, PANF, FADC, FAFC, see below) and the following two letters to the filter type (AC: accumulator only, GL: gliding accumulator, KF: Kalman filter, FE: separate frequency estimation).

Phase Acquisition (PAF0)

Again we may assume that the frequency error is negligible and we only have to determine the phase (or timing error). For example, in a burst transmission system we may have a precise carrier frequency estimate from previous bursts but the phase is unknown at the beginning of the burst.

Phase Acquisition without Frequency Estimation (PANF)

We may also have the situation that we have a frequency error but do not intend to determine a frequency estimate for correction or postfiltering. This excludes a second (or higher) order (Kalman) postfilter from consideration. Then we have phase acquisition under a given frequency error. There is no equivalent case for the feedback synchronization algorithms.

Frequency Acquisition (FADC, FAFC)

When we want to estimate an initial frequency error for correction of the local mixer oscillator frequency (or the sampling frequency for timing synchronization) or to improve the estimate used for correction prior to detection we cannot use an (gliding) accumulator only. Either we must have an additional postfilter, for instance, the Kalman filter discussed in section VII.2.1, or we must use an additional (feedforward) frequency estimation algorithm.

When we correct the oscillator frequency (FAFC), the synchronization algorithm operates without frequency error after acquisition. Since sufficiently small frequency errors have a negligible influence on tracking performance we may use a quantized feedback, i.e. the oscillator frequency is adjusted only when the frequency error exceeds a certain limit. Another approach may be to use a frequency estimate to improve the filtered estimate of phase or timing error used for correction of the decision samples (before demodulation and/or decision feedback) only but to leave the oscillator frequencies uncorrected (FADC). In the case of NDA synchronizers we then have the full influence of the frequency error on the feedforward carrier phase/symbol timing estimation algorithm but no feedback (and, thus, no influence) from the postfilter on the feedforward carrier phase/symbol timing estimation algorithm (performance). In the case of DD synchronizers the initial frequency error will have no influence on the decisions in the tracking mode

but still influence the feedforward carrier phase/symbol timing estimation algorithm (even when it is based on the enhanced decisions).

V.2 Feedback Synchronizer Acquisition Performance

In this section we will concentrate on first order loops or loops having the structure of the Kalman filter introduced in section VII.2.1. As discussed in that section, the Kalman filter in the tracking mode corresponds to a second order loop with integrating loop filter. This filter structure can easily be used in a more generalized sense. This means we need not necessarily use the time varying coefficients obtained by the optimality approach used in section VII.2.1. Instead we could use a coefficient sequence optimized heuristically or determined by simulations of a particular system, i.e. we still use the structure but not the recursions for the K-matrix. A useful example is the two-bandwidth synchronizer which starts with a coefficient set corresponding to a large bandwidth and switches to a coefficient set corresponding to a small bandwidth after lock detection. What is the advantage of such a time variant postfilter approach? The acquisition time strongly decreases with increasing loop noise bandwidth. Therefore, acquisition can be speeded up significantly by using a larger loop noise bandwidth during acquisition at the cost of stronger influence of noise terms. This stronger noise can often be tolerated during acquisition since the loop can often operate at a lower loop signal-to-noise ratio than required in the tracking mode where more stringent cycle skipping and error variance specifications must be met (and, therefore, a narrower loop noise bandwidth is required).

First Order Loops

We will not study the acquisition time of first order loops separately for the individual synchronizer types but rather summarize some basic properties (which partly also have been discussed in the previous section). Since we do not have frequency acquisition with cycle slipping as in a second order loop, acquisition performance is close to the results obtained by linearizing the error detector characteristic (in this case the phase acquisition simply is the step response of a first order lowpass with cutoff frequency $\omega_g=4B_L$). As a rule of thumb, the acquisition time is in the order of magnitude of $1/B_L$. This rule does not apply for large initial frequency errors (which are not acceptable for first order loops anyway since they cause large tracking errors) and it does not apply when the initial phase (or timing) error is close to the unstable null of the error detector characteristic and the initial frequency error is approximately zero (where we get hang-up).

Second Order Loops

When the initial frequency error is approximately zero ('phase acquisition') the acquisition time is in the same order of magnitude as for the first order loop and we also may have hang-up. Frequency acquisition times of the individual synchronizer types will be studied in more detail in the following subsections. For a filter with time varying coefficients (and thus time varying bandwidth) it is difficult to give precise acquisition time figures. When the bandwidth is adapted slowly w.r.t. the loop time constants, we may calculate the worst case acquisition time via the frequency error reduction obtained for each coefficient set (or, equivalently, loop noise bandwidth).

V.2.1 Feedback Carrier Synchronization

NDAFB Carrier Synchronization

With the NDAMLFB we have a sinusoidal phase detector characteristic. Therefore, we have a similar acquisition behaviour as for a phase locked loop (with time varying loop bandwidth), with one exception. Due to the squaring (for BPSK and OQPSK) a frequency error $\Delta\omega$ appears with double frequency $2\Delta\omega$ at the detector output. To obtain the acquisition performance figures for fixed bandwidth we simply have to replace Δf by $2\Delta f$ in eq. (V.1-1) - (V.1-3), i.e. the frequency reduction rate is 1/4 of the rate given in (V.1-2) and the acquisition time is 4 times the value given in (V.1-3). Similarly, for QPSK due to quadrupling we have $4\Delta\omega$ at the detector output. Therefore, we have to introduce a factor of 1/16 in the right hand side of (V.1-2) and a factor of 16 in the right hand side of (V.1-3).

Note that the frequency error reduction rate is 1/4 for BPSK (1/16 for QPSK) since we have to replace Δf by $2\Delta f$ ($4\Delta f$ for QPSK) in **left** and **right** hand side of eq. (V.1-1) or (V.1-2).

Data Aided Feedback Carrier Synchronization

When a preamble is available for acquisition, we may use this known preamble rather than the decisions in a DDMLFB synchronizer. We then have a sinusoidal phase detector characteristic with period of 2π. Note that frequency duplication or quadrupling does not occur. Therefore, (V.1-2) and (V.1-3) directly apply.

DDFB Carrier Synchronization

With decision directed carrier synchronization we have in the high E_S/N_0 limit (i.e. when we neglect decision errors) a phase detector characteristic which is a fraction of a sinusoid. The range of the fraction is $[-\pi/2, +\pi/2]$ for BPSK and OQPSK and $[-\pi/4, +\pi/4]$ for QPSK. The reduced range occurs since the decision device selects the 'nearest' symbol, thus, reducing the phase error inherently to $\pi/2$ for BPSK, OQPSK and to $\pi/4$ for QPSK.

For BPSK and OQPSK we have the same dc term in the phase detector output (with the first order loop approximation used to derive (V.1-2)) as for the basic phase locked loop with sinusoidal characteristic. Therefore, the frequency reduction rate is given by (V.1-2) and the acquisition time by (V.1-3) in the high E_S/N_0 limit.

It is difficult to determine the acquisition time for QPSK using the actual noise free phase detector characteristic. Therefore, we approximate the phase detector characteristic by a sawtooth (with a maximum error of 11% at $\pi/4$) with period $\pi/2$ for the high E_S/N_0 limit. With this approximation we can calculate the acquisition performance based on the results for a sawtooth characteristic with period 2π (eqs. (V.1-4), (V.1-5)) Because of the reduced period we have to modify these equations by introducing a factor of 1/16 (see NDAFB) in the right hand side of (V.1-4). Similarly, the acquisition time is obtained by introducing a factor of 16 in the right hand side of (V.1-5).

For lower signal-to-noise ratios the phase detector characteristic will deviate from the fractionally sinusoidal shape and approach a truely sinusoidal shape with period π (BPSK, OQPSK) or $\pi/2$ (QPSK) in the low E_S/N_0 limit due to decision errors. For QPSK this is shown in Fig. V.2-1. Because of the period of $2\pi/2$ (BPSK, OQPSK) or $2\pi/4$ (QPSK) a frequency error $\Delta\omega$ appears with a frequency of $2\Delta\omega$ (for BPSK and QPSK) or $4\Delta\omega$ (for QPSK) at the phase detector output. Therefore, in the low E_S/N_0 limit we have the same equations (and, thus, performance) as for the NDAML synchronizer.

In Fig. V.2-2a,b the acquisition of a DDFB and a NDAFB synchronizer are shown together with the analytical approximations of acquisition time and frequency error reduction rate discussed above (BPSK, $B_LT=0.005$,

$\Delta f(0)=3B_L$, $\zeta=0.707$ (a) and $\zeta=2.5$ (b), E_S/N_0 = 50dB). We find an excellent agreement for the NDAFB with a large loop damping factor ($\zeta=2.5$) as well as with a small loop damping factor ($\zeta=0.7$). The DDFB acquisition is at the high E_S/N_0 limit for E_S/N_0 = 50dB and slightly below the high E_S/N_0 performance for E_S/N_0 =10dB, as expected. This is due to the fact that the phase error detector characteristic changes with E_S/N_0. For the NDAFB the phase error detector characteristic and thus the acquisition performance is independent of E_S/N_0 as long as the loop signal-to-noise ratio is sufficiently high.

V.2.2. Feedback Symbol Synchronization

As for carrier synchronization we will consider a postfilter having the Kalman-filter structure. The comments on the modification of the filter as discussed for the carrier apply for symbol synchronization as well. With symbol synchronization doubling or quadrupling of a frequency error does not occur. The period of the phase detector characteristic is 1 (symbol period) in all cases. The individual acquisition performance depends on the timing error detector characteristic and the loop noise power spectral density as discussed in the introduction. In Fig. V.2-3a the timing error detector characteristic of the GA for $\alpha=0.5$ is shown as an example for a NDA algorithm characteristic and in Fig. V.2-3b the timing error detector characteristic of the DTTL for $\alpha=0.5$ is shown as an example for a DD algorithm characteristic. We see that the GA characteristic is sinusoidal and independent of E_S/N_0 while the DTTL characteristic depends on E_S/N_0 approaching a sinusoidal shape (with period 1) for low E_S/N_0.

For the NDA symbol synchronizers GA, NDAEL(1/2), NDAML (which all have a sinusoidal timing error detector characteristic) eqs. (V.1-1) - (V.1-3) apply.

For the decision directed symbol synchronizers DDML, DTTL, DDEL, M&M it is more difficult to give a good approximation of the acquisition performance in the high E_S/N_0 limit. While we can derive an upper bound for the frequency error reduction (and, thus, a lower bound on the acquisition time) based on the sawtooth characteristic, a better approximation of the acquisition performance may be obtained by approximating the timing error detector characteristic by a fraction of a

sinusoid, i.e. $g(\epsilon) \sim \sin(\pi\epsilon)$ in the range $-0.5 < \epsilon \leq +0.5$, see Fig. V.2-3b for instance. For the sawtooth approximation of the characteristic the acquisition performance is given by eqs. (V.1-4) and (V.1-5). For the sinusoidal approximation we have to modify eqs. (V.1-2) and (V.1-3). Since the approximation of the timing error detector characteristic is proportional to $\sin(\pi\epsilon)$ (not $\sin(2\pi\epsilon)$) we must replace Δf by $\Delta f/2$ in these equations. This corresponds to introducing a factor of 4 in the right hand side of (V.1-2) (again note that Δf has to be replaced by $\Delta f/2$ in left and right hand side of (V.1-2) yielding a total factor of 4, see NDAFB for carrier!) and a factor of 1/4 in the right hand side of (V.1-3), i.e. we have a four times faster acquisition than for a phase locked loop with sinusoidal characteristic. For lower E_S/N_0 the acquisition performance is between the results for the sawtooth timing error characteristic and the NDA synchronizer performance, approaching the NDA performance (eqs. (V.1-1) - (V.1-3)) in the low E_S/N_0 limit.

In Fig. V.2-2c the acquisition of a DTTL (DD algorithm, $B_L T=0.005$, $\zeta=0.707$ and $\zeta=2.5$, $E_S/N_0=50$dB) and a GA (NDA algorithm, $B_L T=0.001$, $\zeta=0.707$, $E_S/N_0=50$dB) symbol timing synchronizer is shown together with the approximations of acquisition time and frequency error reduction rate discussed above (note that the analytical results for the DTTL are based on the $\sin(\pi\epsilon)$ approximation of the error detector characteristic). We find an excellent agreement in all three cases, i.e. the influence of the loop damping factor again is modelled very accurately. For the DTTL we have shown the high E_S/N_0 limit only. With decreasing E_S/N_0 the frequency error reduction of the DTTL becomes slower and the acquisition time increases as with all decision directed synchronizers (also see carrier synchronization). Acquisition performance of the GA symbol timing synchronizer is independent of E_S/N_0 as for all non data aided synchronizers when the loop signal-to-noise ratio is sufficiently high.

<u>Data Aided Feedback Symbol Synchronization</u>

With a known preamble we might use data-aided synchronization during acquisition. Then, the timing error detector characteristic depends on the preamble structure. The autocorrelation function of the preamble sequence will determine the error detector gain for each interval $[-0.5 < \epsilon\text{-}NT \leq +0.5]$. In general, the characteristic will not be periodic any more.

V.3 Feedforward Synchronizer Acquisition Performance

V.3.1 Feedforward Carrier Synchronization

As discussed in the introduction the main difference is whether we use non data aided or decision directed synchronization. Therefore, we study the various scenarious separately for these two cases. See section V.1.2 for explanation of the terminology.

NDAFF Carrier Synchronization Algorithms

PAF0AC, PANFAC: In these two cases acquisition does not occur at all. A disadvantage is that we have the highest sensitivity to frequency errors in the case of PANFAC (in the case of PAF0AC any frequency error is excluded by definition). The error variance of the phase estimate increases with the frequency error for the PANFAC. The phase estimate is unbiased for the centre sample of each accumulator block. Thus, at the ends of a block we have a bias of $\Delta\omega T(M-1)/2$.

PAF0GL, PANFGL: For the first N samples we will have an acquisition effect since during this time we can base an estimate only on the available samples, i.e. less than N samples. For the time steps $k = 1,...,N$ the estimation error variance will be approximately N/k times its steady state value. The frequency error sensitivity is as for PAxxAC but with initially time varying accumulator length. As can be seen from the discussions in sec. IV.4.2 the influence of the frequency error on the estimation error variance increases with blocklength, and, thus with time for $k = 1,...,N$. With a frequency error (PANFGL) we have a bias of $\Delta\omega T/2$ for N even and BPSK, QPSK. Therefore, only odd N should be used which is sufficient to get a phase estimate for each symbol. For OQPSK and N odd we have a bias of $+\Delta\omega T/2$ and $-\Delta\omega T/2$ for the p and q sample of the center symbol, respectively (note that with N odd we just have the reversed bias when we use each phase estimate to correct the q sample of the symbol before the block center and the p sample of the symbol after the block center). Again, one phase estimate per symbol is already available when we use odd block lengths N only.

Hang-up does not occur in the above cases.

PAF0KF: In this case we determine the update matrix coefficients under the assumption that the initial frequency error is zero but we use a second order postfilter to track long term frequency drifts (a potential application may be a burst transmission where we can set the local oscillator to the correct carrier frequency based on a frequency prediction from previous bursts). Then the resulting coefficients $K_1(k)$, $K_2(k)$ will be such that the first phase estimates will not have an influence on the frequency estimates of the filter, i.e. we start with a pure phase acquisition. The first filtered phase estimate will be the first feedforward carrier phase synchronization algorithm estimate. Acquisition will be finished when the coefficients $K_1(k)$, $K_2(k)$ have approached their steady state values. The probability of hang-up is reduced significantly. To have hang-up the error of the first estimate must have a value close to $\pi/2$ for BPSK and OQPSK or $\pi/4$ for QPSK, i.e. be half the phase difference of two neighbouring symbols. Since further outputs of the feedforward algorithm have less influence on the filtered phase estimate (the coefficients $K_1(k)$ decrease with time, $K_1(k>0) < K_1(0)$), we then will have an extended acquisition period.

FAFCKF: In this case we have a full phase and frequency acquisition which takes the longest time as compared to the above scenarios. For the acquisition with time varying coefficients we will have a shorter acquisition time as compared to a fixed bandwidth synchronizer (see the discussions for the feedback synchronizers).

DDFF Carrier Synchronization Algorithms

When we have a known preamble, i.e. we can use data-aided carrier synchronization, we have the same effects as for the non-data aided feedforward synchronizers except that frequency error doubling or quadrupling does not occur. Thus, we have the best acquisition performance in this case. When we do not have a preamble the situation is different, in general the performance is inferior to the NDA feedforward carrier synchronizer performance. The reason is that DD introduces feedback (see for instance Fig. IV.2-4) and, therefore, also hang-up and other disadvantages as we will discuss below.

PAF0xx: Independent of the type of postfilter we may have hang-up. The reason is that with a phase error of $\pi/2$ (BPSK and OQPSK) or $\pi/4$ (QPSK) the decision error probability is 0.5 and, thus, the mean of the phase estimate will be zero. In Fig. V.3-1 we have shown the mean of the phase estimate for a given phase when the decision samples are not corrected, i.e. when there is a phase error. We have a sawtooth characteristic in the noise free case, approaching a sinusoidal shape in the low E_S/N_0 limit. This means we get biased estimates where the bias is the difference between the noise free characteristic and the applicable characteristic. The bias increases with the initial phase value. Because of this bias in the estimate, the mean phase error will not be zero after correction in the first step (although it will be smaller). With a smaller phase error the bias will be smaller in the second step until it is finally negligibly small. A step corresponds to one update of the postfilter, i.e. to the postfilter 'sampling' time. The number of steps required depends on the initial phase value, the bias value which is assumed to be negligible (theoretically, the bias will approach zero forever but never reach zero) and on E_S/N_0 (since E_S/N_0 determines the deviation between ideal sawtooth and actual characteristic).
This acquisition effect (but not the risk of hang-up) can be reduced by an iterative approach with a block accumulator. The bias is caused by decision errors which try to compensate the (residual) phase error in the samples used for decision. When we store the samples of a block we can correct them based on the phase estimate obtained in the first pass. Since the error is now reduced we will get better decisions and thus a better phase estimate in the second pass. After some passes the bias will be within an acceptable range. Thus, we can obtain a practically unbiased estimate even from the first block of samples at the cost of an increased signal processing effort.

PANFAC, PANFGL: Initially we have a similar situation as for PAF0xx. The signal-to noise ratio is reduced by a frequency error as described for the tracking mode. Since we need a prediction of the phase with DD synchronization algorithms a frequency error causes an additional bias $\Delta\omega T(N-1)/2$ (as already described for the tracking mode) which adds to the bias effect discussed above for the PAF0xx. Therefore, this is not a good operating mode for DD feedforward synchronizer.

FADCKF, FAFCKF: Initially, we do not have a correct frequency estimate in the postfilter. Therefore, we have a bias in the predicted phase and thus the decisions will be based on samples with a residual phase error. This leads to biased phase estimates in the DD synchronizer. The postfilter will use the difference between prediction and estimate to update its phase and frequency estimates. But this difference is not the prediction error (i.e. the difference between prediction and actual phase), it is a function of the prediction error (note that for an accumulator with blocklength N=1 this function is identical to the function shown in Fig. V.3-1). This effect occurs with any type of postfilter. It can only be avoided when a frequency estimate is available which is generated independent from the DD synchronizer estimate (e.g. by an additional frequency estimator). Thus, we have an acquisition performance which is close to that of a feedback synchronizer rather than that of a feedforward synchronizer. Again note that for an accumulator with block length N=1 the function shown in Fig. V.3-1 plays the role of the phase error detector characteristic in the acquisition performance analysis. In particular, we have a sinusoidal characteristic in the low E_S/N_0 limit.

V.3.2 Feedforward Symbol Synchronization

NDAFF Symbol Synchronization

We have the same effects as for the NDAFF carrier synchronization except for the fact frequency error doubling or quadrupling does not occur with symbol synchronizers.

A special case is the NDASQFF$\bar{C}$ algorithm where the timing estimate is obtained as the argument of a complex signal, the spectral line at the symbol rate (since we use the symbol rate spectral line, frequency error doubling does not occur even though the signal is squared). Instead of filtering this argument we might instead directly filter the complex signal (planar filtering). Then, the acquisition performance will be determined by the time constants of this filter. With an initial frequency error, real and imaginary part will be sinusoids having a frequency equal to the frequency error (and a data-dependent amplitude). This means the filter bandwidth must be larger than the maximum frequency error.

Data Aided Feedforward Symbol Synchronization

When we have a known preamble or receive known data (e.g. when we receive our own transmission for network synchronization purposes), we might use data-aided symbol synchronization. The correct timing estimate must be within the range over which the timing estimate is determined. For instance, when the position of the preamble is known initially within ±2 symbols we have to evaluate the symbol synchronization algorithm over at least ±2 symbols. In general we will have a lower noise power spectral density than with the corresponding NDAFF. Thus, we have the best acquisition performance in this case.

DDFF Symbol Synchronization

When we do not have a preamble the situation is different and the performance is in general inferior to the NDA feedforward carrier synchronizer performance.

PAF0xx: Independent of the type of postfilter we may have hang-up. The reason is that with a timing error of 1/2 symbol the decision error probability is 0.5 and, thus, the mean of the timing estimate will be zero. When we display the mean of the estimate over the actual timing error (as we have done for the carrier phase in Fig. V.3-1) we will have a sawtooth characteristic in the noise free case, approaching a sinusoidal shape in the low E_S/N_0 limit. This means we get biased estimates where the bias is the difference between the noise free characteristic and the applicable characteristic. The bias increases with the initial timing error value. Because of this bias in the estimate, the mean timing error will not be zero after correction in the first step (although it will be smaller). With a smaller timing error the bias will be smaller in the second step until it is finally negligibly small. A step corresponds to one update of the postfilter, i.e. to the postfilter 'sampling' time. The number of steps required depends on the initial phase value, the bias value which is assumed to be negligible (theoretically, the bias will approach zero forever but never reach zero) and on E_S/N_0 (since E_S/N_0 determines the deviation between ideal sawtooth and actual characteristic).

We can eliminate this bias by an iterative approach as discussed for carrier synchronization.

PANFAC, PANFGL: Initially we have a similar situation as for PAF0xx. The signal-to noise ratio is reduced by a frequency error as described for the tracking mode. Since we need a prediction of the timing with DD synchronization algorithms a frequency error causes an additional bias $\Delta fT(N-1)/2$ which adds to the bias effect discussed above for the PAF0xx. For small frequency errors this may still be acceptable.

FADCKF, FAFCKF: With the same reasoning as for the carrier synchronizers we have an acquisition performance as with a feedback synchronizer rather than a performance we expect with a feedforward synchronizer. Again, the effect is inherent, independent of the postfilter type. As for carrier synchronization it can only be avoided when an additional frequency estimate is available (which was not generated from the DD synchronizer timing estimates).

V.4 Summary

With feedback carrier or symbol synchronizers acquisition times strongly increase with the initial frequency error. The best performance is obtained when a known preamble is available and the decision directed synchronizer can be replaced by a data-aided synchronizer during acquisition. In the low E_S/N_0 limit the acquisition performance is identical for all feedback synchronizers for a particular modulation scheme, loop damping, and loop noise bandwidth. With increasing E_S/N_0 DD synchronizers perform better. A significant improvement for feedback synchronizers can be obtained by using an appropriate (time varying) postfilter or aided acquisition. When the initial frequency error may be large, aided acquisition should be used (sweep, frequency error estimation).

With feedforward carrier and symbol synchronizers we have a clear advantage of the non data-aided algorithms since with the decision directed algorithms we inherently have memory in the system and, thus, acquisition effects, including hang-up. Again, when a preamble is available we get the best performance by using data-aided synchronization during acquisition. In such a case decision directed synchronization can be used after preamble based acquisition.

An appropriate postfilter is very important with the feedforward synchronizers when we may have large frequency errors. When we must generate a frequency estimate from the phase (or symbol timing) we will introduce feedback via the postfilter. Then, we get similar acquisition effects as with common feedback synchronizers.

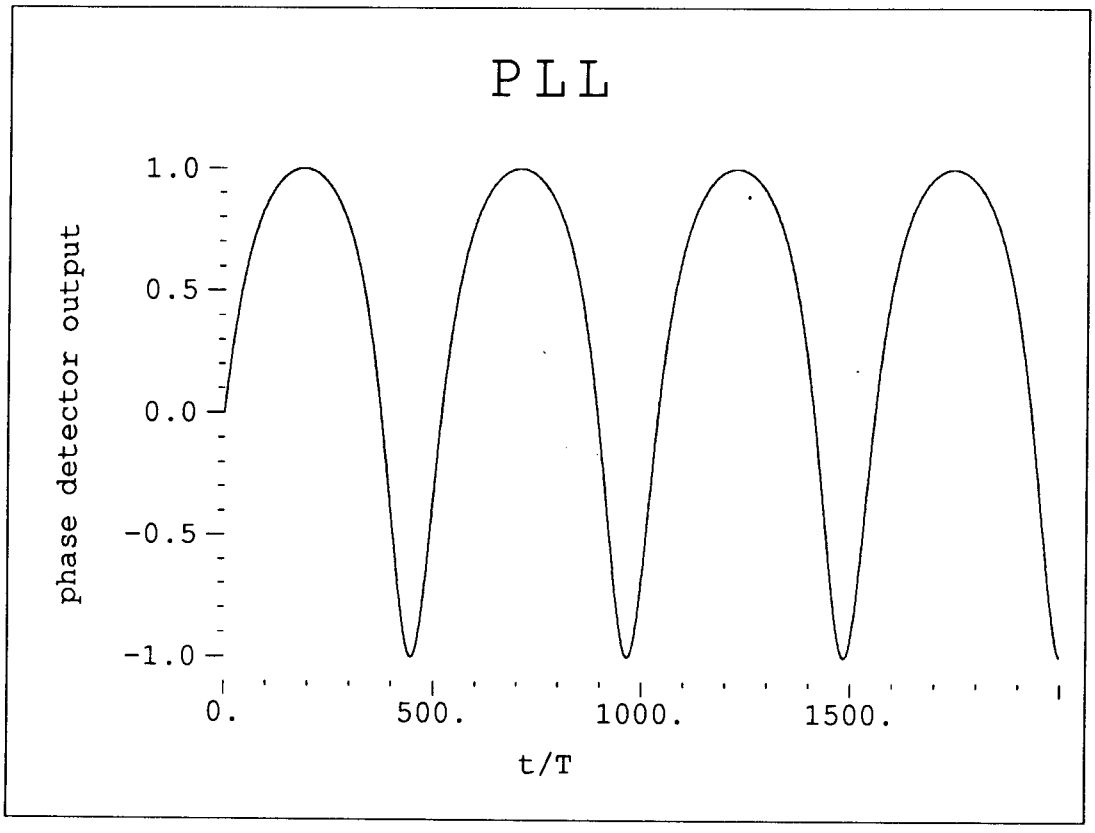


Fig. V.1-1: Phase locked loop 'beat note'

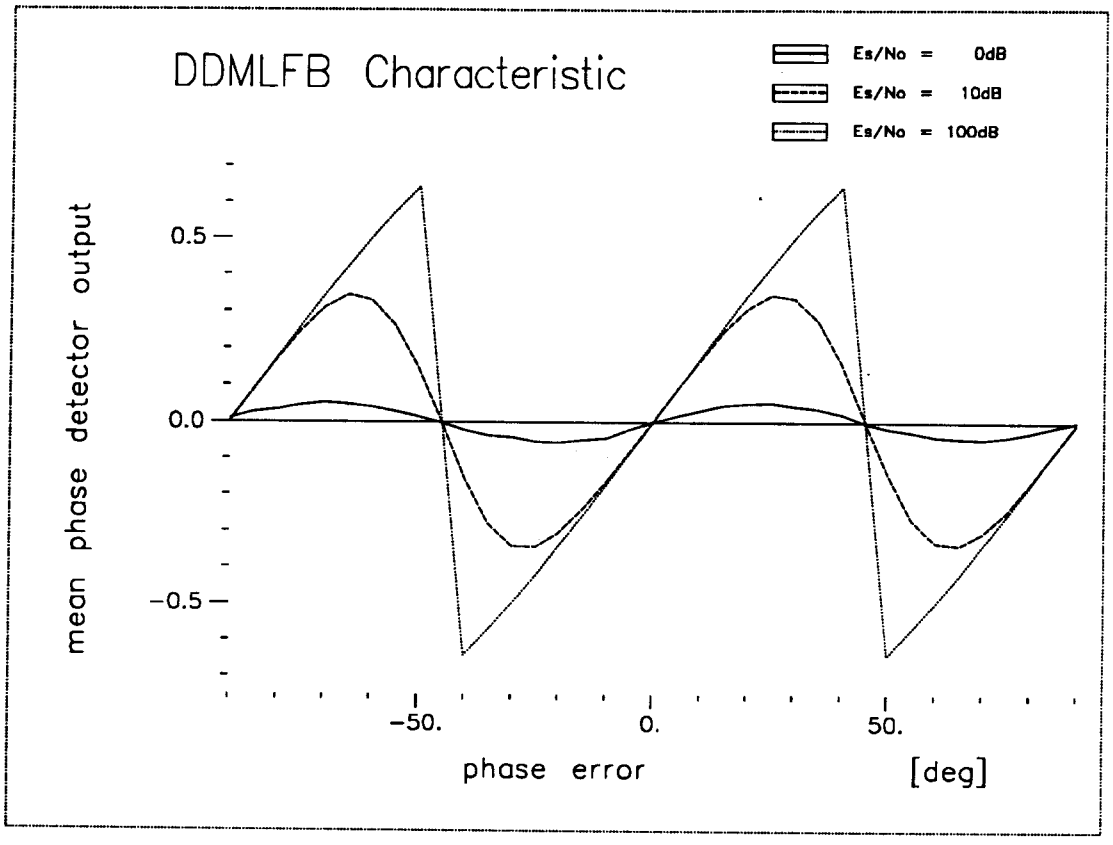


Fig. V.2-1: Phase detector characteristic of QPSK DDMLFB carrier synchronization algorithm, simulation

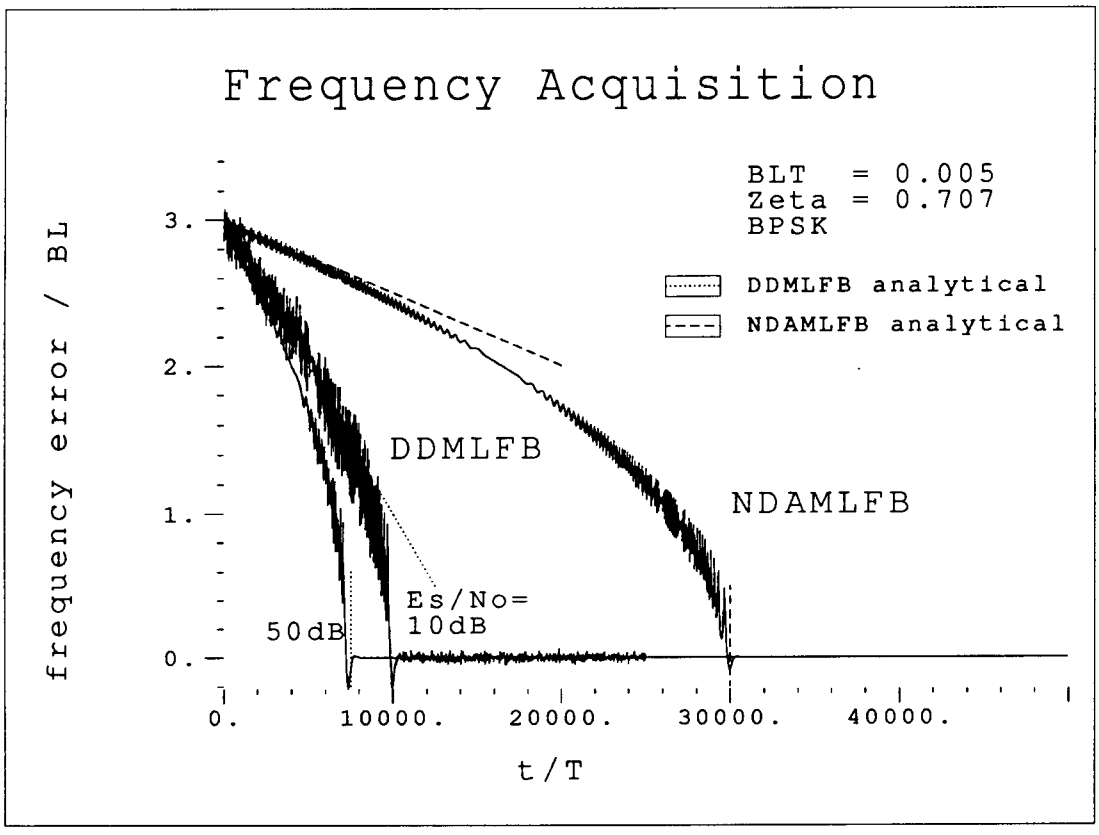


Fig. V.2-2a: Frequency acquisition ($\Delta f/B_L$) of DDMLFB and NDAMLFB, B_LT=0.005, ζ=0.707, E_S/N_0=50dB (and 10dB for DDMLFB). For reference the analytically determined initial frequency error reduction rate and acquisition time are shown. Note that the frequency error is averaged over 50 symbols to remove the oscillations.

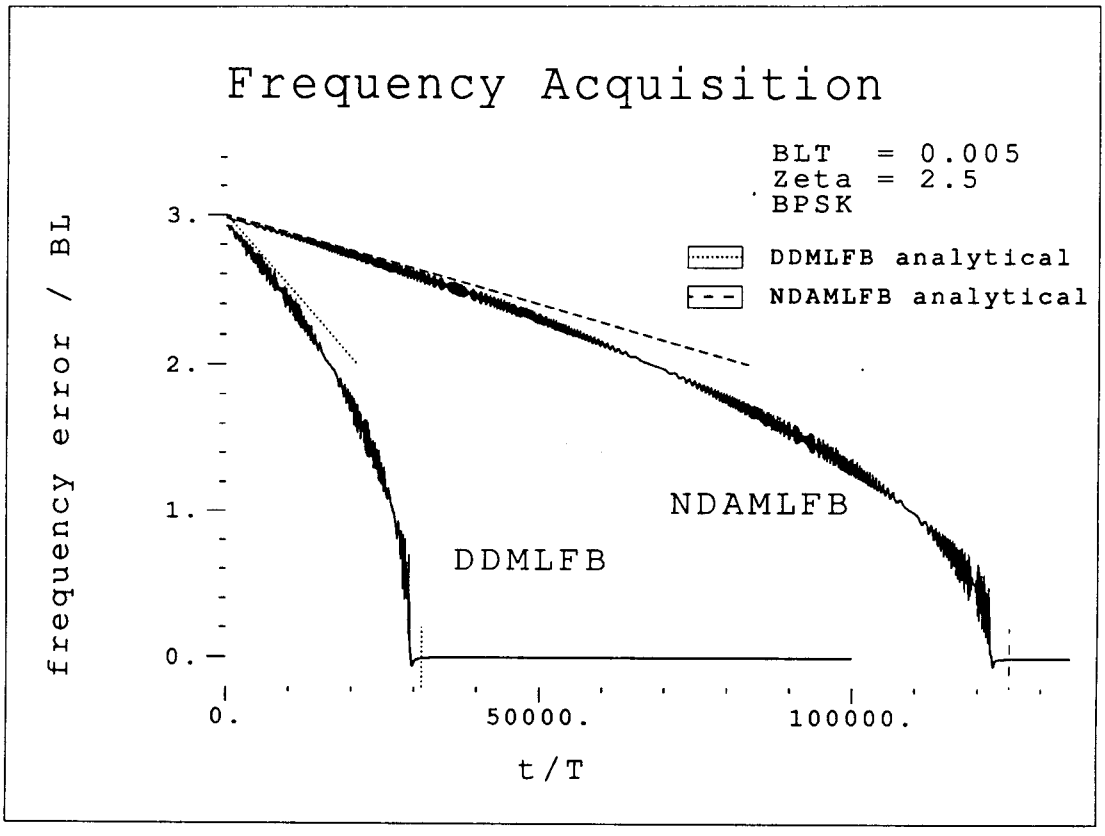


Fig. V.2-2b: Frequency acquisition ($\Delta f/B_L$) of DDMLFB and NDAMLFB, B_LT=0.005, ζ=2.5, E_S/N_0=50dB. For reference the analytically determined initial frequency error reduction rate and acquisition time are shown. Note that the frequency error is averaged over 50 symbols to remove the oscillations.

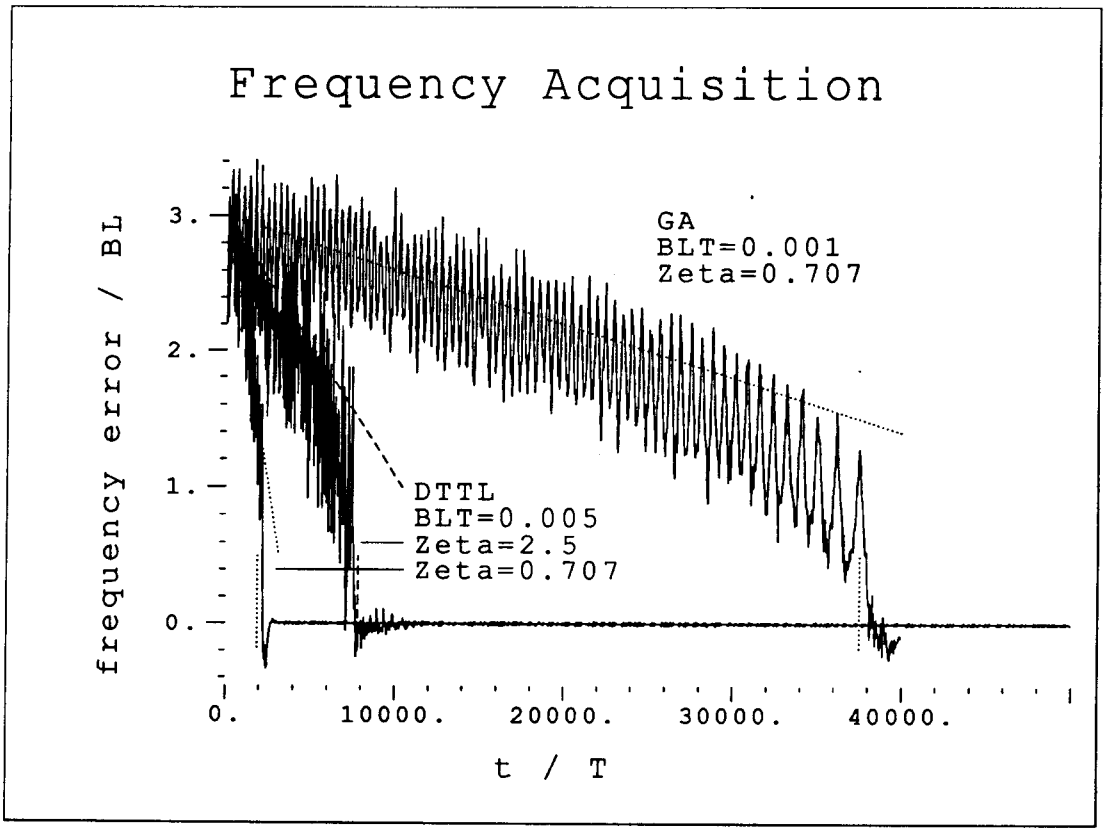


Fig. V.2-2c: Frequency acquisition ($\Delta f/B_L$) of DTTL (B_LT=0.005, ζ=0.7 and 2.5, E_S/N_0=50dB) and GA (B_LT=0.001, ζ=0.7, E_S/N_0=50dB) for QPSK. For reference the analytically determined initial frequency error reduction rate and acquisition time are shown. Note that the frequency error has been averaged over 50 symbols to remove the oscillations.

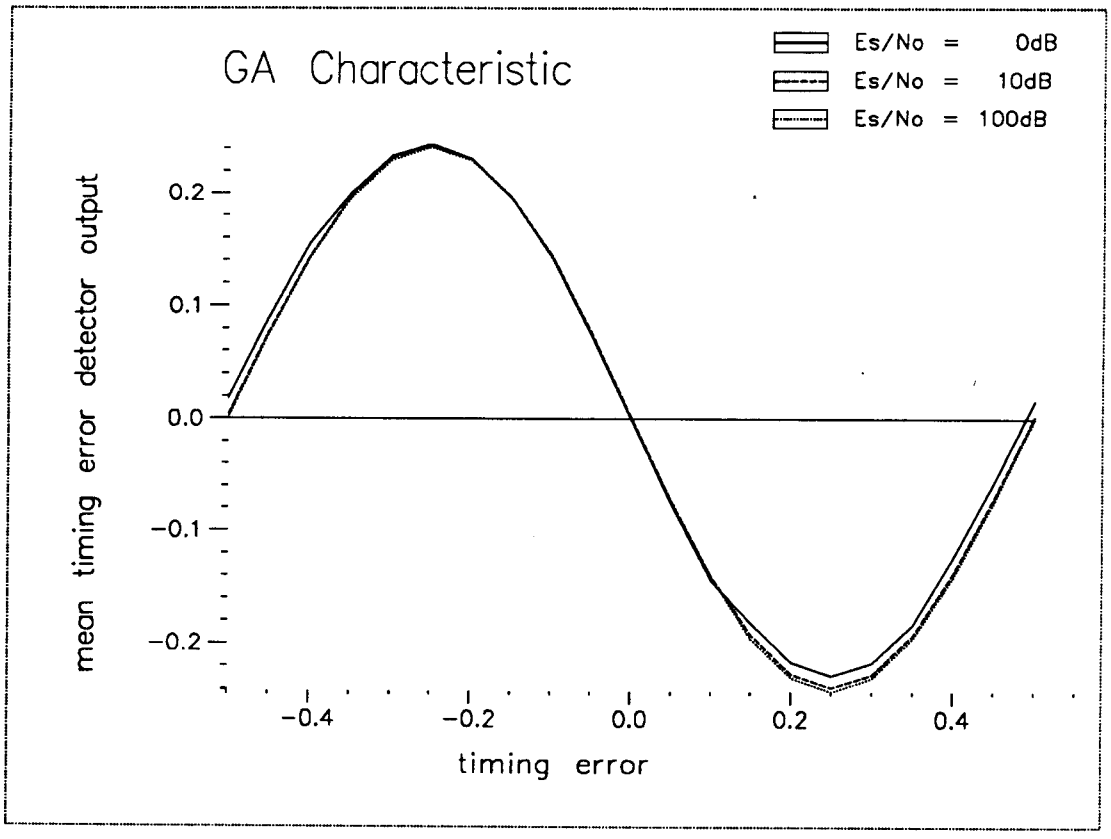


Fig. V.2-3a: Timing error detector characteristic of GA for rolloff $\alpha = 0.5$ and QPSK with E_S/N_0 as parameter, simulation

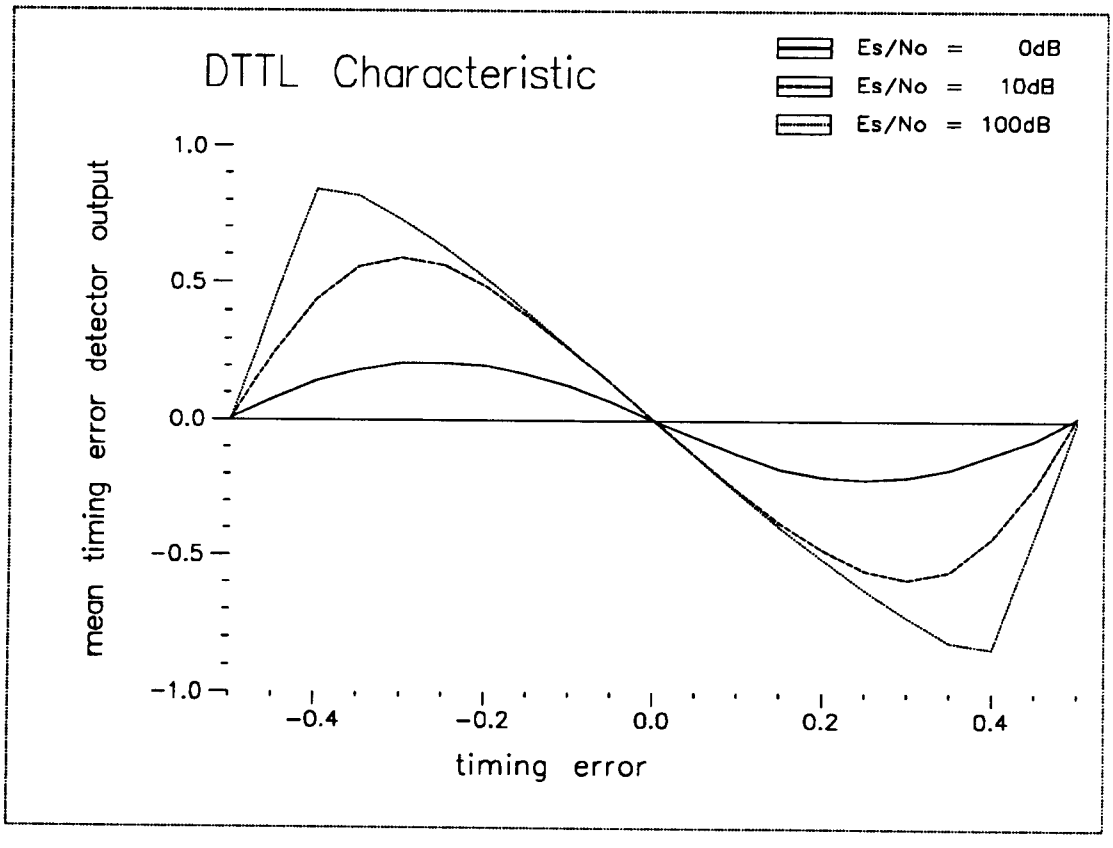


Fig. V.2-3b: Timing error detector characteristic of DTTL for rolloff $\alpha = 0.5$ and QPSK with E_S/N_0 as parameter, simulation

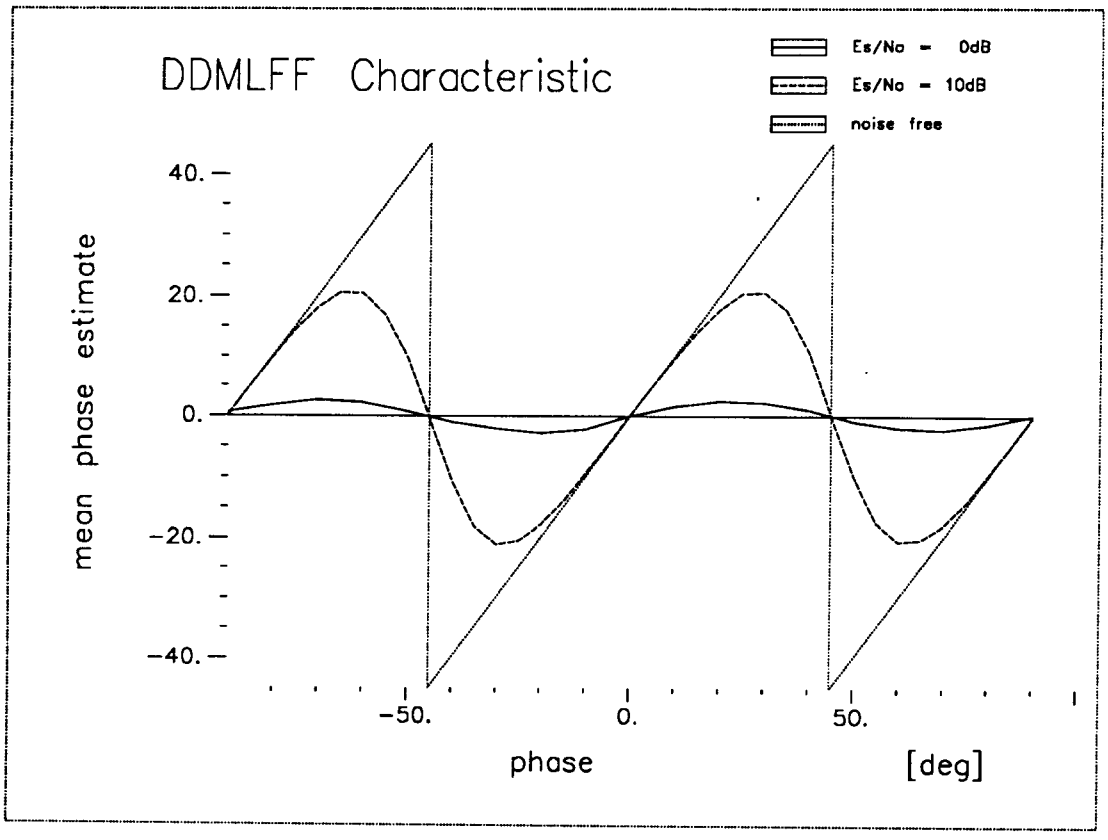


Fig. V.3-1: Mean of phase estimate as a function of phase for QPSK DDMLFF carrier synchronization algorithm (decision samples not corrected), simulation

VI. CYCLE SLIPPING PERFORMANCE

VI.1 PRELIMINARIES

In Part VI of this study we investigate the cycle slipping properties of carrier and symbol synchronizers. Synchronizers should be operated in such a way that cycle slipping is a very rare event. Consequently, studying cycle slips by means of computer simulations is not adequate, because of the excessive requirements on computing time.

Feedback synchronizers operate according to the principle of the phase–locked loop (PLL); powerful analytical tools (Fokker–Planck theory) for studying cycle slips in PLL's are available from the open literature [Mey'90]. Unfortunately, for feedforward synchronizers we have not found any analysis of cycle slips in the literature. Therefore, we have to restrict our attention mainly to feedback synchronizers; some qualitative considerations about cycle slips in feedforward synchronizers are reported in section VI.4.

Cycle slips in PLL's can be studied by means of Fokker–Planck theory. Closed–form analytical results can be obtained only for first–order loops. Consequently, we give analytical results only for first–order feedback carrier and symbol synchronizers. As in Parts II and III of this study, analytical results are obtained under the assumption of "ideal" conditions : additive white Gaussian noise channel, zero frequency error, Nyquist–I pulse at matched filter output, and ideal receiver operations (no quantization, ...).

In many applications, second–order loops are used, because their response to a frequency offset is more favourable than for first–order loops. Some qualitative considerations about cycle slipping in second–order loops are given in section VI.1.4.

VI.1.1 Feedback synchronizer model

Strictly speaking, Fokker–Planck theory applies only to continuous–time systems. Let us consider the equivalent model of a continuous–time first–order PLL, shown in Fig.VI.1–1, where we have used the following notation :

$\psi_c(t)$: phase (or timing) error

$g(\psi_c)$: phase (or timing) error detector characteristic, which is periodic in ψ_c with period p (i.e. $g(\psi_c + p) = g(\psi_c)$); the stable equilibrium points are at $\psi_c = 0, \pm p, \pm 2p, \ldots$, and the slope at the stable equilibrium point is normalized to 1 (i.e. $g(0) = 0$ and $g'(0) = 1$); the unstable equilibrium points are assumed to be at odd multiples of p/2 (i.e. $g(p/2) = 0$).

$N_c(t,\psi_c)$: loop noise, with power spectral density given by $S_c(f;\psi_c)$, which is periodic in ψ_c with period p.

K_c : loop gain ($K_c > 0$).

The subscript c has been introduced to denote quantities related to a continuous–time feedback synchronizer; the corresponding quantities in a discrete–time feedback synchronizer, to be considered in section VI.2.1, will carry the subscript d.

The above model applies to both carrier and symbol synchronizers, when the following correspondences are used :

– carrier synchronization :

$\psi_c = \phi$ (carrier phase error in radians)

$$p = \begin{cases} \pi & \text{BPSK, OQPSK} \\ \pi/2 & \text{QPSK} \end{cases}$$

– symbol synchronization :

$\psi_c = \epsilon$ (timing error measured as fraction of symbol interval)

$p = 1$

VI.1.2 Cycle slips and their effect on the BER.

The PLL from Fig.VI.1–1 has its stable equilibrium points at $\psi_c = 0, \pm p, \pm 2p, \ldots$ Most of the time, the loop noise $N(t,\psi_c)$ causes the error $\psi_c(t)$ to exhibit small fluctuations in the immediate vicinity of a stable equilibrium point (say, the equilibrium point at $\psi_c = 0$). However, occasionally the loop noise gives rise to such large excursions of $\psi_c(t)$, that $\psi_c(t)$ leaves the domain of attraction of the equilibrium point at $\psi_c = 0$, and then moves towards a neighbouring equilibrium point (at $\psi_c = p$ or $\psi_c = -p$). This phenomenon is called a cycle slip.

During a slip, the deviation of $\psi_c(t)$ from the stable equilibrium point is so large, that a burst of bit errors occurs at the output of the receiver's decision device. After a slip has occurred, the synchronizer is at a different equilibrium point. As this new equilibrium point usually affects the operation of the decision device (e.g., after a carrier phase slip the polarity of the signal at the input of the decision device is reversed in the case of BPSK), the receiver must be able to detect that a slip has occurred. Assuming that half of the bits are in error during a slip and as long as the slip has not been detected, the BER due to cycle slipping is approximately given by

$$BER_{slip} \cong \frac{1}{2T_s}\left(\frac{1}{B_L} + T_d\right) = \frac{1}{2B_L T_s}(1 + B_L T_d) \qquad \text{(VI.1.2–1)}$$

where T_s denotes the mean time between slips (equivalently, $1/T_s$ is the cycle slip rate), T_d is the average time needed to detect a slip, and (the order of magnitude of) the duration of a slip is $1/B_L$, where B_L is the one–sided loop bandwidth of the synchronizer. In a properly designed receiver, the contribution of cycle slips to the BER should be much smaller, by several orders of magnitude, than the BER in the absence of synchronization errors.

<u>Uncoded transmission</u>

In the case of a carrier phase slip, the receiver's carrier phase reference after the slip differs by $\pm\pi$ (BPSK, OQPSK) or $\pm\pi/2$ (QPSK) from the carrier phase reference before the slip. The effect of a cycle slip on the BER can be limited to the duration of a slip by using differential encoding/decoding of the information symbols. In this case, the contribution of the cycle slips to the BER is approximated by $1/(2B_L T_s)$. When differential encoding/decoding is not used, the occurrence of a slip can be detected by monitoring a synchronization word which is, once per frame, inserted into the data stream. The average time to

detect a slip is about $T_F/2$, where T_F denotes the frame duration. Hence, the contribution of cycle slips to the BER is approximated by $(1+(T_F B_L/2))/(2B_L T_s)$.

In the case of a slip of the symbol synchronizer, the receiver's symbol count after the slip is wrong by one symbol (repetition or deletion of a symbol), so that the assumed frame boundaries are no longer correct. In this case the effect of a cycle slip extends till the end of the frame containing the slip. The contribution of cycle slipping to the BER is approximated by $(1+(T_F B_L/2))/(2B_L T_s)$, where T_F denotes the duration of a frame.

Coded transmission

The receiver's decoder looks for the valid coded symbol sequence which produces the best fit with the received symbol sequence, and outputs the corresponding information sequence. After a cycle slip of the carrier or symbol synchronizer, the phase or the symbol count of the received symbols are no longer correct, so that the decoder output becomes highly unreliable.

The occurrence of a cycle slip can be detected by monitoring some quantity (such as the metric growth of a Viterbi decoder) which indicates how well the received symbols obey the encoding rule. Alternatively, at the transmitter a synchronization word can be inserted in the coded symbol stream at regular intervals; the occurrence of a slip can be detected by monitoring the synchronization word at the receiver. In both cases, the effect of a cycle slip lasts until its occurrence has been detected. The contribution of cycle slips to the BER is approximated by $(1+T_d B_L)/(2B_L T_s)$, where T_d denotes the average time to detect the occurrence of a slip. In the case of the regulary inserted synchronization word, T_d equals about half the frame duration.

We conclude that the contribution of cycle slips to the BER is well approximated by $1/(2B_L T_s)$, unless the average time to detect the occurrence of a slip is much larger than $1/B_L$.

The reader should be aware that the effect of slips is not limited to just an increase of the BER. For example, slips may give rise to frame synchronization loss when the synchronization word is damaged. The impact of slips depends on the specific application : a lost frame is more harmful to image transmission than to voice transmission.

VI.1.3 Basic analytical results from Fokker–Planck theory.

In many cases of practical interest, the loop noise spectrum $S_c(f;\psi_c)$ of the continuous–time PLL is, for all ψ_c, nearly flat over a frequency interval which is much larger than the loop bandwidth B_L (for a first–order loop, $B_L = K_c/4$), and can be approximated by $S_c(0;\psi_c)$. When this approximation is valid, Fokker–Planck theory can be applied, and a rather complicated closed–form expression for the mean time between slips can be derived. However, for arbitrary $g(\psi_c)$ and $S_c(0;\psi_c)$ the evaluation of this closed–form expression requires a double numerical integration, in which numerical underflow and overflow are hard to avoid. Fortunately, a very accurate approximation can be derived from the exact closed–form expression. This approximation, which forms the starting point for our study of cycle slips, is given by [Moe'85a, Moe'85b, Mey'90]

$$2B_L T_s = \frac{\pi}{2} \frac{\exp(\rho A)}{\sqrt{|g'(p/2)|}} \qquad \text{(VI.1.3–1)}$$

where T_s is the mean time between slips, B_L is the one–sided loop bandwidth, $g'(p/2)$ is the slope at the unstable equilibrium point, ρ is the loop–SNR (i.e. the inverse of the linearized tracking error variance) :

$$\rho = \frac{1}{2B_L S_c(0;0)} = \frac{1}{\mathrm{Var}[\psi_c]} \qquad \text{(VI.1.3–2)}$$

and A is the area indicated in Fig.VI.1–2, given by

$$A = \int_0^{p/2} \frac{g(\psi)d\psi}{S_c(0;\psi)/S_c(0;0)} \qquad \text{(VI.1.3–3)}$$

The approximation (VI.1.3–1) is very accurate for $2B_L T_s > 10$, but overestimates $2B_L T_s$ for $2B_L T_s < 10$ [Lin'73]. What we learn from (VI.1.3–1) is that the mean time between slips is an exponential function of the loop–SNR ρ, and that the most important property of the synchronizer that affects the slipping performance is the area A. When we compare the cycle slipping performance of two synchronizers with areas A_1 and A_2 , respectively, and $A_1 > A_2$, then the first synchronizer is better than the second, by an amount of approximately $10 \log(A_1/A_2)$ dB in loop–SNR.

VI.1.4 Cycle slip rates of second–order loops

Closed form analytical results for cycle slip rates of feedback synchronizers can be obtained for first–order loops only. Sophisticated, series expansion based, numerical methods are needed to determine cycle slip rates (or even 'burst' rates) of second–order loops [Mey'90]. The required effort is much above what a designer might want to invest into the determination of a single performance parameter like the cycle slip rate. Therefore, here we will discuss the effects qualitatively, and give a few numerical results for a second–order phase–locked loop operating on an unmodulated carrier or, equivalently, a synchronizer with sinusoidal detector error characteristic $g(\psi_c)$ and a loop noise power spectral density independent of ψ_c: $S_c(f;\psi_c) = S_c(f;0)$. This will give sufficient insight to upper bound the cycle slip rate of a synchronizer.

In the loop filter of a second–order loop we have an integrating path parallel to the proportional path of the first–order loop. An equivalent blockdiagram of the loop filter is shown in Fig. VI.1–3. Note that the subsequent results also apply to second–order loops with imperfect integrators ('lag–lead' filter) when the steady state phase error due to an initial frequency error is negligible.

With a first–order loop, an initial frequency error $\Delta\omega$ gives rise to a nonzero (mean) phase error in the tracking mode; this mean phase error increases the probability of a slip, because the stable equilibrium point has moved closer to one of the unstable equilibrium points. With a second–order loop the output of the integrating path of the filter (Fig. VI.1–3) will take on a value proportional to $\Delta\omega$, such that the (mean) phase error is zero. Thus, the integrating path provides a frequency estimate $\Delta\hat{\omega}$. In the presence of loop noise this estimate will be 'noisy'. Let us assume $\Delta\omega = 0$ for the moment (after acquisition the initial frequency error is compensated and, thus, has no more influence on loop tracking performance). Due to the noise the integrator will be feigning a frequency error which is not present and, thus, induces a phase error (in the mean conditioned on a positive/negative integrator level) which tries to compensate the virtual frequency error. With a phase error cycle slips are more likely. Therefore, the cycle slip rate of a second–order loop is always higher than the cycle slip rate of a first–order loop for $\Delta\omega = 0$.

Because of the integrator, the cycle slip rate of a second–order loop (with integrating filter, Fig. VI.1–3) is independent of $\Delta\omega$. With a first–order loop we have a (mean) phase error $\sin^{-1}(\Delta\omega/(4B_L))$ in the tracking mode. Therefore, with increasing $\Delta\omega$ we will have a strongly increasing cycle slip rate in a first–order loop.

When, for a given bandwidth, we use a large corner frequency $(1/T_2)$ of the integrating filter, loop damping is low, and the integrator can follow frequency fluctuations more rapidly (and also approach its steady state value more rapidly during acqusition). With a small corner frequency $(1/T_2)$ of the integrating filter loop damping is high, and the integrator reaction is slower. But with a small corner frequency, the frequency estimate at the integrator output is also subject to less noise than for a large corner frequency. Therefore, the cycle slip rate decreases with increasing loop damping factor ζ, approaching first–order loop performance in the limit for $\zeta \to \infty$ [Mey'90].

For a synchronizer with sinusoidal characteristic $g(\psi_c)$ and flat loop noise power spectral density $S_c(f,\psi_c) = S_c(0,0)$, numerical results may be given. For $\zeta = 0.7$ the loop signal–to–noise ratio (i.e. the inverse of tracking error variance in rad^2) must be approximately 3 dB above the loop signal–to–noise ratio of a first–order loop for the same cycle slip rate. For $\zeta = 1$ the difference is approximately 1 dB, going below 0.2 dB for $\zeta = 1.5$ [Mey'90]. It should also be noted that with loops with damping factors $\zeta < 1$, cycle slips tend to appear in bursts, in particular, at low loop signal–to–noise ratios. Instead of isolated cycle slips we then have (isolated) blocks of cycle slips ('bursts'). The average number of cycle slips per burst increases with decreasing loop signal–to–noise ratio.

Because of the integrator in the filter, the second–order loop is able to track a signal over a wide frequency range, but it is able to acquire lock within a narrow range around the current frequency only. Eventually, the noise may drive the integrator value out of this range during a burst of cycle slips, so that the loop drops lock completely [Asc'82]. When this situation is detected, an appropriate reacquisition procedure must be started.

VI.2 APPLICATION OF FOKKER–PLANCK THEORY TO FEEDBACK CARRIER AND SYMBOL SYNCHRONIZERS.

In this section we investigate how the basic result (VI.1.3–1) can be applied to the various carrier and symbol synchronization algorithms, suited for digital implementation. First, we recall that the Fokker–Planck theory has been derived for continuous–time PLL's, whereas the synchronization algorithms of interest operate in discrete time. Secondly, applicability of the Fokker–Planck theory requires that, for all ψ_c, the loop noise spectrum $S_c(f;\psi_c)$ is sufficiently flat over a frequency interval exceeding the loop bandwidth; this is not the case for the self–noise contribution to $S_c(f;\psi_c)$, taken at $\psi_c = 0$, so that for very large E_b/N_o, where self–noise is dominant, Fokker–Planck theory yields questionable results.

VI.2.1 Discrete–time algorithms.

The system equation for a first–order loop, operating in discrete time with updating at the symbol rate 1/T, is given by

$$\psi_d(k+1) = \psi_d(k) - K_d\Big[g(\psi_d(k)) + N_d(k;\psi_d(k))\Big] \qquad \text{(VI.2.1–1)}$$

where the loop noise $N_d(k;\psi_d)$ has a power spectral density $S_d(fT;\psi_d)$; the subscript d is used to denote quantities related to the discrete–time loop. The sequence $\{\psi_d(k)\}$ can be viewed as a sequence of samples of an underlying continuous–time process $\psi_c(t)$: $\psi_d(k) = \psi_c(kT)$. When the loop bandwidth is much smaller than the symbol rate, the process $\psi_c(t)$ satisfies the system equation of a fictitious continuous–time synchronizer :

$$\frac{d\psi_c}{dt} = -K_c\Big[g(\psi_c(t)) + N_c(t;\psi_c(t))\Big] \qquad \text{(VI.2.1–2)}$$

where $N_c(t;\psi_c)$ has a power spectral density $S_c(f;\psi_c)$; the subscript c is used to denote quantities related to the continuous–time equation. The correspondence between the discrete–time equation (VI.2.1–1) and the continuous–time equation (VI.2.1–2) holds for

$$K_c = K_d/T \qquad \text{(VI.2.1–3a)}$$

$$S_c(f;\psi) = \begin{cases} S_d(fT;\psi)\,T & |f|<1/(2T) \\ 0 & \text{otherwise} \end{cases} \qquad \text{(VI.2.1–3b)}$$

The cycle slipping behaviour of the discrete–time synchronizer can now be obtained from studying the cycle slipping of the fictitious continuous–time synchronizer. The mean time between slips is simply obtained by substituting in (VI.1.3–1) the loop bandwidth B_L and loop–SNR ρ of the discrete–time synchronizer ($2B_LT = K_d/2$; $1/\rho = 2B_LTS_d(0;0)$), and computing the area A from (VI.1.3–3), with $S_c(0;\psi)$ replaced by $S_d(0;\psi)$.

From now on, we restrict our attention to discrete–time algorithms, and drop the subscript d.

VI.2.2 Self–noise at equilibrium points.

The loop noise spectrum $S(fT;\psi)$ can be decomposed into an additive noise contribution and a self–noise contribution. In Part II we have pointed out that at $\psi = 0$ (i.e. at the tracking point), the self–noise contribution to the loop noise spectrum vanishes at $fT = 0$. Consequently, for large E_b/N_o, $S(fT;\psi)$ cannot be considered as flat over a large frequency interval for $\psi = 0$.

Let us investigate what happens when we simply replace $S(fT;\psi)$ by $S(0;\psi)$ and apply Fokker–Planck theory. The argument of the exponential function in (VI.1.3–1) is given by

$$\rho A = \frac{1}{2B_L T} \int_0^{p/2} \frac{g(\psi)d\psi}{S(0;\psi)} \qquad \text{(VI.2.2–1)}$$

In the limit for $E_b/N_o \to \infty$, ρA becomes infinite (this is because in the vicinity of $\psi = 0$, $g(\psi)$ and $S(0;\psi)$ are proportional to ψ and ψ^2, respectively, which makes the integrand proportional to $1/\psi$), so that the mean time between slips tends to infinity. In other words, there are no slips in the absence of additive noise. This behaviour is easily verified from the system equation (VI.2.1–1), by noting that the loop noise is identically zero at $\psi = 0$, because its (assumed flat) spectrum is zero at $\psi = 0$. Consequently, when the error ψ reaches a stable equilibrium point, both $g(\psi)$ and the loop noise are zero, so that ψ stays forever at the stable equilibrium point. For many synchronizers, also $S(0;p/2) = 0$ in the absence of additive noise; in this case, when ψ reaches an unstable equilibrium point, it stays there forever. We conclude that the Fokker–Planck theory gives rise to a mean time between slips which increases without limit when E_b/N_o goes to infinity. This is because self–noise at the equilibrium point(s) is ignored, due to its vanishing spectrum at $fT = 0$. When there is no self–noise at the equilibrium point(s), the behaviour predicted by the Fokker–Planck theory is correct.

Unlike the additive noise contribution to the loop noise, the self–noise contribution is bounded. Because of the lowpass nature of the synchronizer loop, the tracking error is essentially determined by the accumulation of the loop noise over an interval of approximately $1/(2B_L T)$ symbols. In spite of the bounded self–noise, the accumulated self–noise can occasionally assume values which are large enough to cause cycle slips. Hence, when self–noise is present at the equilibrium point(s), we expect the mean time between slips to approach a finite limit when E_b/N_o goes to infinity; this behaviour cannot be predicted by the

Fokker–Planck theory because the self–noise spectrum at the equilibrium point(s) vanishes at $fT = 0$.

In order to obtain results which take into account the self–noise at the equilibrium points, we have used the following ad hoc approach. We have replaced the actual loop noise with spectrum $S(fT;\psi)$ by a fictitious loop noise with flat spectrum $S_{eq}(0;\psi)$ such that same variance is obtained when the actual loop noise and the fictitious loop noise are passed through a filter, with the same transfer function as the closed loop. According to section II.1.2, $S_{eq}(0;\psi)$ should be

$$S_{eq}(0;\psi) = S(0;\psi) + C(\psi) \qquad \text{(VI.2.2–2)}$$

where $C(\psi)$ is the following correction term :

$$C(\psi) = K_F(2B_LT) \sum_m |m|.(-R(m;\psi)) \qquad \text{(VI.2.2–3)}$$

In the above, $R(m;\psi)$ is the autocorrelation of the loop noise :

$$R(m;\psi) = E[\, N(k;\psi)N(k+m;\psi)\,] \qquad \text{(VI.2.2–4)}$$

and K_F depends on the closed–loop transfer function (K_F=2 for a first–order loop). The resulting filtered loop noise variance is given by

$$\begin{aligned} Var &= (2B_LT)\, S_{eq}(0;\psi) \\ &= (2B_LT)\, S(0;\psi) \;+\; K_F(2B_LT)^2 \sum_m |m|\,(-R(m;\psi)) \end{aligned} \qquad \text{(VI.2.2–5)}$$

which is a generalization of eq.(II.1.2–4) to arbitrary ψ.

Because of the usually small values of $2B_LT$, the influence of the correction term $C(\psi)$ is important only for those values of ψ yielding $S(0;\psi) = 0$, i.e. at the stable and unstable equilibrium points. Therefore, we have simplified the correction term $C(\psi)$ to a linear interpolation between $C(0)$ and $C(p/2)$:

$$C(\psi) \cong C(0) + (C(p/2) - C(0)).|\psi|/2p \qquad \text{(VI.2.2–6)}$$

We have numerically verified that the effect on the cycle slip rate is negligible when using (VI.2.2–6) instead of (VI.2.2–3).

VI.2.3 Approximation for small E_b/N_o.

According to (VI.1.3–1, VI.1.3–3), for computing the mean time between slips we need the loop–SNR ρ, the phase (or timing) error detector characteristic $g(\psi)$ and the ratio $S(0;\psi)/S(0;0)$, which represents the DC–value of the loop noise spectrum, normalized by its value at the tracking point $\psi = 0$. Self–noise at the equilibrium points can be taken into account by using $S_{eq}(0;\psi)$, given by (VI.2.2–2), instead of $S(0;\psi)$.

Whereas ρ is simply the inverse of the linearized phase (or timing) error variance, which has been evaluated in part II of this study, the quantities $g(\psi)$ and $S(0;\psi)$ are more difficult to evaluate. In the case of the quadratic NDA carrier and symbol synchronizers, we have been able to obtain the following general result for $g(\psi)$ and $S(0;\psi)$:

$$g(\psi) = \frac{p}{2\pi}\sin(2\pi\psi/p) \qquad \text{(VI.2.3–1a)}$$

$$\frac{S(0;\psi)}{S(0;0)} = 1 + B^2\sin^2(2\pi\psi/p) \qquad \text{(VI.2.3–1b)}$$

where B^2 depends on the specific algorithm, the shape of the matched filter output pulse, and vanishes when E_b/N_o goes to zero. The above result indicates that the phase (or timing) error detector characteristic $g(\psi)$ is sinusoidal with period p, and that $S(0;\psi)/S(0;0)$ is periodic in ψ with period p/2, and reaches its maximum value of $1 + B^2$ at $\psi = p/4$. In case of DD algorithms, we have not been able to calculate $g(\psi)$ and $S(0;\psi)/S(0;0)$ exactly (with the exception of DDML carrier synchronization for BPSK and QPSK). The main complication is the occurrence of quadrature interference and intersymbol interference for $\psi \neq 0$, and its effect on the decisions.

Fortunately, a reasonable approximation, valid for any synchronizer (DD or NDA), can be obtained for small E_b/N_o. Indeed, it has been observed [Mey'90] that for small E_b/N_o, the following approximation holds :

$$g(\psi) \cong \frac{p}{2\pi}\sin(\frac{2\pi}{p}\psi) \qquad \text{(VI.2.3–2a)}$$

$$\frac{S(0;\psi)}{S(0;0)} \cong 1 \qquad \text{(VI.2.3–2b)}$$

In other words, the phase (or timing) error detector characteristic becomes sinusoidal with period p, and the loop noise spectrum $S(0;\psi)$ becomes independent of the error ψ. Using the above approximations in (VI.1.3–1), one simply obtains :

$$2B_L T_s \cong \frac{\pi}{2} \exp\left(\frac{\rho p^2}{2\pi^2}\right) = \frac{\pi}{2} \exp\left(\frac{p^2}{2\pi^2} \frac{1}{\mathrm{Var}[\psi]}\right) = \frac{\pi}{2} \exp\left(\frac{p^2}{2\pi^2} \frac{1}{2B_L T \; S_{eq}(0;0)}\right) \tag{VI.2.3–3}$$

Within the accuracy of the above approximation, synchronizers yielding nearly the same (linearized) tracking error variance give rise to nearly the same cycle slip performance.

Let us investigate (VI.2.3–3) for the cases of carrier and symbol synchronization.

– Carrier synchronization

a) BPSK, OQPSK ($p = \pi$)

$$2B_L T_s \cong \frac{\pi}{2} \exp\left(\frac{\rho}{2}\right) = \frac{\pi}{2} \exp\left(\frac{1}{2} \frac{1}{\mathrm{Var}[\psi]}\right) = \frac{\pi}{2} \exp\left(\frac{1}{2} \frac{1}{2B_L T \; S_{eq}(0;0)}\right) \tag{VI.2.3–4a}$$

b) QPSK ($p = \pi/2$)

$$2B_L T_s \cong \frac{\pi}{2} \exp\left(\frac{\rho}{8}\right) = \frac{\pi}{2} \exp\left(\frac{1}{8} \frac{1}{\mathrm{Var}[\phi]}\right) = \frac{\pi}{2} \exp\left(\frac{1}{8} \frac{1}{2B_L T \; S_{eq}(0;0)}\right) \tag{VI.2.3–4b}$$

– Symbol synchronization ($p = 1$)

$$2B_L T_s \cong \frac{\pi}{2} \exp\left(\frac{\rho}{2\pi^2}\right) = \frac{\pi}{2} \exp\left(\frac{1}{2\pi^2} \frac{1}{\mathrm{Var}[\epsilon]}\right) = \frac{\pi}{2} \exp\left(\frac{1}{2\pi^2} \frac{1}{2B_L T \; S_{eq}(0;0)}\right) \tag{VI.2.3–4c}$$

Comparing (VI.2.3–4a) and (VI.2.3–4b), it follows that, for a given phase error variance (in rad^2), the cycle slip rate for QPSK is much larger than for BPSK and OQPSK. Indeed, for QPSK the stable equilibrium point is only $\pi/4$ away from the unstable equilibrium points, whereas for BPSK and OQPSK the

difference between stable and unstable equilibrium points is $\pi/2$. In order to compare (VI.2.3–4c) with (VI.2.3–4a) and (VI.2.3–4b) in a fair way, we should first convert the timing error ϵ (which is defined as the timing error normalized to the symbol interval T) to rad. This involves the introduction of the timing error $\epsilon' = 2\pi\epsilon$, such that $\mathrm{Var}[\epsilon'] = 4\pi^2\,\mathrm{Var}[\epsilon]$ (in rad^2). This yields

$$2B_L T_s \cong \frac{\pi}{2}\exp\left(\frac{2}{\mathrm{Var}[\epsilon']}\right) \qquad \text{(VI.2.3–4d)}$$

Hence, for a given tracking error variance in rad^2, the symbol synchronizer exhibits less cycle slips than the carrier synchronizers, because its stable equilibrium point is π away from the unstable equilibrium points.

The contribution of cycle slips to the BER becomes critical at small E_b/N_o, for which (VI.2.3–3) is a good approximation. For moderate and large E_b/N_o, the specific shapes of $g(\psi)$ and $S(0;\psi)/S(0;0)$ can be rather different from (VI.2.3–2), so that the approximation (VI.2.3–3) might be questionable. However, for moderate and large E_b/N_o, the contribution of cycle slips to the BER is many orders of magnitude smaller than the total BER, so that the effect of cycle slips becomes negligible. We conclude that (VI.2.3–3) is a simple and useful approximation which predicts the cycle slip performance in the region of small E_b/N_o, where the effect of cycle slips might be critical.

VI.3 CYCLE SLIPS RESULTS FOR FEEDBACK SYNCHRONIZERS

In this section we present numerical results for $1/(2B_LT_s)$, the contribution of cycle slips to the BER, and compare it to $Q(\sqrt{2E_b/N_o})$, the BER in the absence of tracking errors. Cycle slips are expected to be harmful for those values of E_b/N_o where $1/(2B_LT_s)$ has the same order of magnitude as $Q(\sqrt{2E_b/N_o})$. Unless noted otherwise, results hold for $2B_LT = 0.01$ and $K_F = 2$ (i.e. for a first–order loop).

The cycle slip performance for carrier synchronizers is evaluated under the assumption of perfect timing; similarly, perfect carrier recovery is assumed when computing the cycle slip performance of symbol synchronizers. This approach, which ignores the coupling between the carrier and the symbol synchronizer during slips, is valid when cycle slip rates are small.

For small E_b/N_o, the cycle slip performance is well approximated by (VI.2.3–3). In this region of E_b/N_o, the cycle slip performance comparison of various synchronizers can be derived from their tracking performance : synchronizers with a similar tracking error variance give rise to a similar cycle slip rate.

The cycle slip performance of the DD synchronizers is obtained from the approximate expression (VI.2.3–3); exceptions are the DDML carrier synchronizers for BPSK and QPSK, for which the exact $g(\psi)$ and $S(0;\psi)$ are used in (VI.2.2–1) and (VI.1.3–1). The cycle slip performance of the NDA synchronizers is computed from (VI.1.3–1), making use of the exact $g(\psi)$ and of $S_{eq}(0;\psi)$, the loop noise power spectral density with the correction for self–noise at the equilibrium points.

Computer simulation results have been obtained for a few cases, by counting the number of slips that occur during a run of 10^7 symbols. For $2B_LT = 0.01$, this means that values of $2B_LT_s$ larger than 10^5 cannot be measured, and values larger than 10^4 are not very reliable (less than 10 slips occur).

VI.3.1 Feedback carrier synchronizers

– BPSK (Fig.VI.3–1)

The DDML and NDAML synchronizer yield a similar cycle slip performance (which is independent of the rolloff α), because their tracking error variances are nearly the same; the NDAML synchronizer is slightly better than the DDML synchronizer, by about 1 dB in E_b/N_o. For E_b/N_o>0 dB, the effect of cycle slips is negligibly small.

– QPSK (Fig.VI.3–2)

The DDML and NDAML synchronizer yield a very similar cycle slip performance (which is independent of the rolloff α), because their tracking error variances are nearly the same; they differ by less than 0.5 dB in E_b/N_o. For E_b/N_o>2 dB, the effect of cycle slips is negligibly small.

– OQPSK (Fig.VI.3–3)

The DDML and NDAML synchronizer yield a substantially different cycle slip performance, especially for small rolloff. The better cycle slip performance of the DDML synchronizer is a consequence of its smaller tracking error variance, as compared to the NDAML synchronizer. As the tracking performance of the NDAML synchronizer becomes considerably worse with decreasing rolloff, the same behaviour is observed for its cycle slip performance. For E_b/N_o>0 dB, the effect of cycle slips is negligibly small for the DDML synchronizer, irrespective of the rolloff, and also for the NDAML synchronizer, for moderate (α = 0.5) and large (α = 1) rolloff. For small (α = 0.1) rolloff, the NDAML synchronizer slips cycles so often that its contribution to the BER cannot be ignored.

Computer simulations have been performed for the NDAML synchronizer. For α = 0.5 and E_b/N_o = –5 dB, the simulation result coincides with the analytical result. For α = 0.1 and E_b/N_o = 2 dB, much more cycle slips occurred than predicted by analysis; as explained at the end of section III.4, in this case the true tracking error variance is much larger than the linearized tracking error variance, so that also much more cycle slips can be expected.

We have not derived numerical results for other carrier synchronizers such as the V&V synchronizer and the NDAMLT̄B synchronizer (feedback versions !). In the cases of BPSK and QPSK, the V&V synchronizers yield a similar tracking error variance as the corresponding NDAML synchronizer, and therefore, they have a similar cycle slip performance. The same observation holds for the NDAMLT̄B synchronizer. In the case of OQPSK, the V&V synchronizer gives

rise to a similar tracking error variance as the NDAML synchronizer, for moderate ($\alpha = 0.5$) and large ($\alpha = 1$) rolloff, so that their cycle slip performances are expected to be similar too; for small ($\alpha = 0.1$) rolloff, the tracking error variance of the V&V synchronizer is much smaller than for the NDAML synchronizer, so that its cycle slip performance will be considerably better.

VI.3.2 Feedback symbol synchronizers

In this section we consider the cycle slip performance of symbol synchronizers, assuming BPSK transmission. In the case of (O)QPSK transmission, the tracking error variance at a given E_b/N_o is half as large as for the corresponding synchronizer for BPSK transmission; consequently, the loop–SNR ρ for (O)QPSK is twice as large as for BPSK, so that (VI.1.3–1) yields for (O)QPSK a value of $2B_LT_s$ which is roughly the square of the value for BPSK.

– <u>DD symbol synchronizers</u> (Figs. VI.3–4 and VI.3–5)

The DDML, DDEL(λ=1/2) and DTTL synchronizers have a similar tracking performance, and, therefore, also a similar cycle slip performance. Performance becomes worse with decreasing rolloff. For E_b/N_o>0 dB, the contribution of cycle slips to the BER is negligibly small.

The DDEL(λ=1) and mM&M synchronizer have identical tracking performance, which is very close to that of the M&M synchronizer. Consequently, these three synchronizers yield a similar cycle slip performance. Performance becomes worse with decreasing rolloff. For α = 0.1 and α = 0.5, we need E_b/N_o>2 dB before the contribution of cycle slips to the BER is negligibly small; for α = 1, the effect of cycle slips cannot be ignored.

Computer simulations have been performed for the M&M synchronizer (see Fig. VI.3–5). For α = 0.5 and E_b/N_o = 3 dB, no slips have been detected during 10^7 symbols, indicating that $2B_LT_s > 10^5$. Taking into account that the analytical result follows from the approximate expression (VI.2.3–4c), the agreement between analysis and simulation is satisfactory.

– <u>NDA symbol synchronizers</u> (Fig. VI.3–6)

The NDAEL(λ=1/2) and GA synchronizers yield an identical cycle slip performance, because they are in fact identical synchronizers, with interchanged stable and unstable equilibrium points. Indeed, for the NDAEL(λ=1/2) synchronizer the timing error detector output is given by (see Table T.5)

$$\begin{aligned} u_\tau(k;\theta,\hat\tau) &= p_R(k;\hat\tau,\theta)[p_R(k+\tfrac{1}{2};\hat\tau,\theta) - p_R(k-\tfrac{1}{2};\hat\tau,\theta)] \\ &\quad + \quad q_I(k;\hat\tau,\theta)[q_I(k+\tfrac{1}{2};\hat\tau,\theta) - q_I(k-\tfrac{1}{2};\hat\tau,\theta)] \\ &= p_R(k;\hat\tau,\theta)[p_R(k;\hat\tau+\tfrac{T}{2},\theta) - p_R(k;\hat\tau-\tfrac{T}{2},\theta)] \\ &\quad + \quad q_I(k;\hat\tau,\theta)[q_I(k;\hat\tau+\tfrac{T}{2},\theta) - q_I(k;\hat\tau-\tfrac{T}{2},\theta)] \end{aligned}$$

whereas for the GA synchronizer one has

$$u_\tau(k;\theta,\hat\tau) = p_R(k+\tfrac{1}{2};\hat\tau,\theta)[p_R(k;\hat\tau,\theta) - p_R(k+1;\hat\tau,\theta)] + q_I(k+\tfrac{1}{2};\hat\tau,\theta)[q_I(k;\hat\tau,\theta) - q_I(k+1;\hat\tau,\theta)]$$
$$= p_R(k;\hat\tau+\tfrac{T}{2},\theta)[p_R(k;\hat\tau,\theta) - p_R(k;\hat\tau+T,\theta)] + q_I(k;\hat\tau+\tfrac{T}{2},\theta)[q_I(k;\hat\tau,\theta) - q_I(k;\hat\tau+T,\theta)]$$

In the above, $p_R(k;\tilde\tau,\theta)$ and $q_I(k;\tilde\tau,\theta)$ denote the real and imaginary part of the complex–valued phase–rotated matched filter output samples $p(k;\tilde\tau)\exp(-j\theta)$ and $q(k;\tilde\tau)\exp(-j\theta)$, respectively. Now, evaluating the NDAEL(λ=1/2) timing error detector output at $\hat\tau + T/2$ yields :

$$u_\tau(k;\theta,\hat\tau+\tfrac{T}{2}) = p_R(k;\hat\tau+\tfrac{T}{2},\theta)[p_R(k;\hat\tau,\theta) - p_R(k;\hat\tau+T,\theta)] + q_I(k;\hat\tau+\tfrac{T}{2},\theta)[q_I(k;\hat\tau,\theta) - q_I(k;\hat\tau+T,\theta)]$$

which is, within a sign reversal, the GA timing error detector output. In other words, changing the polarity of the feedback turns the NDAEL(λ=1/2) synchronizer into the GA synchronizer, and vice versa. Consequently, they have the same cycle slip performance.

The NDAML, NDAEL(λ=1/2) and GA synchronizers have a similar tracking performance, and, hence, a similar cycle slip performance. The somewhat better cycle slip performance of the NDAML synchronizer is due to the lower degree of peaking of $S(0;\psi)$ at $\psi = p/4$, as compared to the NDAEL(λ=1/2) and GA synchronizer. For E_b/N_o>0 dB, the contribution of cycle slips to the BER is negligibly small for $\alpha = 0.5$ and $\alpha = 1$; for $\alpha = 0.1$ the effect of cycle slips cannot be ignored.

We have not evaluated cycle slip performance for the NDAML$\overline{C}$B/Q, NDAEL(λ=1/2)$\overline{C}$B/Q, GA$\overline{C}$B/Q and NDANLFB$\overline{C}$B/Q synchronizers.

– The NDANLFB$\overline{C}$B synchronizer has a similar tracking performance as the M&M synchronizer for BPSK, and, therefor, a similar cycle slip performance is expected. The NDANLFB$\overline{C}$Q synchronizer yields a tracking error variance which is about twice as large as for the M&M synchronizer for QPSK transmission. Hence, for a given E_b/N_o, the NDANLFB$\overline{C}$B and NDANLFB$\overline{C}$Q synchronizer have about the same tracking performance, so that their cycle slip performances are expected to be similar.

– The NDAML$\overline{C}$Q, NDAEL(λ=1/2)$\overline{C}$Q and GA$\overline{C}$Q synchronizers are mathematically equivalent to the NDAML, NDAEL(λ=1/2) and GA synchronizers for QPSK, and, therefore, have the same cycle slip performance. The NDAML$\overline{C}$B, NDAEL(λ=1/2)$\overline{C}$B and GA$\overline{C}$B synchronizers yield a tracking

error variance which is similar to that of the NDAML, NDAEL(λ=1/2) and GA synchronizers for BPSK; consequently their cycle slip performances are expected to be similar.

VI.3.3 Miscellaneous

In this subsection, the following items are considered : the accuracy of the approximation (VI.2.3–3), the effect of the loop bandwidth on cycle slip performance, and an example of cycle slip computation.

Accuracy of (VI.2.3–3)

For most DD synchronizers, the exact $g(\psi)$ and $S(0;\psi)$ are very difficult to calculate, because of the occurrence of quadrature interference or ISI, and their effect on the receiver's decisions. Therefore, the approximate expression (VI.2.3–3) has been used to compute the cycle slip rate.

In the case of DDML carrier synchronization for BPSK and QPSK, the correct $g(\psi)$ and $S(0;\psi)$ can be computed, and the cycle slip performance calculated from (VI.1.3–1). Fig. VI.3–7 compares the cycle slip performances obtained from (VI.1.3–1) and (VI.2.3–3). The difference between these results is less than 1 dB in E_b/N_o.

Effect of loop bandwidth

It follows from (VI.2.2–1) that the effect of the loop bandwidth on the cycle slip rate is considerable. Indeed, halving the loop bandwidth doubles ρA, so that $2B_LT_s$ from (VI.1.3–1) is approximately squared; for instance, $2B_LT_s$ changes from 10^5 to about 10^{10}. The effect of changing the loop bandwidth is illustrated in Fig. VI.3–8. The corresponding mean time between slips at E_b/N_o = 0 dB is given in Table VI.3–1.

Figs. VI.3–9 to VI.3–11 show that using $2B_LT = 0.003$ considerably improves the cycle slip performance for those synchronizers having an excessive cycle slip rate at $2B_LT = 0.01$ (compare with Figs. VI.3–3b, VI.3–5b and VI.3–6b).

Example of cycle slip computation

Let us consider the M&M symbol synchronizer for BPSK, operating at $E_b/N_o = 5$ dB, with $\alpha = 1$. We will compute the cycle slip performance from (VI.2.3–4c), for $2B_LT = 0.01$ and $2B_LT = 0.003$.

From Fig. II.3–9, we see that the normalized loop noise power spectral density $S(0;0)$ is about –0.8 dB (or 0.832) at $E_b/N_o = 5$ dB. As there is no self–noise at the equilibrium point, the linearized tracking error variance $\mathrm{Var}[\epsilon]$ equals $2B_LT\,S(0;0)$. From (VI.2.3–4c) we obtain the following results :

a) $2B_LT = 0.01$

$2B_LT_s = 693$ or $-\log(2B_LT_s) = -2.84$

Hence, T_s is about $7\ 10^4$ symbols, which corresponds to 70 ms when transmitting at 1Msymb/s.

b) $2B_LT = 0.003$

$2B_LT_s = 1.03\ 10^9$ or $-\log(2B_LT_s) = -9.01$

Hence, T_s is about $3.4\ 10^{11}$ symbols, which corresponds to almost 10 hours when transmitting at 1Msymb/s.

Note that the values for $-\log(2B_LT_s)$ are in agreement with Figs VI.3–5b and VI.3–10.

DDML carrier synchronization, QPSK transmission		
$2B_LT$	T_s/T @ $E_b/N_o = 0$ dB	T_s @ $E_b/N_o = 0$ dB, 1 Msymb/s
0.02	$5\ 10^3$	5 ms
0.01	$6\ 10^5$	0.6 s
0.005	$1.1\ 10^{10}$	3 hours

Table VI.3–1

VI.4 FEEDFORWARD SYNCHRONIZERS

VI.4.1 Introduction

An important question is whether we have cycle slips also in feedforward synchronizers. The answer is not just simply yes or no. The difficulty is already in the nature of the phase. For the sake of simplicity we start with a feedforward carrier synchronizer for an unmodulated carrier. With two phase estimates determined at two arbitrary instants in time and their corresponding actual carrier phase values (measured for instance via a noise free reference path) we cannot find out whether there was a cycle slip in between or not. The reason is that phase (and phase error) can be measured modulo 2π only (we will use the basic interval $[+\pi, -\pi)$ here, meaning that $+\pi$ belongs to the interval, but $-\pi$ does not). But when measuring cycle slips we take the complete phase (error) trajectory into account, also counting cycles (i.e. transitions through $\pm\pi$).

Therefore, with phase estimates available at discrete instants in time it is not possible to define and measure cycle slips reasonably when we do not specify how the trajectory is generated from the samples. The reasonable assumption may be hidden in a synchronizer control program, in the overall synchronizer structure, or defined by the postprocessing of the phase estimates. When we have a 'dense' spacing of the phase estimates, e.g. with a gliding accumulator we may obtain one estimate per data symbol, the extension may be a very natural one – the difference between consecutive estimates is assumed to be less than π, e.g. when after a phase estimate of $+170^{\circ}$ we obtain a phase estimate of -175° we assume that this actually is $+185^{\circ} = -175^{\circ} + 360^{\circ}$. Thus, in the postprocesing we generate an extension of the basic interval $[+\pi, -\pi)$.

VI.4.2 Feedforward carrier synchronizers

Before we study the different carrier synchronization algorithms we have to discuss the interaction between detection and synchronization in the receiver in more detail. When we do not have known symbols in the data sequence there is always an ambiguity in the detection, e.g. for QPSK the scatter diagram may be rotated by an arbitrary multiple of $\pi/2$. In the general case we have at the input of the decision device

$$d_k = [\, a_k + j\, f(\underline{b})\,] \exp(\, j(\theta - \hat{\theta} + 2\pi m/M)) + n_k \;; \quad m \text{ integer} \tag{VI.4.2–1}$$

In the above, $\underline{b}$ denotes the vector consisting of all quadrature symbols, $f(\underline{b})$ is given by

$$f(\underline{b}) = \begin{cases} 0 & \text{BPSK} \\ \sum\limits_m b_{k-m}\, g(mT - \frac{T}{2}) & \text{OQPSK} \\ b_k & \text{QPSK} \end{cases} \tag{VI.4.2–2}$$

and M=4 for QPSK and M=2 for BPSK and OQPSK. The ambiguity can either be resolved by inserting known symbols (e.g. a preamble, a unique word also used for block synchronization, etc.) or its influence is removed using differential decoding (with appropriate encoding on the transmit side). When, after operating around zero, the carrier phase <u>error</u> reaches $+2\pi/M$ (or $-2\pi/M$) we again have a stable operating point, i.e. we have observed a cycle slip. Therefore, a drift of the phase error tracking point by $2\pi/M$ corresponds to a cycle slip. Without differential decoding we need ambiguity resolution before we again have correctly decoded data symbols, i.e. we have to determine the new value of m (which is the old value of m ±1).

This ambiguity is also of importance for the postprocessing of the phase estimates. With the nondata–aided algorithms the phase estimates are limited to $[+\pi/M, -\pi/M)$ since the output of the argument function, which is limited to $[+\pi, -\pi]$, is divided by M to finally obtain the phase estimate. The ambiguity must either be resolved for each phase estimate separately or we must observe the phase estimate <u>trajectory</u> to maintain the correct reference position (as given by m). With decision–directed algorithms we may use decisions based on previous

phase estimates to determine new phase estimates. Then, ambiguity resolution is automatically maintained until a $2\pi/M$ cycle slip occurs. These effects will be discussed in more detail in the following subsections.

Nondata–aided carrier synchronizers

Let us first assume the case that the accumulator filter outputs one phase estimate per block of K symbols. This means that we can tolerate only a small carrier frequency error in the down converted signal, otherwise the phase error at the beginning and at the end of the accumulation interval may be too large. Note that the estimate is unbiased for the centre of the accumulation interval.

A special case now is burst transmission with blocks of length K, i.e. we only have one phase estimate per burst. Since we usually also have some kind of preamble or known symbols we can have separate ambiguity resolution in each block. In such a system it does not make sense to define cycle slips, which would be the cycles lost or added between consecutive bursts. As soon as we use more than one estimate per burst, burst transmission and continuous data transmission are subject to the same (cycle slip) effects. The effects may only differ quantitatively.

Let us further assume that the frequency difference of the down converted signal is zero, i.e. the local oscilator frequency perfectly matches the received signal carrier frequency. Then we have a constant phase at the input of the carrier synchronizer.

As discussed above, with the NDA type synchronization algorithms we obtain a phase estimate in the range $[+\pi/M, -\pi/M)$ where M is 4 for QPSK and M=2 for BPSK, OQPSK. Now assume that the phase to be estimated is close to the boundary, e.g. slightly below π/M. Due to noise the estimated phase will vary around its correct value which also includes possible phase estimates at values close to $-\pi/M$. But values close to $-\pi/M$ correspond to a different reference position than values close to $+\pi/M$, i.e. we have a rotation of the scatter diagram (by $2\pi/M$) between such consecutive blocks or, equivalently, different values for m in (VI.4.2–1). With differential encoding this yields one symbol error on the block boundary, without differential encoding we must have ambiguity resolution in each block to avoid complete loss of information in the corresponding blocks.

When we process the phase estimates of each block independently (with separate ambiguity resolution), the notion of a cycle slip does not make sense since we explicitly exclude the full trajectory from consideration. This is not always an acceptable solution. Therefore, in general, we must try to extend the basic range $[+\pi/M, -\pi/M)$, for example by an algorithm which assumes that a

value close to $-\pi/M$ following a value close to $+\pi/M$ actually is a value slightly above $+\pi/M$, as shown in Fig. VI.4–1. Using a gliding accumulator provides a denser spacing and, thus, eases the extension. A postfilter as discussed in Part IV predicts the next phase estimate based on the previous measurements (internally using a frequency estimate) and extends the basic range by assuming that the absolute value of the prediction error is less than π/M (this uniquely defines the extension). The extension is easily implemented by applying a sawtooth nonlinearity with period $2\pi/M$ to the prediction error.

In all these cases we generate a trajectory which allows continuation outside the range $\pm\pi/M$. Having a phase estimate trajectory, the phase error trajectory is obtained by subtracting the actual phase trajectory. When the phase error trajectory exceeds $\pm 2\pi/M$, having started around zero after ambiguity resolution, a cycle slip has occurred.

Note that the cycle slip phenomenon already occurs even when we assume zero frequency error. Knowing that there is no frequency error we could limit the phase estimate range after ambiguity resolution to avoid cycle slips. While this might be a good strategy for multiple phase measurements per burst (with small frequency error), this is not practical for continuous data transmission. This shows that cycle slipping performance depends not only on the synchronization algorithm used but also on the postprocessing which may be very sophisticated and tailored to particular situations.

Decision–directed carrier synchronizers

With the decision–directed algorithms we do not have a pure feedforward algorithm since we have a decision feedback as shown in Fig. VI.4–2 in principle. As with the NDA algorithms stable operating points occur with a period of $2\pi/M$. But now we inherently have memory in the system : decisions are based on a predicted phase value, and the decisions are used to determine new phase estimates. During acquisition the synchronizer will take on a particular reference position, i.e. a particular value for m in (VI.4.2–1), which is maintained as long as a cycle slip does not occur. Therefore, the phase estimate is obtained modulo 2π and, thus, the extension must be based on the interval $[+\pi, -\pi)$. For instance, with the postfilter described in section IV the sawtooth nonlinearity has period 2π.

Although the basic interval is $[+\pi, -\pi)$, the phase error is periodic with $2\pi/M$. The reason is that for a phase error above π/M the decision unit will, in the mean, decide for wrong symbols such that the absolute value of the phase error is reduced $2\pi/M$. With the reference rotated by $2\pi/M$ we obtain a stable operating point for a phase error of $2\pi/M$: a cycle slip has occured. Note that without differential decoding ambiguity resolution is required before valid symbols are obtained again. As discussed with the NDA synchronization algorithms, cycle slipping performance of the feedforward synchronizers depends not only on the synchronization algorithm but also on the postprocessing which may be very sophisticated.

VI.4.3 Feedforward symbol synchronizers

In the case of symbol synchronization we do not have the problem of ambiguity resolution, the periodicity of all timing detectors equals 1 (note that the timing estimate is assumed to be normalized to the symbol duration T). Therefore, there is no more difference in principle between DD and NDA algorithms.

As with the carrier synchronization we must extend the basic interval of [+0.5, –0.5) when we process multiple timing estimates to avoid loss or duplication of symbols (= timing error cycle slips). Only with 'single shot' measurements is such an extension not required, but then we only get the deviation from a correct sampling instant. The information about the start of the transmission must be obtained from an additional preamble detection (frame synchronization).

The most important point is that when we process more than one timing estimate we will have to extend the basic interval [+0.5, –0.5) similarly as for the feedforward carrier synchronizers. Thus, we generate a timing trajectory and as a consequence have a timing error trajectory which is subject to symbol timing cycle slips (symbol slips). The actual cycle slip rate depends on the timing synchronization algorithm used for timing estimation <u>and</u> the postprocessing of the estimates.

VI.5 CONCLUSIONS

In a feedback synchronizer, the loop noise occasionally pushes the phase (or timing) error from a stable equilibrium point towards a neighbouring stable equilibrium point; this phenomenon is called a cycle slip. Cycle slips give rise to error bursts at the output of the decision device. Denoting the loop bandwidth and the mean time between slips by B_L and T_s, the contribution of cycle slips to the BER is in the order of $1/(2B_L T_s)$. This contribution should be smaller, by several orders of magnitude, than the BER corresponding to perfect synchronization.

Fokker–Planck theory shows that the mean time between slips in a first–order feedback loop is an exponentionally increasing function of the loop–SNR.

For $2B_L T = 0.01$, we have evaluated $1/(2B_L T_s)$, resulting from various first–order feedback carrier and symbol synchronizers. In most cases, the resulting cycle slips have a negligible effect on the BER for $E_b/N_o > 0$ dB. Important exceptions, for which the cycle slips cannot be ignored, are :

– The NDAML carrier synchronizer for OQPSK, operating at small rolloff.

– The quadratic NDA symbol synchronizers, operating at small rolloff.

– The DDEL(λ=1), M&M, mM&M and NDANLFB$\overline{C}$B/Q symbol synchronizers, operating at large rolloff.

For these synchronizers, an acceptable cycle slip performance can be obtained by reducing the synchronizer bandwidth, because this increases the loop–SNR.

The cycle slipping behaviour of second–order feedback synchronizers depends on the loop damping ζ. For $\zeta < 1$, cycle slips tend to occur in bursts, especially at low signal–to–noise ratios; these bursts can cause drop of lock. For $\zeta > 1$, cycle slips are independent isolated events, as for a first–order loop, and the cycle slip rate is much the same as for a first–order loop.

Cycle slipping in feedforward synchronizers depends not only on the specific synchronizer which makes a phase or timing estimate, but also on the postprocessing of these estimates. As no analysis methods are available from the literature, only qualitative considerations have been presented.

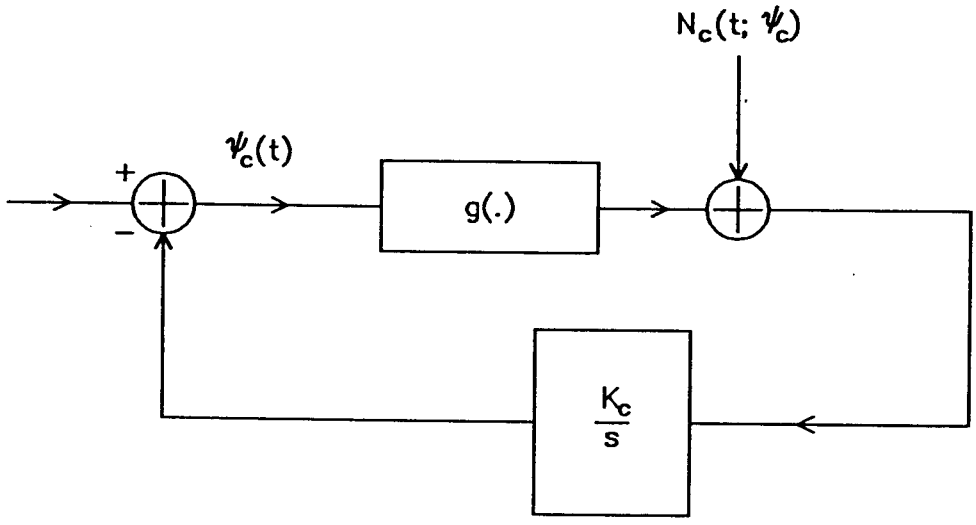


Fig. VI.1–1 : Equivalent model of first–order PLL

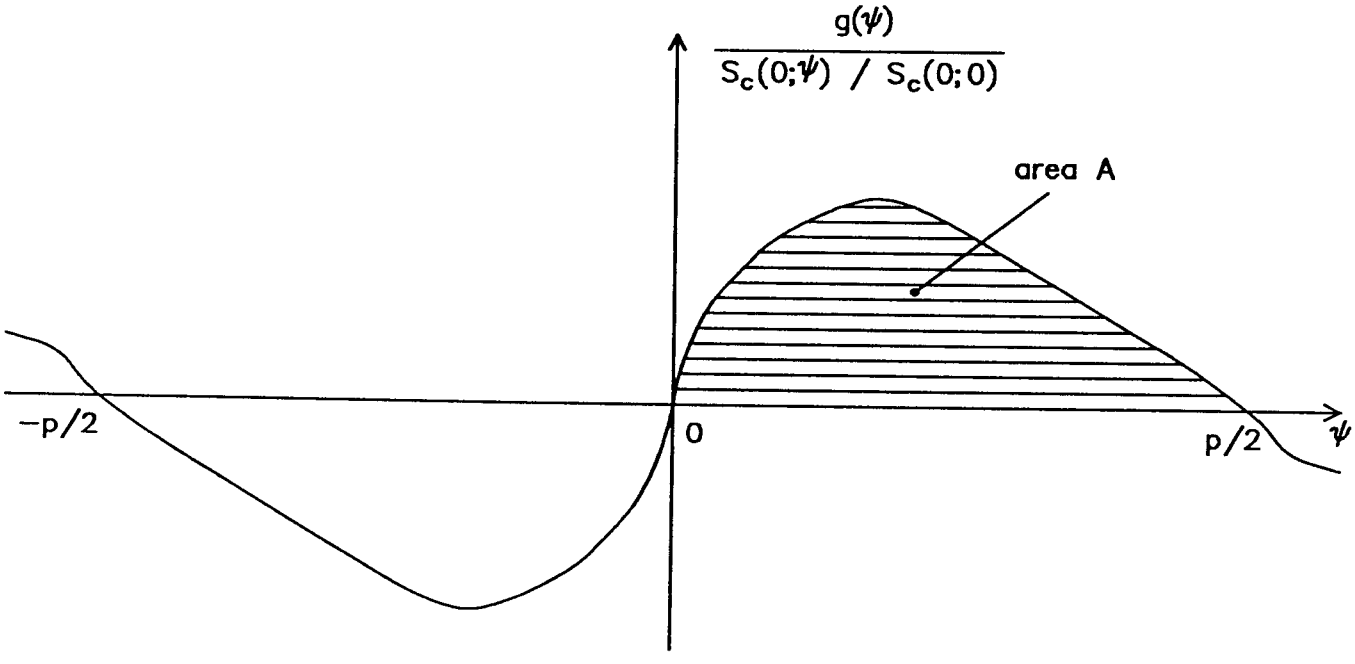
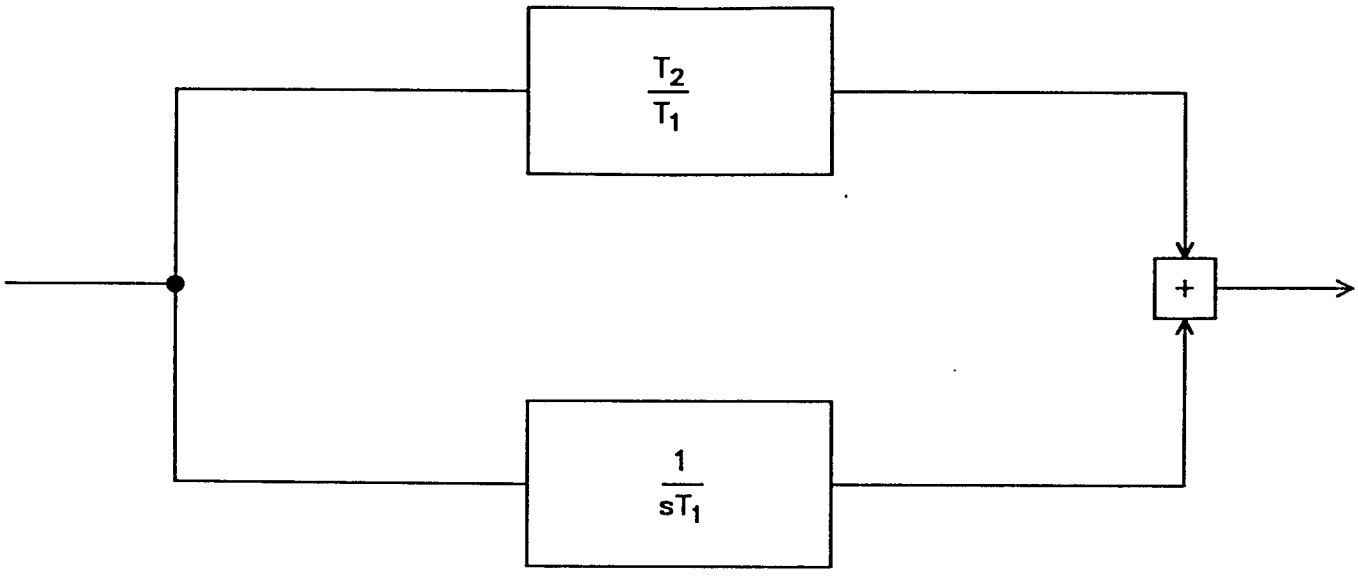


Fig. VI.1–2 : Interpretation of the area A



$$F(s) = \frac{1 + ST_2}{ST_1}$$

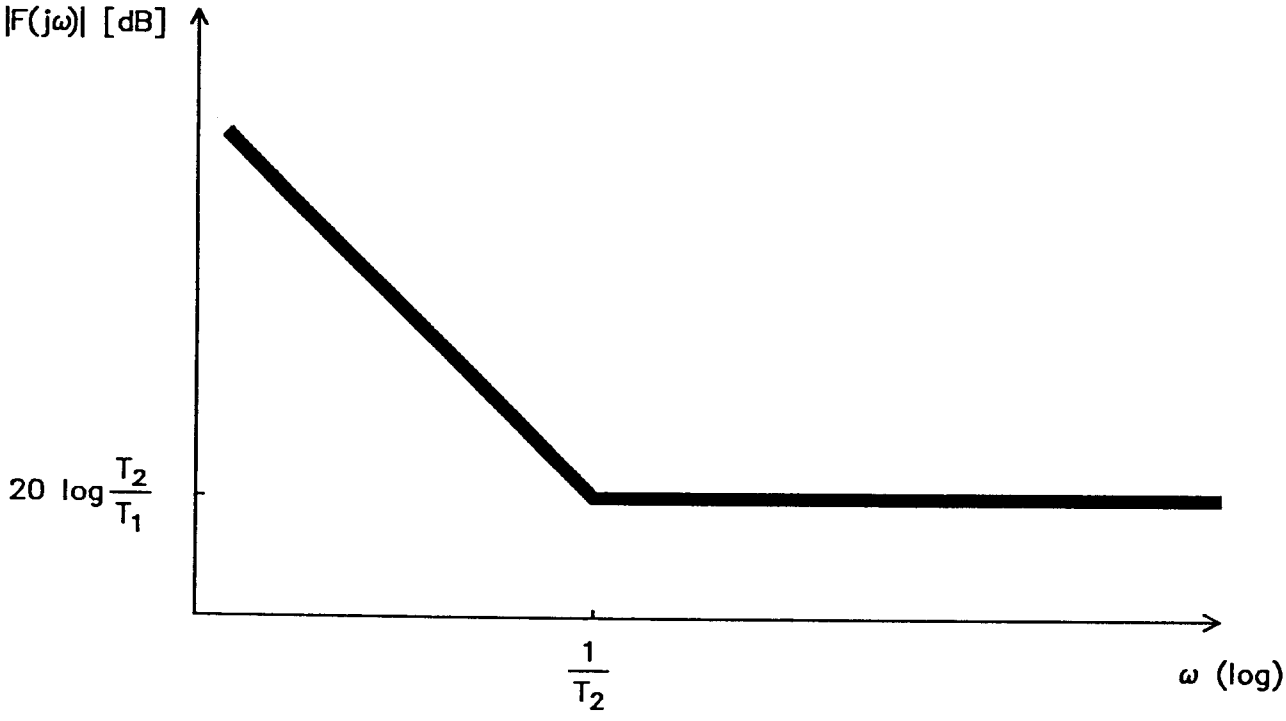


Fig. VI.1–3 : Integrating loop filter

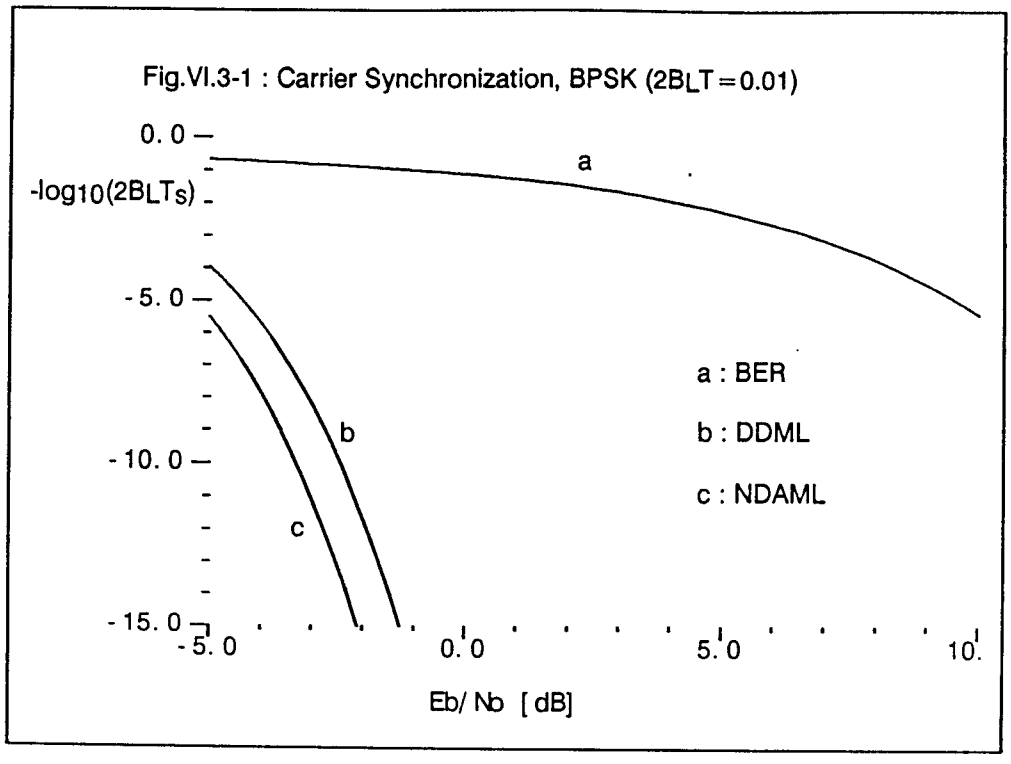


Fig.VI.3-1 : Carrier Synchronization, BPSK (2BLT=0.01)
0. 0
-log10(2BLTs)
-5. 0
-10. 0
-15. 0
-5. 0
0. 0
5. 0
10.
Eb/ No [dB]
a
b
c
a : BER
b : DDML
c : NDAML

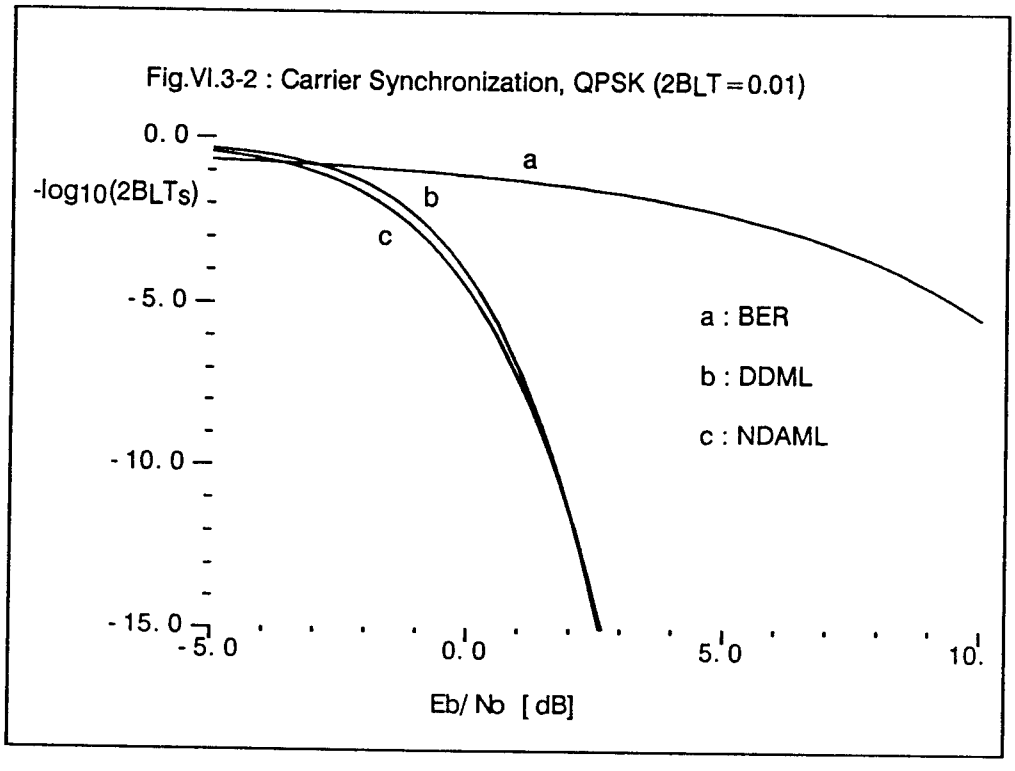


Fig.VI.3-2 : Carrier Synchronization, QPSK (2BLT=0.01)
0. 0
-log10(2BLTs)
-5. 0
-10. 0
-15. 0
-5. 0
0. 0
5. 0
10.
Eb/ No [dB]
a
b
c
a : BER
b : DDML
c : NDAML

Fig.VI.3-3a : DDML Carrier Synchronization, OQPSK ($2B_LT = 0.01$)

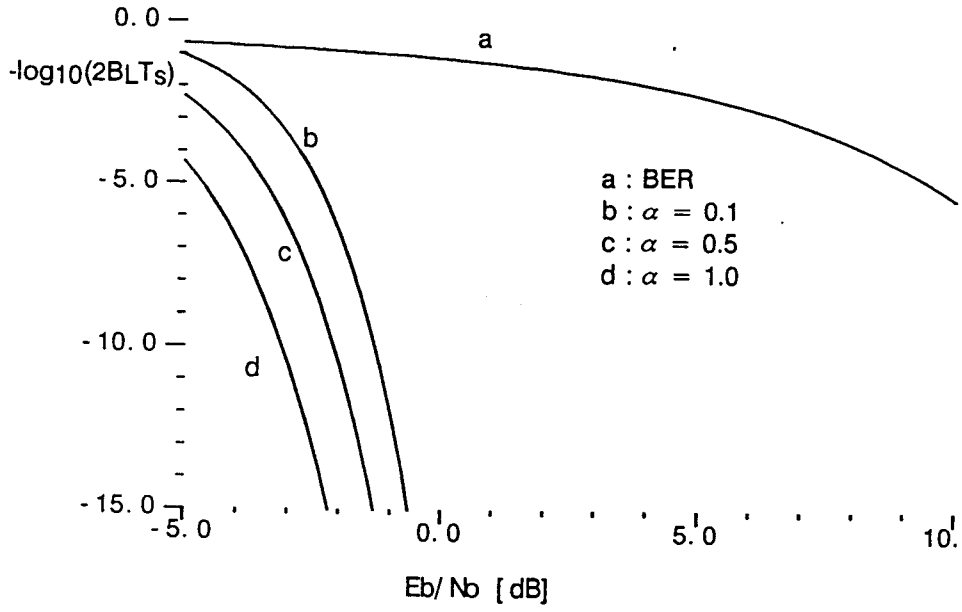


Fig.VI.3-3b : NDAML Carrier Synchronization, OQPSK ($2B_LT = 0.01$)

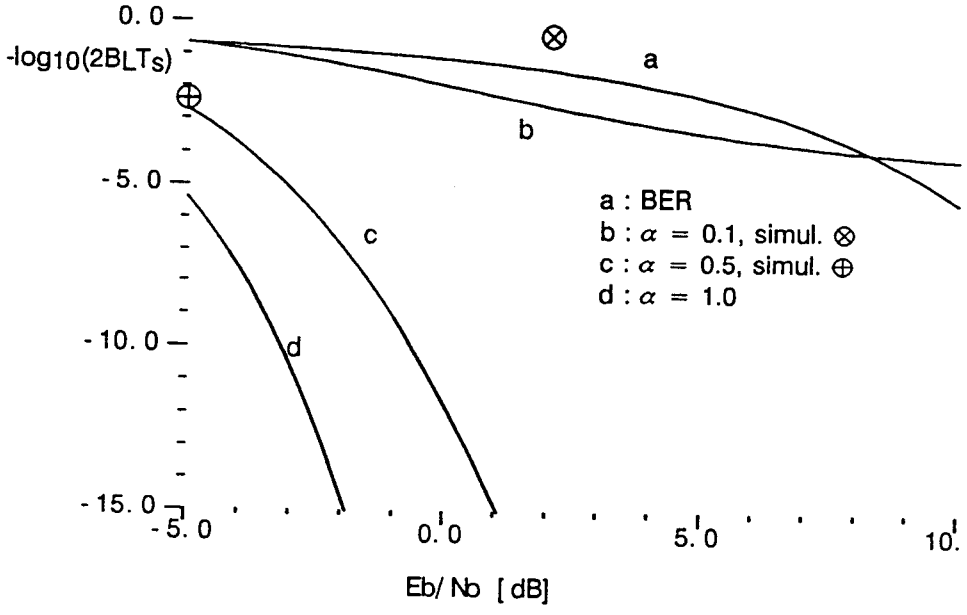


Fig.VI.3-4a : DDML Symbol Synchronization, BPSK ($2B_LT = 0.01$)

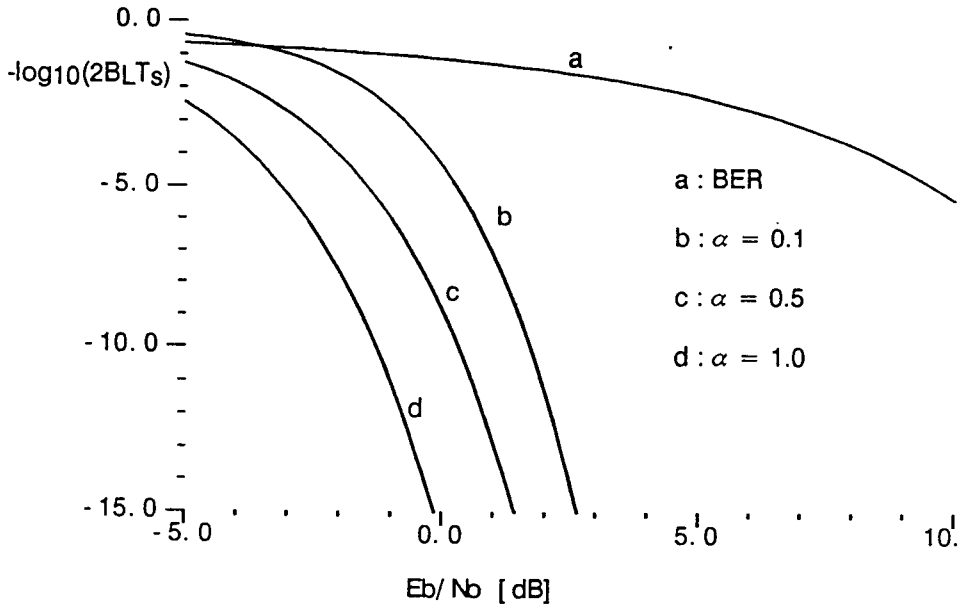
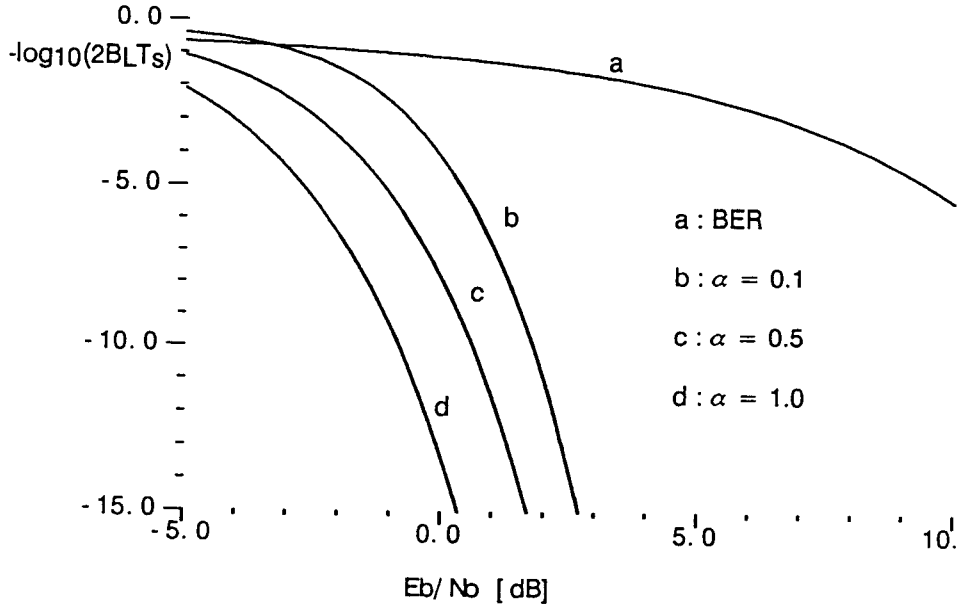


Fig.VI.3-4b : DDEL($\lambda = 1/2$) Symbol Synchronization, BPSK ($2B_LT = 0.01$)



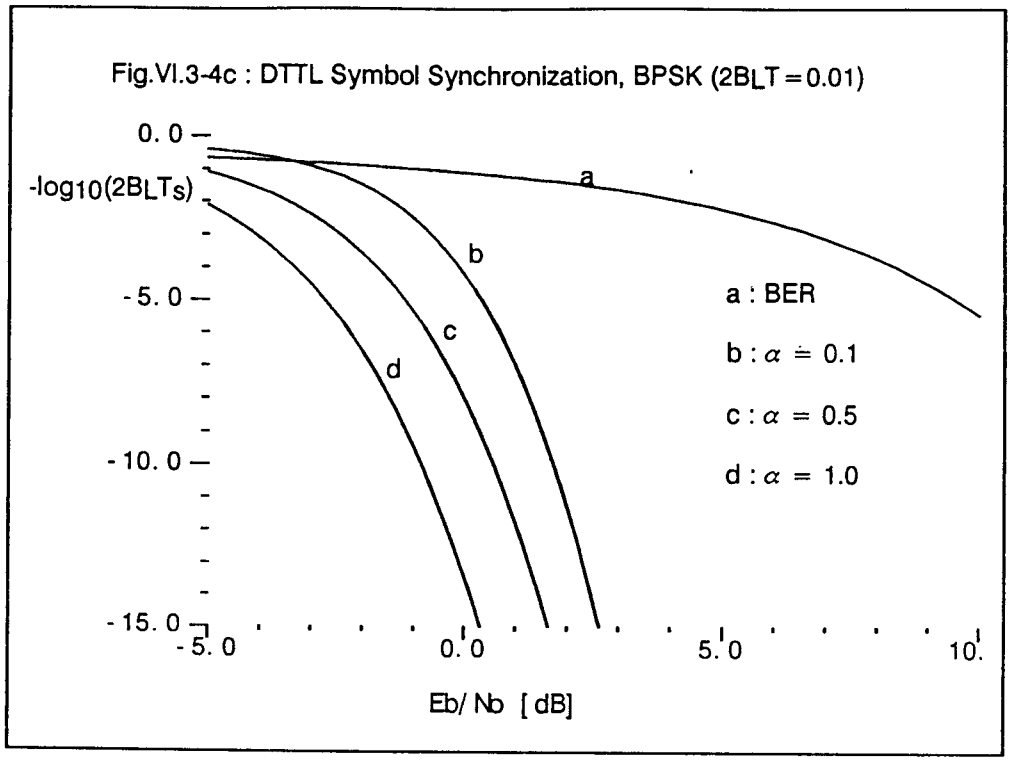


Fig.VI.3-4c : DTTL Symbol Synchronization, BPSK (2BLT = 0.01)
0. 0
-log10(2BLTs)
-5. 0
-10. 0
-15. 0
-5. 0
0. 0
5. 0
10.
Eb/ No [dB]
a
b
c
d
a : BER
b : α = 0.1
c : α = 0.5
d : α = 1.0

Fig.VI.3-5a : DDEL($\lambda = 1$) and mM&M Symbol Synchronization, BPSK ($2B_LT = 0.01$)

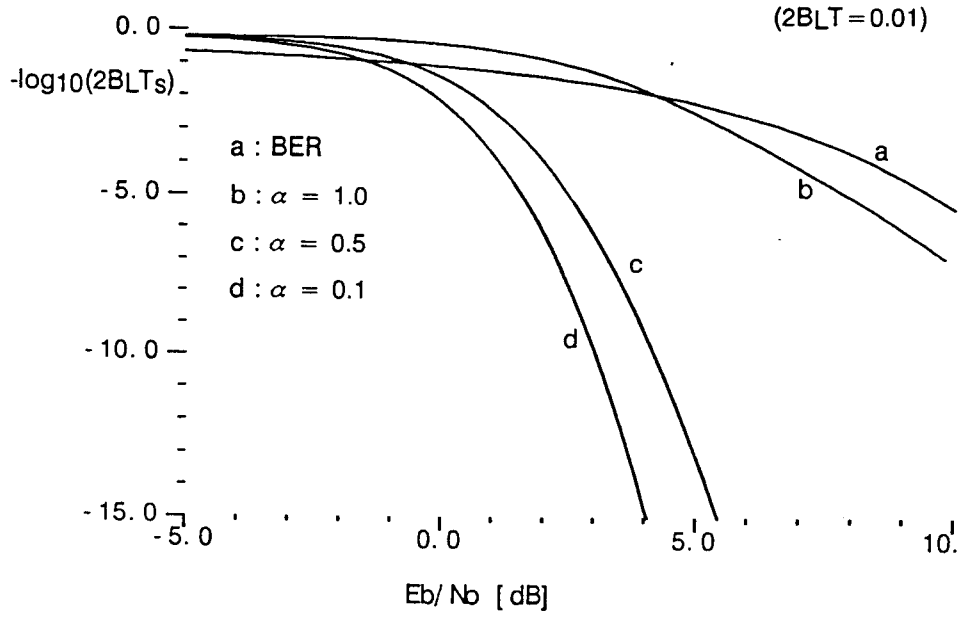
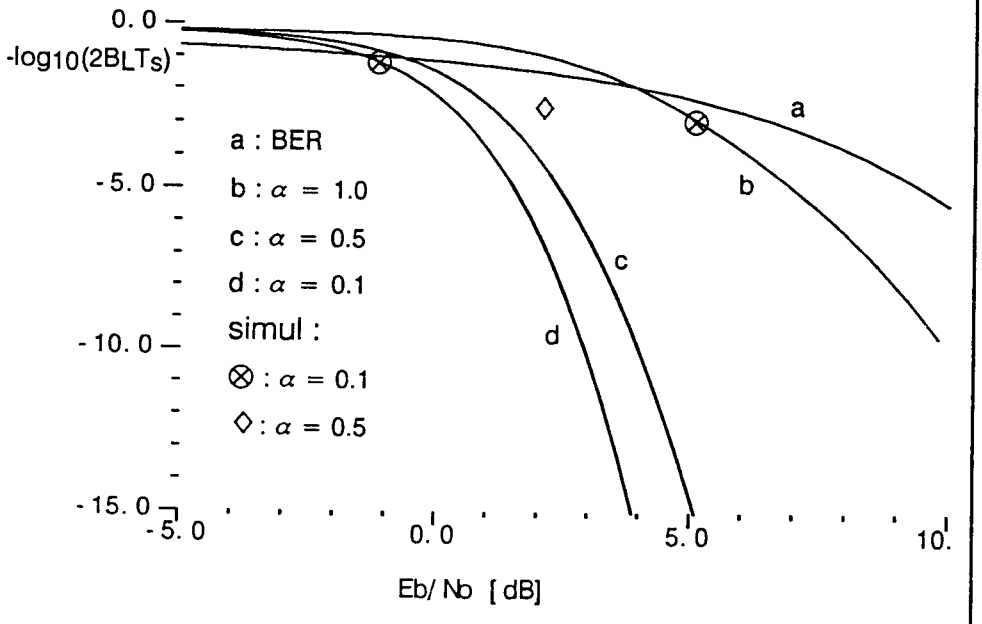


Fig.VI.3-5b : M&M Symbol Synchronization, BPSK ($2B_LT = 0.01$)



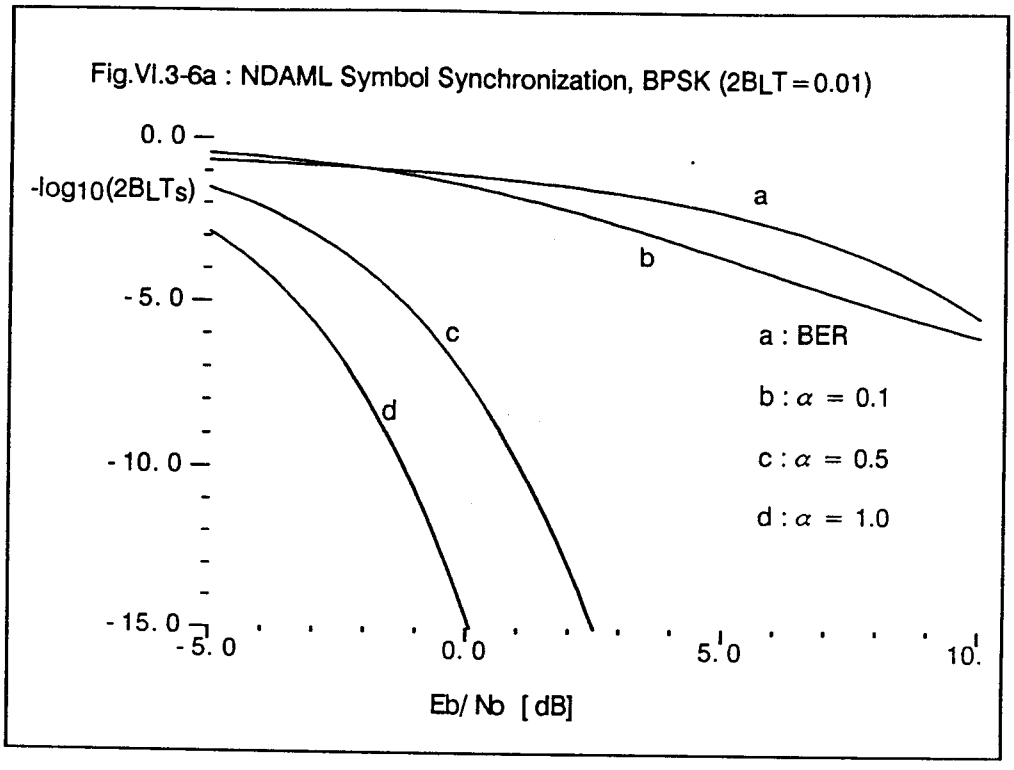


Fig.VI.3-6a : NDAML Symbol Synchronization, BPSK (2BLT=0.01)
0. 0
-log10(2BLTs)
-5. 0
-10. 0
-15. 0
-5. 0
0. 0
5. 0
10.
Eb/ No [dB]
a
b
c
d
a : BER
b : α = 0.1
c : α = 0.5
d : α = 1.0

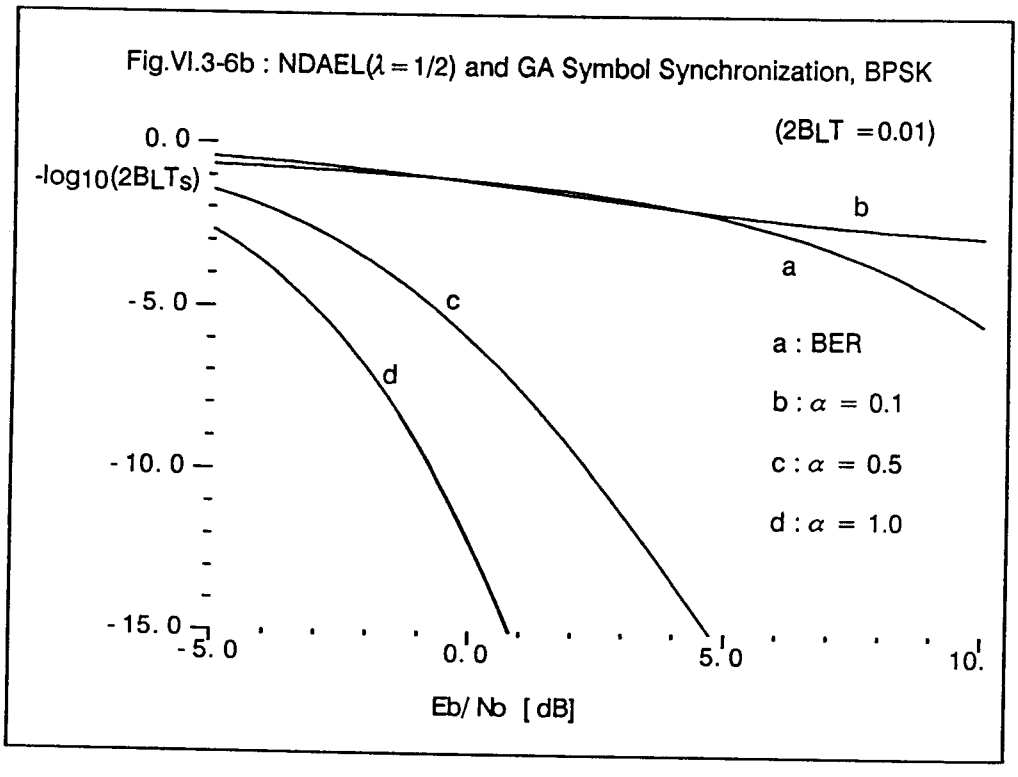
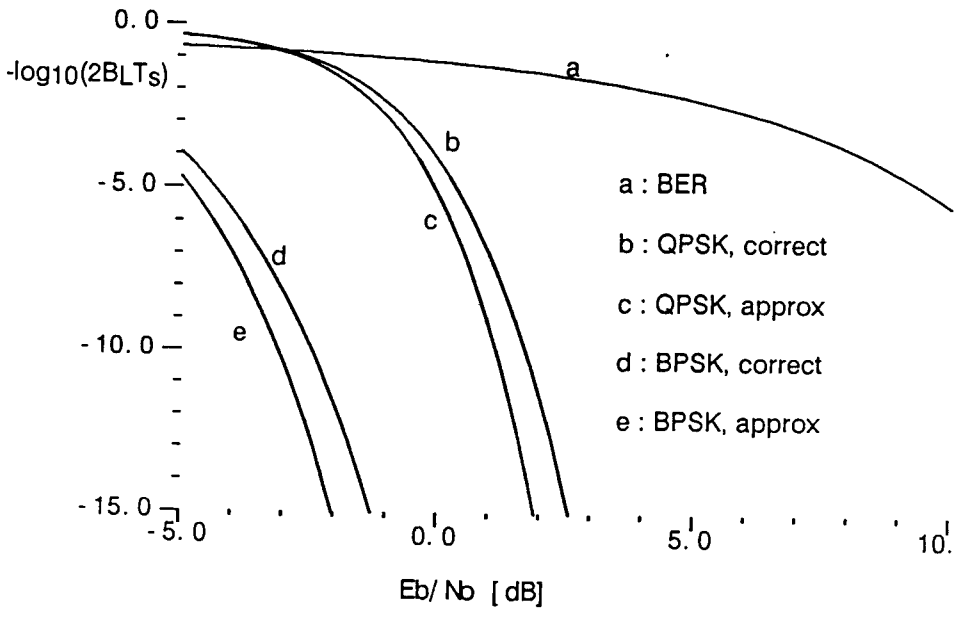


Fig.VI.3-6b : NDAEL(λ=1/2) and GA Symbol Synchronization, BPSK
(2BLT =0.01)
0. 0
-log10(2BLTs)
-5. 0
-10. 0
-15. 0
-5. 0
0. 0
5. 0
10.
Eb/ No [dB]
a
b
c
d
a : BER
b : α = 0.1
c : α = 0.5
d : α = 1.0

Fig.VI.3-7 : DDML Carrier Synchronization (2BLT = 0.01)



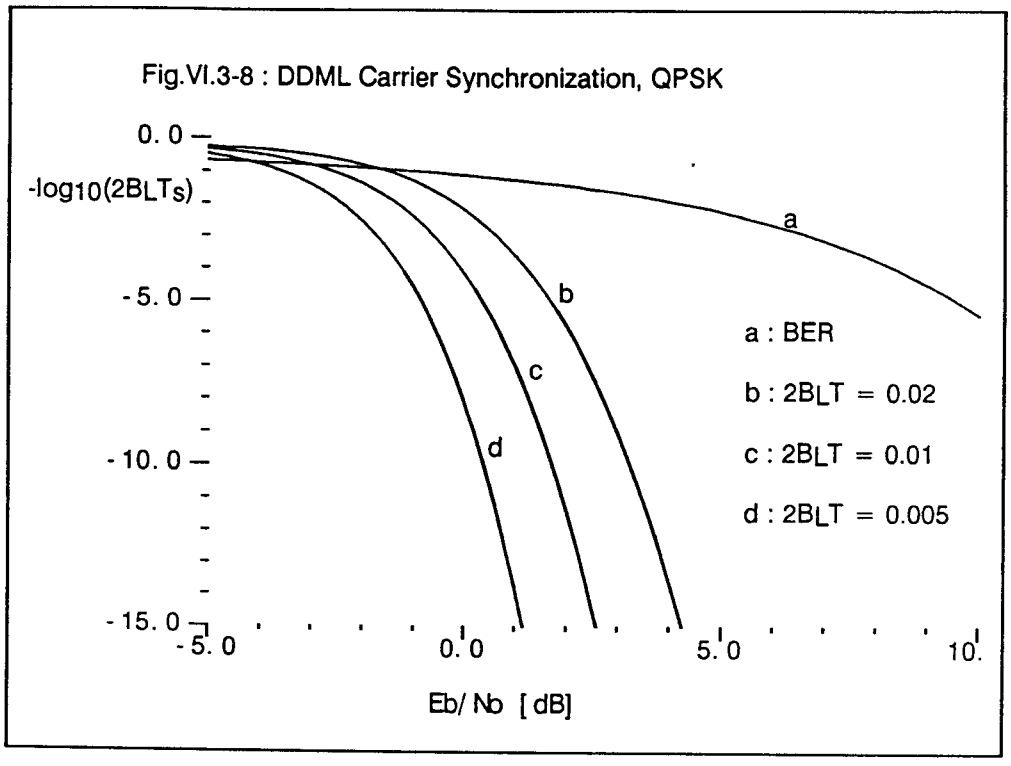


Fig.VI.3-8 : DDML Carrier Synchronization, QPSK
0. 0
-log10(2BLTs)
- 5. 0
- 10. 0
- 15. 0
- 5. 0
0. 0
5. 0
10.
Eb/ No [dB]
a
b
c
d
a : BER
b : 2BLT = 0.02
c : 2BLT = 0.01
d : 2BLT = 0.005

Fig.VI.3-9 : NDAML Carrier Synchronization, OQPSK ($\alpha = 0.1$, 2BLT = 0.003)

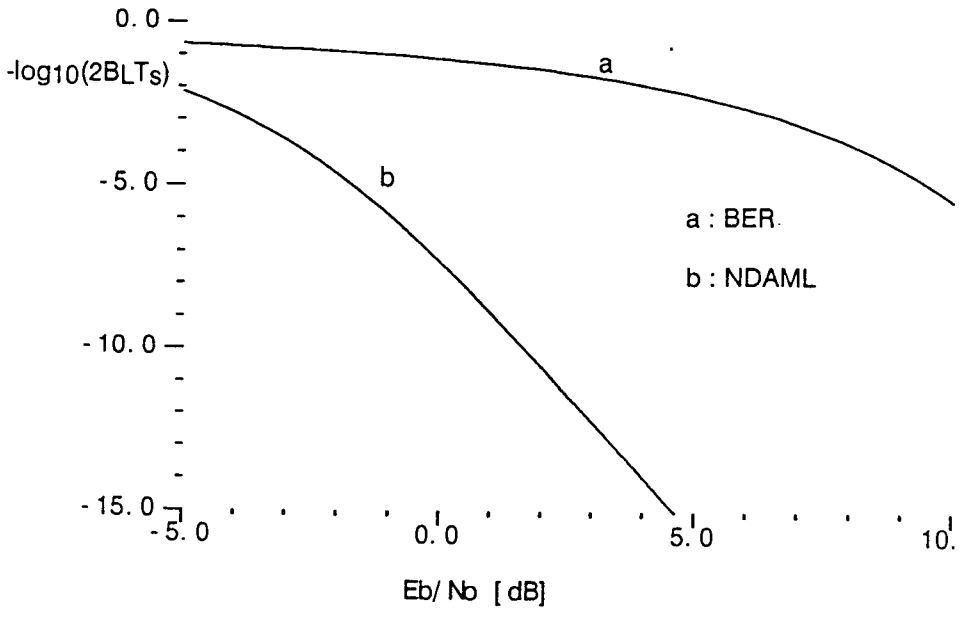
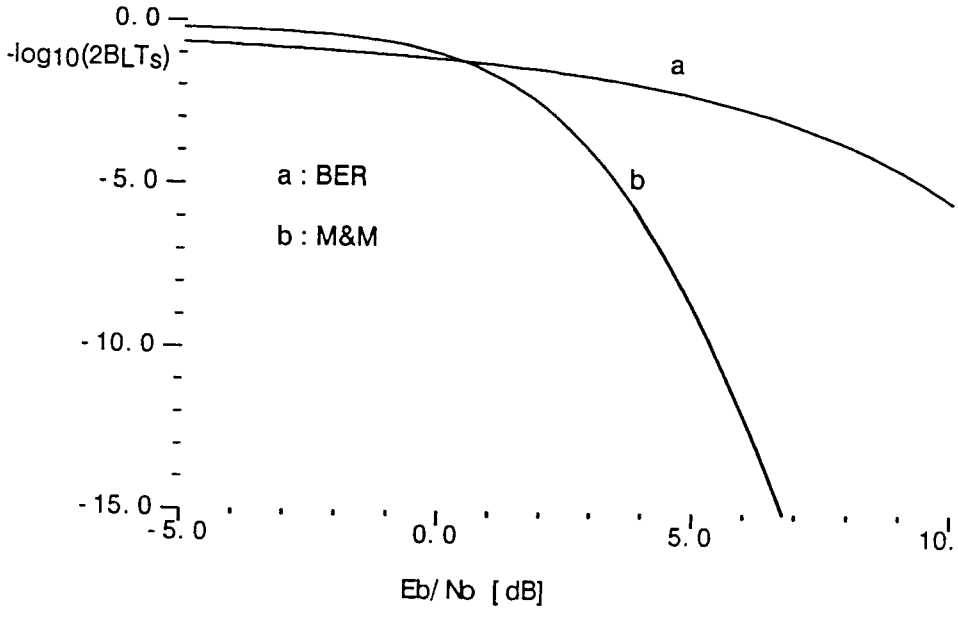


Fig.VI.3-10 : M&M Symbol Synchronization, BPSK ($\alpha = 1.0$, 2BLT = 0.003)



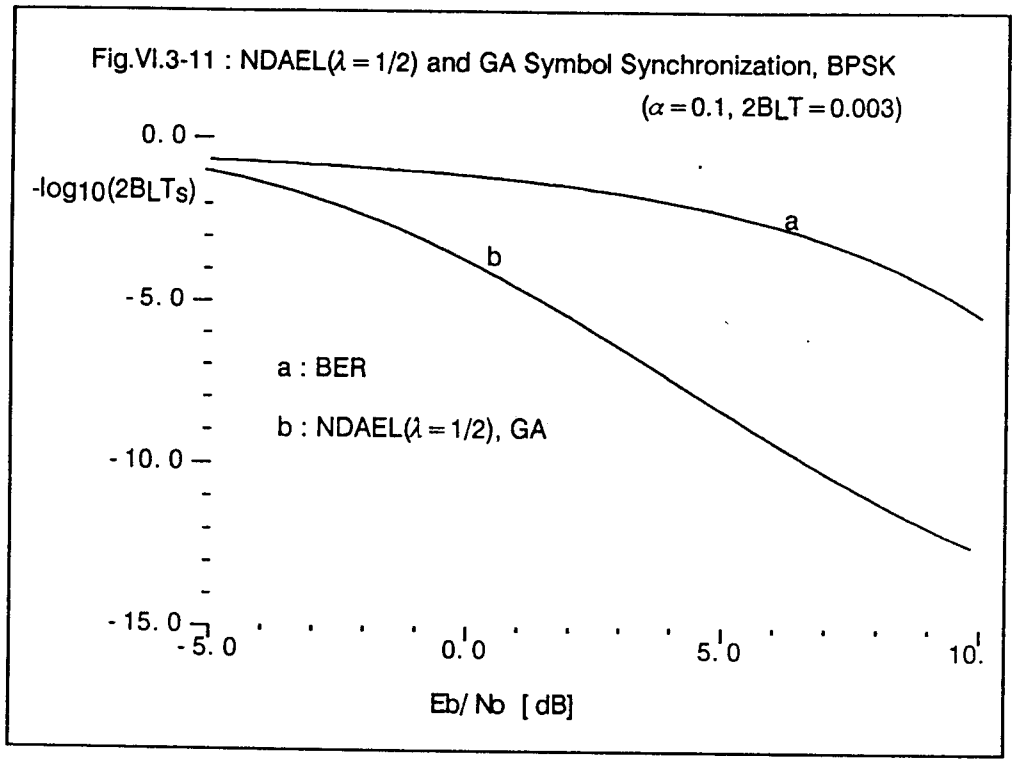


Fig.VI.3-11 : NDAEL(λ = 1/2) and GA Symbol Synchronization, BPSK
(α = 0.1, 2BLT = 0.003)
0. 0
-log10(2BLTs)
- 5. 0
- 10. 0
- 15. 0
a : BER
b : NDAEL(λ = 1/2), GA
a
b
- 5. 0
0. 0
5. 0
10.
Eb/ No [dB]

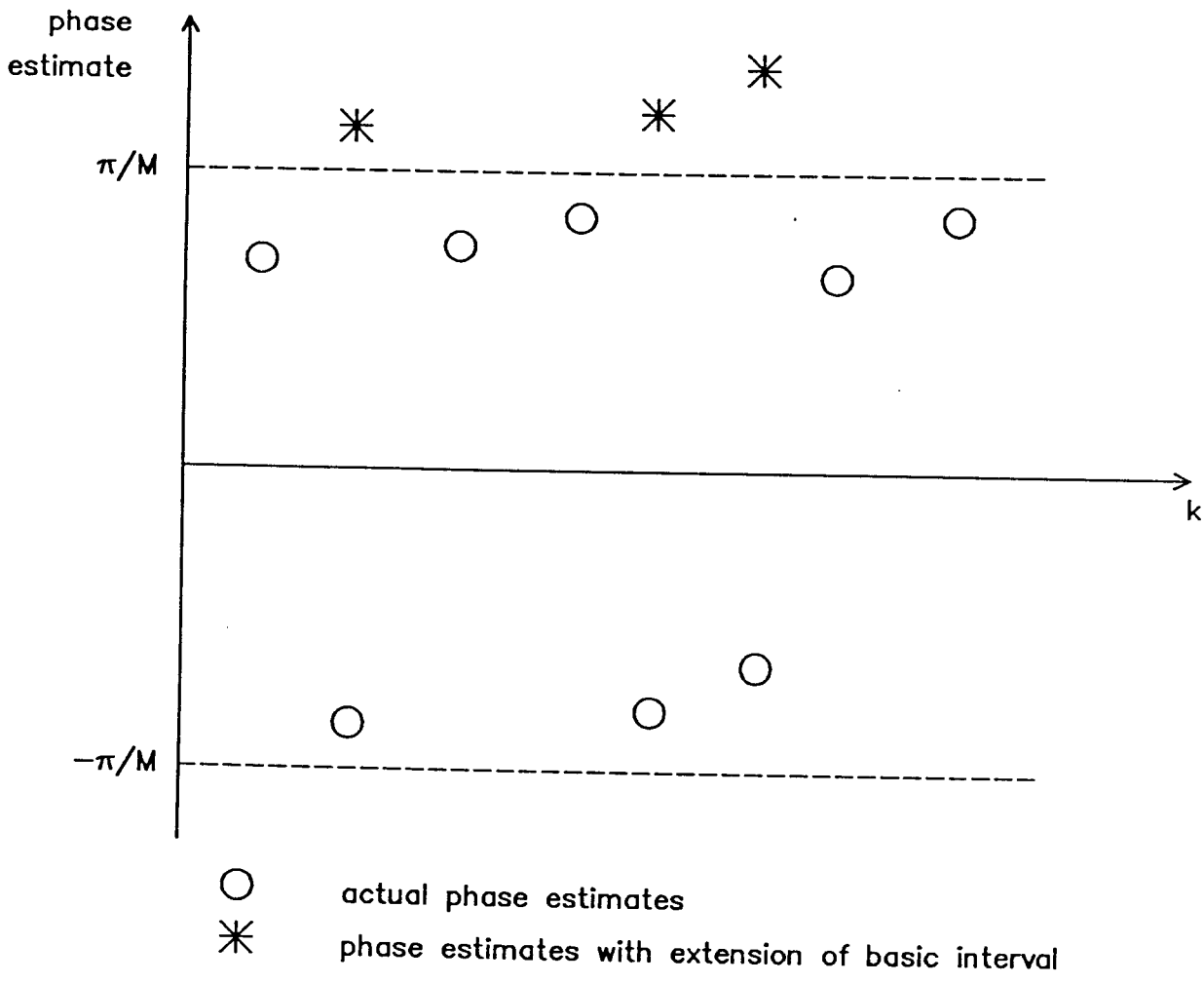


Fig. VI.4–1 : Extension of basic interval

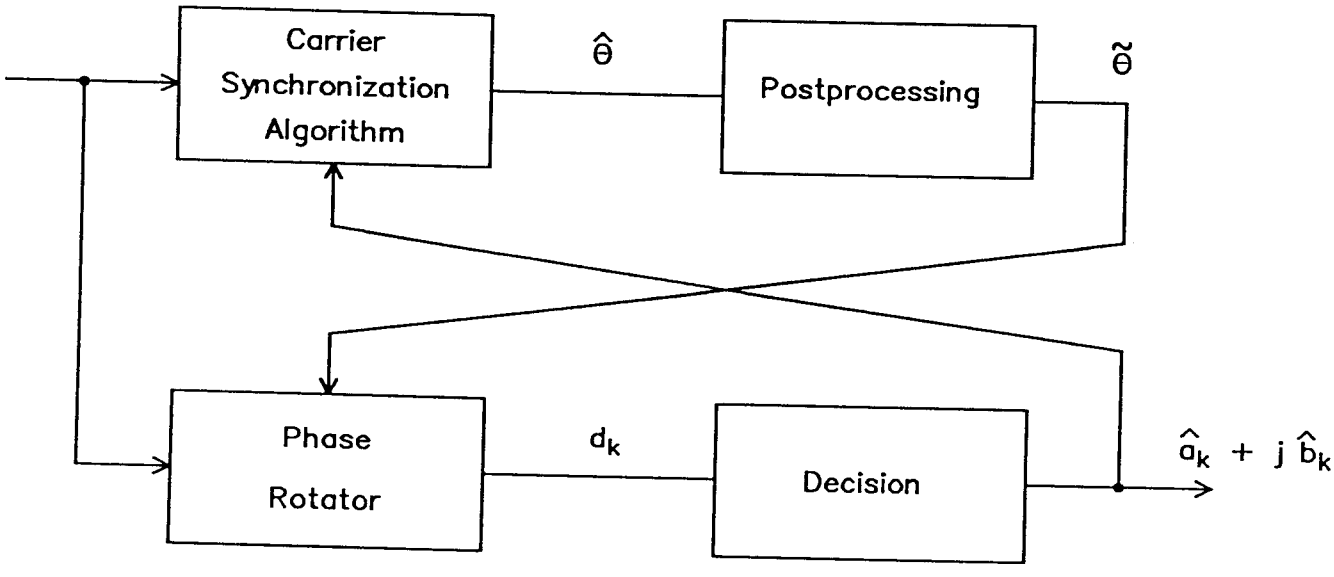


Fig. VI.4–2 : Decision Directed Carrier Synchronization

VII. TRACKING PERFORMANCE UNDER NONIDEAL CHANNELS CONDITIONS

VII.1 Preliminaries

VII.1.1 Introduction

In this part we will study the influence of linear and nonlinear distortions which may be present on the channel. Note that we take on the digital synchronizer's point of view, i.e. the analog receiver frontend is also part of the channel. The influence of distortions caused by digital synchronizer imperfections will be studied in the next part (implementation aspects).

There is a variety of distortions that may occur on the channel. We restrict ourselves to some important distortions here. When a distortion type is not considered it is out of the scope of this study, but this does not mean that it may not be of importance for any application. Nevertheless we start with a systematic approach to classification of distortions. One important grouping is by linear and nonlinear distortions.

<u>Linear distortions</u>

Linear distortions can be additive distortions and multiplicative distortions. Most important additive distortions are noise (as considered in part II, for instance) and interfering signals (co-channel/adjacent channel interference).
Analysis of interference is limited to a few special cases with idealized conditions; the influence of interferers on synchronizer performance can only be determined assuming a particular, well defined scenario. We will not study the influence of interferers on synchronizer performance here.
The most general model of a multiplicative distortion is a time-varying transfer function (or equivalently a time-varying channel impulse reponse). A very important representative of this type of distortion is the "fading channel". Here we will consider only transfer functions which are not a function of time. An example of such a distortion is an intermediate frequency (IF) filter in the receiver.

Nonlinear distortion

We again could introduce a grouping here, e.g. memoryless nonlinear distortions and nonlinear distortions with memory or distortions introduced before the noise is added and after the noise is added. But this is of limited use for the analysis. Therefore, we will first discuss sources of nonlinear effects and select typical nonlinear distortions.

Also, the combined effects of nonlinear and linear distortions may be different from their separate effects. Here we will only study the effects of nonlinear distortions on their own.

In digital transmission there are various sources of nonlinear effects. Mixers and amplifiers may show nonlinear effects in particular during overload due to strong input signals or strong interferers. In satellite transmission major sources of nonlinear distortions are high power amplifiers (HPA) using for example travelling wave tubes (TWT). Finally, for amplitude control a limiter may be used.

Nonlinear effects due to interferers can only be analyzed for a particular situation. Nonlinear effects in mixers and input amplifiers can be controlled efficiently, so that their influence may be kept small. Therefore, we will concentrate on HPA's, and, in particular on travelling wave tubes.

As in the other parts simulation results are indicated by symbols and analytical results are shown as lines. Exceptions are spectra and time functions which are shown as lines for simulation, too.

VII.1.2 Linear distortions

As discussed above, in this part we will only consider linear distortions which are not functions of time and, thus, correspond to a fixed transfer function $H_D(f)$. A further classification of such distortions is very helpful for the analysis.

When studying the system in the complex baseband $H_D(f)$ in general corresponds to a complex impulse response $h_D(t)$. This occurs for instance when the receiver signal is influenced by an IF filter which is not symmetrical with respect to the received signal carrier frequency (note that by assuming one transfer function $H_D(f)$ we have excluded a mismatch in inphase and quadrature channel matched filter from consideration).

The matched filter output signal is

$$p(t) = x_L(t) \odot h(-t) \odot h_D(t)$$

$$= \left\{ \left[\sum_m a_m g(t-mT-\tau) + j \sum_m b_m g_Q(t-mT-\tau) \right] e^{j\theta} + n(t) \right\}$$

$$\odot\ h_D(t) \qquad \text{(VII.1.2-1)}$$

where $g_Q(t) = 0$ for BPSK, $g_Q(t) = g(t)$ for QPSK, and $g_Q(t) = g(t-T/2)$ for OQPSK. Denoting the real part of $h_D(t)$ by $h_{DR}(t)$ and the imaginary part by $h_{DI}(t)$

$$h_D(t) = h_{DR}(t) + j\, h_{DI}(t) \qquad ; \qquad h_{DR}(t),\ h_{DI}(t) \text{ real}$$

the matched filter output is

$$p(t) = \left[\sum_m a_m [g(t-mT-\tau) \odot h_{DR}(t)] - \sum_m b_m [g_Q(t-mT-\tau) \odot h_{DI}(t)] \right.$$

$$\left. + j \sum_m b_m [g_Q(t-mT-\tau) \odot h_{DR}(t)] + j \sum_m a_m [g(t-mT-\tau) \odot h_{DI}(t)] \right] e^{j\theta}$$

$$+ n(t) \odot h_D(t) \qquad \text{(VII.1.2-2)}$$

Note that (VII.1.2-2) is only correct when the variations of θ are small for the length of the impulse response $h_D(t)$. From (VII.1.2-2) we see that the real part of $h_D(t)$ causes identical filtering of the inphase signal and the quadrature phase signal. The imaginary part of $h_D(t)$ causes cross-talk between inphase and quadrature signal. Also, the symmetry of $h_D(t)$ is of importance. When $h_D(t)$ is an even function, i.e. $h_D(t) = h_D(-t)$, then $g(t) \odot h_D(t)$ is also an even function ($g(t) = h(t) \odot h(-t)$ always is an even function). When we shift the timing reference by half a symbol with OQPSK ($g'_Q(t) = g_Q(t+T/2) = g(t)$) the quadrature channel symbol pulse is also an even function of time t. In such a case the optimum sampling instants (for decision) of the distorted signal are in the same positions as the optimum sampling instants of the undistorted signal. To be precise, this is only correct as long as the maximum of g(t) is still at t = 0. Otherwise we have two maxima left and right from t=0 which

means a very severe channel distortion, (e.g. multipath fading), and requires different receiver algorithms. It is therefore not considered here.

Based on these discussions we can define four basic types of distortions. Any linear distortion which can be represented by a fixed transfer function can be regarded as a superposition of these basic distortions. Note that the decomposition

$$h_D(t) = \left[h_{DRE}(t) + h_{DRO}(t)\right] + j\left[h_{DIE}(t) + h_{DIO}(t)\right]$$

$$h_{DRE}, h_{DIE} \text{ real, even} \quad ; \quad h_{DRO}, h_{DIO} \text{ real, odd}$$

is unique.

Thus, we have the following types of distortions:

1.) Intersymbol interference in inphase and quadrature channel, even symbol pulse symmetry - represented by $h_{DRE}(t)$
2.) Intersymbol interference in inphase and quadrature channel, odd symmetric component in symbol pulses - represented by $h_{DRO}(t)$
3.) Crosstalk between inphase and quadrature channel, even symmetry in cross-interfering symbol pulses - represented by $h_{DIE}(t)$
4.) Crosstalk between inphase and quadrature channel, odd symmetry in cross-interfering symbol pulses - represented by $h_{DIO}(t)$

The influence of any of these distortions on synchronizer performance depends on the actual shape of $h_D(t)$. This shape may be arbitrarily complicated, e.g. the gain ripple and group delay variations introduced by a Chebycheff filter. On the other hand, when we want to compare the sensitivity of synchronizers to each of these distortion types we may take a basic representative for each type. One possible, good way is to define representatives in the frequency domain.

Tab. VII.1-1 recalls some properties of the Fourier-transform which are of importance in this case. We find that

1.) $h_{DRE}(t)$ corresponds to $H_D(f) = H_D(-f) = H_D^*(-f)$ real and even

2.) $h_{DRO}(t)$ corresponds to $H_D(f) = -H_D(-f) = H_D^*(-f)$ imaginary and odd

3.) $h_{DIE}(t)$ corresponds to $H_D(f) = H_D(-f) = -H_D^*(-f)$ imaginary and even

4.) $h_{DIO}(t)$ corresponds to $H_D(f) = -H_D(-f) = -H_D^*(-f)$ real and odd

It is obvious that $h_D(t)$ always should have a real and even component ($h_{DRE}(t)$, a Dirac pulse in the ideal case), otherwise we have a very severe distortion (except for the trivial case that type 3 occurs only which corresponds to an interchange of inphase and quadrature channel).

Impulse response	Transfer function	ISI	Xtalk
real, even	real, even	even	-
real, odd	imaginary, odd	odd	-
imaginary, even	imaginary, even	-	even
imaginary, odd	real, odd	-	odd

Table VII.1-1: Symmetry relations between a Fourier transform pair and impact on ISI and crosstalk
(even: $S(x) = S(-x)$; odd: $S(x) = -S(-x)$).

Distortion type	Type	ISI	Xtalk
symmetric gain	1	even	-
asymmetric gain	4	even	even
asymmetric phase / *	1+2	even+odd	-
symmetric phase / **	1+3	even	odd

Table VII.1-2: Distortion representatives and corresponding distortions
* symmetric group delay
** asymmetric group delay

The representatives used here are shown in Fig. VII.1-1a,b and the relations are summarized in Tab. VII.1-2.

The symmetric (even) gain distortion, where the gain in dB is quadratic with the frequency, yields a distortion of type 1, i.e. even ISI. An asymmetric (odd) gain distortion, where the gain is linear with the frequency, yields a distortion of type 4, i.e. odd crosstalk. With the representative shown in Fig. VII.1-1a we also have an even ISI (Type 1) component since the gain is linear in dB, i.e. the gain measured in a linear scale is not exactly linear with frequency. When the gain asymmetry is not too strong, i.e. a few dB at most, type 4 will dominate.

Since the group delay $\tau_{gr} = -d\psi/d\omega$, a symmetric (even) group delay distortion $\tau_{gr}(f)$ corresponds to an asymmetric (odd) phase distortion $\psi(f)$ and vice versa. The most basic asymmetric phase distortion is a phase linear with frequency f. But this is a trivial case which simply corresponds to a delay (τ_{gr}=constant). Here we have used a symmetric group delay distortion, where τ_{gr} is proportional to $|f|$ (solid line in Fig. VII.1-1b). Another possible example is τ_{gr} proportional to f^2. In this case the group delay is continuously differentiable. Since both are artificial examples the non-continuously differentiable group delay of the type used here should be acceptable for demonstrating the influence of a symmetric group delay distortion. The symmetric group delay distortion corresponds to an asymmetric phase distortion and, thus, to a combination of type 1 and type 2 distortions, i.e. even and odd intersymbol interference without crosstalk.

Finally, the dashed line in Fig. VII.1-1b corresponds to an asymmetric group delay distortion ($\tau_{gr} \sim f$) or symmetric phase distortion $\psi(f) \sim f^2$, which is a combination of distortions of type 1 and type 3, i.e. even symmetric intersymbol interference (ISI) and even symmetric crosstalk (X-talk). Now we can give an additional motivation for proposing the $\tau_{gr} \sim |f|$ group delay distortion in the previous paragraph. We have the same ISI component in both cases and the same absolute magnitude of the phase.

Fig. VII.1-2 to Fig. VII.1-5 show the matched filter output eye-patterns for QPSK with the four distortion representatives introduced here. Since the

distortions of Fig. VII.1-2, VII.1-3 are pure inchannel distortions (i.e. ISI only, no crosstalk) the eye-pattern is identical for BPSK, QPSK and OQPSK (except for a $\sqrt{2}$ larger amplitude for BPSK).

It is also interesting to observe the eye patterns of the even symmetric crosstalk component (Fig. VII.1-4b) and the odd symmetric crosstalk component (Fig. VII.1-5c,d) without the in-channel signal. For the even symmetric crosstalk we have maximum levels at the decision instants and small levels in the middle between the sampling instants for decision. For the odd crosstalk we have the opposite situation, i.e small levels arround the decision instants and maximum levels halfway between the decision instants. The small levels occur because the crosstalk transfer function is 0 at $fT=0$ in both cases, i.e. we have a differentiating effect, the crosstalk cannot have low frequency componenents. Therefore, even when consecutive symbols are identical the crosstalk distortion decays once per symbol.

In Fig. VII.1-4c and VII.1-5e,f the ISI component of the two distortions with crosstalk are shown. We see that the ISI is small, the crosstalk dominates in both cases (Note the extended scale in Fig. VII.1-5c,d).

<u>BPSK</u>

For the two distortions with crosstalk the eye-patterns are different for the three modulation schemes. Since we do not have a quadrature phase signal with BPSK, we do not have crosstalk effects in the in-channel eye-pattern and only observe the in-channel distortions type 1 and 2, respectively. In the quadrature phase component we observe the inphase channel signal distorted by a filter with the (crosstalk) impulse response $h_{DIE}(t)$. Similarly with odd symmetric crosstalk we again observe the even symmetric intersymbol interference in the inphase channel only. In the quadrature phase component we have the inphase signal but now distorted by a filter with the odd symmetric (crosstalk) impulse response $h_{DIO}(t)$.
Although we have no quadrature phase data channel, the occurrence of a signal on the quadrature phase component may be of importance for carrier synchronizer performance. Even with BPSK the quadrature phase component is used in carrier synchronization algorithms.

QPSK

For QPSK we have in the inphase channel a superposition of the inphase data stream and an attenuated version of the quadrature phase data stream. Therefore, with even crosstalk the interference is strongest at the decision instants where the interfering quadrature phase channel has its largest amplitude while the distortion is small at the sampling instance shifted T/2 from the decision instant (because of the 0 at fT=0 of the crosstalk transfer function, see above). With the odd crosstalk shown in Fig. VII.1-5a,b the crosstalk from the quadrature phase channel is smaller at the decision instants, the odd crosstalk symbol pulse is zero at t = 0, only the tails of the pulse contribute.

OQPSK

For OQPSK the quadrature phase channel is shifted by half a symbol. Now with even crosstalk the distortion by the quadrature phase channel is small arround the decision instants of the inphase channel (the even crosstalk symbol pulse is small arround ±T/2) and vice versa. With odd symmetric crosstalk the crosstalk pulse is large arround ±T/2, the distortion by the quadrature phase channel is strongest at the decision instants of the inphase channel and vice versa. Therefore, we may expect that the crosstalk distortions interchange their role which we actually find in Fig. VII.1-6 and Fig. VII.1-7. With even symmetric crosstalk we now have a similar figure as for the odd symmetric crosstalk with QPSK (see Fig. VII.1-6 and Fig. VII.1-5) and with odd symmetric crosstalk we have a similar eye pattern as for the even symmetric crosstalk with QPSK (see Fig. VII.1-7 and Fig. VII.1-4).

Finally, it must be noted that the position of the linear distortion in the channel also plays a role. With a nonlinearity in the channel the position of the distortion with respect to the nonlinearity has strong influence on performance. Here we will discuss the linear distortion only and not the combination of linear and nonlinear distortions. Another important point is the position of the linear distortion with respect to the noise. When the noise is added before the linear distortion it is influenced by $h_D(t)$, too. For the respresentative distortions here, we have two different situations. While the gain distortions influence the noise power spectral density this is not the case for the group delay distortions for which the statistics, in particular the noise power spectral density, are identical after and before the distortion. Therefore, in this case the position of the linear distortion is irrelevant. In

the analysis of the subsequent sections we have assumed that the linear distortion is introduced after the noise is added which is always the case when the receiver is the source of the distortion.

With the two gain distortions the matched filter output samples have a different noise variance and power spectral density. The eye-patterns for the gain distortions, i.e. Fig. VII.1-2, 5 and 7 have been normalized such that we have the same noise variance in the samples as without gain distortion. The difference is small for asymmetric gain distortions of 1 and 3 dB (< 0.2 dB) and the symmetric gain distortion of 1 dB (< 0.5 dB) but significant (~ 2 dB) for the symmetric gain distortion with 3 dB (i.e. the gain which in dB is quadratic with the frequency equals 3 dB at $f = 1/(2T)$).

VII.1.3 Nonlinear Distortions

Fig. VII.1-8 shows the very basic elements of a satellite link. The signal is transmitted from ground to satellite (uplink). Here the signal is distorted by the uplink noise. In the satellite the signal is amplified, converted to the down link frequency and transmitted back to ground (or to a space terminal). We will only consider the effect of the on-board TWT.

<u>TWT</u>

When the signal bandwidth is small compared to the TWT bandwidth, the influence of the TWT can be modelled by a memoryless nonlinearity operating on the complex envelope. The output signal amplitude is a nonlinear function of the input signal amplitude (AM/AM conversion) and the output signal phase is shifted with respect to the input signal phase by an amplitude dependent phase offset (AM/PM conversion).

Fig. I.4-4 shows the AM-AM and the AM-PM characteristics of a typical TWT (AEG Telefunken, Ku-band TWT for OTS, data provided by ESA/ESTEC). The curves given are approximations of the measured characteristics by a rational polynomial approximation. Two cases are considered in the following analysis: the "worst" case of operation at

saturation (0 dB backoff) and operation 3dB below saturation (3 dB input backoff ≈ 0.5 dB output backoff).

For the understanding of the various effects it is very helpful to write down the signal at the TWT input

$$x_u(t) = \sqrt{E_{Bi}/T}\left[\sum_m a_m h_I(t-mT-\tau) + \sum_m jb_m h_Q(t-mT-\tau)\right]e^{j\theta} + n_u(t)$$

(VII.1.3-1)

Where h_I, h_Q are the received uplink data pulse shapes normalized to have unit power, E_{Bi} is the input energy per bit, and $n_u(t)$ is the uplink noise. Since the influence of the TWT is independent of the timing offset τ and a phase offset θ both will be omitted in the following equations. The power P_i and the phase $\varphi_i(t)$ of this signal is

$$P_i(t) = E_{Bi}/T\left[\left(\sum_m a_m h_I(t-mT) + Re\{n_u(t)\}\right)^2 + \left(\sum_m b_m h_Q(t-mT) + Im\{n_u(t)\}\right)^2\right]$$

$$\varphi_i(t) = \arg\left\{\frac{\sum_m b_m h_Q(t-mT) + Im\{n_u(t)\}}{\sum_m a_m h_I(t-mT) + Re\{n_u(t)\}}\right\}$$

The output signal of the TWT then is

$$s_d(t) = \sqrt{P_0(P_i(t))}\,\exp[j(\varphi_i(t)+\psi(P_i(t)))]$$

The uplink noise causes a random AM and PM modulation of the TWT output signal. This is a different problem which is out of the scope of this study. Therefore, we neglect the uplink noise influence in the further analysis. Thus,

$$P_i(t) = E_{Bi}/T\left[\left(\sum_m a_m h_I(t-mT)\right)^2 + \left(\sum_m b_m h_Q(t-mT)\right)^2\right]$$

(VII.1.3-2)

and

$$s_d(t) = \sqrt{P_0(P_i(t))}\ \left\{\frac{\sqrt{E_{Bi}/T}}{\sqrt{P_i(t)}}\ \left[\ \sum_m a_m h_I(t-mT) + \sum_m jb_m h_Q(t-mT)\ \right]\ e^{j\psi(P_i(t))}\right.$$

With hard keyed PSK the signal power is constant and the TWT simply amplifies the input signal and adds a constant phase offset $\psi(P_i)$. But with bandlimited signals the power is not constant. The influence of the TWT for different transmission schemes can be studied qualitatively from eq. (VII.1.3-2).

Let us start with BPSK, i.e. $b_m = 0$. Assuming in a first approach that the symbol pulse is zero outside [0, T] we find that in the TWT output signal we have a different symbol pulse

$$\bar{h}_I(t) \sim \sqrt{P_0(h_I^2(t)\ E_{Bi}/T)}$$

and a time-varying phase rotation

$$\psi(h_I^2(t)\ E_{Bi}/T)$$

Thus, a filter matched to the symbol pulses at the TWT output now would be a complex matched filter. With a bandlimited signal the symbol pulses extend over several symbol intervals. Therefore, the signal amplitude $\sqrt{P_i(t)}$ depends on the adjacent symbols. The pulse shape distortion now also depends on the adjacent symbols. For instance, when we have a transition because neighbouring symbols are different, the signal amplitude will cross zero. We may consider this situation as having a modified symbol pulse which corresponds to the expected value (with respect to the data symbols) of the symbol pulse and an additive intersymbol interference (ISI) which is nonlinear, i.e. not just additive with respect to the data symbol sequence. Note that maximum likelihood sequence estimation, i.e. a 'Viterbi-Equalizer', is required for optimum detection. In this part root cosine rolloff filters with

linear phase are assumed both for transmit pulse shaping and as receive matched filter.

With (O)QPSK the signal power depends on both the inphase and quadrature data symbols so that with a separation approach as for BPSK (average pulse shape and ISI) we find that quadrature data causes intersymbol interference on the inphase signal and vice versa.

An analytical evaluation of the influence of the TWT on synchronizer performance cannot be given. Therefore, throughout the section simulation results are used which are complemented by a qualitative analysis based on the above equations and remarks.

VII.1.4 Feedback and feedforward synchronizer

So far, we have seen that feedback and feedforward synchronizers have the same tracking performance under ideal conditions. A feedforward synchronizer maximizes a function of the input signal. The corresponding feedback synchronizer adjusts the derivative of this function (and not of any of the signals) to zero. Therefore, as long as the derivatives with respect to the parameters carrier phase and timing can be linearized around the tracking point, the performance of corresponding feedback and feedforward synchronizers are identical (for corresponding loop filters, see section I.3.2. Since this is still the case, feedback and feedforward synchronizer performance are identical even under nonideal channel conditions.

VII.1.5 Self noise measurement

In the analysis for ideal conditions it was shown that the self-noise causes a contribution to the phase error variance proportional to $(2B_LT)^2$. The factor of proportionality was determined analytically, this is not possible here. Therefore, we must look for a different method to determine this factor, i.e. for an appropriate measurement.

Assuming an arbitrary power spectral density $X(fT)$, $(0 \leq fT \leq 1)$ we can use a series expansion

$$X(fT) = \sum_{i=0}^{\infty} a_i \, (fT)^2$$

Since the power spectral density is even $a_1 = a_3 = a_{2n+1} = 0$. Neglecting terms of order $i > 2$

$$X(fT) = a_0 + a_2 \, (fT)^2 \qquad \text{(VII.1.5-1)}$$

Thus, for a closed loop (or feedforward filter) transfer function $H(j2\pi fT)$ the phase error variance is

$$\sigma^2 = 2 \int_0^{0.5} X(fT) \; |H(j\omega T)|^2 d(fT)$$

$$2a_0 \int_0^{0.5} |H(j\omega T)|^2 d(fT) + 2a_2 \int_0^{0.5} (fT)^2 \; |H(j\omega T)|^2 d(fT) \qquad \text{(VII.1.5-2)}$$

For a first order loop

$$H(j2\pi fT) = \frac{1}{1 + \frac{j2\pi fT}{4B_L T}}$$

the error variance (VII.1.5-2) is obtained as

$$\sigma^2 = a_0 \, (2B_L T) \, \frac{2}{\pi} \text{ arc tan } \frac{\pi}{4B_L T} +$$

$$+ \frac{a_2}{2\pi^2} (2B_L T)^2 \, (2 - (2B_L T) \, \frac{4}{\pi} \text{ arc tan } \frac{\pi}{4B_L T} \,)$$

For a narrowband (i.e. $B_L T << 1$) synchronizer arc $\tan(\pi/(4B_L T)) \approx \pi/2$ and

$$\sigma^2 \approx a_0(2B_L T) + K_F \frac{a_2}{2\pi^2} (2B_L T)^2 \qquad \text{(VII.1.5-3)}$$

with $K_F = 2$. Comparing (VII.1.5-3) with the variance for ideal conditions (II.1.2-2) we find the following corresponding terms

$$a_o \hat{=} S(fT = 0)/K_{\theta\theta}^2$$

$$\frac{a_2}{2\pi^2} \hat{=} \frac{1}{K_{\theta\theta}^2} \sum_m |m|(-R_m)$$

For a second order loop, the same correspondences are found and the same K_F as in section II.1.2 results.

Note that the integrals in (II.1.5-2) are bounded even for first order loops because the upper limit of the integral is fT = 0.5 (sampled system). This is not the case for time-continuous systems where the situation is more complicated.

To analyse the influence of the TWT on the self-noise we have to measure

$$N_2 = \frac{a_2}{2\pi^2}$$

With a measured power spectral density S(fT) the minimum mean square estimate of a_2 is found from

$$\frac{d}{da_2} \int_0^z [S(fT)/K_{\theta\theta}^2 - (a_0+a_2(fT)^2)]^2 \, d(fT) = 0$$

as

$$N_2 = \frac{a_2}{2\pi^2} = \frac{1}{2\pi^2} \frac{5}{z^5} \left[\int_0^z (fT)^2 S(fT)/K_{\theta\theta}^2 \, d(fT) - \frac{z^3}{3} S(fT)/K_{\theta\theta}^2 \right]$$

(VII.1.5-4)

In theory z = 0.5 but the power spectral density falls off from a parabolic curve for z -> 0.5, see Fig. VII.1-9. Therefore, with z = 0.25 we get a pessimistic value for N_2 which is an upper bound on the synchronizer self-noise performance. Note that the integral may be obtained by passing the phase detector output through a differentiator and an ideal lowpass filter with cutoff frequency fT = 0.25 and then estimating the variance. $S(fT = 0)/K_{\theta\theta}^2$ has been determined using a narrowband ideal lowpass filter as before.

VII.2 Tracking performance results for carrier synchronization - linear distortions

VII.2.1. Qualitative analysis

All synchronizers for BPSK and QPSK which assume timing to be known operate on the sample taken at the decision instant. For OQPSK the samples in the corresponding synchronizers are spaced by T/2. Therefore, the samples taken in the middle between the decision instants must also be considered. Also the influence of the distortions is different as discussed in section VII.1.2. We will first study BPSK and QPSK.

<u>BPSK, QPSK</u>

With a symmetric gain distortion even symmetric intersymbol interference (ISI) is introduced at the decision instant. As was already found with the TWT, BPSK and QPSK carrier synchronizers are rather insensitive to this type of distortion. It can be considered as additional noise at the decision instant. Therefore, when the ISI variance is small compared to the channel noise variance the performance degradation should be negligible.

With even symmetric ISI and even symmetric crosstalk which is introduced by an asymmetric group delay distortion as shown in Fig. VII.1-4a we have at the decision instants the in-channel ISI and the crosstalk. By comparison of Fig. VII.1-4 and Fig. VII.1-2 we find that we have a similar situation as for pure even symmetric intersymbol interference only. But now the variances of both distortions (ISI and crosstalk) must be added and used to estimate the influence of the distortion. As discussed above the sensitivity of carrier synchronizers to this distortion is low.

For odd symmetric crosstalk the interference by the actual symbol of the quadrature phase channel in the inphase channel (and vice versa) is zero, we have a distortion by the tails of the pulses of the neighbouring symbols only. Therefore, as also can be seen in Fig. VII.1-5a,b the crosstalk at the decision instant is very small, a significant performance degradation should not occur in this case. This was also confirmed by simulations of DDML and NDAML detector performance.

Finally, with odd symmetric intersymbol interference the optimum sampling instants for decision are shifted from the position of the undistorted case by an amount which depends on the asymmetry. Different symbol synchronization algorithms track the shift of the optimum sampling instant to a different degree so that we have a net error between tracking point and optimum sampling instant for decision. At the optimum sampling instant we have for moderate group delay distortions as in Fig. VII.1-3a an interference level which is similar to that for a moderate even symmetric ISI and, therefore, no significant performance degradations.

OQPSK

For OQPSK two points are different. First the carrier synchronizers where timing is assumed to be known operate on a two-sample-per-symbol basis (p,q). Second we have a half bit shift of the quadrature phase channel, so that for crosstalk the influence of the quadrature interference is different. While the distortion of the second sample taken with a half bit offset is not very strong for the two ISI cases, we find an interesting situation for the crosstalk. Because of the half symbol delay in the quadrature channel, the odd symmetric crosstalk (Fig. VII.1-7) now yields a similar eye-pattern as the even symmetric crosstalk in the case of QPSK (Fig. VII.1-4). Correspondingly, the even symmetric crosstalk (Fig. VII.1-6) in the OQPSK case and the odd symmetric crosstalk (Fig. VII.1-5) in the QPSK case yield a similar eye-pattern. Therefore, we can expect that the performance is also interchanged.

Tracking point

While ISI does not have an influence on the mean of the carrier phase estimate (we still have a real symbol pulse, the ISI distortions have zero mean), with crosstalk we may have a different mean of the carrier phase estimate. For odd crosstalk the imaginary part of the symbol pulse is zero at $t=0$ but with even crosstalk we have a complex symbol pulse having a phase $\psi(t)$ which is different from 0 for $t=0$, i.e. $\psi(0)\neq 0$. Therefore, for BPSK and QPSK we have a mean phase shift with even crosstalk. Analogously, for OQPSK we have a mean phase shift with odd crosstalk. For the distortion levels studied here the mean phase shift is very small.

VII.2.2 Carrier synchronizer performance

So far self-noise in carrier synchronizers was not of importance. Therefore, here we will neglect the influence of linear distortions on self-noise in the analysis.

In all cases the influence of the linear distortions considered is negligibly small. For example, in Fig. VII.2-1a,b results are shown for DDML with QPSK and OQPSK. We only find a very slight degradation for the 3 dB symmetric and asymmetric gain distortion in Fig. VII.2-1a.

Fig. VII.2-2 shows the influence of asymmetric gain distortions on NDAML with QPSK and Fig. VII.2-3 shows the influence of group delay distortions on DDML with OQPSK. Slight deviations are only visible because of the extended scale and may be within the simulation uncertainty range.

Results for V&V and carrier synchronizers independent of timing are not shown but these synchronizers are as robust as the shown synchronizers with respect to linear distortions.

The conclusion from the simulation studies is that carrier phase tracking is very robust against linear distortions, the influence in general is negligibly small.

VII.3 Tracking performance results for symbol synchronization - linear distortions

VII.3.1 Qualitative analysis

To use the eye-patterns given in section VII.1 we first have to discuss which sampling instants are processed by the various synchronization algorithms. With the arguments given in section VII.1.4 we consider the feedback versions only. Samples taken around the optimum decision instants are used for DDEL(λ = 1), M&M, mM&M. DDEL(λ = 1/2), DDTL and NDAEL(λ = 1/2), GA also uses samples taken in the middle between the optimum decision instants. Special cases are DDML, NDAML and DDMMSE which all use the derivative of the matched filter output for processing, i.e. we also have to take the influence of the linear distortion on the derivative of the symbol pulse into account.

Note that as for carrier synchronization, QPSK and OQPSK have to be studied separately because the influence of the linear distortion is different. With BPSK the crosstalk is not of interest since we do not have a signal in the quadrature phase channel (noise only).

Finally, some general remarks on the influence of the linear distortions on the optimum decision instant samples and the samples taken in the middle between optimum decisions will be given. Here we have to study the different modulation schemes separately.

<u>BPSK</u>

As discussed above the crosstalk is not of importance with BPSK. With the symmetric gain distortion we have ISI at the decision instants. From Fig. VII.1-2a,b (which after multiplication by $\sqrt{2}$ also applies for BPSK) we find that for the 3 dB gain distortion we have a significant degradation of the noise margin for the decisions. Therefore, we will have much higher decision error probability. This is important for the DD type synchronizers. For the odd symmetric ISI the synchronizer should adjust the timing such that the samples for decision are taken close to the optimum instant which is now shifted with respect to the undistorted operation. The decision error probability then is not increased significantly for moderate values of the

corresponding group delay distortion (see Fig. VII.1-3a). Actually, the different symbol synchronizers have different tracking points as will be discussed below.

The influence of even symmetric and odd symmetric intersymbol interference on the samples halfway between the decision instants is small even for the symmetric gain distortion and for moderate symmetric group delay distortion.

QPSK

For even symmetric ISI and odd symmetric ISI the same discussions as for BPSK apply. The even symmetric crosstalk causes mainly additional noise at the decision instant. As long as these variations are sufficiently small compared to the channel noise, the performance degradation may be expected to be negligible. For the odd crosstalk we only have a small influence at the decision instants, while samples in the middle between the optimum decision instants may be strongly distorted.

OQPSK

For even and odd symmetric ISI we have the same influence as for QPSK. But now we have for the even symmetric crosstalk a similar behaviour as for QPSK with odd symmetric crosstalk and vice versa. Therefore, it can be expected that for OQPSK the performance degradations with even symmetric crosstalk are similar to the performance degradations for QPSK and odd symmetric crosstalk and for OQPSK with odd symmetric crosstalk they are similar to the results for QPSK and even symmetric crosstalk.

Tracking point

So far we have discussed the degradation with respect to tracking error variance. Another important aspect is the influence on the tracking point of the symbol synchronizer. The crosstalk is a zero mean distortion (inphase and quadrature channel symbols are assumed to be independent) for the timing error synchronizer. With even symmetric ISI we still have a symmetric symbol pulse, the tracking point remains at $\tau=0$ for all synchronizers. With odd symmetric ISI the symbol pulse has an asymmetric shape as shown in principle in Fig. VII.3-13. In this case the tracking point depends on the type of synchronizer (but not on the modulation scheme

since the shift is caused by an 'in-channel' distortion). Odd symmetric ISI occurs with a symmetric group delay distortion. This aspect will be discussed in the next section in more detail.

To summarize, it is sufficient to study the influence of ISI for BPSK and QPSK only, OQPSK is identical to QPSK in this case. Also, for the signal-to-noise ratios considered here we can expect that the influence of even symmetric crosstalk is small. Therefore, we will mainly study even symmetric and odd symmetric ISI and odd crosstalk for QPSK and, for the reasons discussed above, even crosstalk with OQPSK.

VII.3.2 Symbol synchronizer performance

In this section we will discuss the influence of the intersymbol interference on the normalized loop noise power spectral density at $fT = 0$. Because of the large variety of parameters and the parameter range that would be covered, it is impractical to give a complete survey of the actual performance degradations for the various cases. Therefore, we will concentrate on a few illustrative values. Because so far we have found a strong sensitivity in the low to moderate E_s/N_0 range, we mainly study $E_S/N_0 = 5$ dB.

Since in previous parts and sections the self noise was found to be of minor importance for the operating conditions considered here and also the sensitivity of the linear distortions on the self noise is low, it is not studied further here.

<u>Symmetrical gain distortion (even symmetric ISI)</u>

The influence of a symmetric gain distortion on the different synchronizer slopes and loop noise power spectral densities is shown in Fig. VII.3-1 to Fig. VII.3-10 (except for figures with extension c which show results for asymmetric gain only). Except for the M&M synchronizer we only find an influence on the slope, the normalized loop noise power spectral density is not influenced significantly. Note that in Fig. VII.5-6a and Fig. VII.5-6b the results for $E_s/N_0 = 5$dB have been shifted right and left so that they become visible separately. Actually, they are on the theoretical curves within

the measurement uncertainty except for α=0. For rolloff α=0 we find for DTTL a slight influence of a gain distortion on synchronizer performance with BPSK and QPSK, with the loss being higher for QPSK than for BPSK. A possible explanation is, that at the given E_s/N_0 value the QPSK decision signal-to-noise ratio is 3 dB below the BPSK decision SNR. Therefore, the additional noise due to even symmetric ISI is less significant. For the M&M (Fig. VII.3-8a,b,c) we find a significant influence of a gain distortion for rolloff 1 only. With 3 dB gain distortion the degradation is very strong for BPSK as well as for QPSK.

To summarize, except for the M&M synchronizer and rolloff α=1.0 the symbol synchronizer algorithms are very insensitive to even symmetric ISI at the signal-to-noise ratios considered here. The additional noise due to ISI at the decision instant is dominated by the channel noise.

Asymmetric gain distortion
(even symmetric ISI + odd symmetric crosstalk)

In Fig. VII.5 -Fig. VII.5-10 also is shown the influence of an asymmetric gain distortion on the performance of the different symbol synchronizers. As discussed before, for OQPSK we have a situation which is similar to the even symmetric crosstalk for QPSK. Since the symbol synchronizers are very insensitive to this type of distortion, we can expect the same for OQPSK and odd crosstalk. This is indeed found in Fig. VII.3-2c, 4c, 6c, 8c, 10c.

But also for BPSK and QPSK we do not find a significant influence for any synchronizer (DDML and NDAML are not given but also do not show a performance degradation). Therefore we can conclude that symbol synchronizers are very insensitive to the studied asymmetric gain distortion.

Asymmetric group delay distortion
(Even symmetric ISI + even crosstalk)

For OQPSK we can expect an influence on performance similar as for BPSK and QPSK with odd crosstalk. From the above results we can conclude that we do not have a significant performance degradation. For BPSK and QPSK we have a similar situation as with even ISI only. Since the symbol synchronizers were found to be insensitive to ISI we can expect that we do not have a significant performance degradation. This was indeed found in

the simulations. Since the results do not show a deviation from the theoretical results for ideal conditions they are not shown here.

Symmetric group delay distortion

Even symmetric ISI and odd symmetric ISI

Due to the odd symmetric ISI the optimum sampling instant is shifted. Now it is an interesting question whether the synchronizer tracks this optimum decision instant when the signal is severely distorted. Tab. VII.3-1 shows the tracking points for different synchronizers for the severe distortion shown in Fig. VII.1-3b obtained by simulation. The minimum error probability is obtained arround 0.16 to 0.06. There is no clear tendency for a particular synchronizer to be best in all cases. Simulations indicate that we have a flat minimum, i.e. slight deviations of the synchronizer tracking point from the optimum sampling instant for demodulation do not cause a significant demodulation performance loss.

The tracking point of some synchronizers can be determined analytically from the symbol pulse. The tracking points are indicated in Fig. VII.3-13b. The DDML tracks the maximum of the pulse. The center of the pulse where it has a width of T is the tracking point of DDEL(λ=1/2) and DTTL. The center of the pulse where it has a width of 2T is the tracking point of DDEL(λ=1), M&M, and mM&M. The theoretical tracking points agree quite well with the simulation results.

	Mean [T]		
	$\alpha = 0$	$\alpha = .5$	α=1.0
Minimum $P_e \approx$	0.16	0.16	0.06
DDML	0.18 (0.13)	0.21 (0.15)	0.22 (0.20)
NDAML	-	0.25	0.21
DTTL	0.18 (0.13)	0.20 (0.15)	0.21 (0.18)
M&M	0.14 (0.10)	0.08 (0.08)	-0.13 (-0.03)
GA	-	0.26	0.22

Table VII.3-1: Mean of timing error detector output for symmetric group delay distortion $\tau_{gr}(f) = 0.5 \quad |f|T$, $E_s/N_0 = 5$dB, (O)QPSK, simulation results and analytically determined tracking points (in parentheses)

To summarize, the most significant influence of linear distortions is the shift of the tracking point occuring with odd symmetric ISI. In all other cases the influence of linear distortions is small or negligible. The only critical exception is the M&M for α=1.0 where even the influence of symmetrical distortions may be strong.

VII.4. Tracking performance results for carrier synchronization - nonlinear distortions

VII.4.1 Mean phase shift

Ideal reference phase shift

From the TWT characteristics (Fig. I.4-4) and eq. VII.1.3-2 it is obvious that there is a phase offset between TWT input and output signal. For 3 dB input backoff we read off from Fig.I.4-4 $\psi = -0.7$ (-40°) and for 0 dB $\psi = -0.89$ (-51°). But the TWT input signal does not have constant power. When it is below the average power, the phase offset is smaller and when it is above the average power, the phase offset is larger.

Fig.VII.4-1 shows the AM-PM conversion over input power in a linear scale. We might expect that the mean phase shift is below the phase shift at the average power level ('nominal phase shift') since the slope decreases with increasing input power.

In simulations the TWT input and output signal both have been passed through a filter matched to the undistorted signal and the mean phase shift at the decision instants has been measured. The complex conjugate of the filtered undistorted signal has been multiplied by the filtered TWT output. This complex product has been averaged. The argument of the result is given in Tab. VII.4-1 (reference). These mean phase shifts are larger than the nominal phase shift for BPSK and QPSK and slightly smaller than the nominal phase shift for OQPSK. Note that we have a different mean phase shift when we consider the continuous signal instead of the decision instants.

There are complicated interactions but an explanation may be that in the matched filter the phase difference is weighted by the (matched) pulse amplitude. For a large pulse amplitude the contribution to the output signal is large and for a small amplitude it is small. Therefore, the mean phase shift is stronger for stronger input power fluctuations (i.e. for strong fluctuations around the average power).

	Input backoff	Mean phase shift Reference, DDML [rad]/[deg]	NDAML [deg]
BPSK			
$\alpha = 0.0$	0 dB	-0.933/-53.4	-51.7
$\alpha = 0.5$	0 dB	-0.923/-52.9	-52.6
$\alpha = 1.0$	0 dB	-0.925/-53.0	-52.7
BPSK			
$\alpha = 0.0$	3 dB	-0.760/-43.5	-42.0
$\alpha = 0.5$	3 dB	-0.744/-42.6	-42.4
$\alpha = 1.0$	3 dB	-0.741/-42.5	-42.2
QPSK			
$\alpha = 0.0$	0 dB	-0.899/-51.5	-51.3
$\alpha = 0.5$	0 dB	-0.906/-51.9	-51.5
$\alpha = 1.0$	0 dB	-0.908/-52.0	-51.5
QPSK			
$\alpha = 0.0$	3 dB	-0.724/-41.5	-41.0
$\alpha = 0.5$	3 dB	-0.723/-41.4	-41.0
$\alpha = 1.0$	3 dB	-0.724/-41.5	-41.2
OQPSK			
$\alpha = 0.0$	0 dB	-0.863/-49.4	-
$\alpha = 0.5$	0 dB	-0.874/-50.0	-50.2
$\alpha = 1.0$	0 dB	-0.885/-50.7	-50.7
OQPSK			
$\alpha = 0.0$	3 dB	-0.687/-39.4	-
$\alpha = 0.5$	3 dB	-0.691/-39.6	-40.3
$\alpha = 1.0$	3 dB	-0.697/-39.9	-39.6

Table VII.4-1 : Mean phase shift (determined with $E_s/N_0 = 20$ dB)

Note that the reference values are $-0.89/-51.0^{o}$ for 0 dB and $-0.70/-40.1^{o}$ for 3 dB input backoff (0.01 rad $\hat{=}$ 0.57^{o}).

Power fluctuations occur mainly when consecutive data symbols are different. Therefore, we expect a larger mean phase shift for BPSK than for QPSK where the transitions are smoother in the average. Finally, with OQPSK zero crossings in one channel (e.g. inphase data symbols) occur when the other channel (e.g. quadrature data symbols) is at its maximum, so that the power fluctuations are smoothed further. In Tab. VII.4-1 we indeed find the maximum phase shift for BPSK and a smaller phase shift for QPSK and OQPSK. The influence of the rolloff is small.

DDML

The mean phase shift, i.e. the carrier phase error for which the average of the phase detector output is zero, is shown in Fig. VII.4-2 - VII.4-4 for BPSK, QPSK and OQPSK for various rolloffs. Because of symmetry, decision errors do not influence the mean phase shift. Also, the noise is purely additive. Thus, the mean phase shift is independent of E_s/N_o. It is larger than the nominal phase shift (indicated by a solid line) for BPSK and QPSK with a smaller difference for QPSK, and it is (almost) identical with the nominal phase shift for OQPSK. Without decision errors the remodulated decisions equal the noise-free ideal reference used to determine the values in Tab. VII.4-1. Therefore, the mean phase shift is identical for DDML and ideal reference.

NDAML, V&V

For NDAML and V&V we have the same results. Again, it was found by simulation that the phase shift is independent of E_S/N_o. Also, the values are very close to the values obtained for DDML (except for OQPSK and rolloff = 0 where the NDAML does not operate) - see Tab. VII.4-1.

VII.4.2 Slopes

For BPSK and QPSK the slopes were found to be independent of TWT backoff and rolloff and identical to the slopes under ideal conditions. Therefore, the simulation results will not be given (see the normalized loop noise power spectral density $S(fT = 0)/K^2_{\theta\theta}$ in the next section for verification) except for the V&V. For OQPSK and NDAML the slopes are also independent of the TWT backoff. The situation is different for DDML as can be seen from Fig. VII.4-5a.

Up to now it was argued that OQPSK is influenced less by the nonlinearity. Therefore, it is very surprising that the DDML for OQPSK shows a strong influence of the TWT while for QPSK we have no major degradation. The degradation is caused by decision errors. The scatter diagram Fig. VII.4-5b shows that the TWT introduces significant intersymbol interference (ISI) which adds to the noise on the decision samples. Note that the imaginary part of $q(k;\hat{\tau})$ is shown over the real part of $p(k;\hat{\tau})$, i.e. the OQPSK signal after removal of the T/2 time shift of the quadrature channel. The ISI decreases with increasing rolloff so that we may expect smaller degradations for larger rolloff which indeed is found in Fig. VII.4-5a.

To study the degrading effects in more detail noise has been added to the distorted signal first, then, this signal was fed to the DDML detector directly and, in parallel, after phase rotation by 0.1 rad to a second DDML phase detector. The difference of both detector outputs is shown in Fig. VII.4-5c, and, in extended scale, in Fig. VII.4-5d. We see that the difference fluctuates arround a mean of 0.1 but frequently has negative peaks due to decision errors. Since these peaks all have a large negative amplitude the mean is reduced. Thus, also the slope is reduced. To verify this, the undistorted signal before the TWT and the distorted signal after the TWT have been corrupted by the same noise samples in a simulation. The signals have been fed to parallel DDML phase detectors. When the decisions from the synchronizer with the undistorted signal where used in the detector with the TWT distorted signal the performance of the latter was close to the performance under ideal conditions. On the other hand, using the decisions from the DDML detector with the TWT distorted signal in the detector with the undistorted signal yields a similar performance degradation as in Fig. VII.4-5a.

For the V&V synchronizer the slopes are given in Fig. VII.4-10a,b,c,d for BPSK, QPSK with f(u) = u and OQPSK with f(u) = 1 and f(u) = u. We again find that the influence of the TWT is small even for an input backoff of 0 dB. For OQPSK there is almost no difference between 3 dB and 0 dB input backoff except for rolloff $\alpha = 0$ which is an interesting case because the other non-data aided algorithm, NDAML, does not operate with $\alpha = 0$.

VII.4.3 Loop noise

Since with the travelling wave tube some kind of intersymbol interference (ISI) is introduced we may expect that the loop noise power spectrum is not flat. Therefore, we will first study $S(fT=0)/K_{\theta\theta}^2$ to obtain results for the constant part of the loop noise power spectrum. In a second step the shape of the spectrum is studied similarly as for ideal conditions.

Fig. VII.4-6 - VII.4-9 show the normalized loop noise power spectral density at fT = 0 for DDML and NDAML for all three modulation schemes considered in this study. It can been seen, that except for DDML with OQPSK there is no visible difference to the analytical results for ideal conditions. The normalized loop noise power spectral density at fT = 0 for V&V with f(u) =u is shown in Fig. VII.4-11a,b,d for all three modulation schemes and with f(u) = 1 for OQPSK in Fig. VII.4-11c. The results are also very close to the simulation results for ideal conditions. The two reasons for this result are that there is virtually no influence of the TWT operating point on the synchronizer slopes, and that the additional self-noise caused by the TWT is zero at fT = 0. Since for OQPSK we have found that the slopes of the DDML synchronizer are significantly smaller for a rolloff $\alpha = 0$ we also have a larger normalized power spectral density.

VII.4.4 Self-noise

The self-noise terms N_2 determined by simulation as described in section VII.1.5 are given in Tab. VII.4-2 for OQPSK. Since ISI effects now may also occur for BPSK and QPSK, the corresponding values of N_2 have also been determined by simulations. The values of N_2 that were found are so small that they can be neglected.

We must be cautious when we compare the values obtained for OQPSK with the analytical results obtained for ideal conditions. There, a different approach is used. The self-noise figures are a series expansion of the loop noise variance, while we here use a best-fit approximation of a measured spectrum. Therefore, we cannot expect that both approximations yield identical results (for identical (ideal) conditions). Yet, the values should not differ much (when the approximation error is small). Therefore, it is instructive to show the analytical, ideal case results for comparison. We find that the values for the TWT obtained by simulation and the analytically determined values are in the same order of magnitude. Wile the difference is small for DDML we have a factor of 2-3 for NDAML.

Yet, the comparison of results for different backoff shows, that the influence of the TWT is negligible. With the same arguments as given in the section assuming ideal conditions, we may conclude that the self-noise is negligible in satellite communication even when a TWT is used.

	Backoff	N_2	ideal case (analytical)
DD ML			
$\alpha = 0.1$	0 dB	0.280	0.368
$\alpha = 0.5$	0 dB	0.179	0.203
$\alpha = 1.0$	0 dB	0.107	0.125
DD ML			
$\alpha = 0.1$	3 dB	0.269	
$\alpha = 0.5$	3 dB	0.177	
$\alpha = 1.0$	3 dB	0.106	
NDA ML			
$\alpha = 0.1$	0 dB	316.0	147.
$\alpha = 0.5$	0 dB	9.08	3.24
$\alpha = 1.0$	0 dB	1.54	0.50
NDA ML			
$\alpha = 0.1$	3 dB	332.0	
$\alpha = 0.5$	3 dB	9.58	
$\alpha = 1.0$	3 dB	1.57	
V&V, f(u)=1			
$\alpha = 0.1$	0 dB	3.53	
$\alpha = 0.5$	0 dB	1.26	
$\alpha = 1.0$	0 dB	0.50	
V&V, f(u)=1			
$\alpha = 0.1$	3 dB	3.65	
$\alpha = 0.5$	3 dB	1.29	
$\alpha = 1.0$	3 dB	0.49	
V&V, f(u)=u			
$\alpha = 0.1$	0 dB	8.17	
$\alpha = 0.5$	0 dB	1.96	
$\alpha = 1.0$	0 dB	0.72	
V&V, f(u)=u			
$\alpha = 0.1$	3 dB	7.50	
$\alpha = 0.5$	3 dB	1.86	
$\alpha = 1.0$	3 dB	0.84	

Tab. VII.4-2 : Self-noise performance of carrier synchronizers (OQPSK), simulation

VII.4.5 Conclusions

It was found that the travelling wave tube introduces a mean phase shift on the signal which slightly depends on modulation scheme (BPSK, QPSK, OQPSK) and rolloff, and which strongly depends on the backoff level. For a given modulation scheme and backoff all carrier synchronizer types indicate practically the same mean phase shift as found using an ideal reference. Also, there is no significant tracking performance degradation, i.e. the influence of the TWT on carrier synchronization is small, except for OQPSK DDML.

Enhanced self-noise effects were not found. So they still remain irrelevant for satellite communication in this particular case.

VII.5 Tracking performance results for symbol synchronizers

VII.5.1 Introduction

For most of the symbol synchronizers the carrier phase error must be corrected. As was discussed in section VII.4.1, all carrier synchronizers estimate almost identical mean phase shifts for the same backoff, rolloff and modulation scheme. Therefore, prior to the symbol synchronizers the signal is rotated back by the corresponding mean phase shift as given in Tab. VII.4-1.

For the influence of the TWT on the pulse shape, see the discussions in section VII.1.3. As for the carrier synchronizers, simulation has been used to study the influence of nonlinear distortions.

VII.5.2 Synchronizer performance neglecting self-noise

In contrast to the ideal case we now have different curves for OQPSK and QPSK since the influence of the TWT on OQPSK and QPSK may be different.

VII.5.2.1 Slopes

<u>DDML</u>

We find a significant influence of the TWT on the DDML synchronizer performance (Fig. VII.5-1a,b,c). For BPSK and QPSK we have similar results, the slope is strongly degraded. The degradation increases when the TWT operating point approaches the saturation level. We observe that the strongest degradations occur for a rolloff $\alpha = 1$, degradations are small for $\alpha = 0.5$ and increase again for $\alpha = 0$. For OQPSK we have almost no degradation for rolloff factors of 0.5 and higher, only for rolloff zero we observe a significant degradation which is still smaller than for BPSK and QPSK.

NDAML

For NDAML (Fig. VII.5-3) separate curves for BPSK and QPSK slopes are not given, the differences are marginal. Again we observe an increasing degradation for α -> 1. For BPSK and QPSK the degradation at $\alpha = 1$ is roughly 50% higher than for DDML. As for DDML the degradation for offset QPSK is very small, slightly increasing for α -> 1. Note that the slope was found to be independent of the signal-to-noise ratio as for ideal conditions.

DTTL, DDEL(λ=1/2)

As a representative of this group the DTTL synchronizer has been simulated. We observe for the DTTL (Fig. VII.5-5a,b,c) similar results as for DDML, i.e. degradation is strong for α -> 1 and for α -> 0 and small for $\alpha \approx 0.5$. For $\alpha = 1$ the degradations are approximately 40% below the degradations for DDML, for other values of the rolloff the difference is less significant. Again, the degradations increase when the TWT operating point approaches saturation level.

M&M, mM&M, DDEL(λ=1)

As a representative of this group the mM&M synchronizer has been simulated. With the mM&M algorithm (Fig. VII.5-7a,b,c) we observe an opposite effect than before. Now for rolloff $\alpha = 1$ performance is even improved. Note that contrary to e.g. the DDML, for the mM&M the slope increases with decreasing rolloff. Again, we observe similar effects for BPSK and QPSK. We observe a slight improvement for rolloff $\alpha = 1$, a slight degradation for rolloff $\alpha = 0.5$, and a strong degradation (in particular around 5 dB E_s/N_0) for rolloff $\alpha = 0$.

With OQPSK for the mM&M we have ideal case performance for rolloff $\alpha = 1$ a slight degradation for $\alpha = 0.5$ and a degradation as for QPSK for $\alpha = 0$.

GA, NDAEL(λ=1/2)

As a representative of this group the GA synchronizer has been simulated. For BPSK and QPSK we have a degradation which increases with rolloff and is slightly larger for 0 dB input backoff as compared to 3 dB input backoff (Fig. VII.5-9a). For OQPSK the degradation is very small.

VII.5.2.2 Loop Noise

Performance degradations are mainly due to the slope degradations so that the discussions for the slopes can be directly applied to the loop noise.

DDML

For DDML, Fig. VII.5-2a,b,c, we find degradations of up to 4 dB for all modulation schemes with rolloff $\alpha = 0$ at $E_S/N_0 = 5$ dB. For high E_S/N_0 the degradation decreases. For rolloff $\alpha = 1$ we have a small degradation with BPSK and QPSK while for OQPSK the results do not differ from the ideal case figures. For rolloff $\alpha = 0.5$ we do not have degradations for any modulation scheme.

NDAML

For NDAML, Fig. VII.5-4a,b,c, we have a degradation of tracking performance for BPSK and QPSK which is almost independent of the rolloff and practically no degradation for OQPSK. Here only rolloff 1 and 0.5 are shown, for rolloff zero NDAML does not operate.

DTTL

For DTTL, Fig. VII.5-6a,b,c, we observe mainly strong degradations for rolloff zero (up to 6 dB). For rolloff = 1 the degradation is much smaller. Again we have less degradation for QPSK and no degradations for OQPSK except for small rolloff where we still observe a degradation of approx. 3 to 4 dB.

mM&M

For the mM&M, Fig. VII.5-8a,b,c, we observe a slight improvement for rolloff $\alpha = 1$ and a slight degradation around $E_S/N_0 = 5$ dB for $\alpha = 1$ and $\alpha = 0.5$.

GA

For the GA synchronizer, Fig. VII.5-9b,c,d, we have slight degradations for rolloff $\alpha = 1$ and $\alpha = 0.5$ with BPSK and QPSK, which are almost constant over E_S/N_0. With OQPSK we do not observe a performance loss.

Summary

For OQPSK we have no visible performance degradation compared to the ideal case.

There is a common effect for the synchronizers which use decisions, in particular, DDML and DTTL. The degradation is moderate for high E_S/N_0 and stronger for moderate E_S/N_0. This can be explained from the fact that for high E_S/N_0 the decision error probability is so small, that a slight increase does not influence the slope. Therefore, a slope degradation is mainly due to pulse misshaping and intersymbol interference caused by the TWT. In the moderate E_S/N_0 region the slope strongly depends on decision errors. Therefore, in this range an increased decision error rate due to the influence of the TWT further degrades performance.

VII.5.3 Self-noise

As for the loop noise we now have to consider OQPSK separately because the influence of the TWT may be different for QPSK and OQPSK. Self-noise performance has been determined as discussed in section VII.5.1.5. Tab. VII.5-1 shows the factor N_2 for DDML and NDAML with TWT for all three modulation schemes. When we compare the values of N_2 for 0 dB backoff and 3 dB input backoff we find little difference. Therefore, we may conclude that the influence of the TWT on the self-noise performance is small - the normalized loop noise power spectrum was found to be significantly different for 0 dB and 3 dB backoff. When we compare this table with the self-noise performance for ideal conditions shown in Tab. II.3-1 we find that N_2 with TWT is larger by a factor of 3 to 5, except for rolloff = 1 where the difference is larger. This large difference may be due to the measurement uncertainty since the values of N_2 are very small.

With DDML and the NDAML we have considered a synchronizer with small self-noise and a synchronizer with strong self-noise. In both cases we found a similar trend in the values of N_2. The influence of the TWT was found to be small. Therefore, it can be concluded that the influence of the TWT for the other synchronizers will be small as well.

An interesting exception may be the case that a synchronizer is primarily free of self-noise. Therefore, the influence of the TWT on self-noise

cancellation in DDML and NDAML synchronizers has been studied as well. The corresponding values of N_2 are shown in Tab. VII.5-2. Note that the self-noise cancellation filter was adjusted to the undistorted case, i.e. without TWT. We find in Tab.VII.5-2 that due to the TWT self-noise cancellation is no longer perfect. But the values of N_2 are by orders of magnitude below the values without self-noise cancellation.

As a conclusion for all these cases the same arguments as in section II.3-4 apply. For realistic values of $2B_LT$ the tracking error variance in a satellite communication environment (say, $E_s/N_o \leq 10$ dB) is still dominated by the additive noise contribution (possible exceptions are the quadratic NDAML symbol synchronizers for small rolloff).

VII.5.4 Conclusions

What has been predicted from the influence of the travelling wave tube has been found in the loop noise figures. The degradations are strongest for BPSK, smaller for QPSK and often negligible for OQPSK. For DDML and NDAML we have similar influence of the TWT and for DTTL we have a strong influence for a small rolloff. For the modified Müller&Mueller we observe the strongest degradations for small rolloff while we even have an improvement for rolloff=1.

	Backoff	N_2		
		BPSK	QPSK	OQPSK
DD ML				
$\alpha = 0.1$	0 dB	2.46	1.04	1.2
$\alpha = 0.5$	0 dB	0.52	0.25	0.28
$\alpha = 1.0$	0 dB	0.08	0.05	0.023
DD ML				
$\alpha = 0.1$	3 dB	2.28		
$\alpha = 0.5$	3 dB	0.52		
$\alpha = 1.0$	3 dB	0.07		
NDA ML				
$\alpha = 0.1$	0 dB	65.4	31.0	30.6
$\alpha = 0.5$	0 dB	1.28	0.61	0.61
$\alpha = 1.0$	0 dB	0.13	0.07	0.029
NDA ML				
$\alpha = 0.1$	3 dB	73.0		
$\alpha = 0.5$	3 dB	1.29		
$\alpha = 1.0$	3 dB	0.10		

Table VII.5-1: Self-noise performance of symbol synchronizers (simulation)

	Backoff	N_2		
		BPSK	QPSK	OQPSK
DD ML				
$\alpha = 0.1$	0 dB	0.019	0.034	0.025
$\alpha = 0.5$	0 dB	0.014	0.018	0.0073
$\alpha = 1.0$	0 dB	0.002	0.009	0.0005
NDA ML				
$\alpha = 0.1$	0 dB	0.199	0.34	0.12
$\alpha = 0.5$	0 dB	0.013	0.035	0.014
$\alpha = 1.0$	0 dB	0.004	0.011	0.0008

Table VII.5-2: Self-noise performance of symbol synchronizers with self-noise cancellation (simulation)

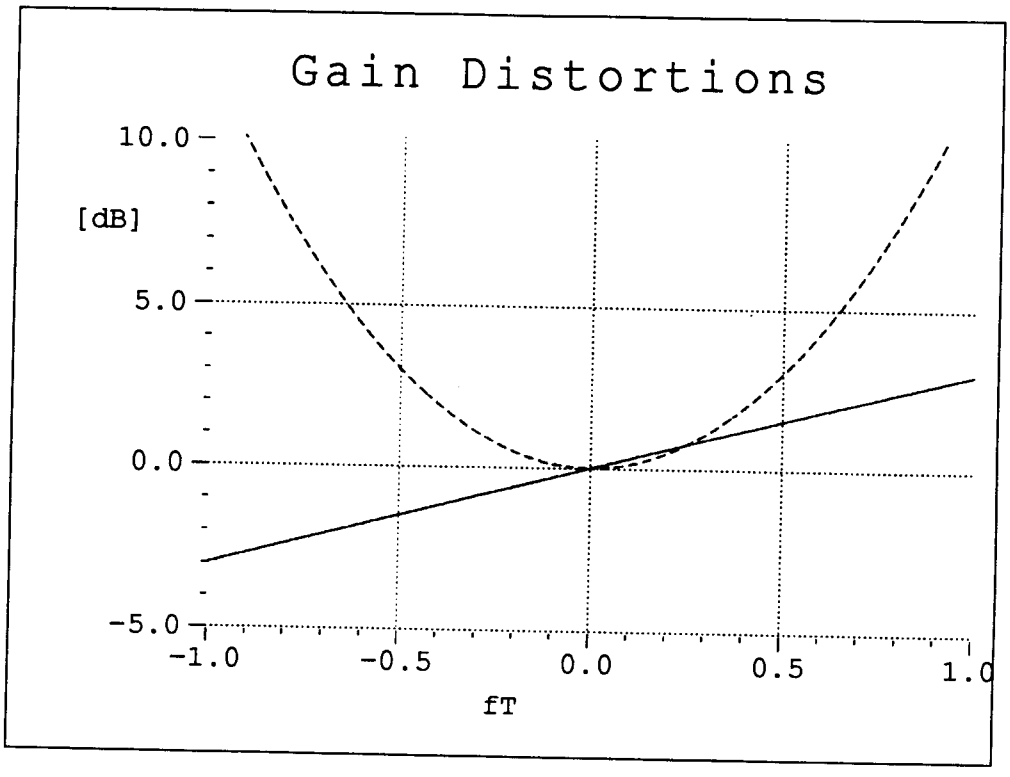


Fig. VII.1-1a: Linear distortions: symmetric gain distortion - gain in dB ~ f^2 (dashed line), asymmetric gain distortion - gain in dB ~ f (solid line)

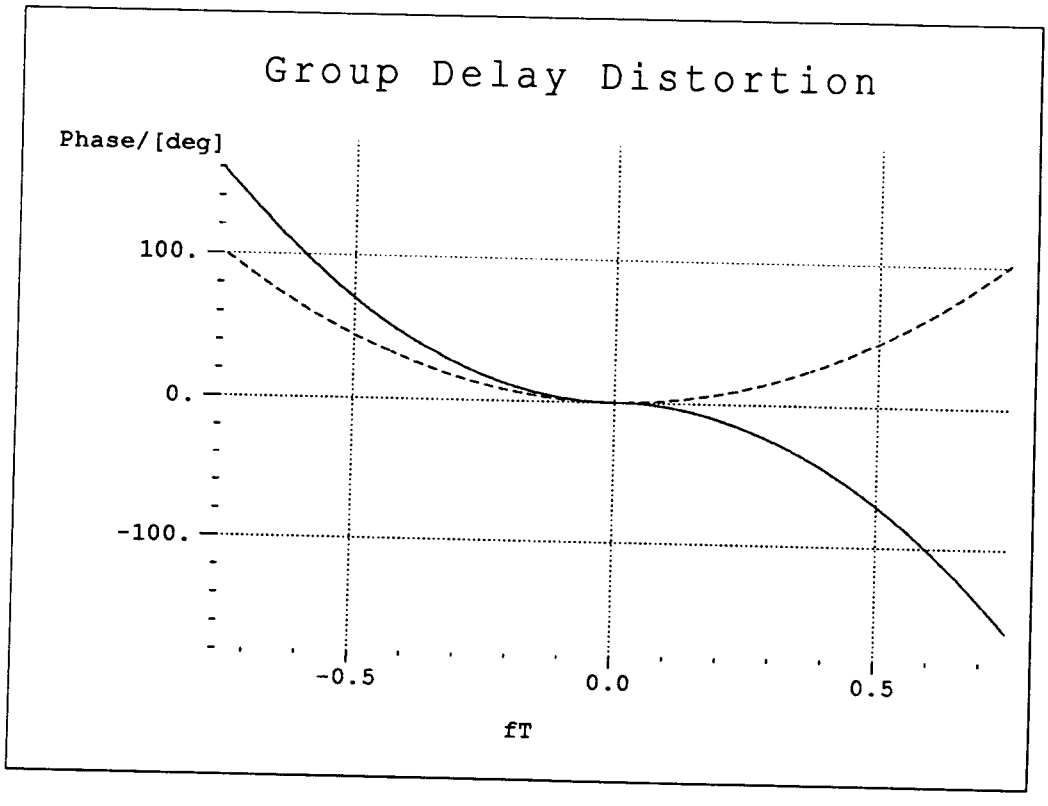


Fig. VII.1-1b: Linear distortions: symmetric group delay distortion ~ |f| (solid line), asymmetric group delay distortion ~ f (dashed line)

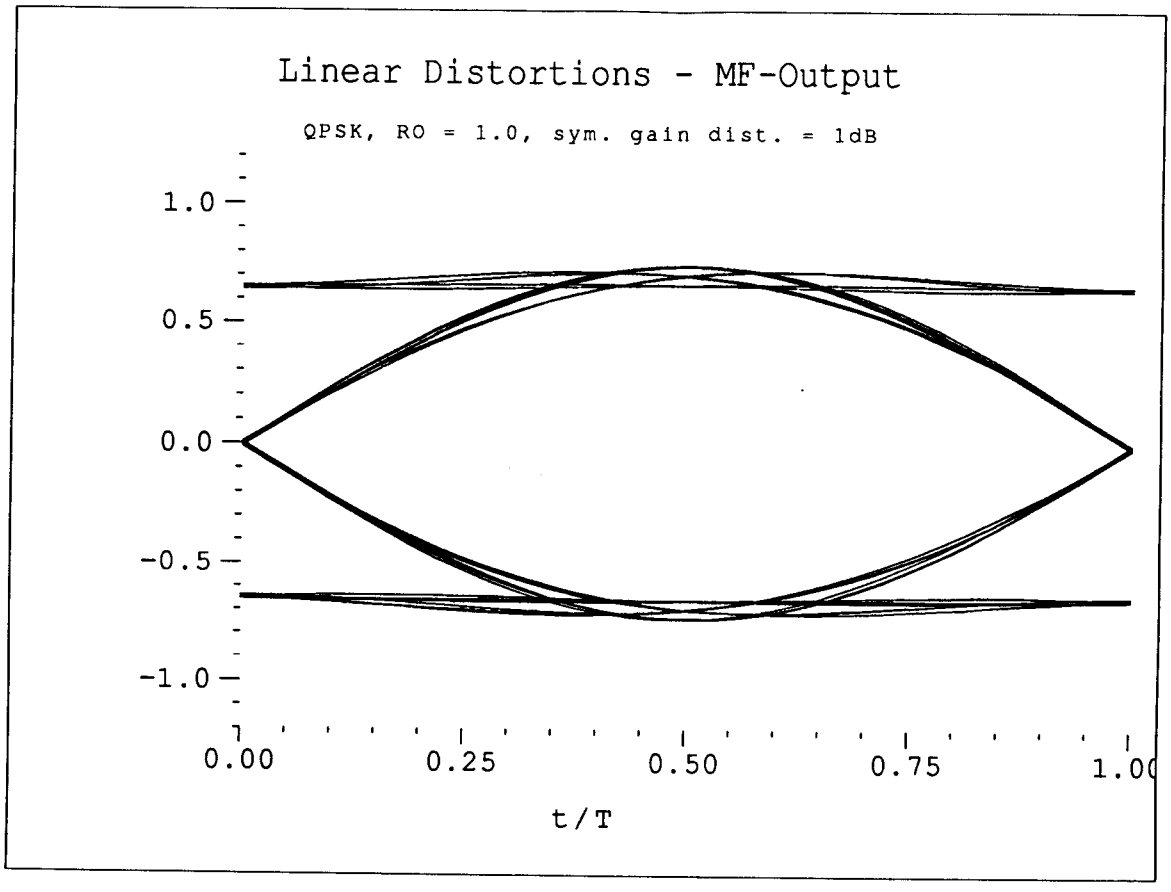


Fig. VII.1-2a: Matched filter output eye diagram (QPSK), symmetric gain distortion, gain in dB $\sim f^2$, 1dB at fT=0.5, causes even symmetric ISI

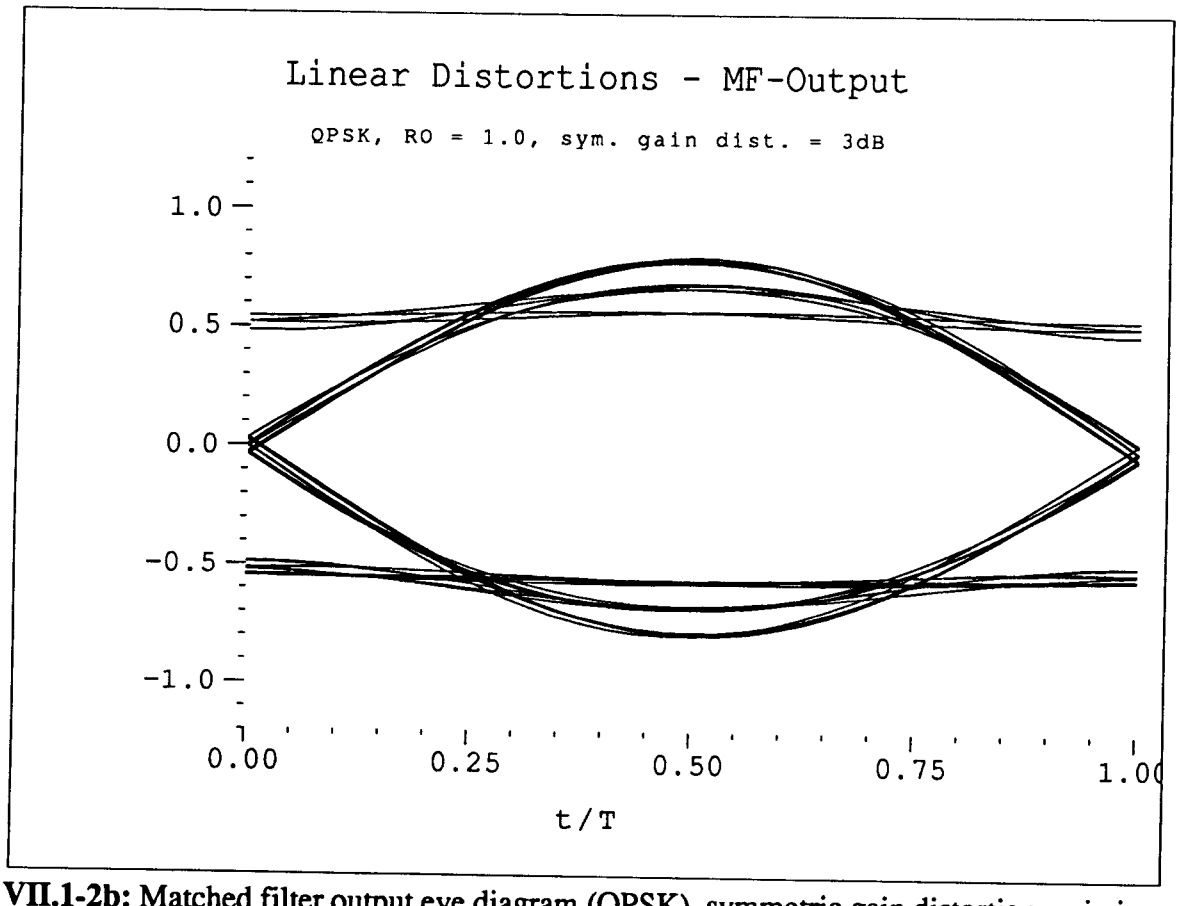


Fig. VII.1-2b: Matched filter output eye diagram (QPSK), symmetric gain distortion, gain in dB $\sim f^2$, 3dB at fT=0.5, causes even symmetric ISI

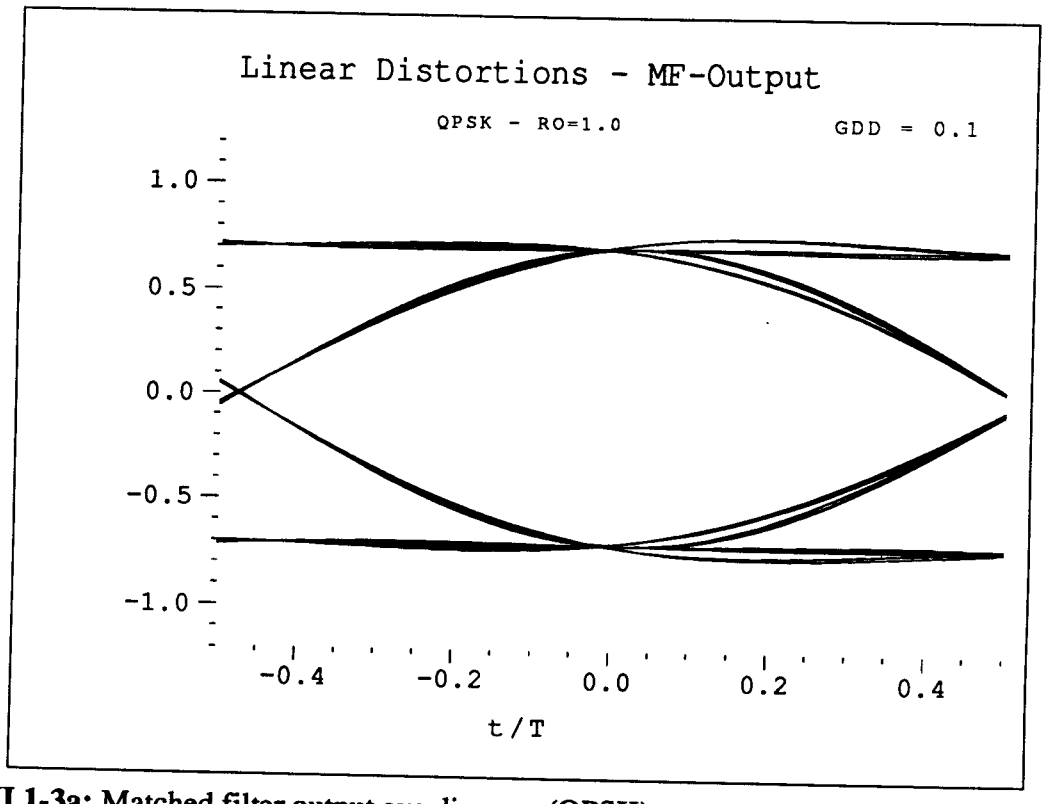


Fig. VII.1-3a: Matched filter output eye diagram (QPSK), symmetric group delay distortion ~ |f| , $\Delta\tau/T=0.1$, causes even symmetric ISI plus odd symmetric ISI

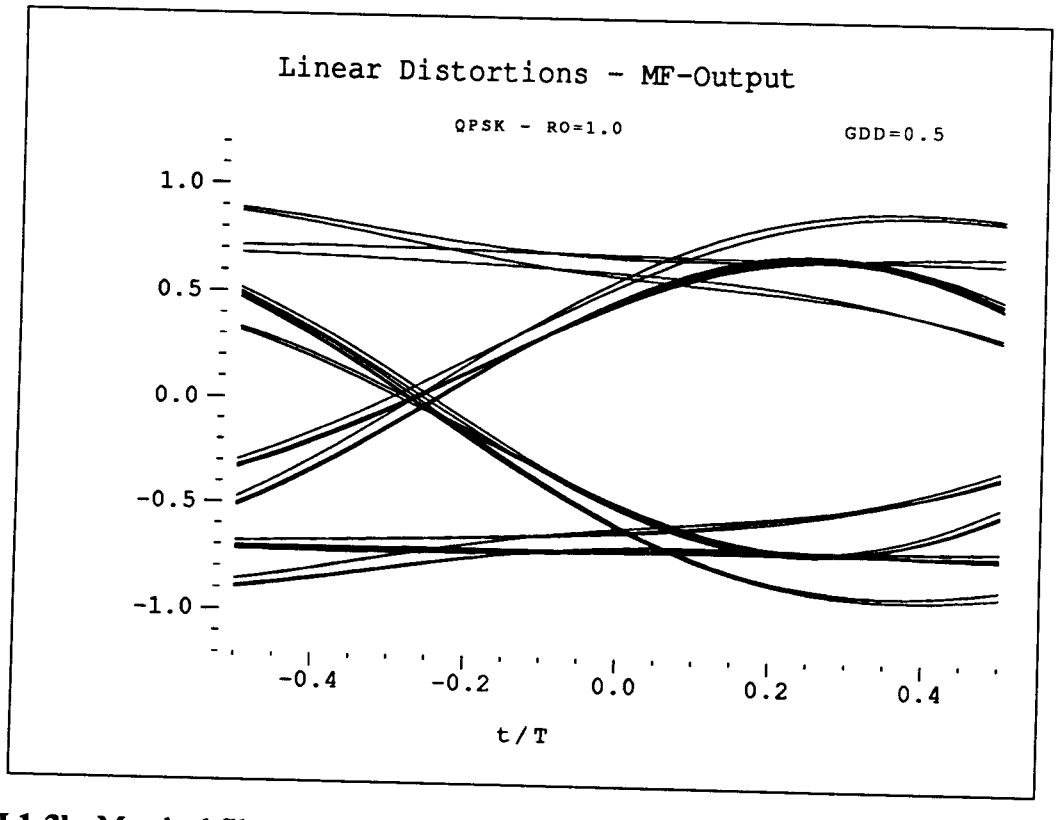


Fig. VII.1-3b: Matched filter output eye diagram (QPSK), symmetric group delay distortion ~ |f| , $\Delta\tau/T=0.5$, causes even symmetric ISI plus odd symmetric ISI

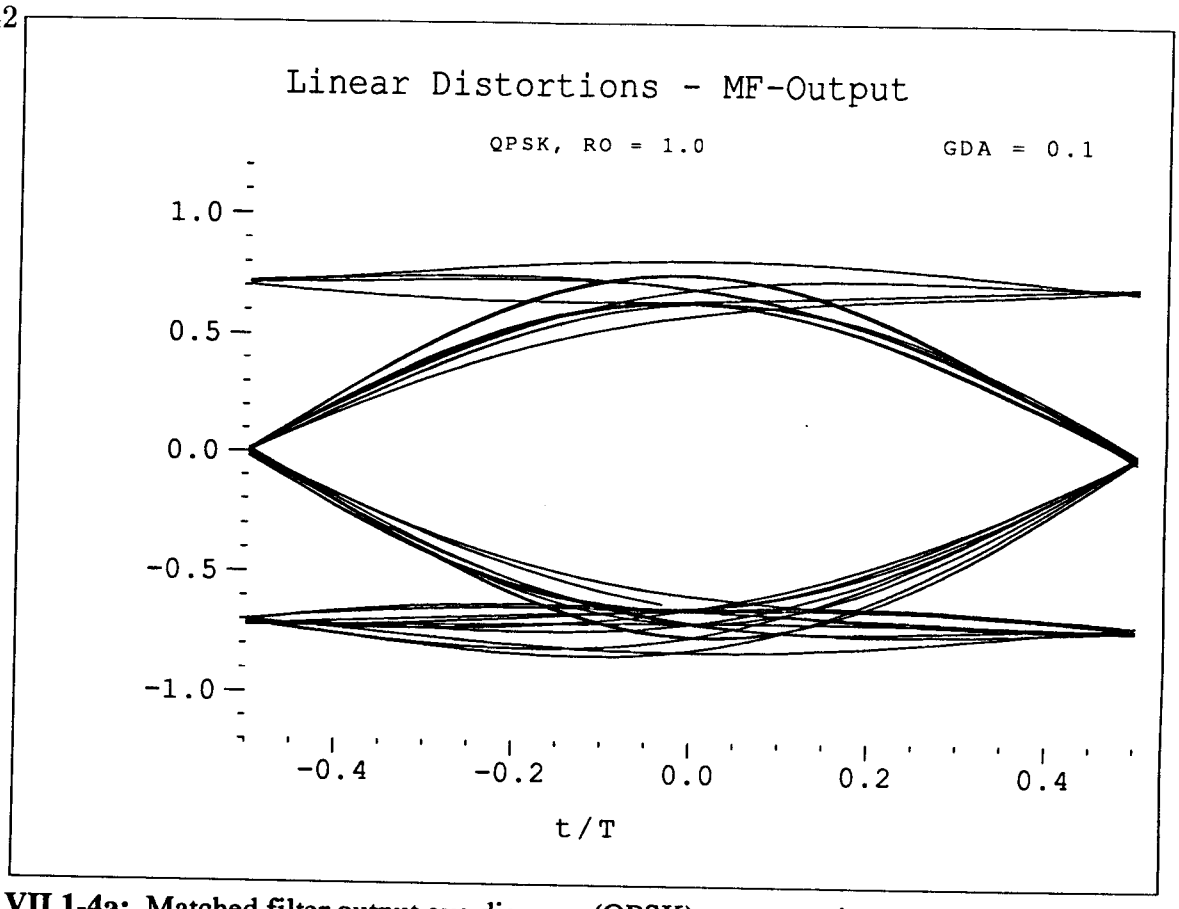


Fig. VII.1-4a: Matched filter output eye diagram (QPSK), asymmetric group delay distortion ~ f , $\Delta\tau/T=0.1$, causes (small) even symmetric ISI plus even symmetric Xtalk

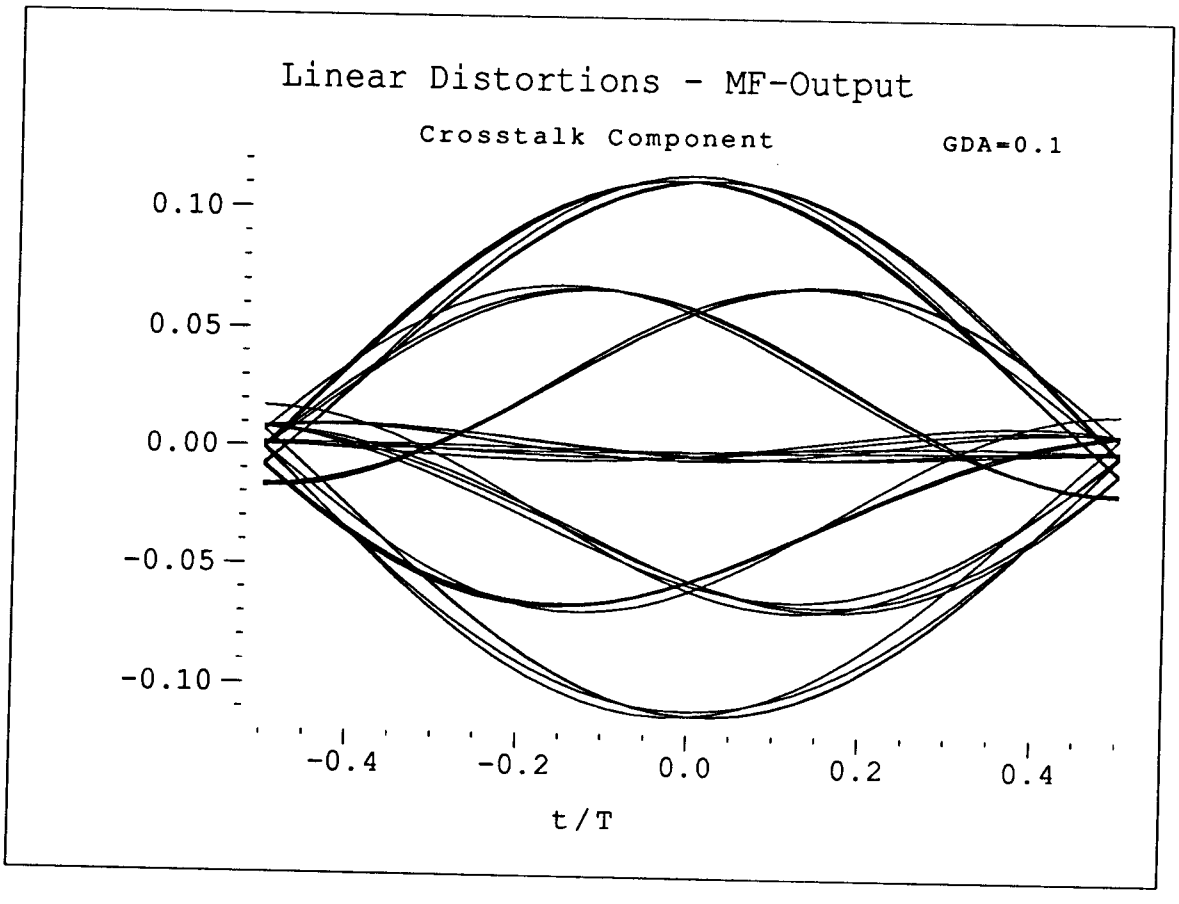


Fig. VII.1-4b: Matched filter output eye diagram (QPSK), asymmetric group delay distortion ~ f , $\Delta\tau/T=0.1$, cross talk component only

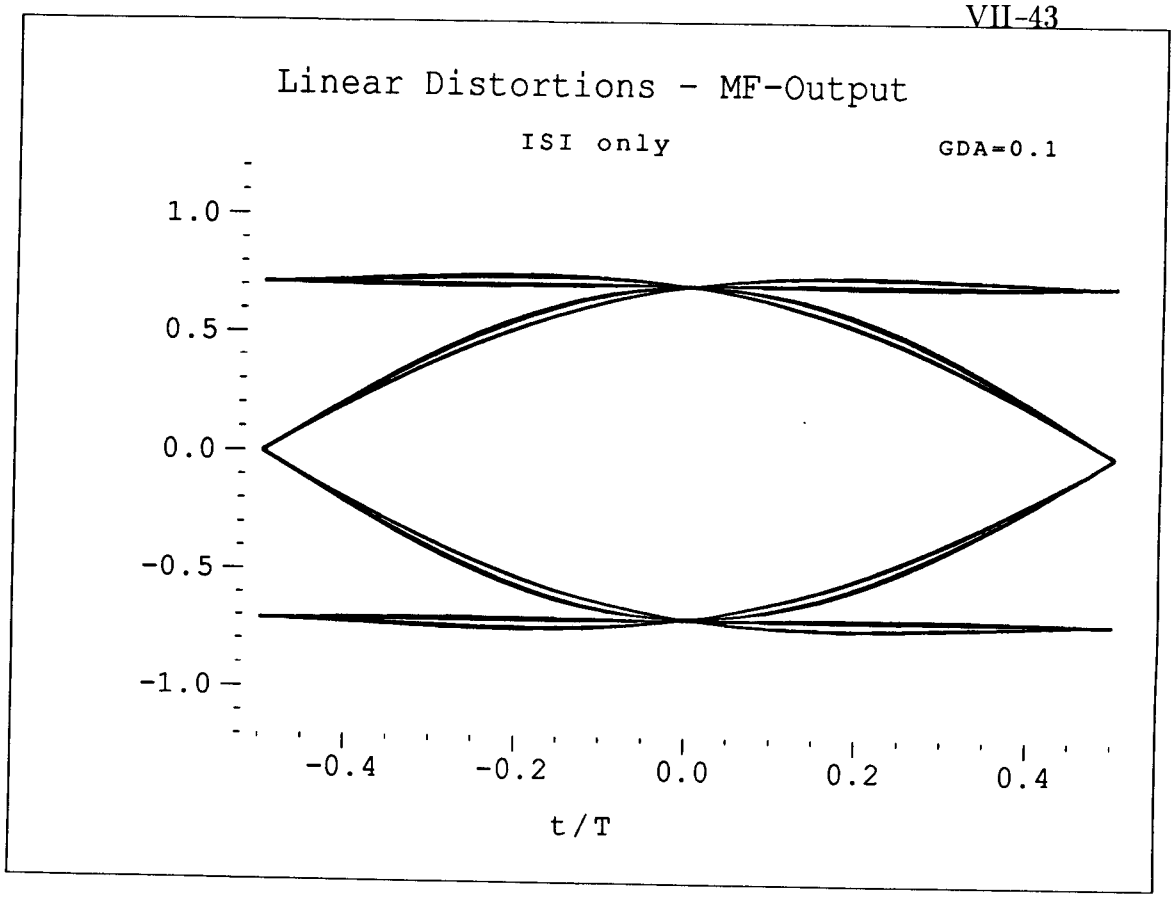


Fig. VII.1-4c: Matched filter output eye diagram (BPSK => ISI only), asymmetric group delay distortion ~ f , $\Delta\tau/T=0.1$

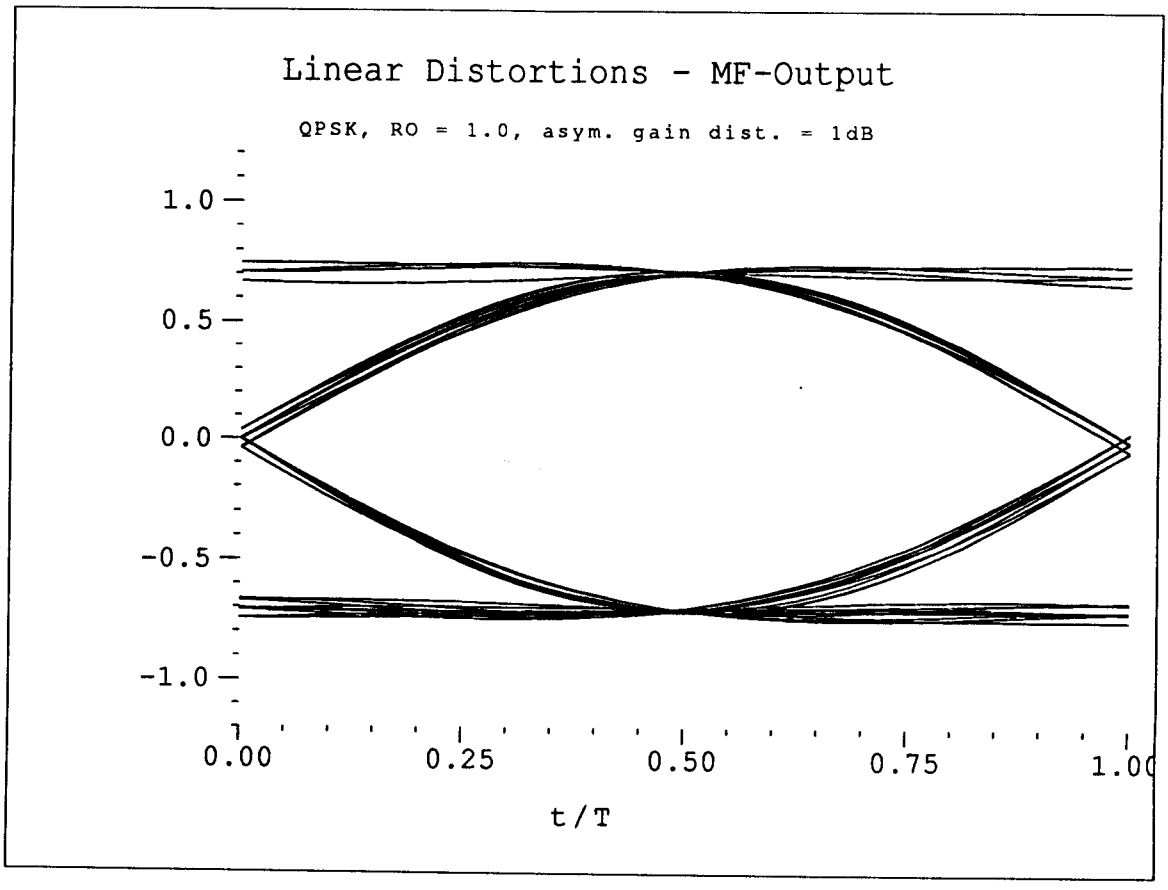


Fig. VII.1-5a: Matched filter output eye diagram (QPSK), asymmetric gain distortion, gain in dB ~ f, 1dB over 1/T, causes (small) even symmetric ISI plus odd Xtalk

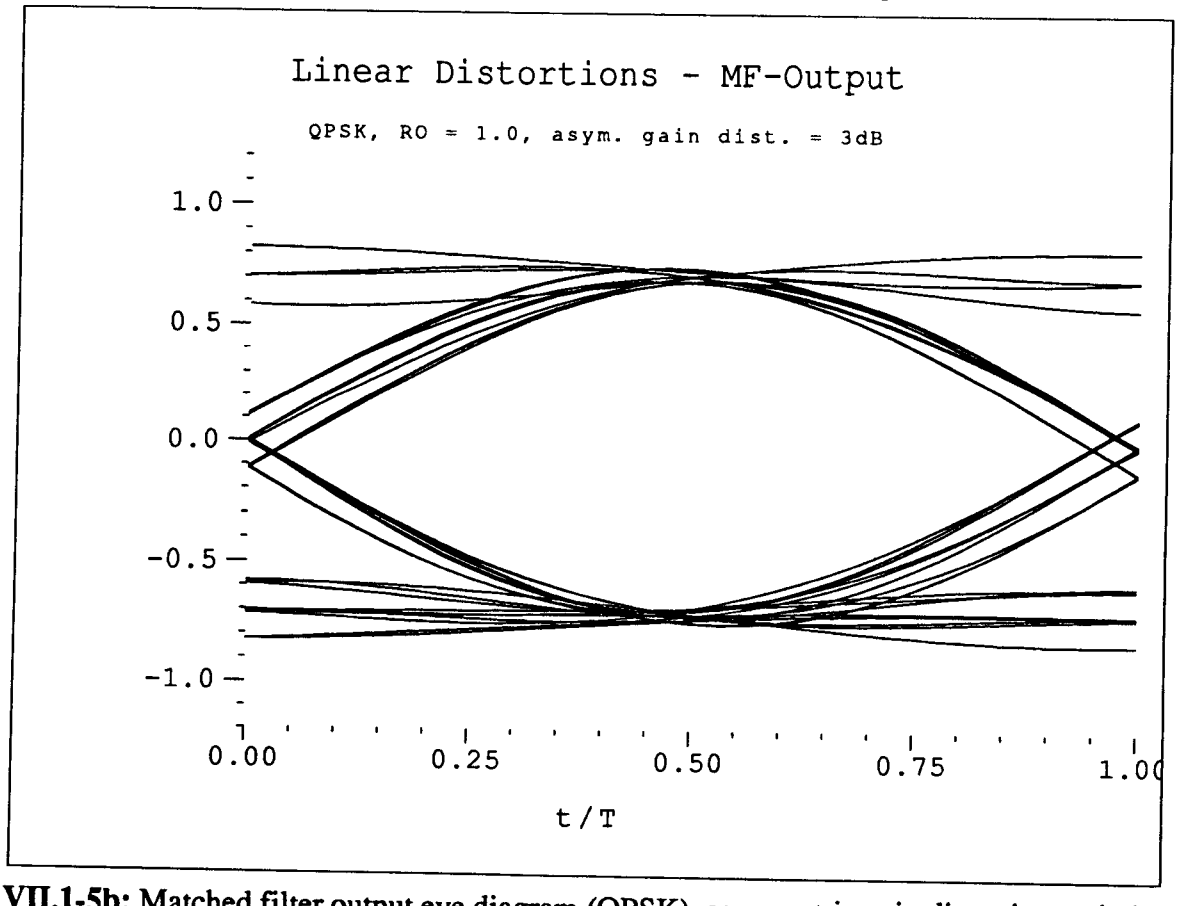


Fig. VII.1-5b: Matched filter output eye diagram (QPSK), asymmetric gain distortion, gain in dB ~ f, 3dB over 1/T, causes (small) even symmetric ISI plus odd Xtalk

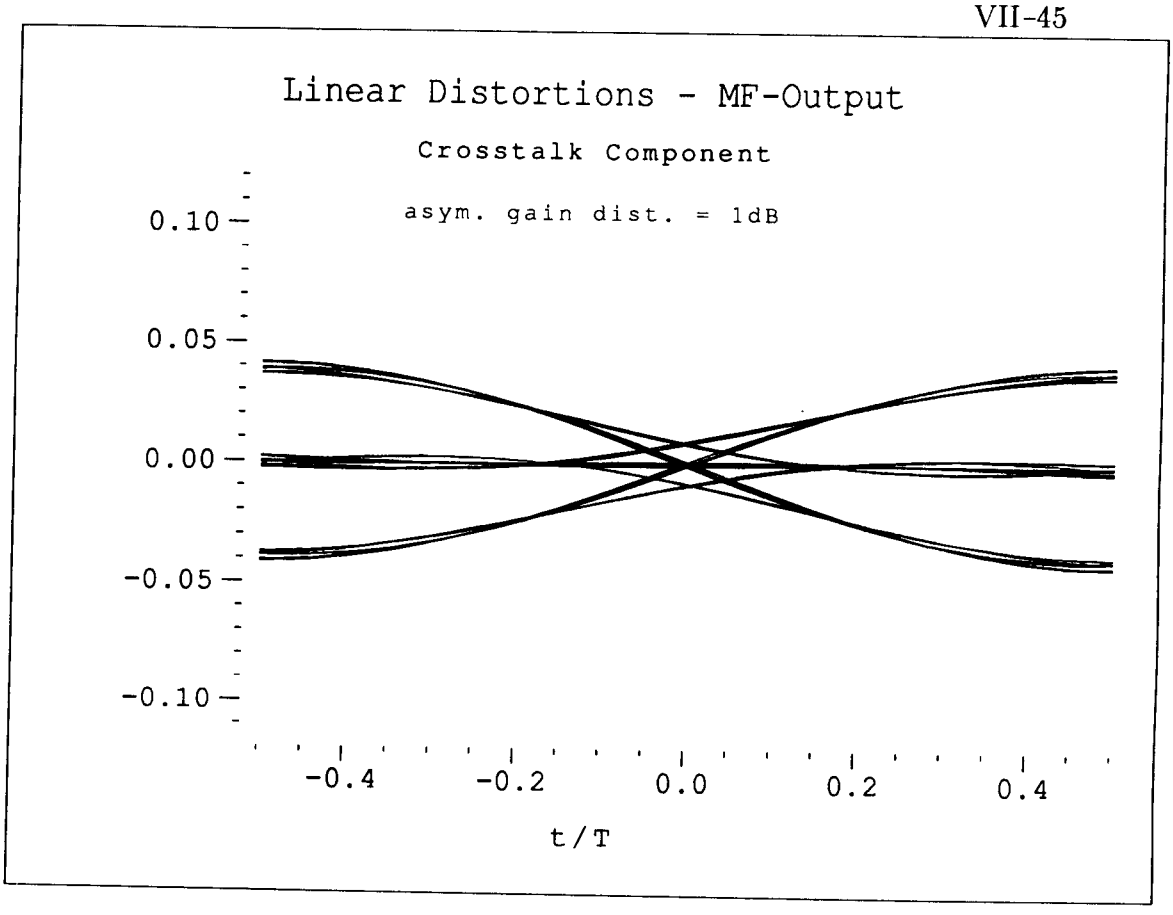


Fig. VII.1-5c: Matched filter output eye diagram (QPSK), asymmetric gain distortion, gain in dB ~ f , 1dB over 1/T, cross talk component only

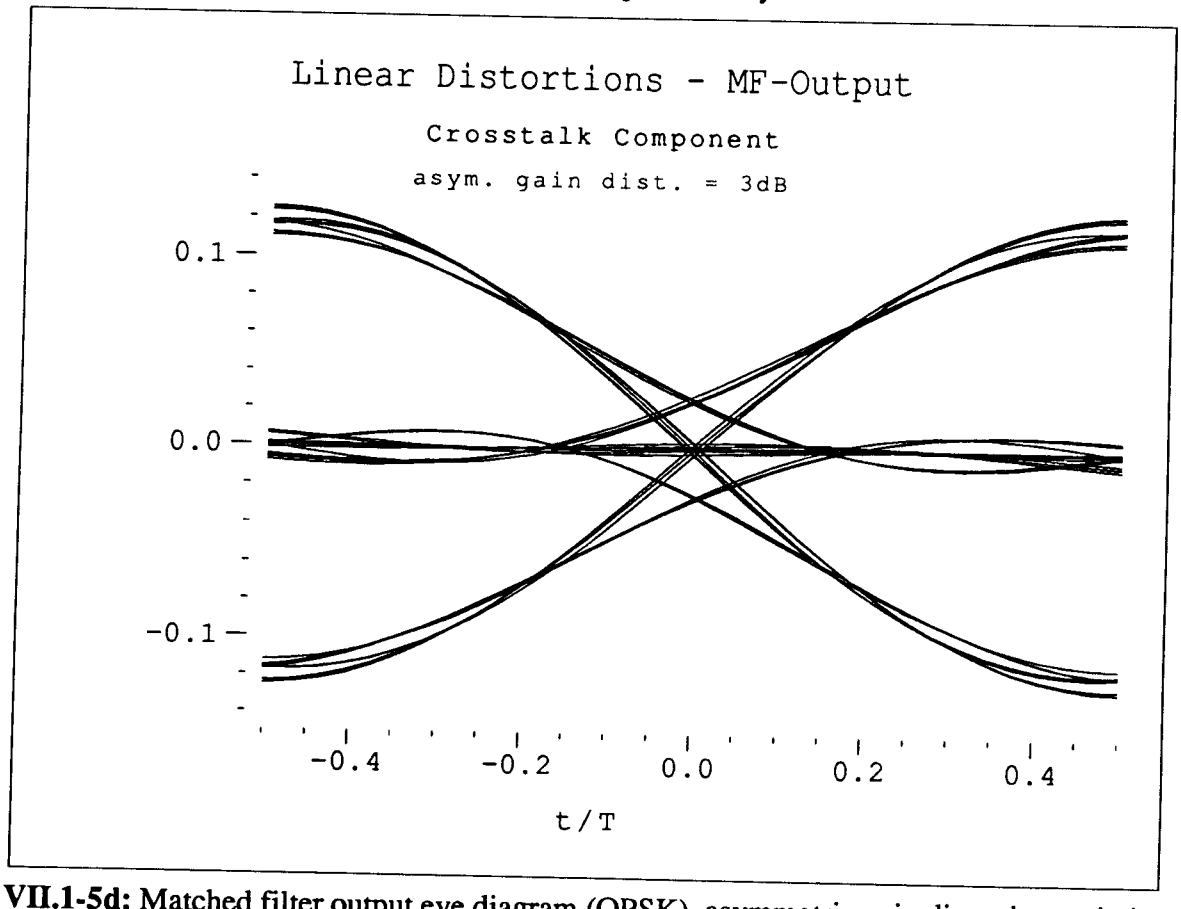


Fig. VII.1-5d: Matched filter output eye diagram (QPSK), asymmetric gain distortion, gain in dB ~ f , 3dB over 1/T, cross talk component only

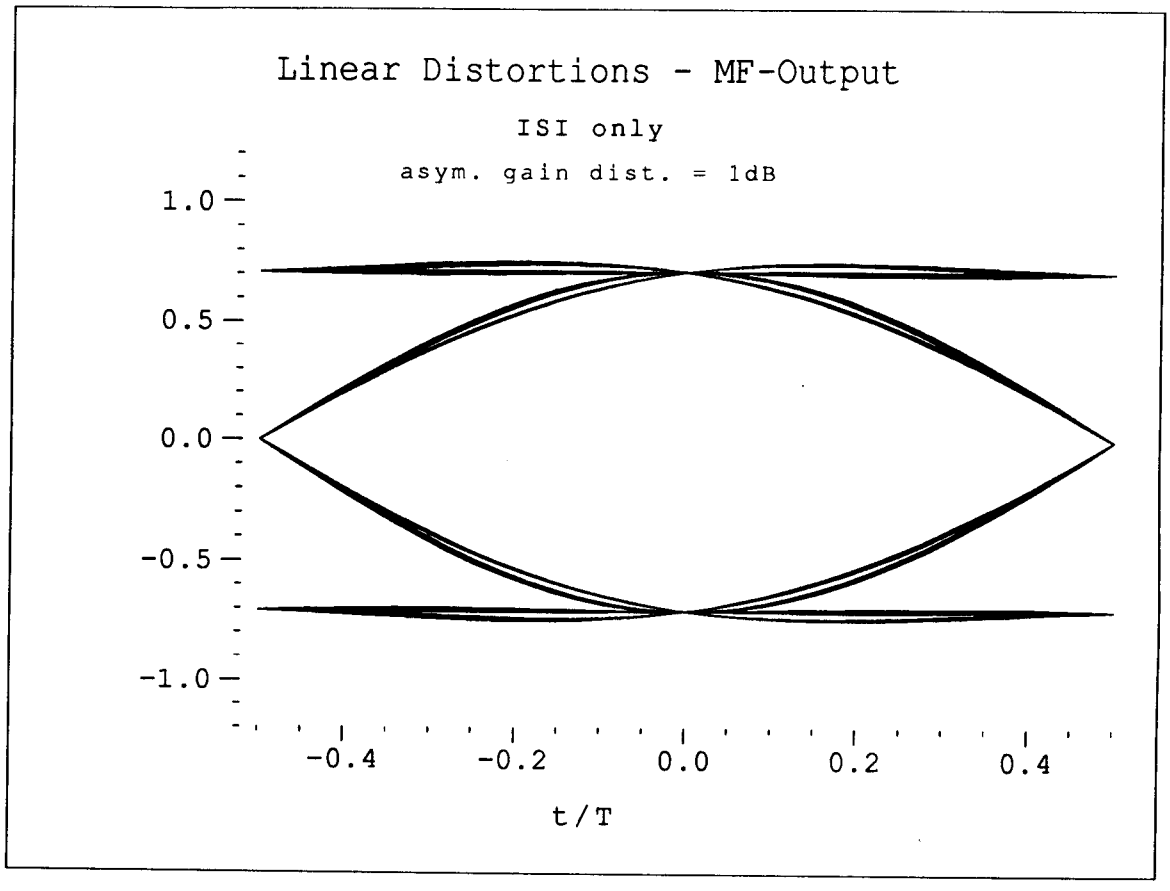


Fig. VII.1-5e: Matched filter output eye diagram (BPSK => ISI only), asymmetric gain distortion, gain in dB ~ f , 1dB over 1/T

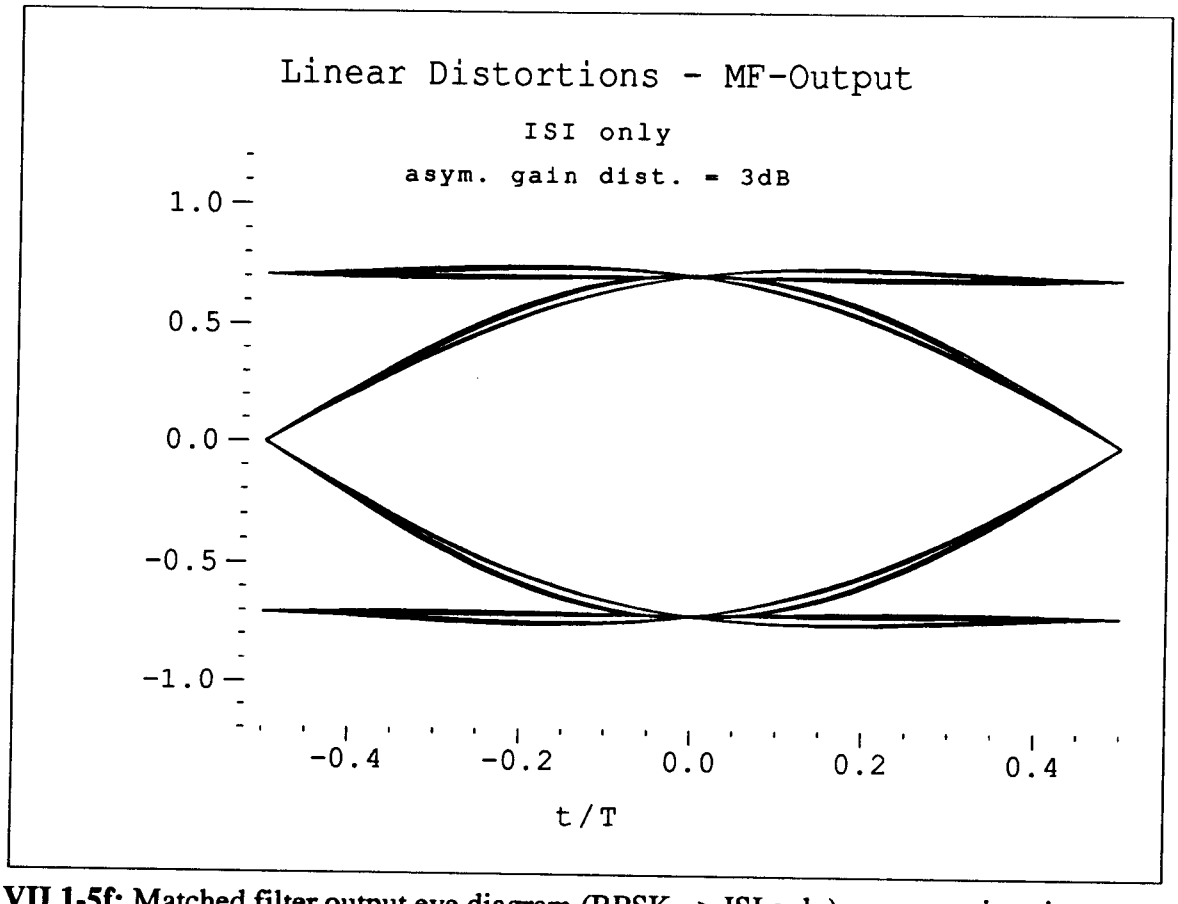


Fig. VII.1-5f: Matched filter output eye diagram (BPSK => ISI only), asymmetric gain distortion, gain in dB ~ f , 3dB over 1/T

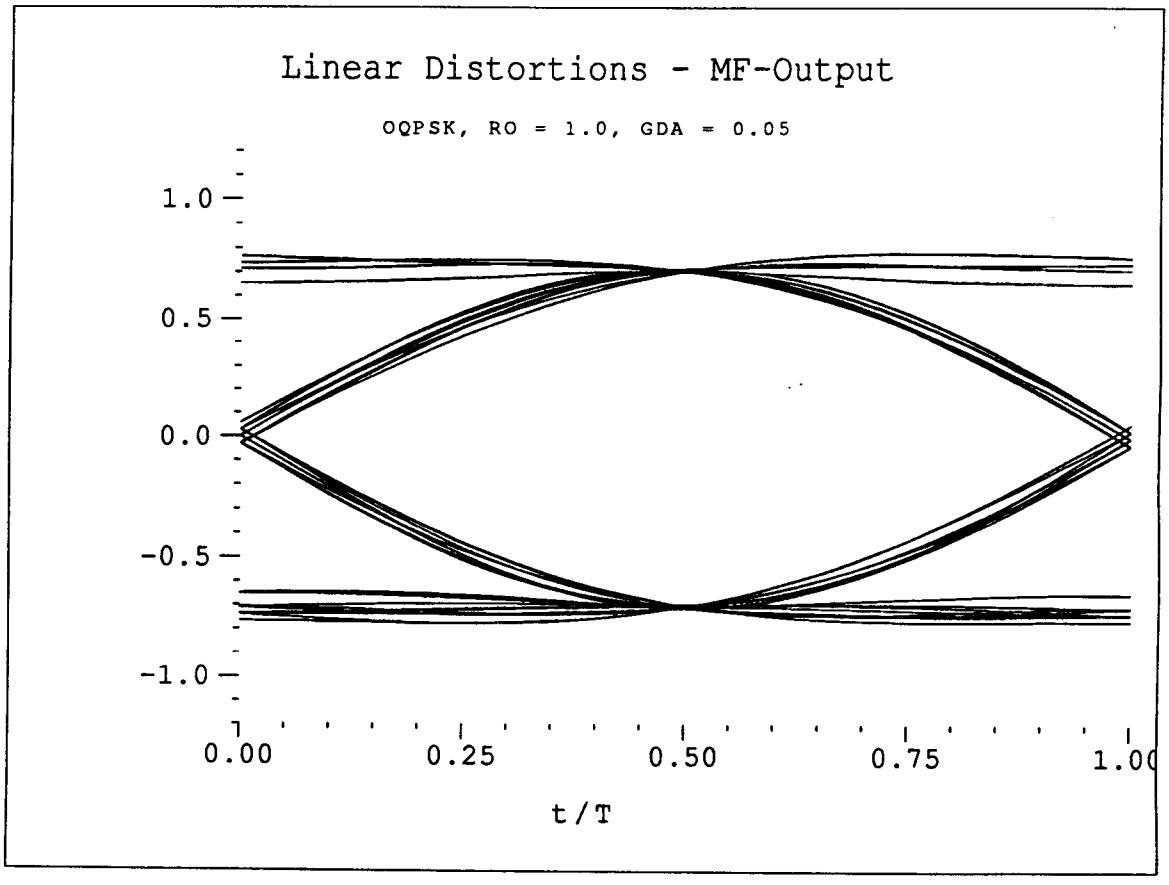


Fig. VII.1-6a: Matched filter output eye diagram (OQPSK), asymmetric group delay distortion ~ f , $\Delta\tau/T=0.05$

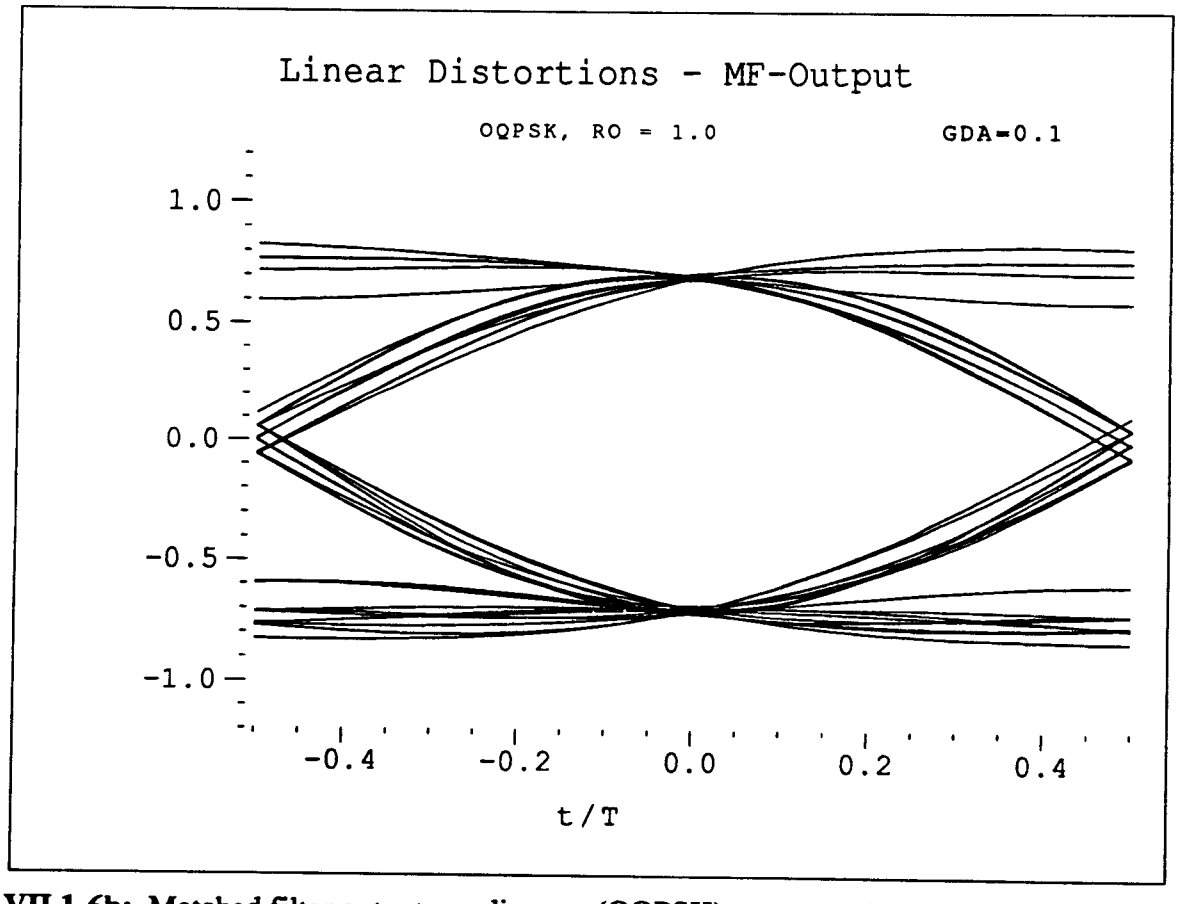


Fig. VII.1-6b: Matched filter output eye diagram (OQPSK), asymmetric group delay distortion ~ f , $\Delta\tau/T=0.1$

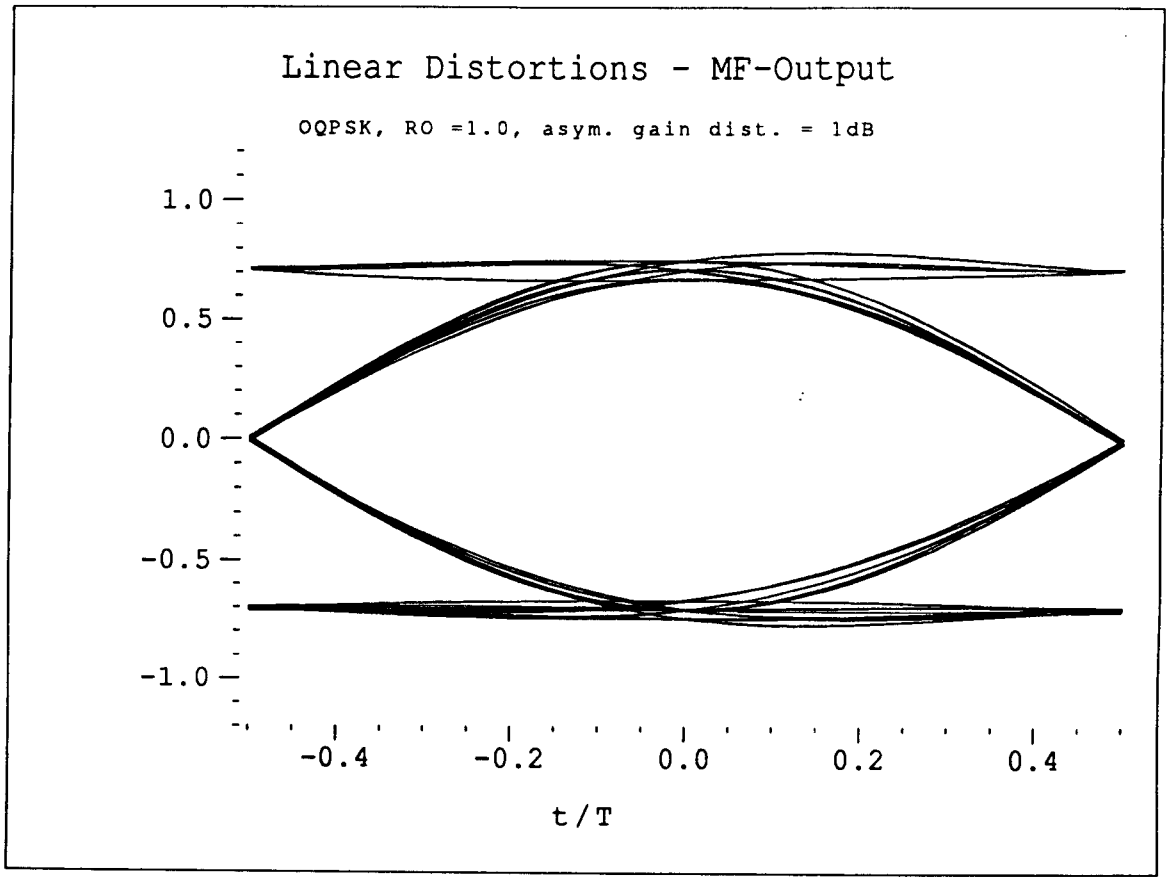


Fig. VII.1-7a: Matched filter output eye diagram (OQPSK), asymmetric gain distortion, gain in dB ~ f, 1dB over 1/T

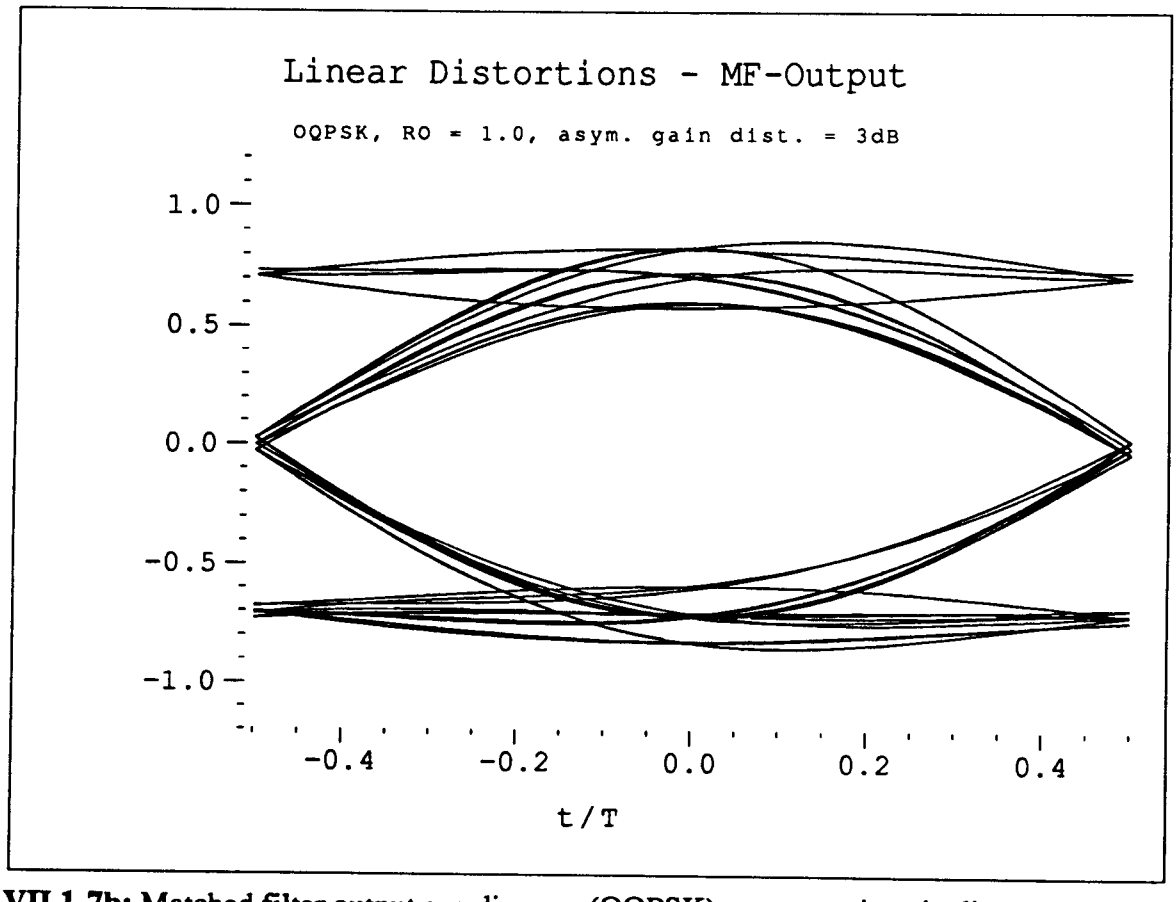


Fig. VII.1-7b: Matched filter output eye diagram (OQPSK), asymmetric gain distortion, gain in dB ~ f, 3dB over 1/T

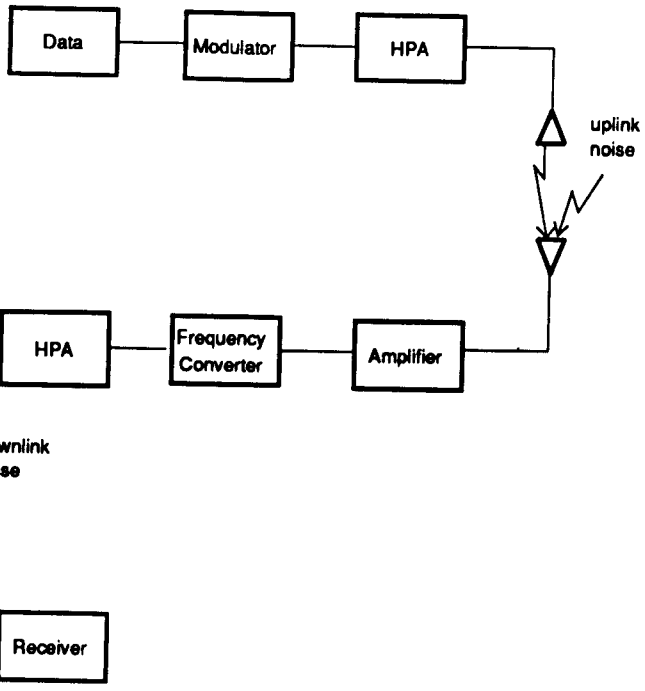


Fig. VII.1-8: Basic elements of a satellite link

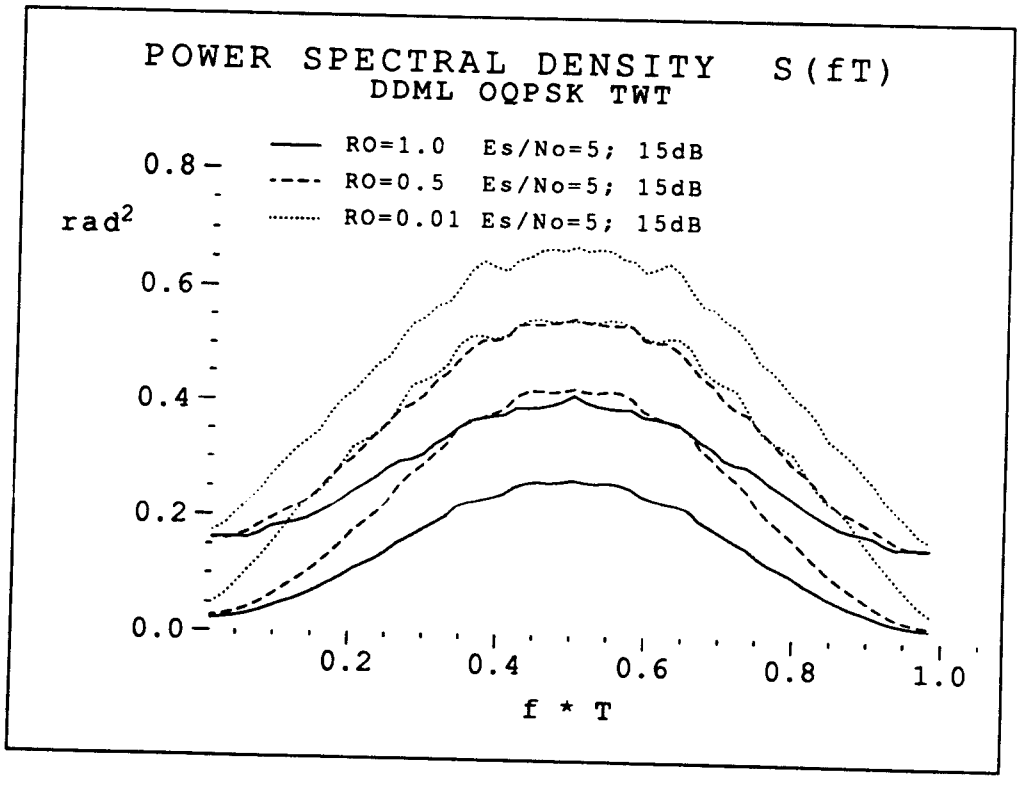


Fig. VII.1-9: Carrier: noise power spectral density of DDML for OQPSK with TWT

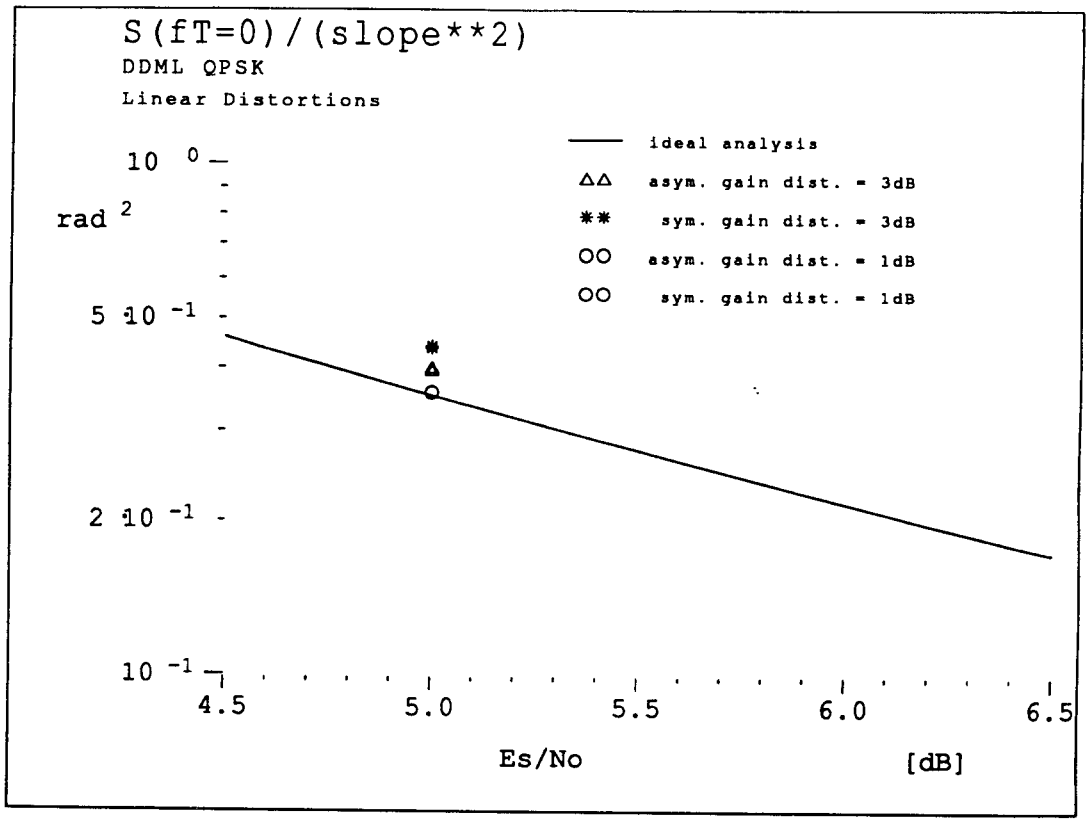


Fig. VII.2-1a: Carrier: DDML normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\theta\theta}$) for QPSK with symmetric and asymmetric gain distortions as a function of E_S/N_0

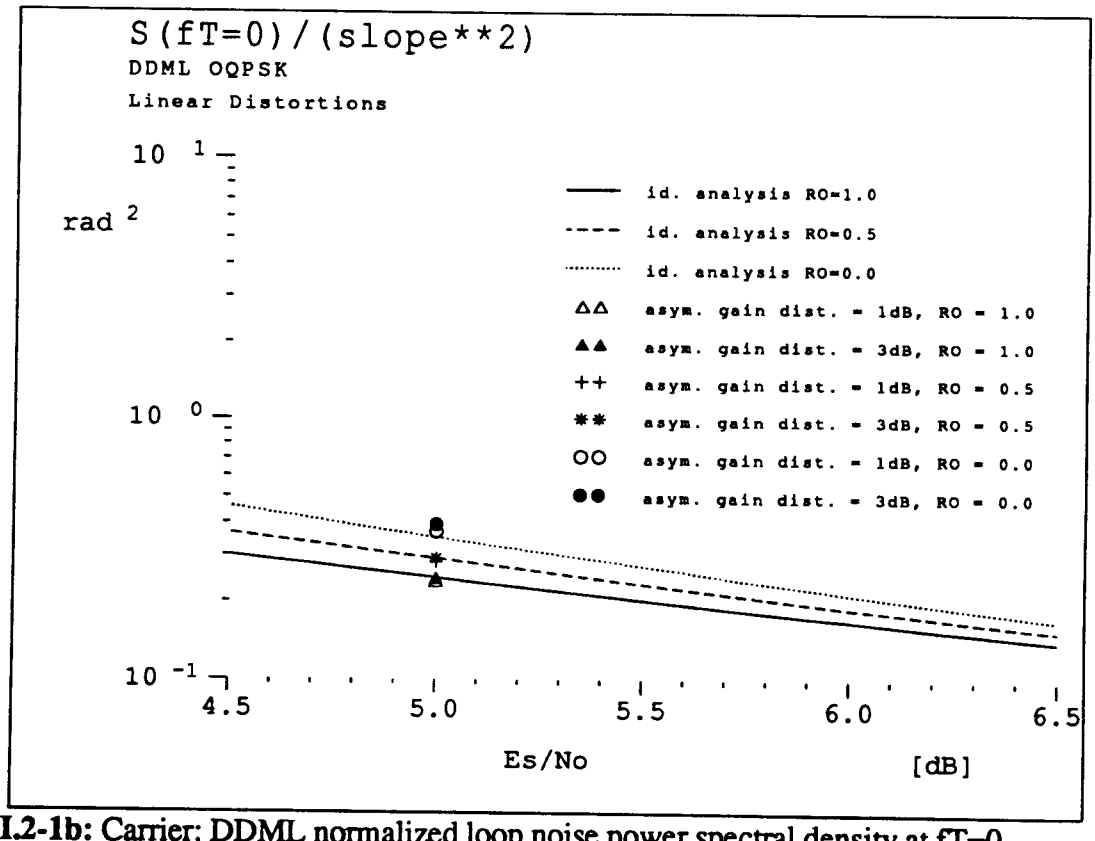


Fig. VII.2-1b: Carrier: DDML normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\theta\theta}$) for OQPSK with asymmetric gain distortion as a function of E_S/N_0

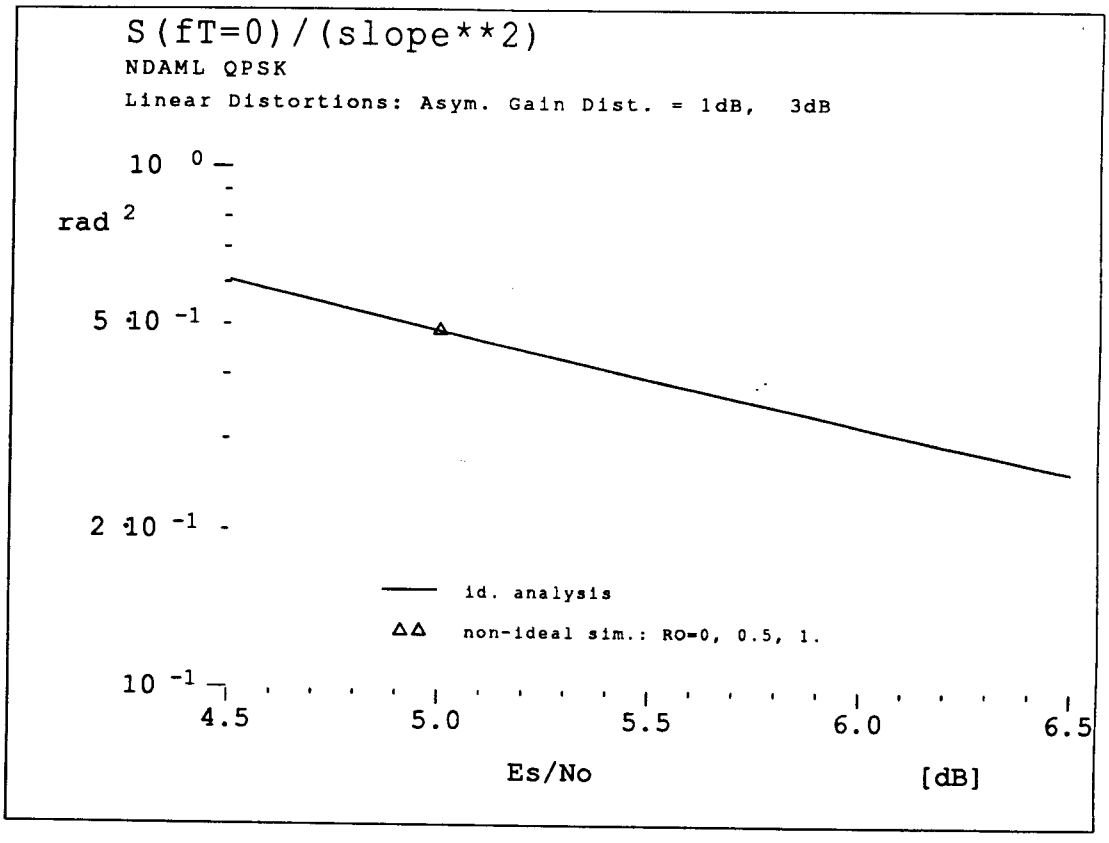


Fig. VII.2-2: Carrier: NDAML normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\theta\theta}$) for QPSK with asymmetric gain distortion as a function of E_S/N_0

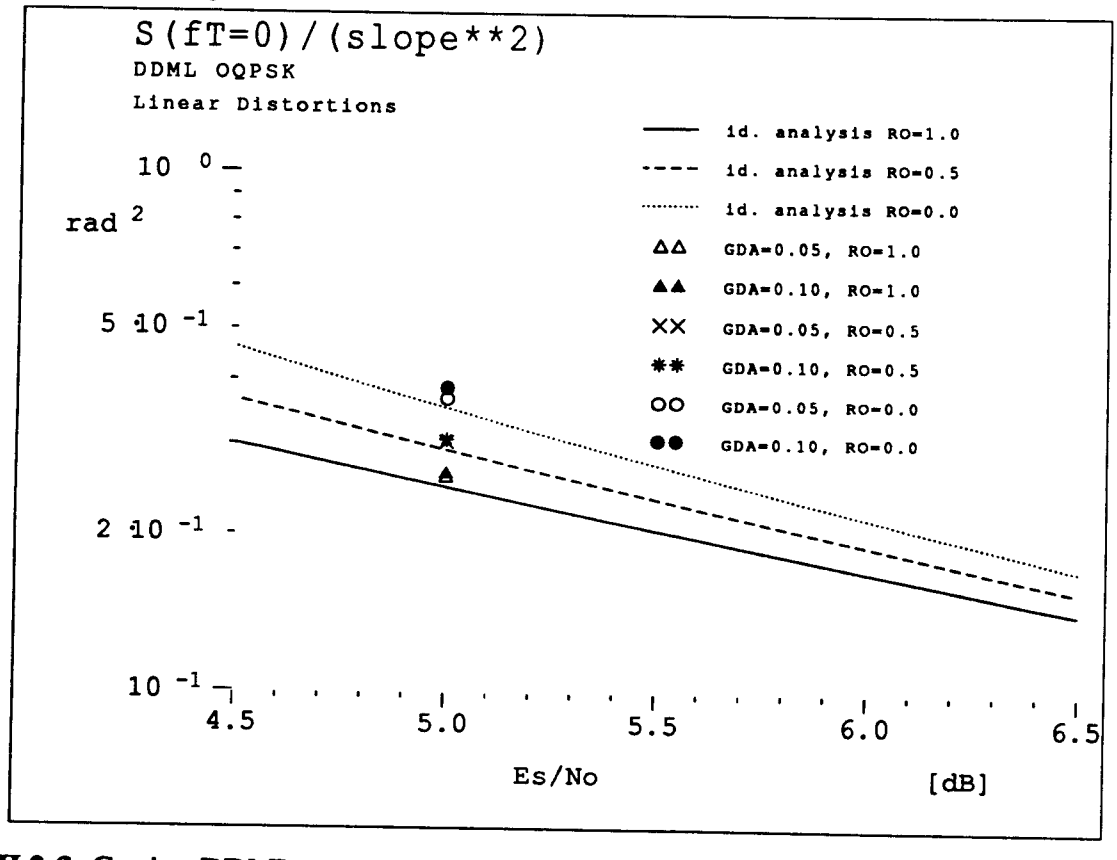


Fig. VII.2-3: Carrier: DDML normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\theta\theta}$) for OQPSK with symmetric group delay distortion as a function of E_S/N_0

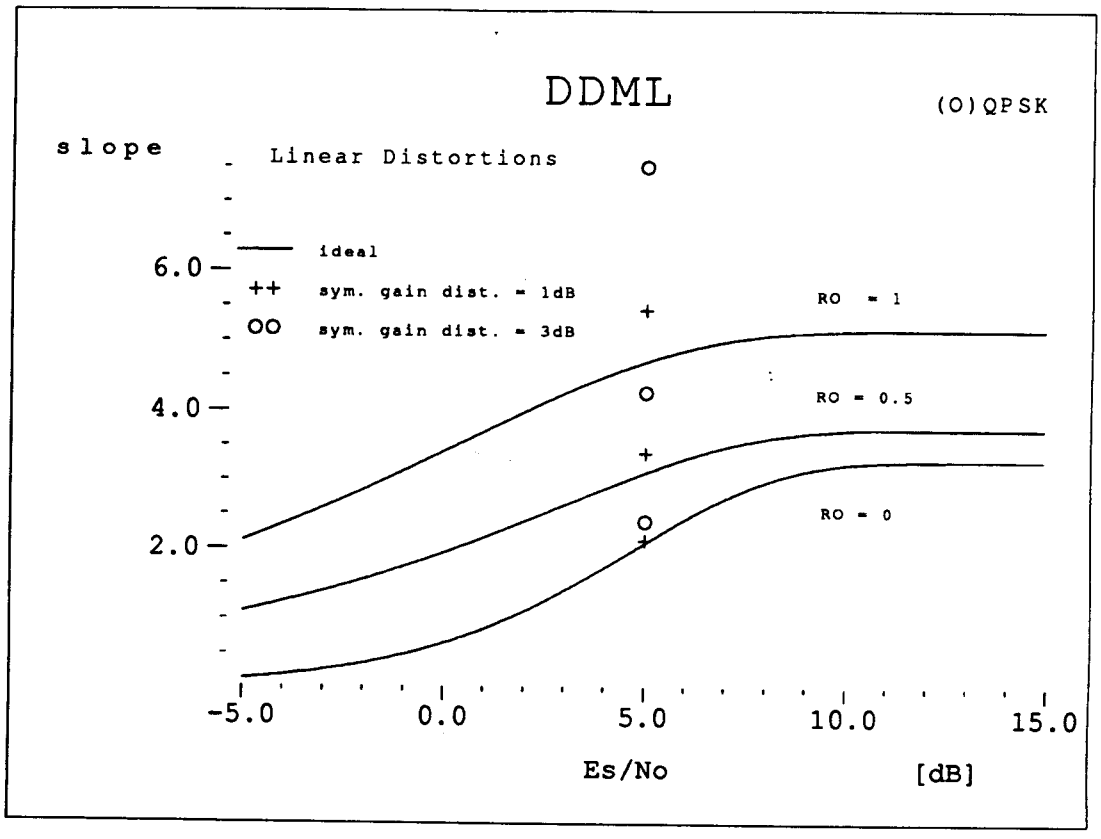


Fig. VII.3-1: DDMLFB timing error detector slopes $K_{\tau\tau}$ for (O)QPSK with symmetric gain distortion as a function of E_S/N_0

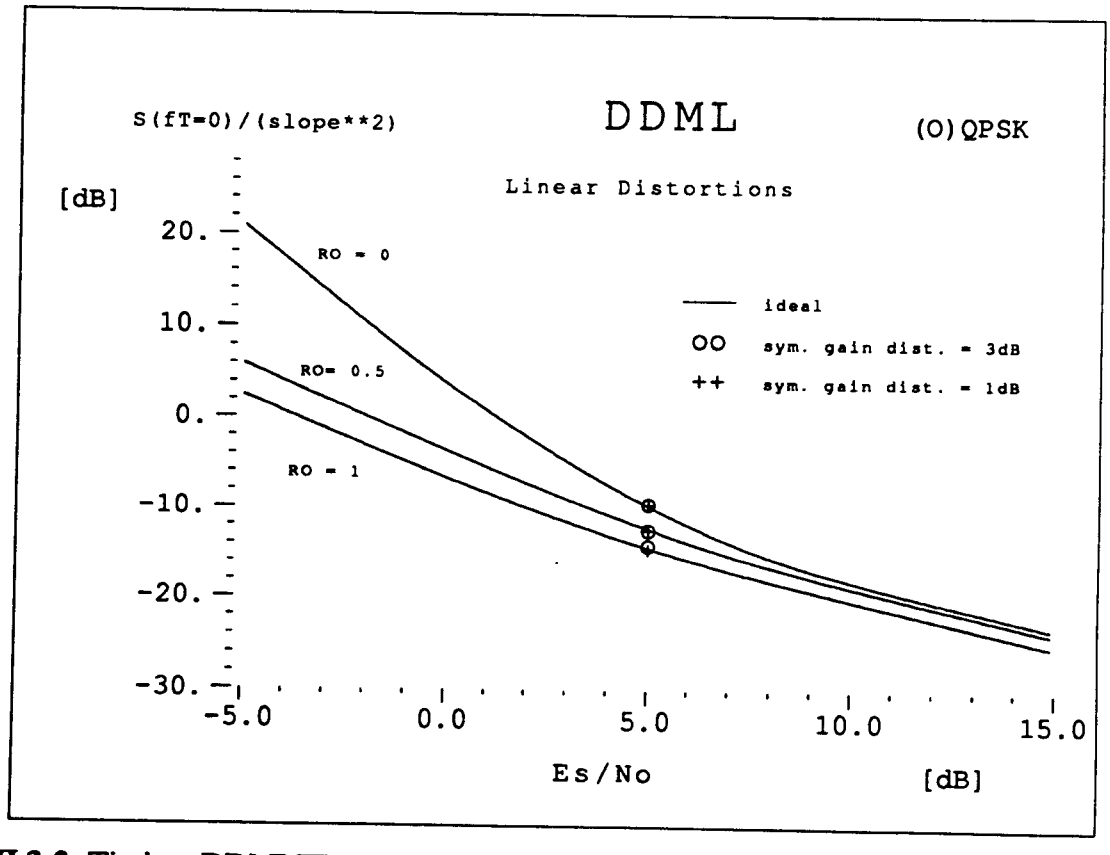


Fig. VII.3-2: Timing: DDMLFB normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\tau\tau}$) for O(Q)PSK with symmetric gain distortion as a function of E_S/N_0

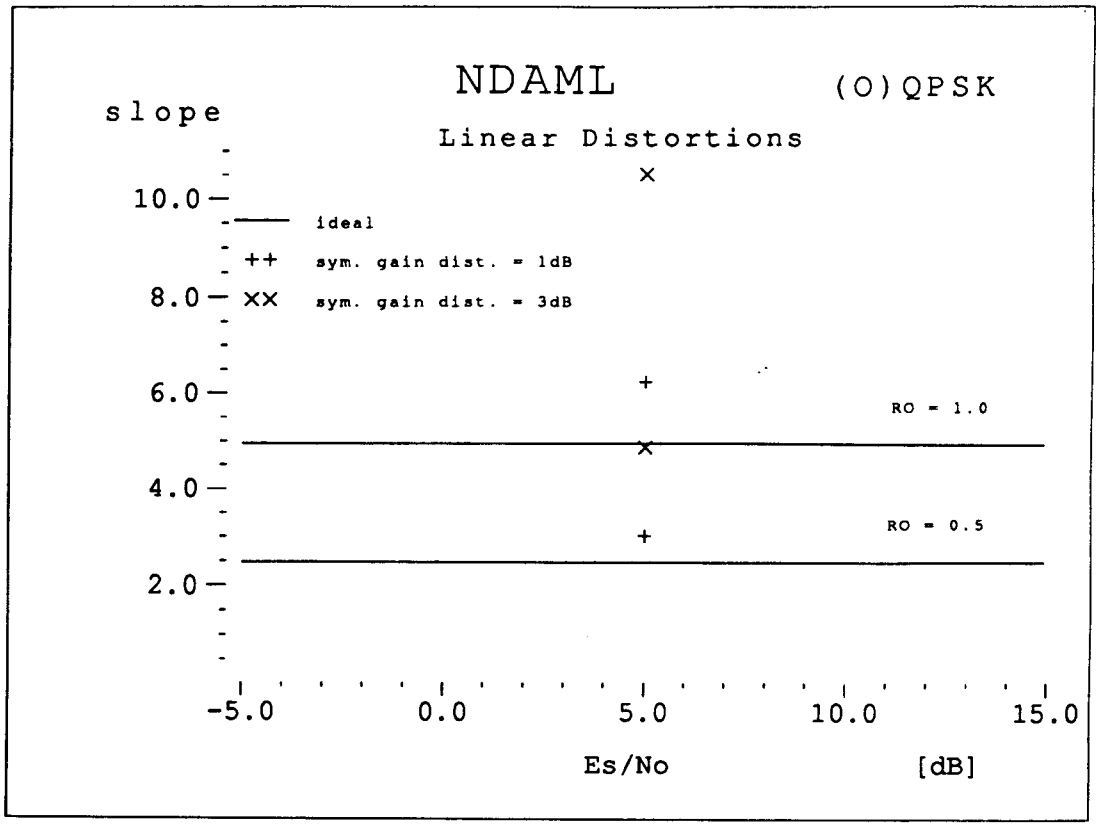


Fig. VII.3-3: NDAMLFB timing error detector slopes $K_{\tau\tau}$ for (O)QPSK with symmetric gain distortion as a function of E_S/N_0

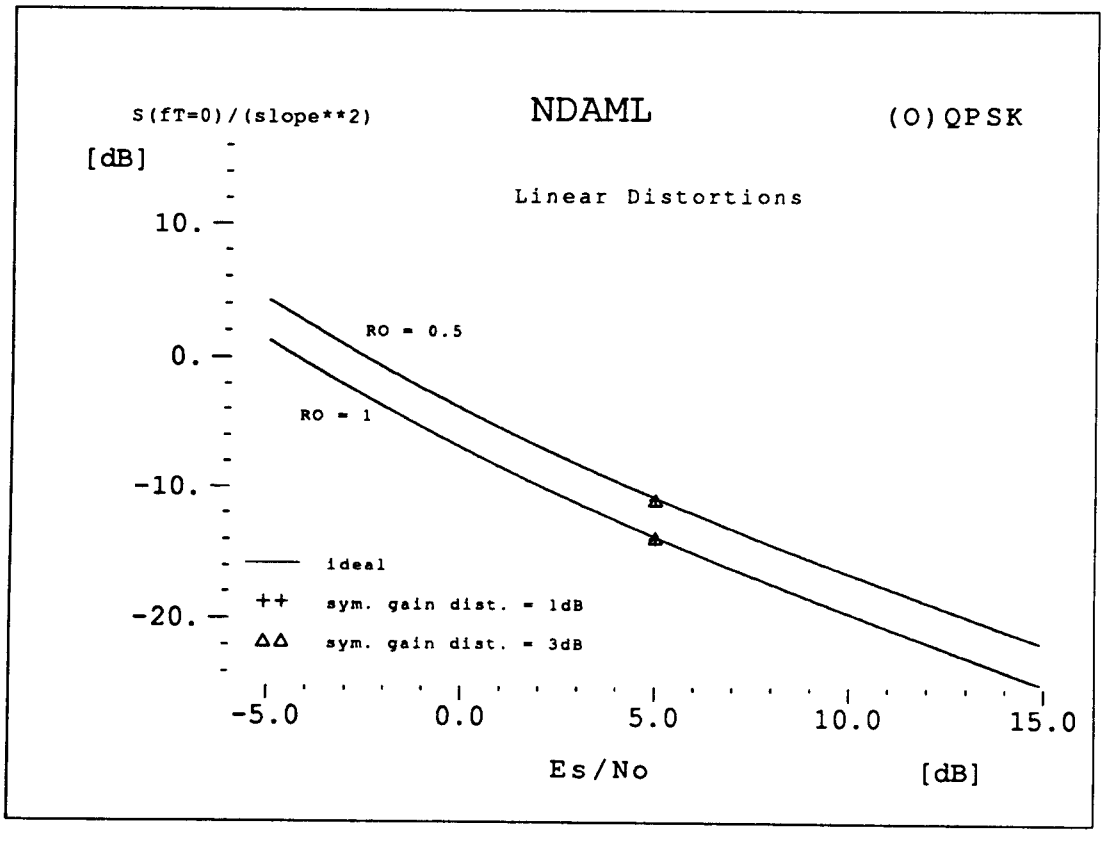


Fig. VII.3-4: Timing: NDAMLFB normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\tau\tau}$) for O(Q)PSK with symmetric gain distortion as a function of E_S/N_0

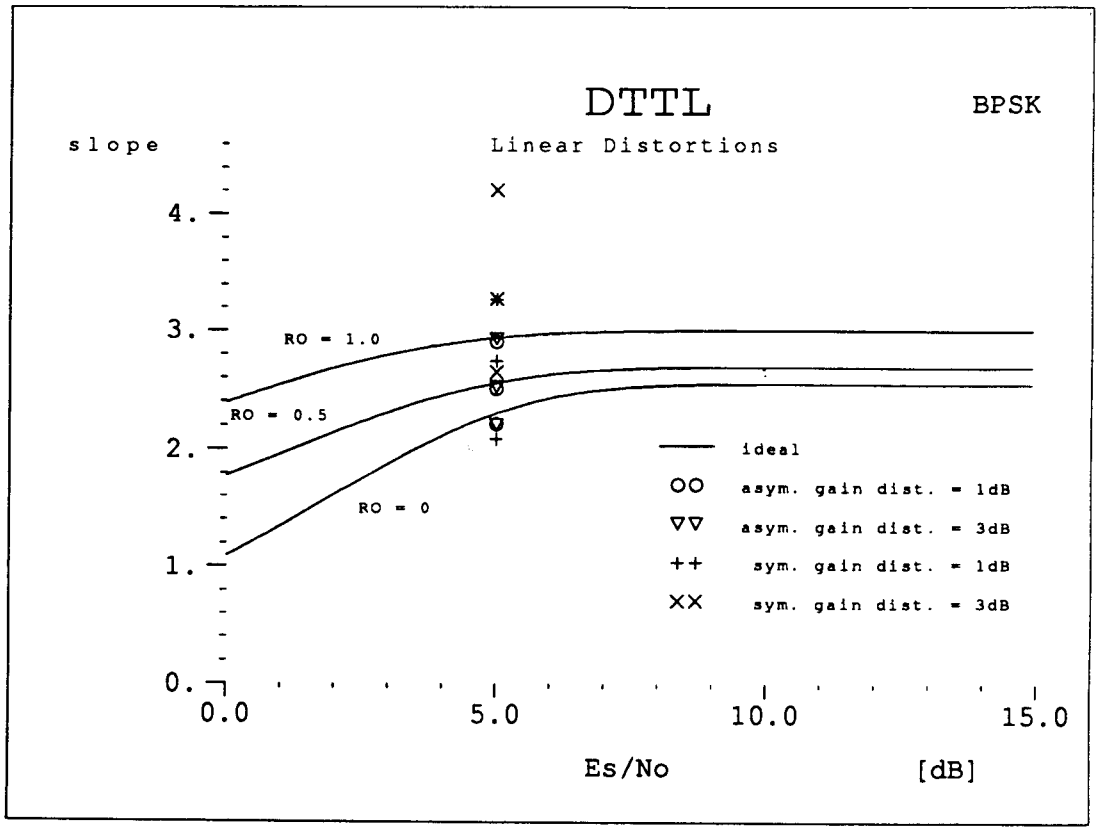


Fig. VII.3-5a: DTTL timing error detector slopes $K_{\tau\tau}$ for BPSK with symmetric and asymmetric gain distortion as a function of E_S/N_0

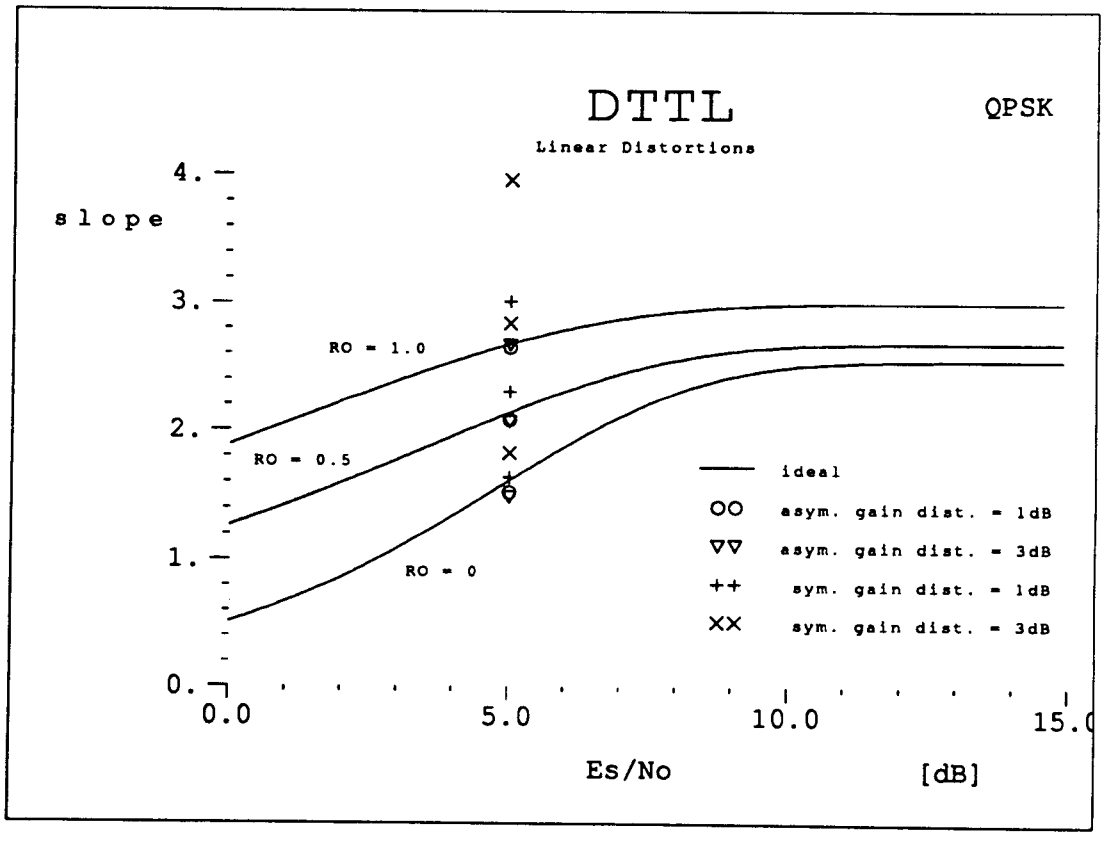


Fig. VII.3-5b: DTTL timing error detector slopes $K_{\tau\tau}$ for QPSK with symmetric and asymmetric gain distortion as a function of E_S/N_0

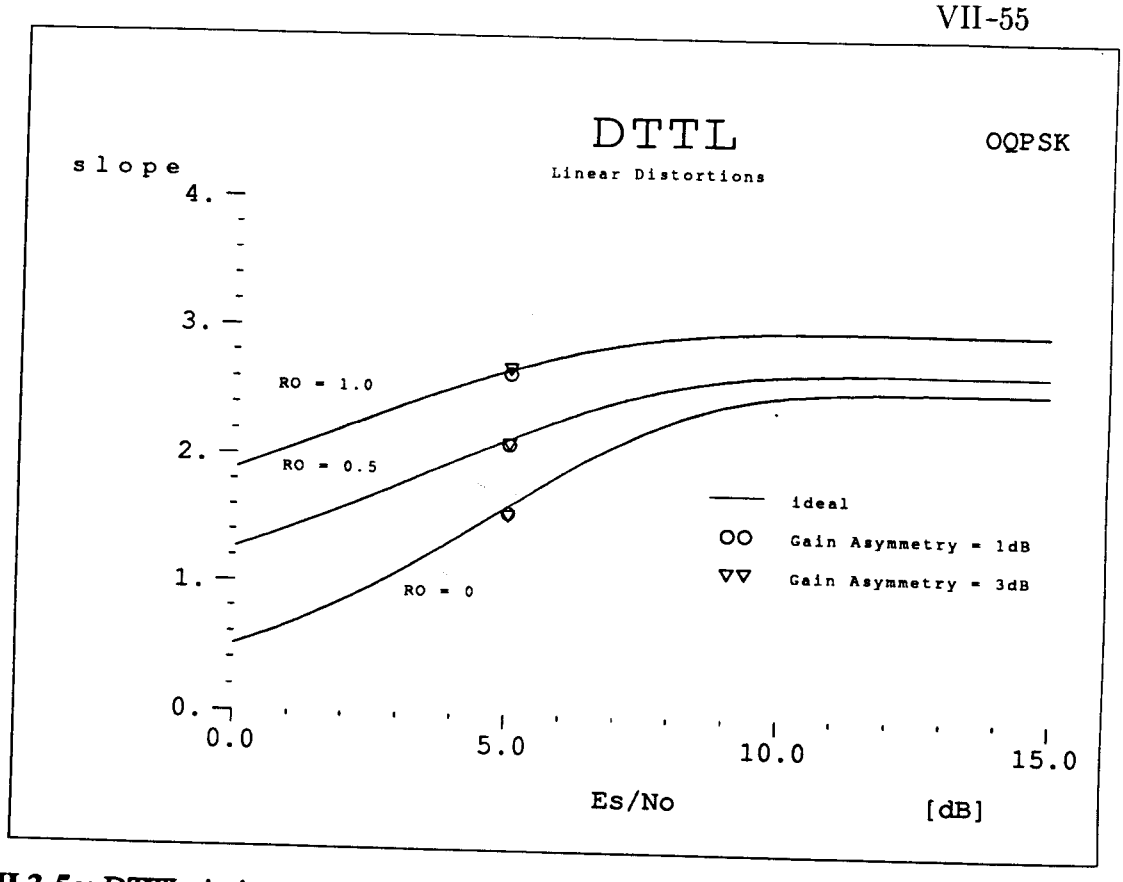


Fig. VII.3-5c: DTTL timing error detector slopes $K_{\tau\tau}$ for OQPSK with asymmetric gain distortion as a function of E_S/N_0

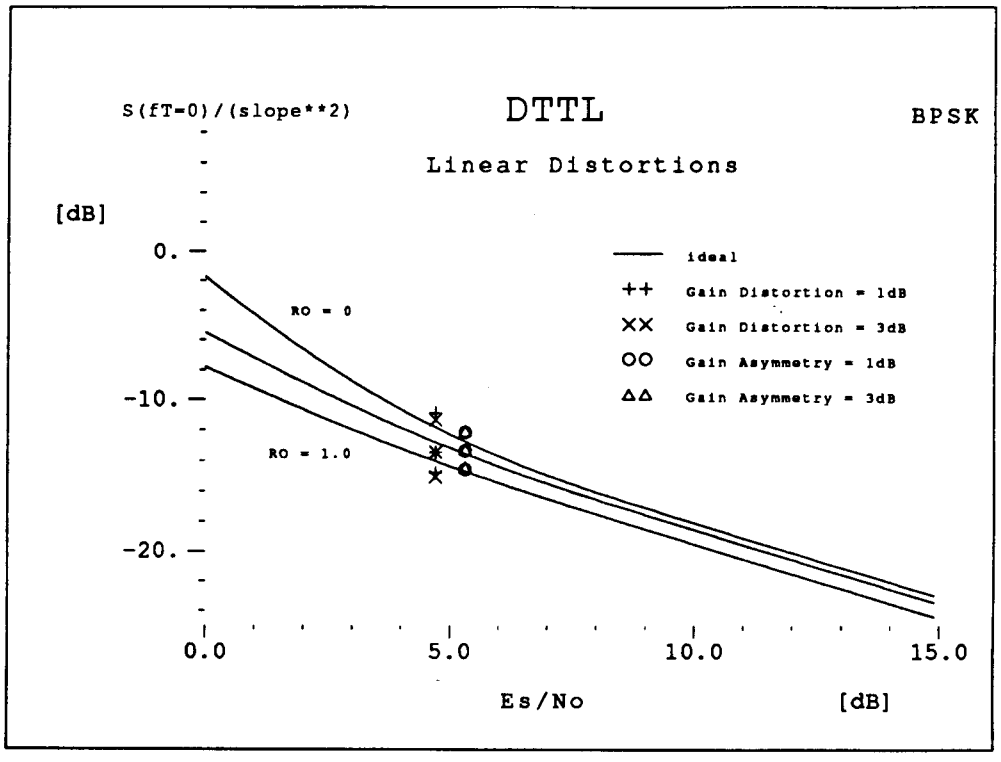


Fig. VII.3-6a: Timing: DTTL normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\tau\tau}$) for BPSK with symmetric and asymmetric gain distortion as a function of E_S/N_0 (simulation results are for E_S/N_0 = 5dB but shown shifted left and right for better readibility).

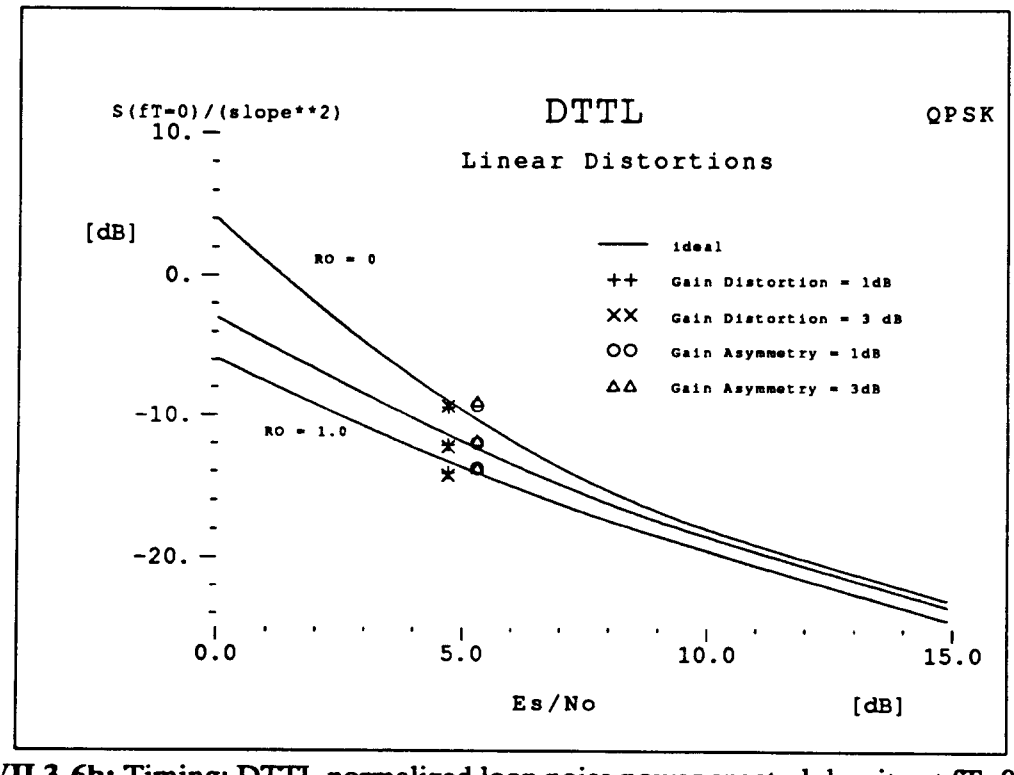


Fig. VII.3-6b: Timing: DTTL normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\tau\tau}$) for QPSK with symmetric and asymmetric gain distortion as a function of E_S/N_0 (simulation results are for E_S/N_0 = 5dB but shown shifted left and right for better readibility)

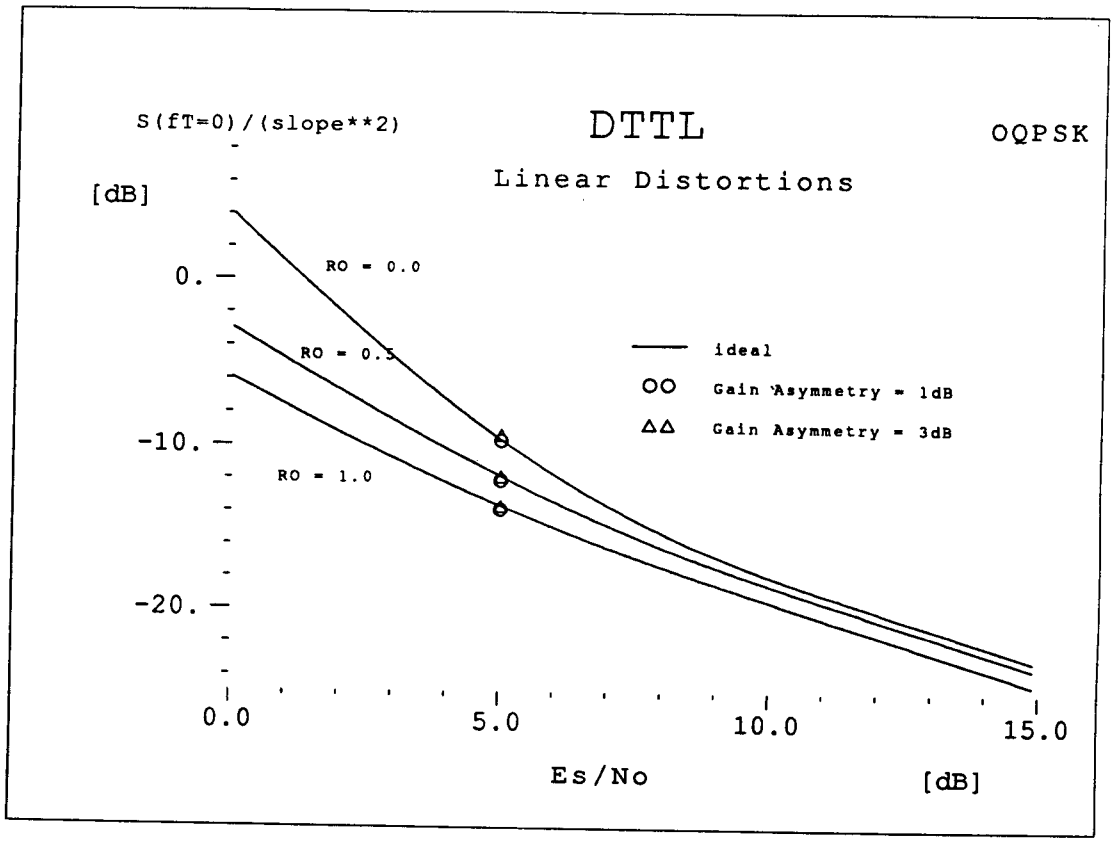


Fig. VII.3-6c: Timing: DTTL normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2\tau\tau$) for OQPSK with asymmetric gain distortion as a function of E_S/N_0

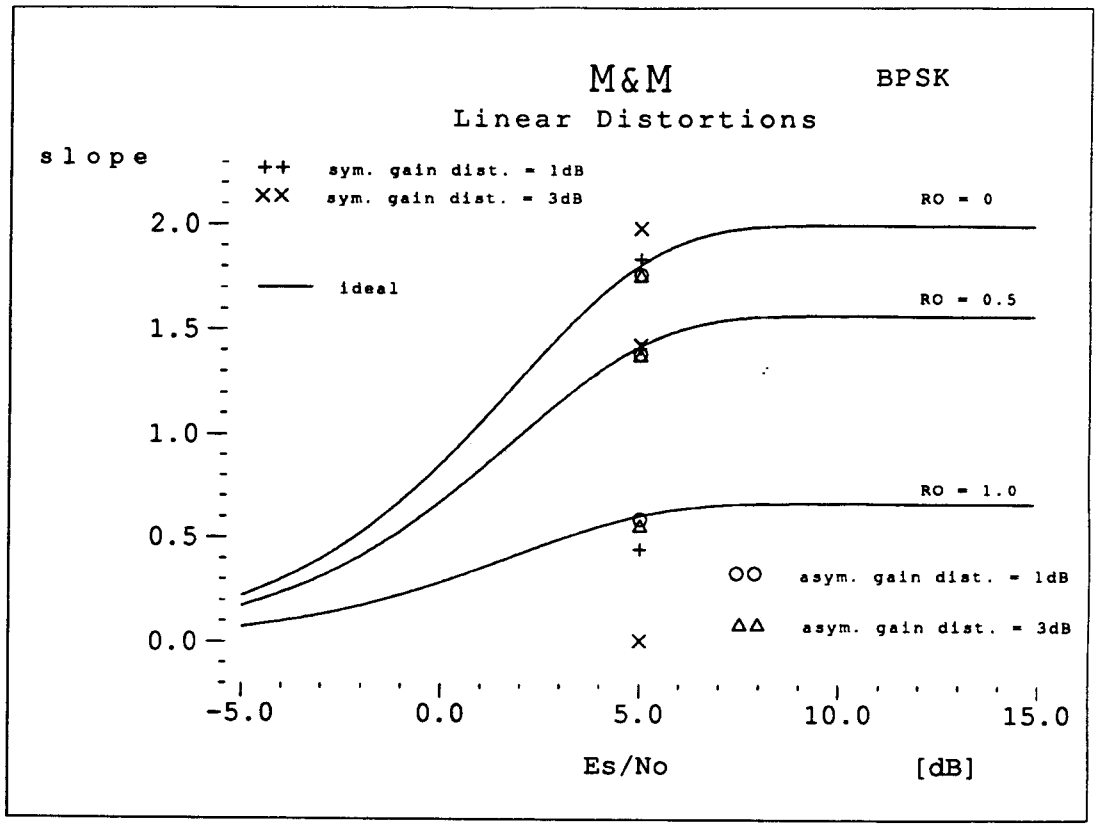


Fig. VII.3-7a: M&M timing error detector slopes $K_{\tau\tau}$ for BPSK with symmetric and asymmetric gain distortion as a function of E_S/N_0

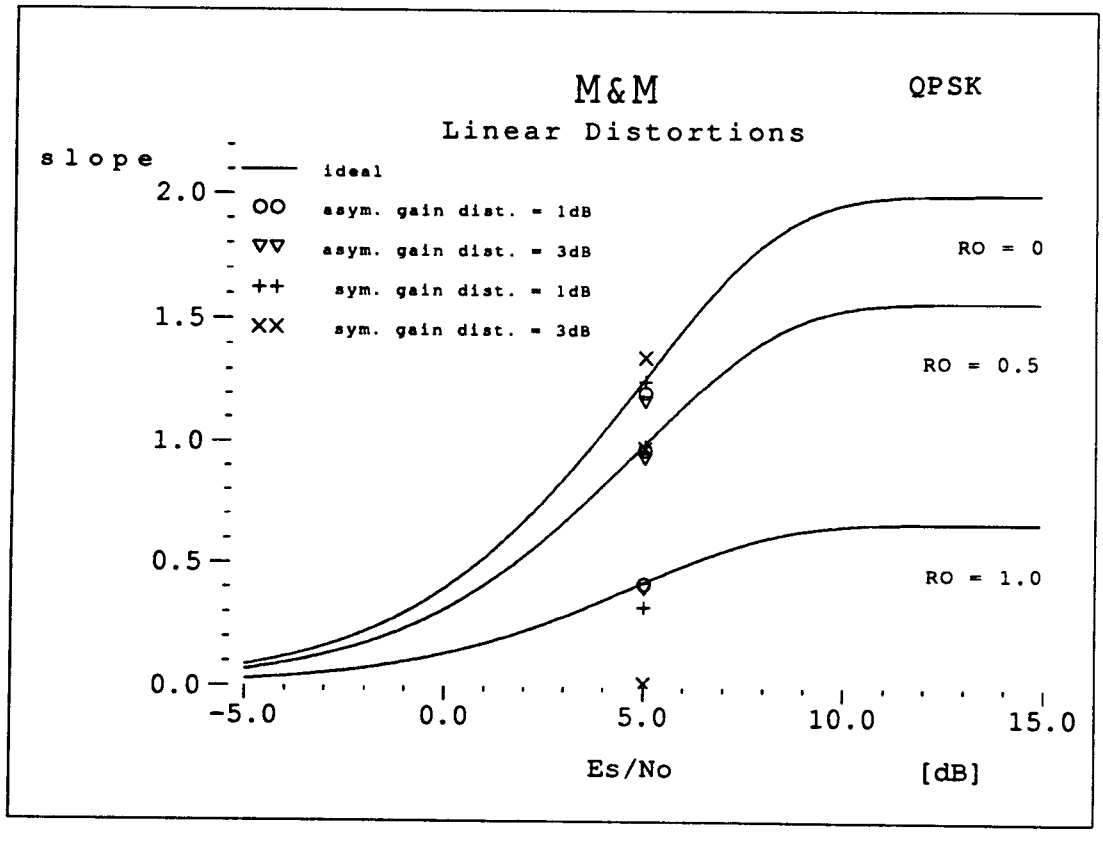


Fig. VII.3-7b: M&M timing error detector slopes $K_{\tau\tau}$ for QPSK with symmetric and asymmetric gain distortion as a function of E_S/N_0

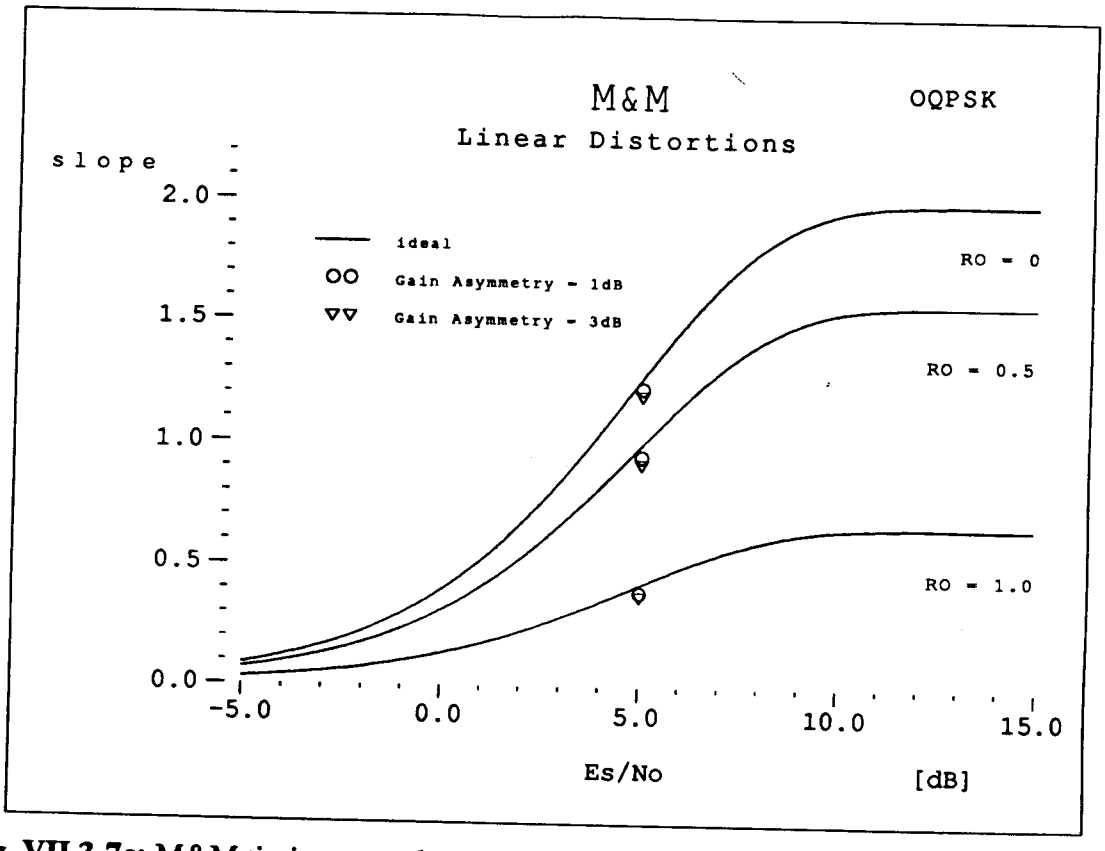


Fig. VII.3-7c: M&M timing error detector slopes $K_{\tau\tau}$ for OQPSK with asymmetric gain distortion as a function of E_S/N_0

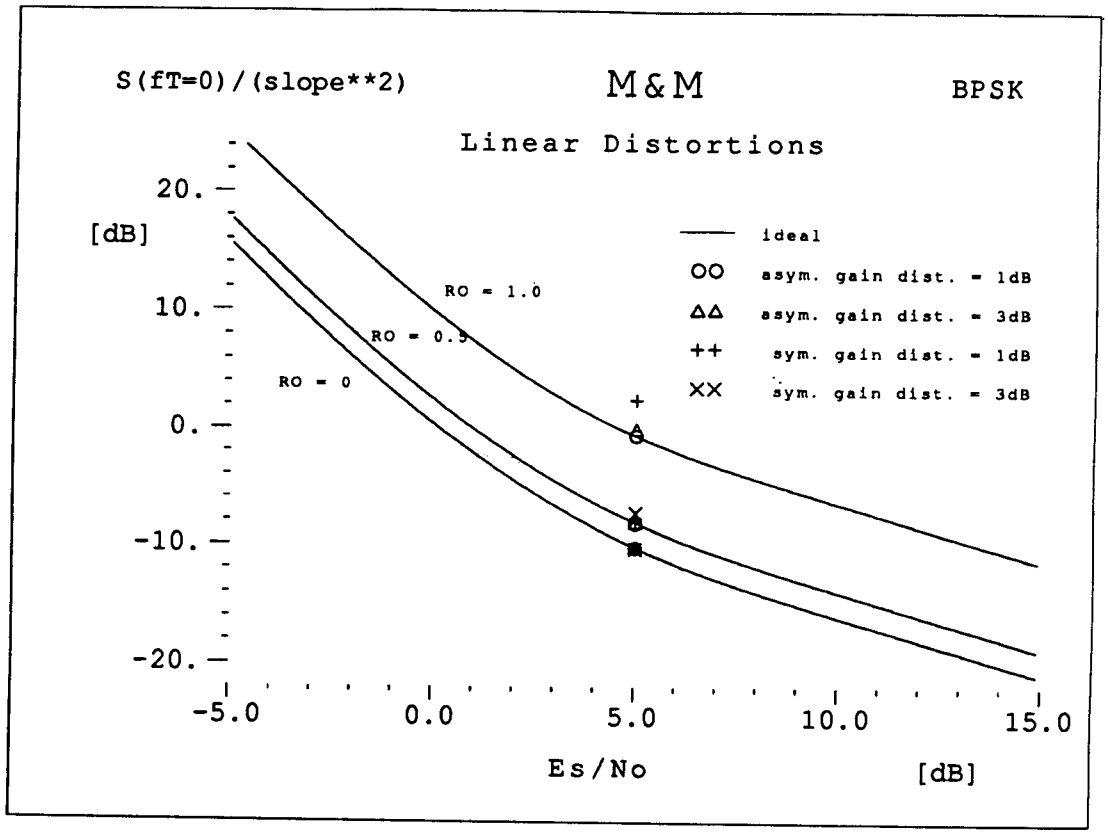


Fig. VII.3-8a: Timing: M&M normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\tau\tau}$) for BPSK with symmetric and asymmetric gain distortion as a function of E_S/N_0

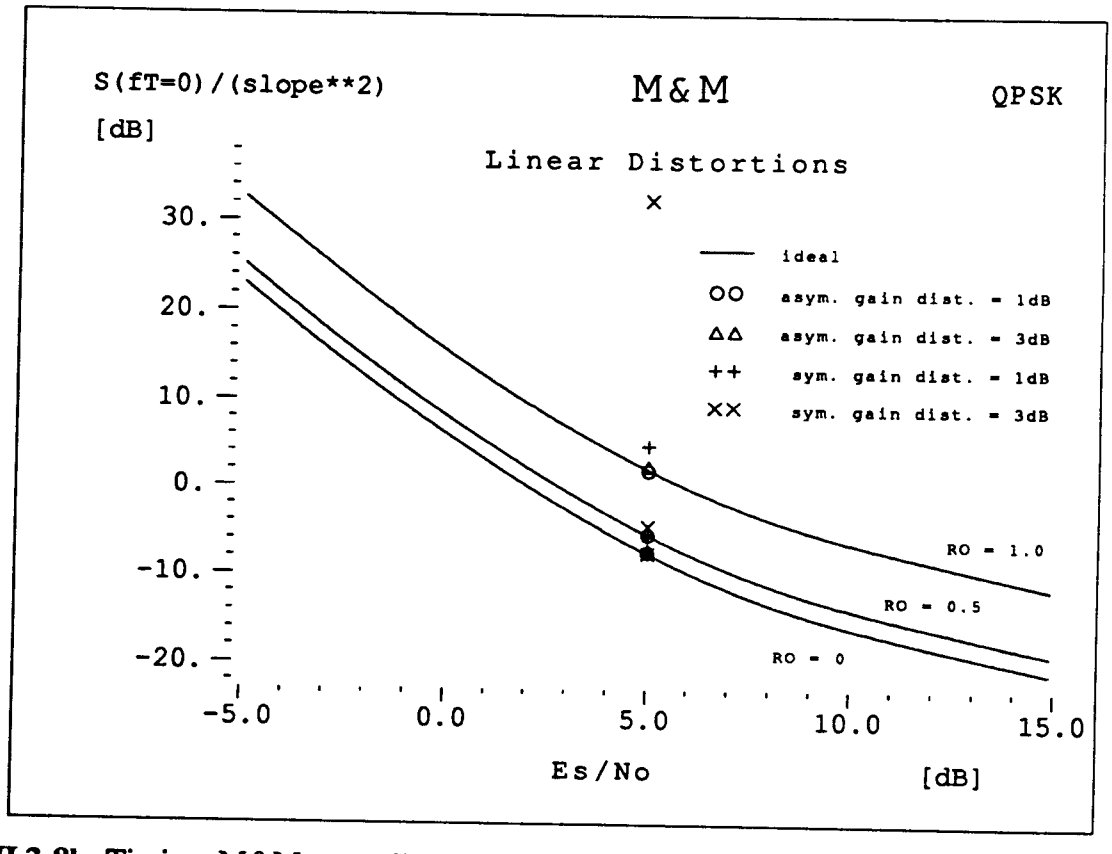


Fig. VII.3-8b: Timing: M&M normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\tau\tau}$) for QPSK with symmetric and asymmetric gain distortion as a function of E_S/N_0

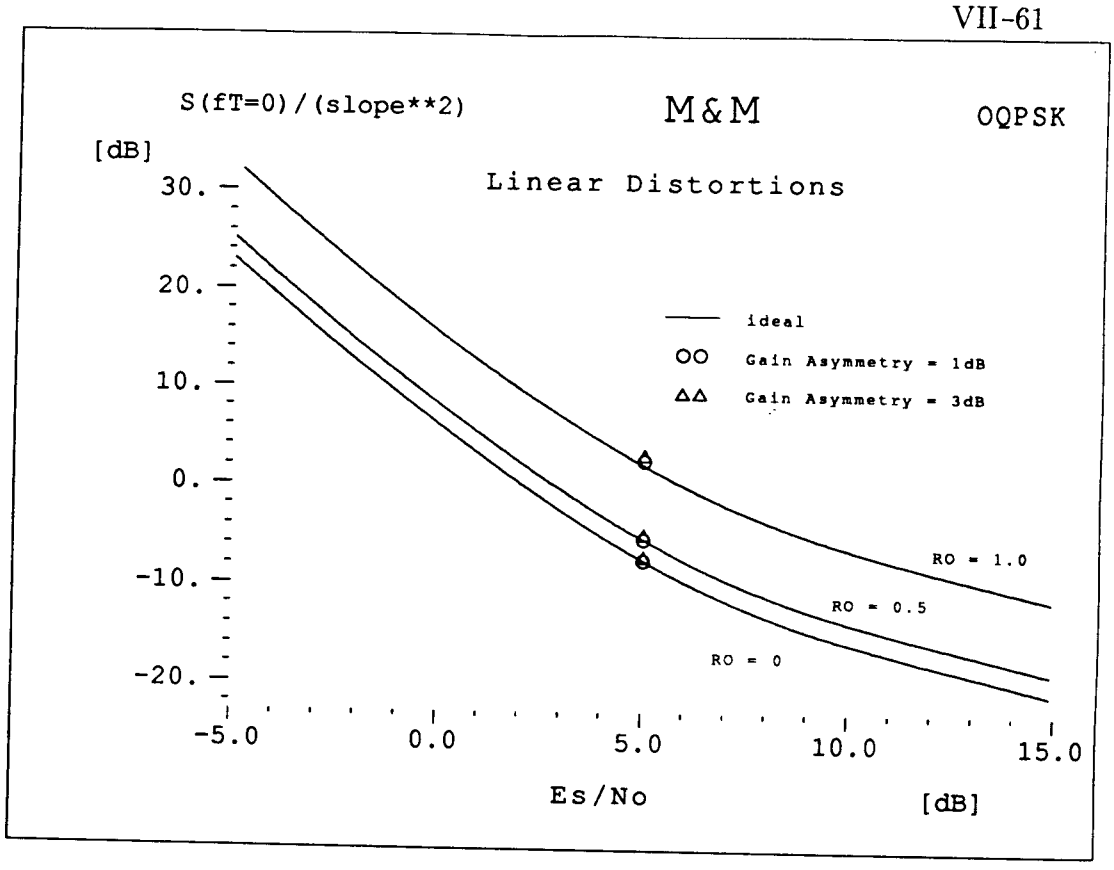


Fig. VII.3-8c: Timing: M&M normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\tau\tau}$) for OQPSK with asymmetric gain distortion as a function of E_S/N_0

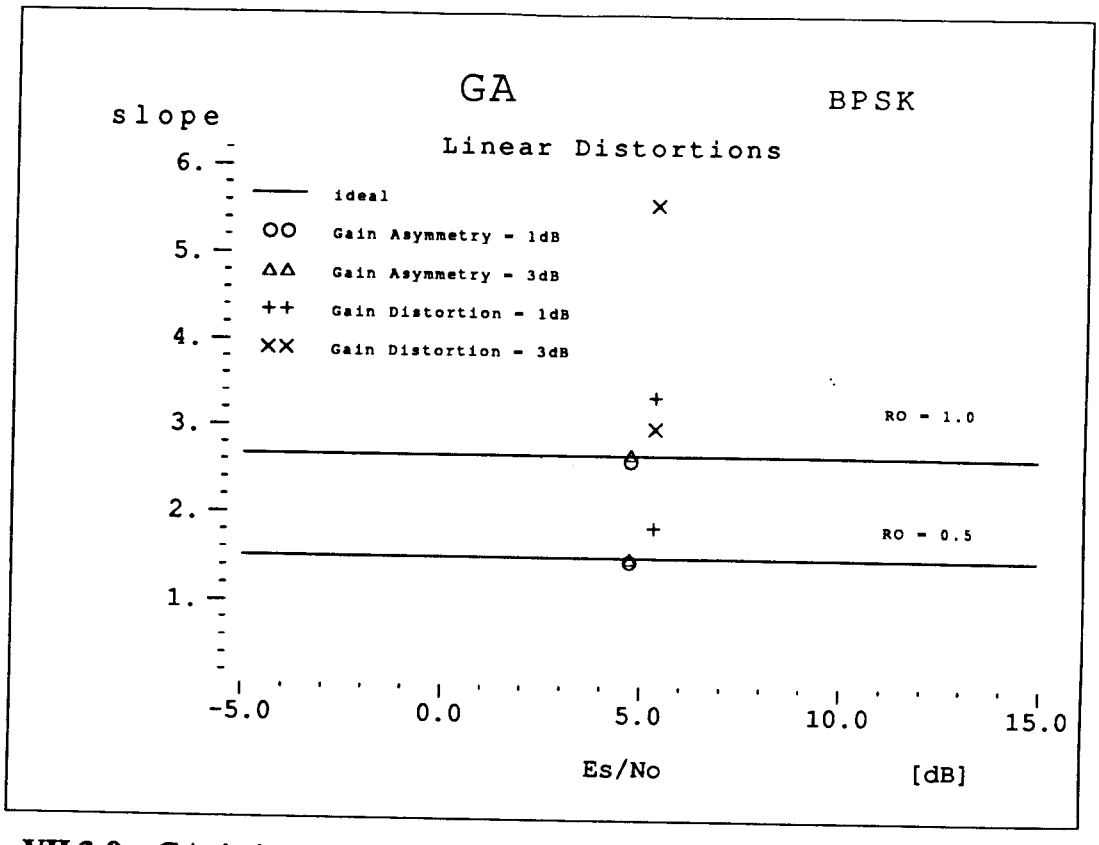


Fig. VII.3-9a: GA timing error detector slopes $K_{\tau\tau}$ for BPSK with symmetric and asymmetric gain distortion as a function of E_S/N_0 (simulation results are for E_S/N_0 = 5dB but shown shifted left and right for better readibility).

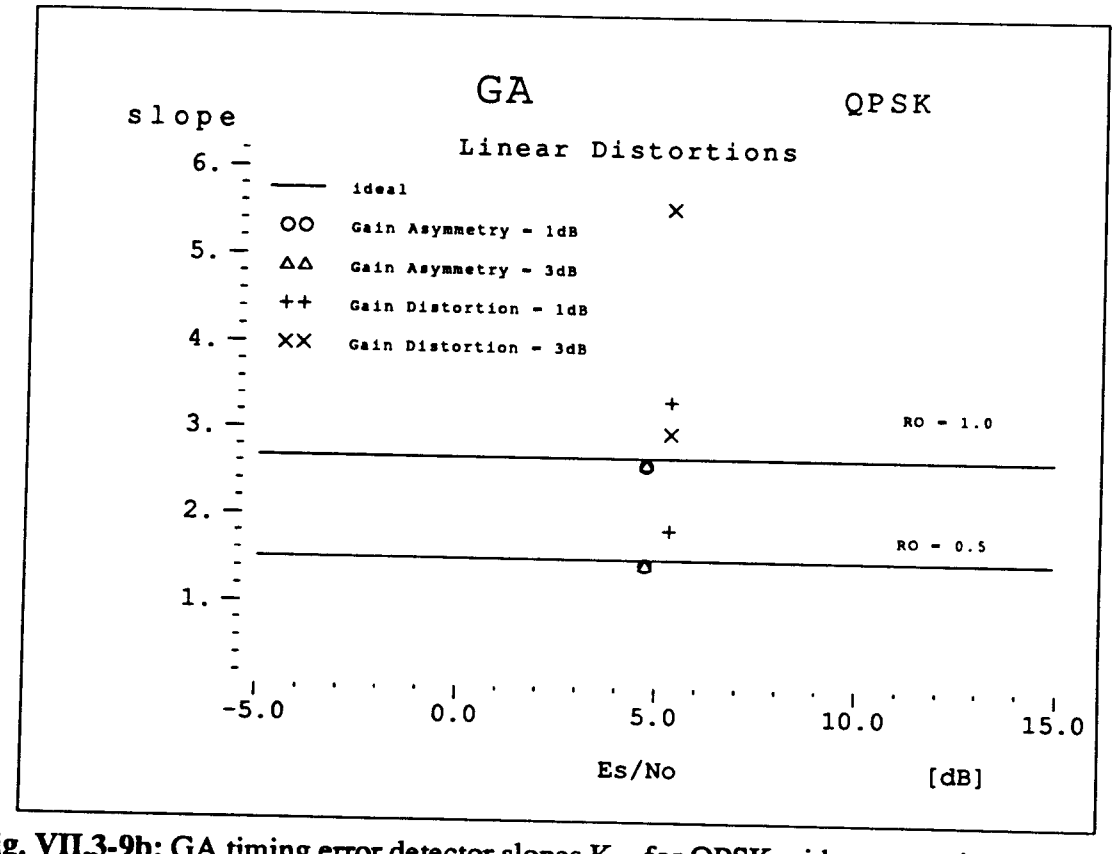


Fig. VII.3-9b: GA timing error detector slopes $K_{\tau\tau}$ for QPSK with symmetric and asymmetric gain distortion as a function of E_S/N_0 (simulation results are for E_S/N_0 = 5dB but shown shifted left and right for better readibility).

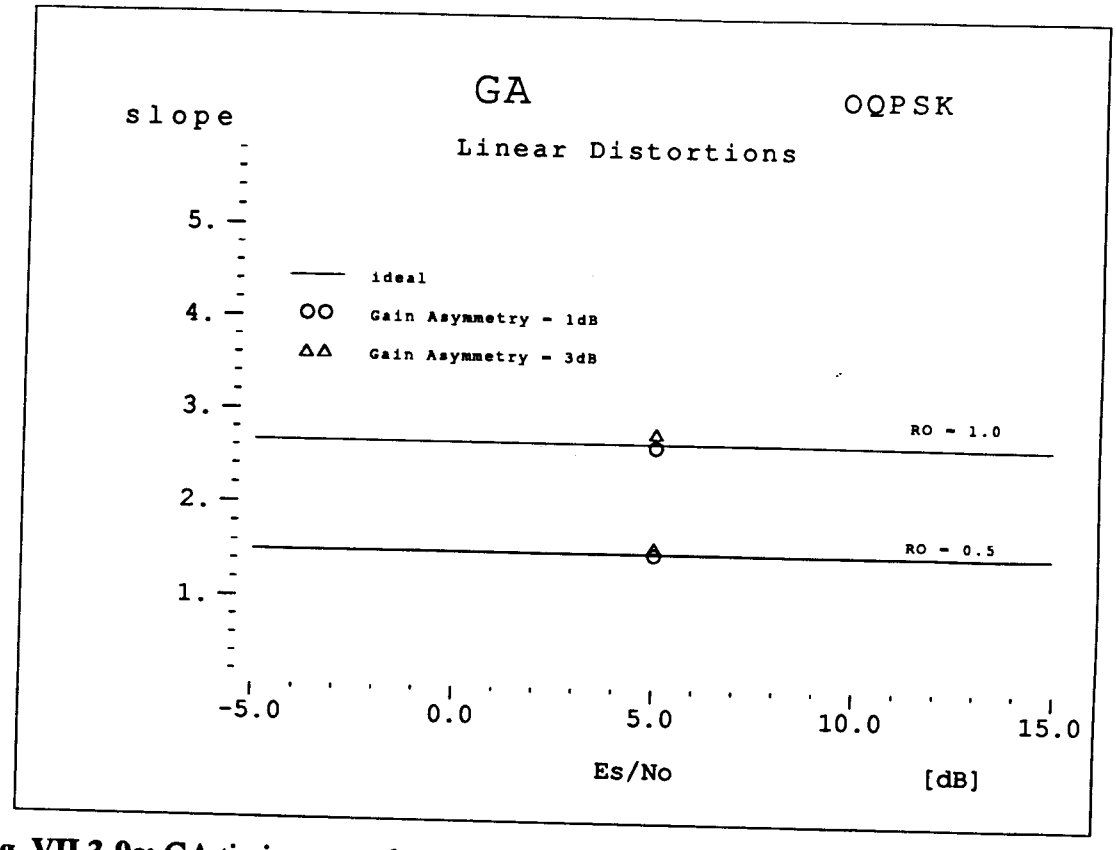


Fig. VII.3-9c: GA timing error detector slopes $K_{\tau\tau}$ for OQPSK with asymmetric gain distortion as a function of E_S/N_0

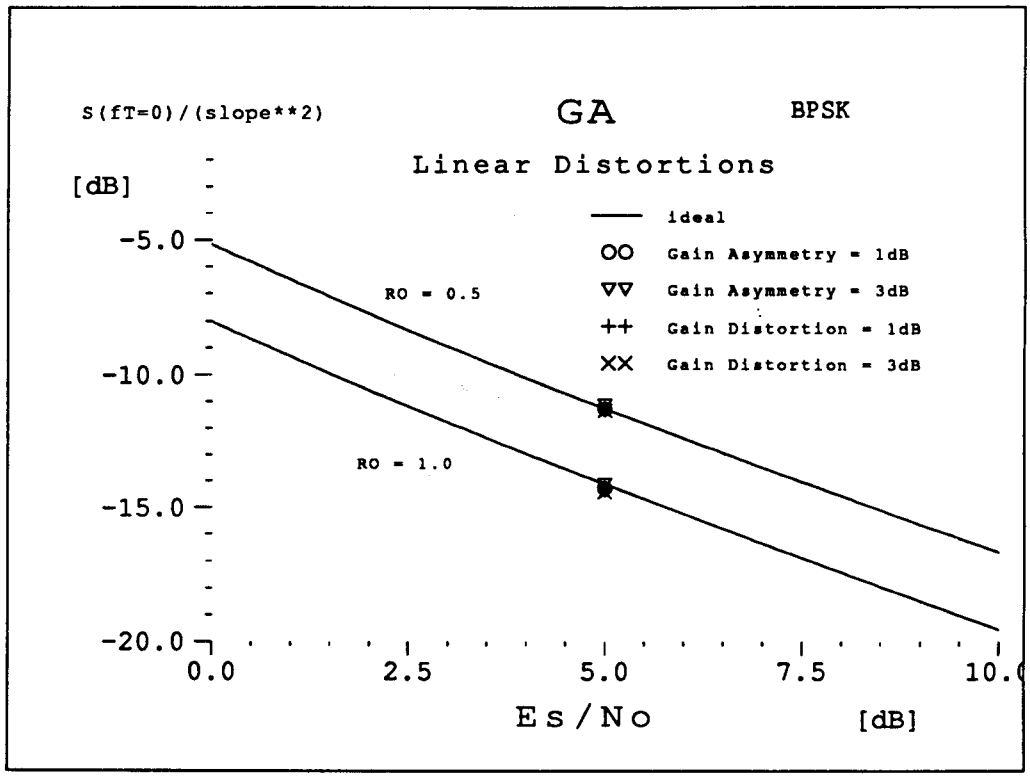


Fig. VII.3-10a: Timing: GA normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\tau\tau}$) for BPSK with symmetric and asymmetric gain distortion as a function of E_S/N_0

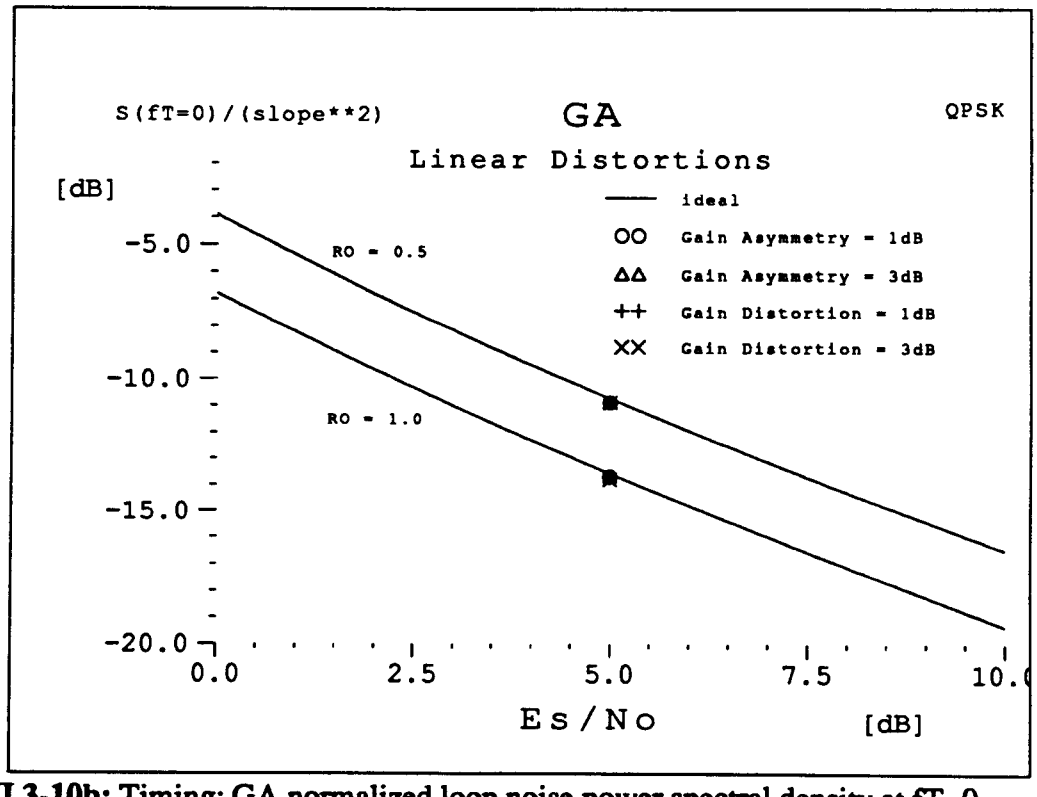


Fig. VII.3-10b: Timing: GA normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\tau\tau}$) for QPSK with symmetric and asymmetric gain distortion as a function of E_S/N_0

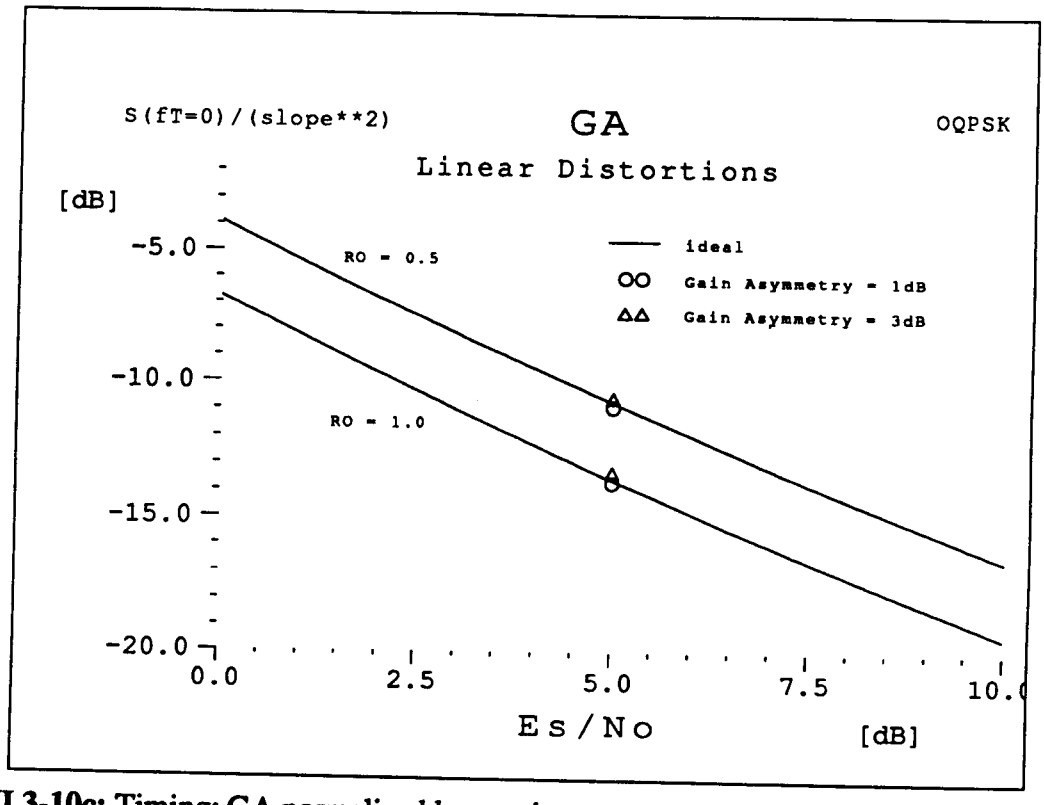


Fig. VII.3-10c: Timing: GA normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\tau\tau}$) for OQPSK with asymmetric gain distortion as a function of E_S/N_0

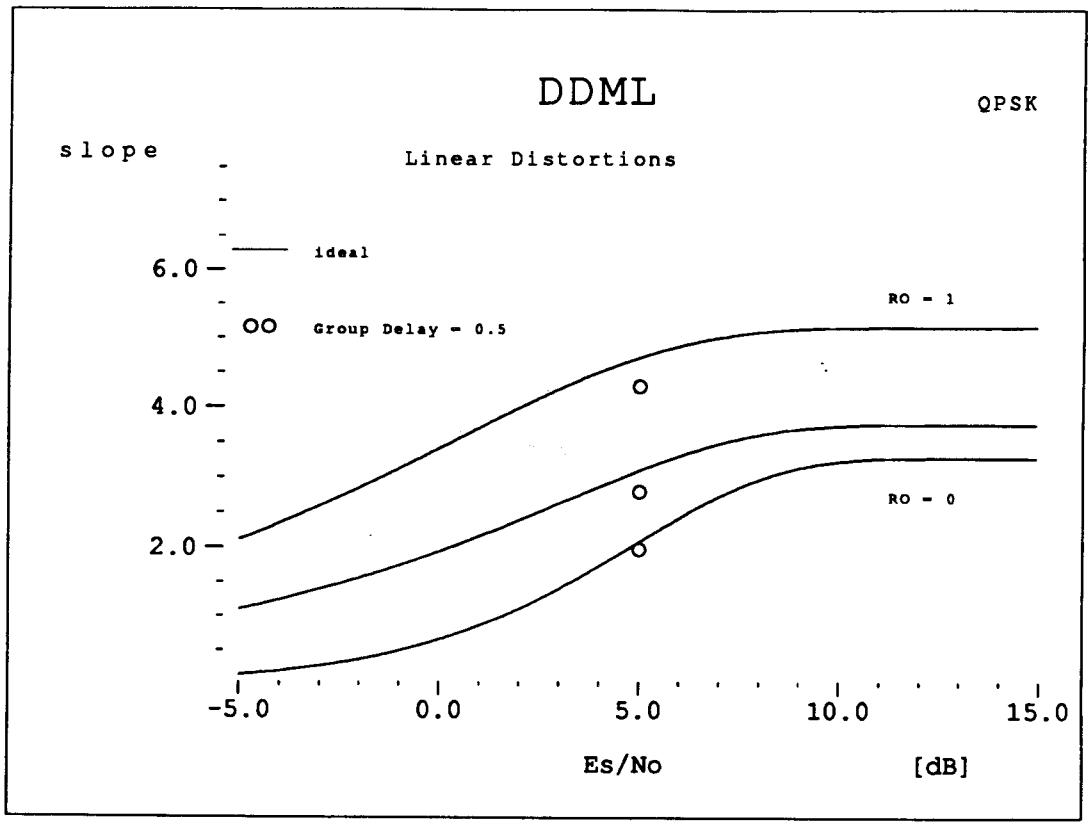


Fig. VII.3-11a: DDML timing error detector slopes $K_{\tau\tau}$ for QPSK with symmetric group delay distortion as a function of E_S/N_0

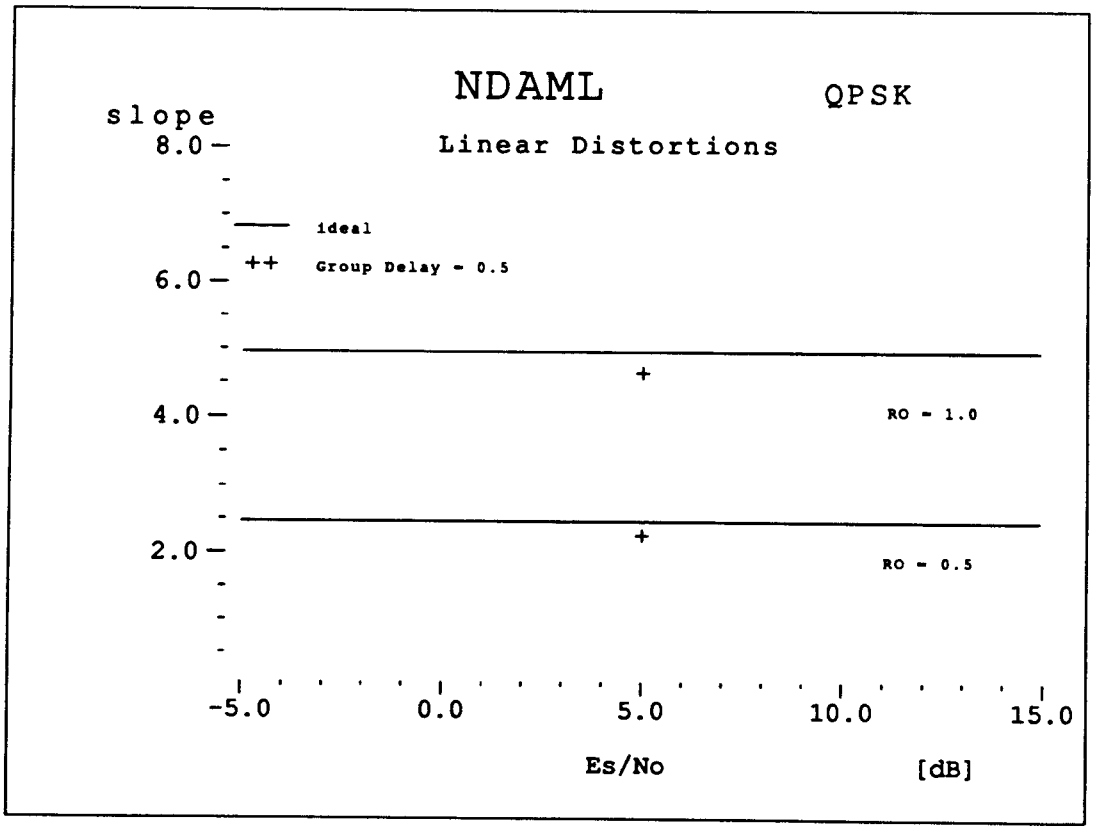


Fig. VII.3-11b: NDAML timing error detector slopes $K_{\tau\tau}$ for QPSK with symmetric group delay distortion as a function of E_S/N_0

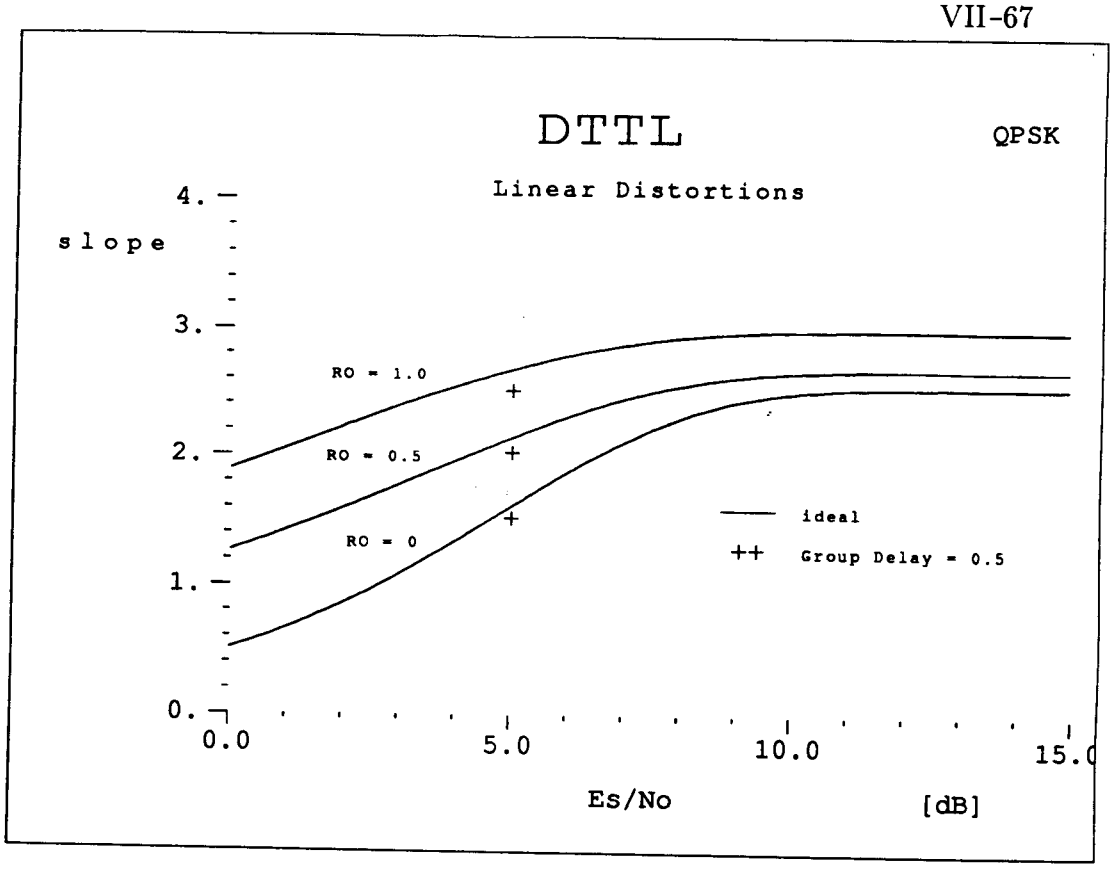


Fig. VII.3-11c: DTTL timing error detector slopes $K_{\tau\tau}$ for QPSK with symmetric group delay distortion as a function of E_S/N_0

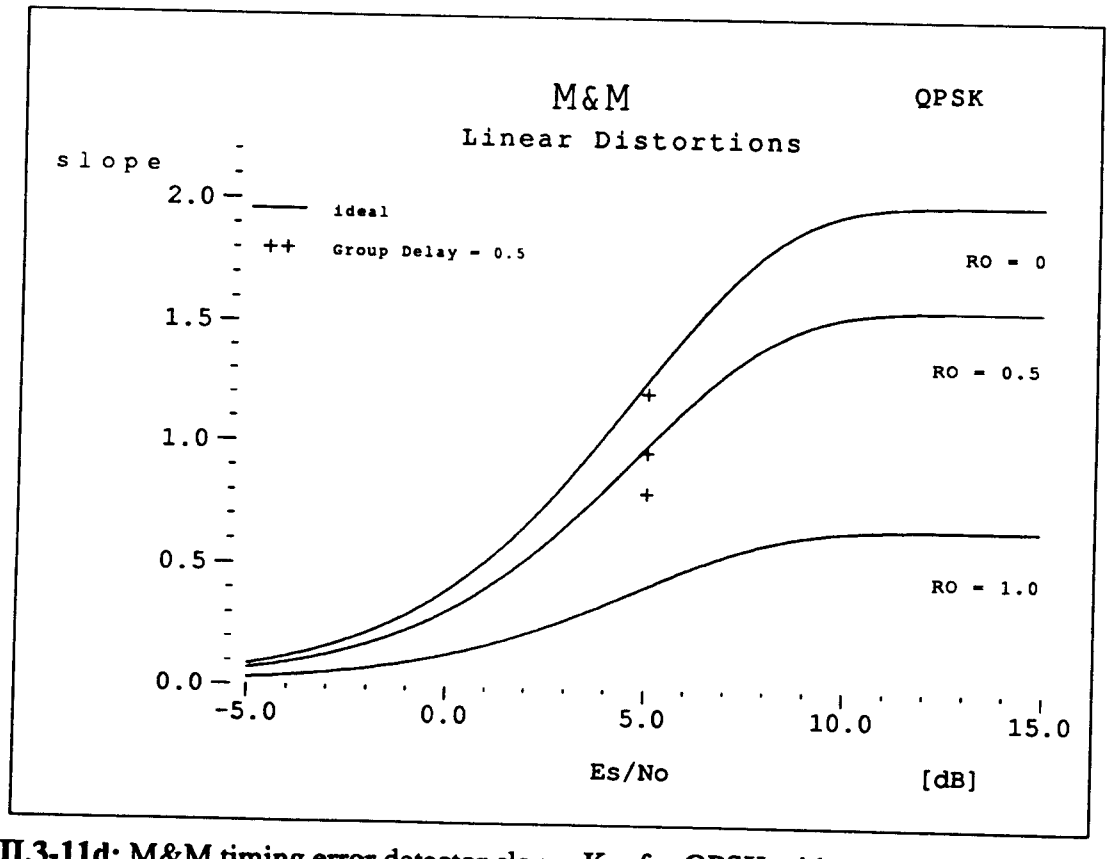


Fig. VII.3-11d: M&M timing error detector slopes $K_{\tau\tau}$ for QPSK with symmetric group delay distortion as a function of E_S/N_0

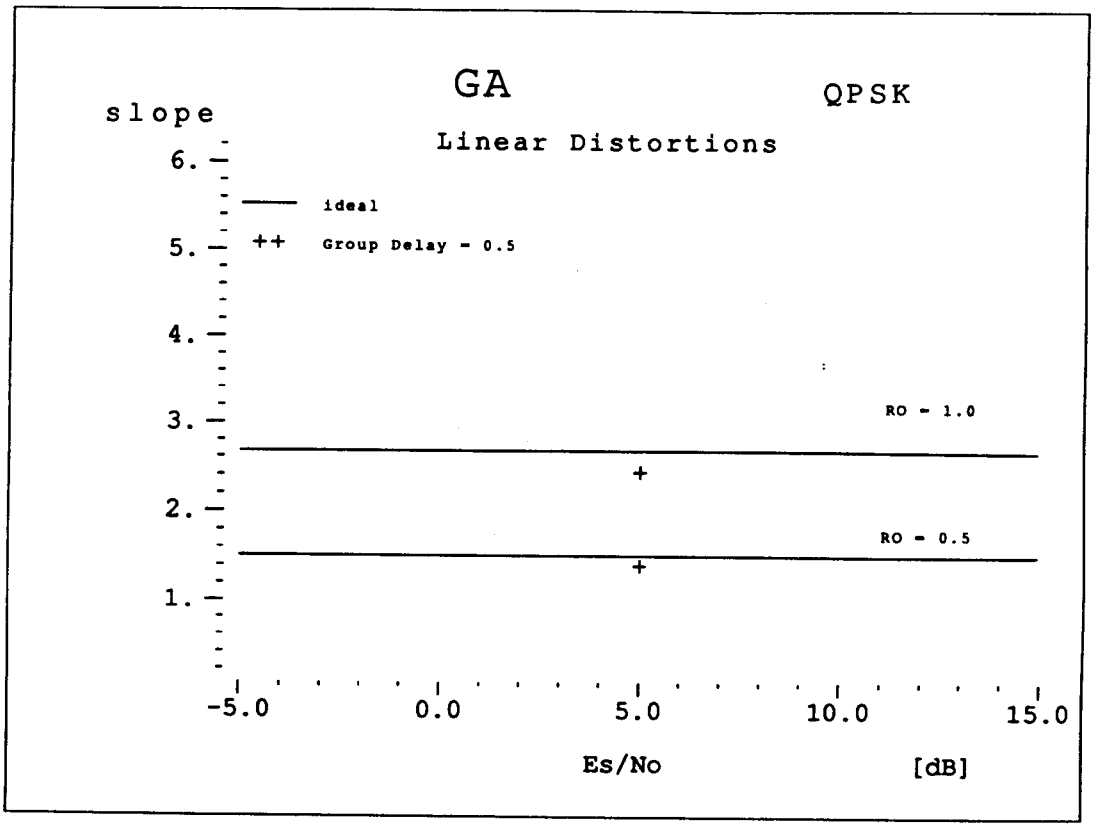


Fig. VII.3-11e: GA timing error detector slopes $K_{\tau\tau}$ for QPSK with symmetric group delay distortion as a function of E_S/N_0

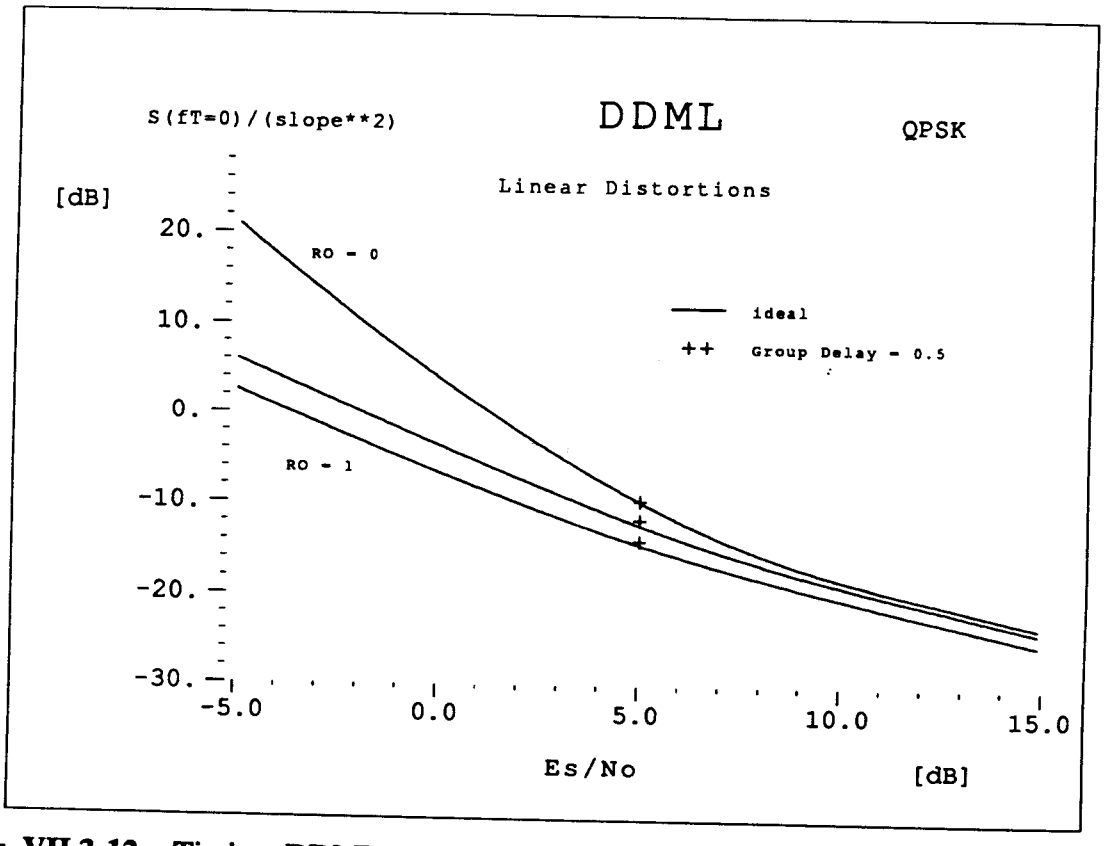


Fig. VII.3-12a: Timing: DDML normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\tau\tau}$) for QPSK with symmetric group delay distortion as a function of E_S/N_0

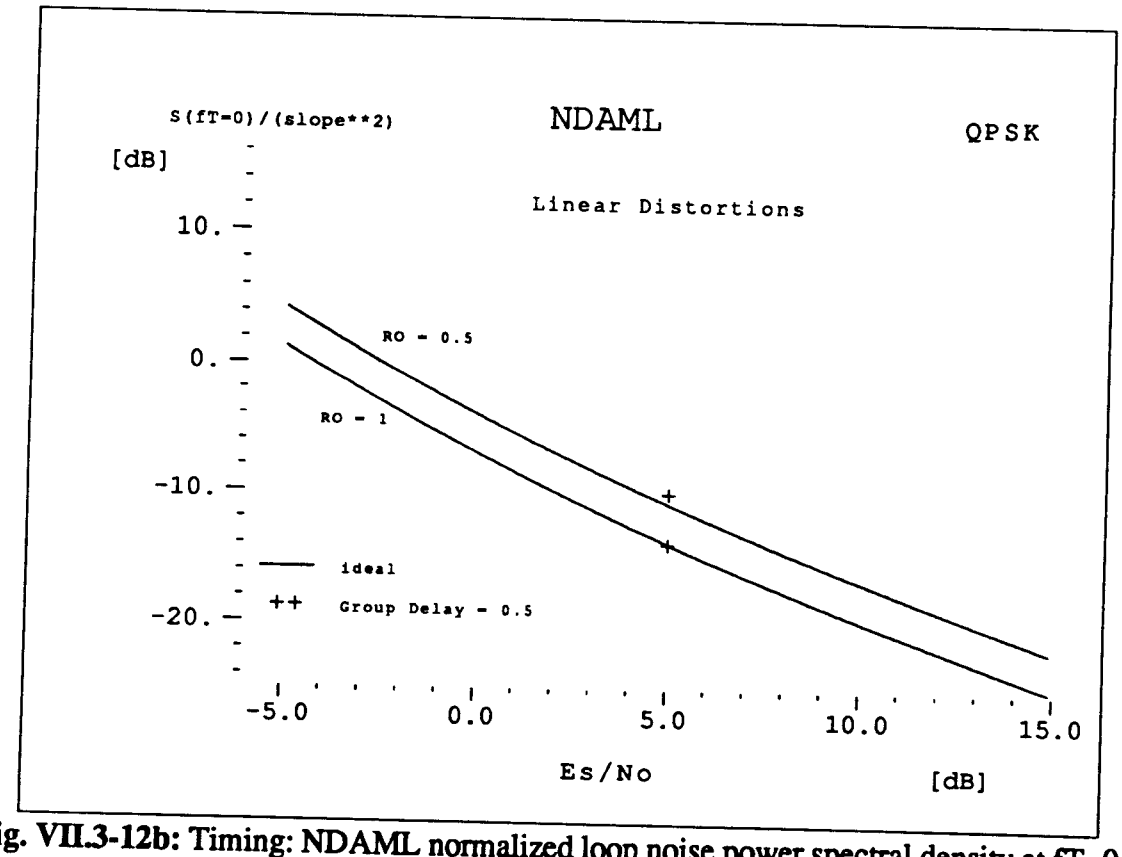


Fig. VII.3-12b: Timing: NDAML normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\tau\tau}$) for QPSK with symmetric group delay distortion as a function of E_S/N_0

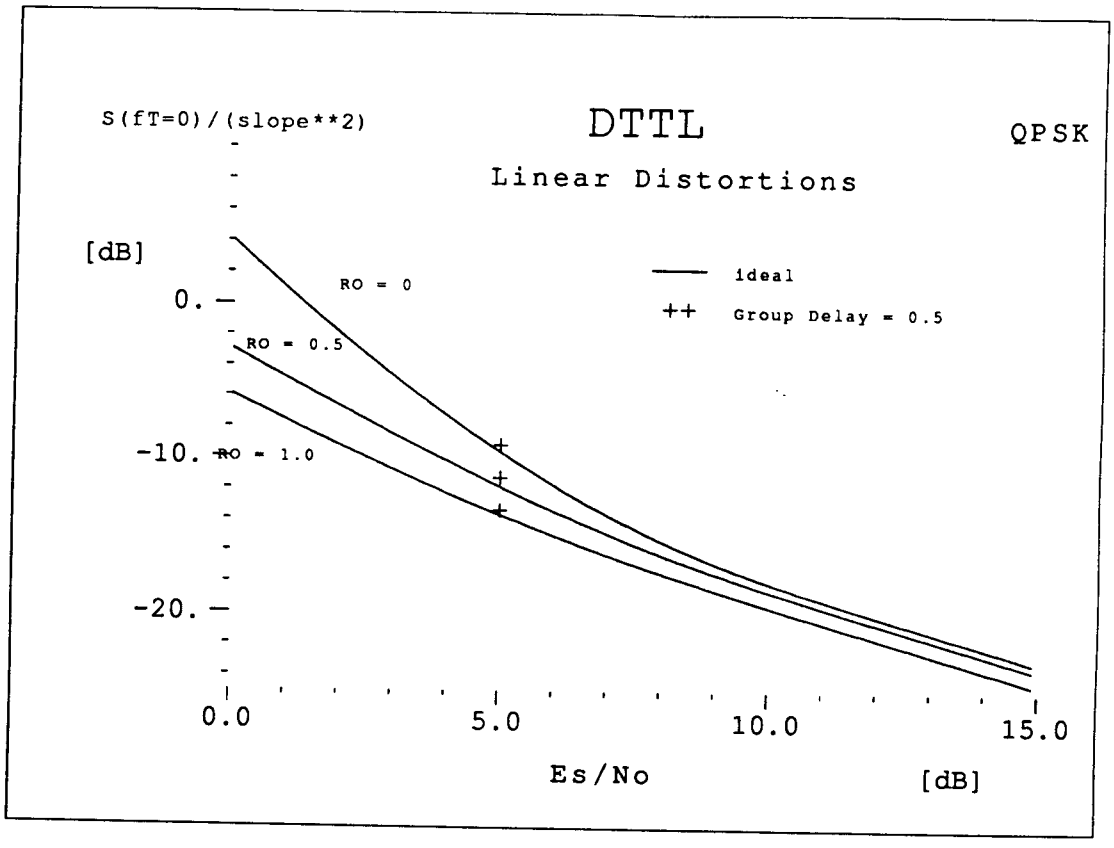


Fig. VII.3-12c: Timing: DTTL normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\tau\tau}$) for QPSK with symmetric group delay distortion as a function of E_S/N_0

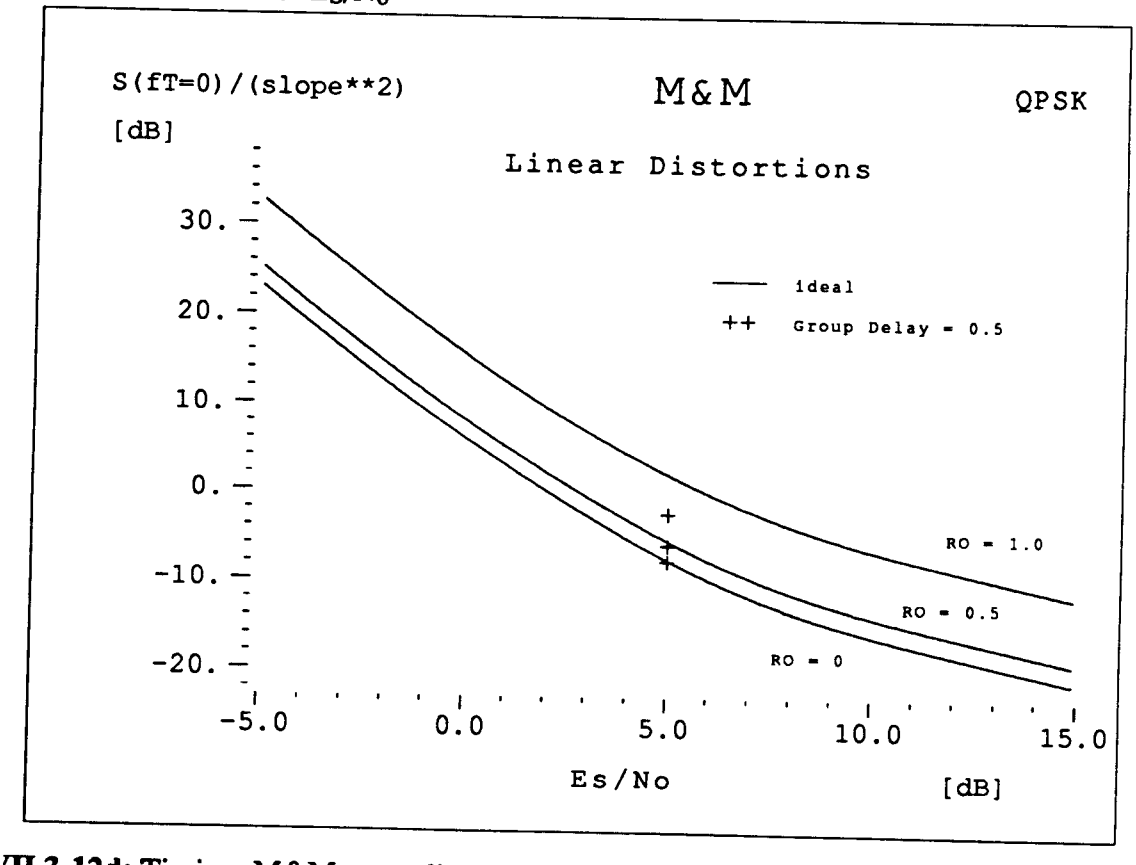


Fig. VII.3-12d: Timing: M&M normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\tau\tau}$) for QPSK with symmetric group delay distortion as a function of E_S/N_0

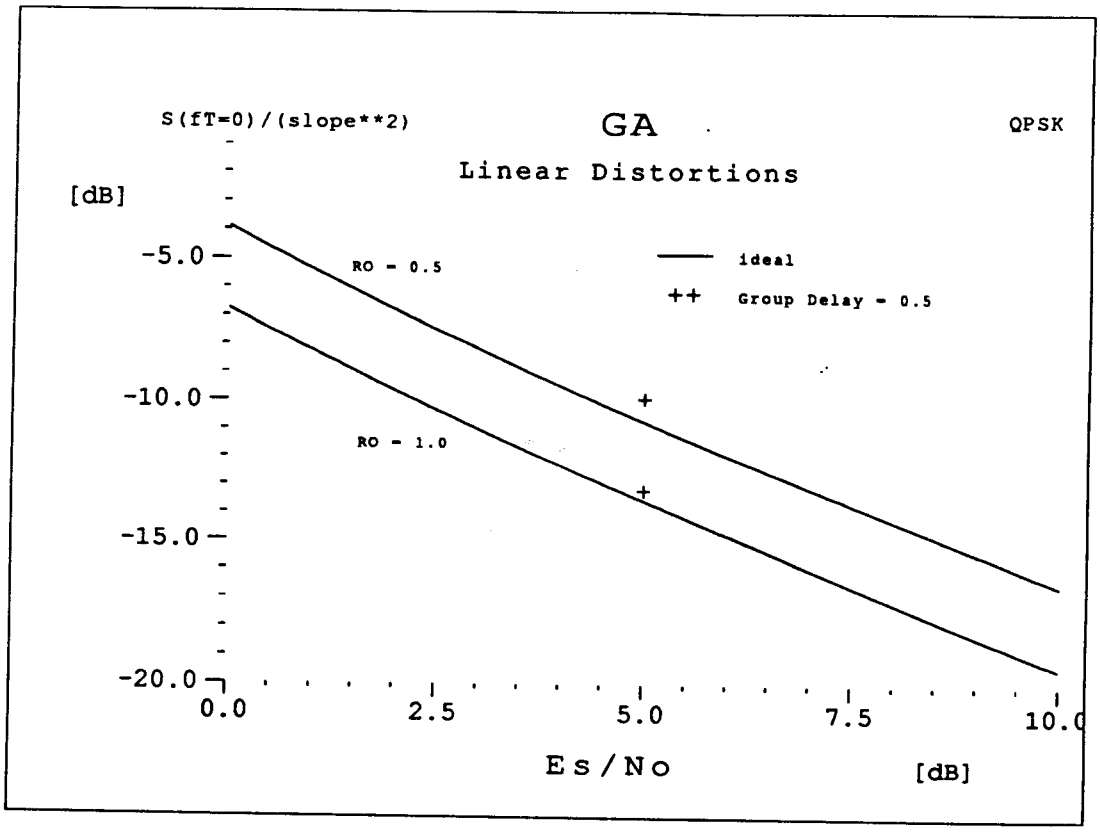


Fig. VII.3-12e: Timing: GA normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\tau\tau}$) for QPSK with symmetric group delay distortion as a function of E_S/N_0

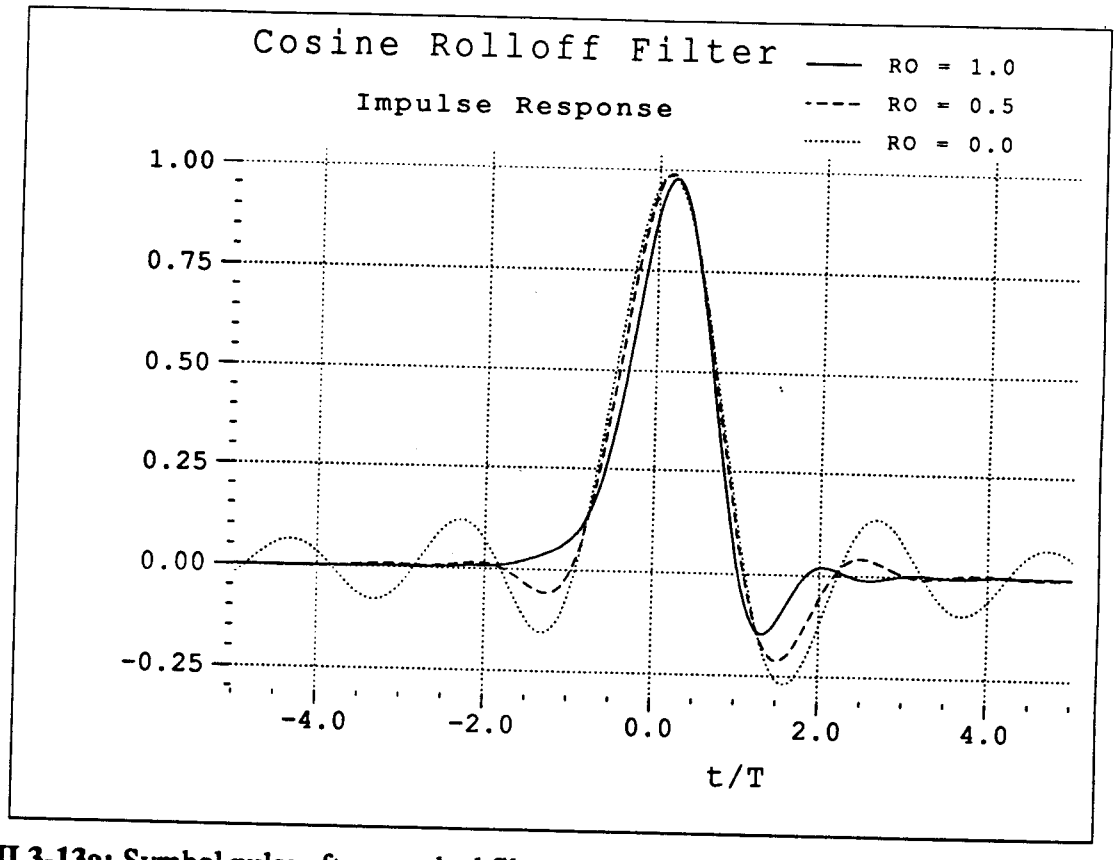


Fig. VII.3-13a: Symbol pulse after matched filter, cosine rolloff filter, symmetric group delay distortion, $\Delta\tau/T=0.5$

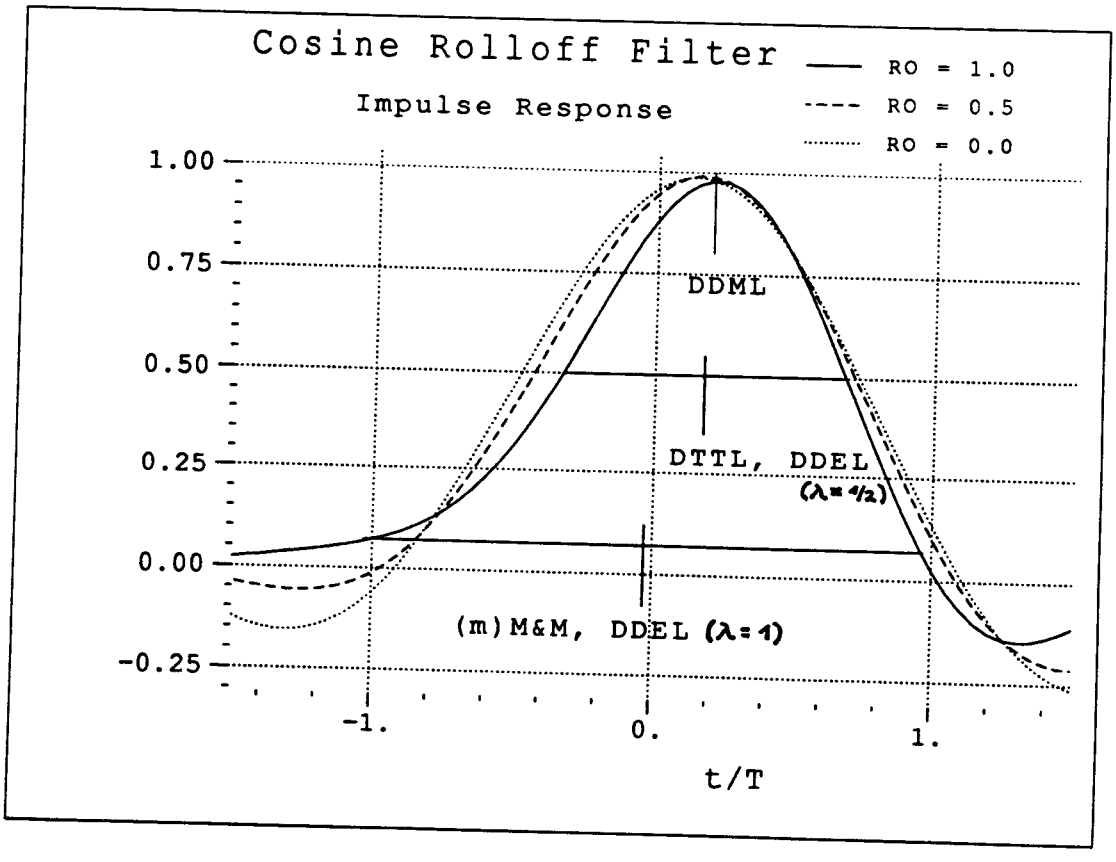


Fig. VII.3-13b: Same pulses as in Fig. VII.3-13a, expanded scale, analytically determined tracking points of different symbol synchronizers indicated for rolloff $\alpha=1$

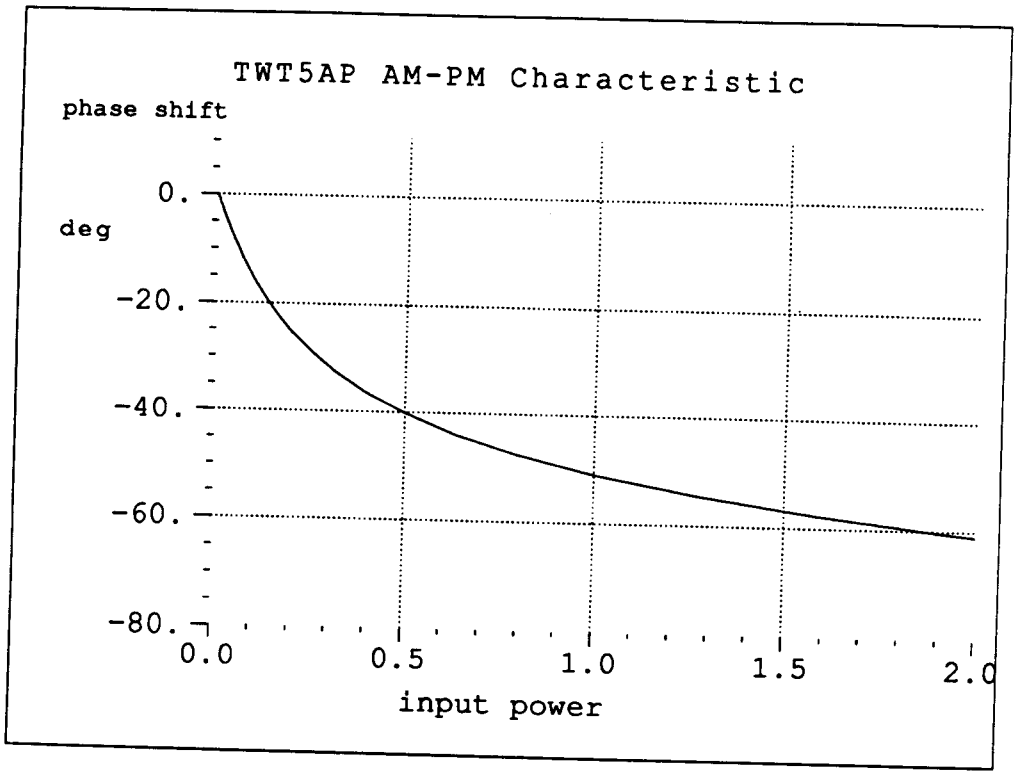


Fig. VII.4-1: AM-PM characteristic of TWT assumed in this study, both axes linear scaled

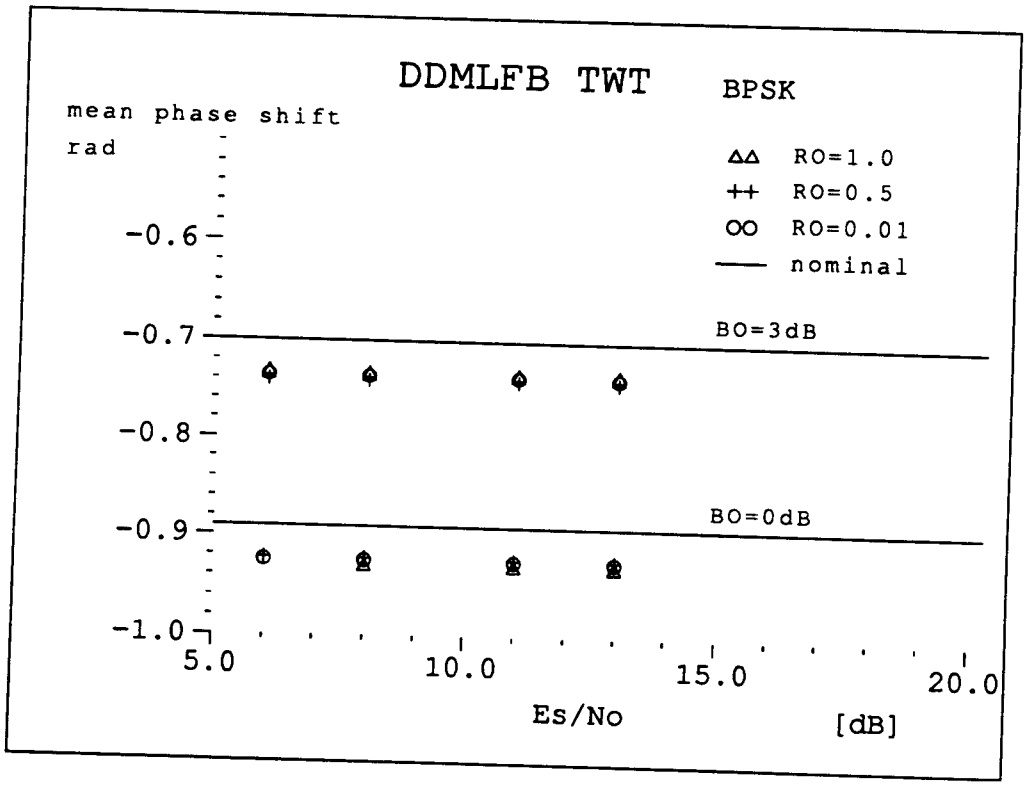


Fig. VII.4-2: Carrier: mean phase shift of DDMLFB for BPSK

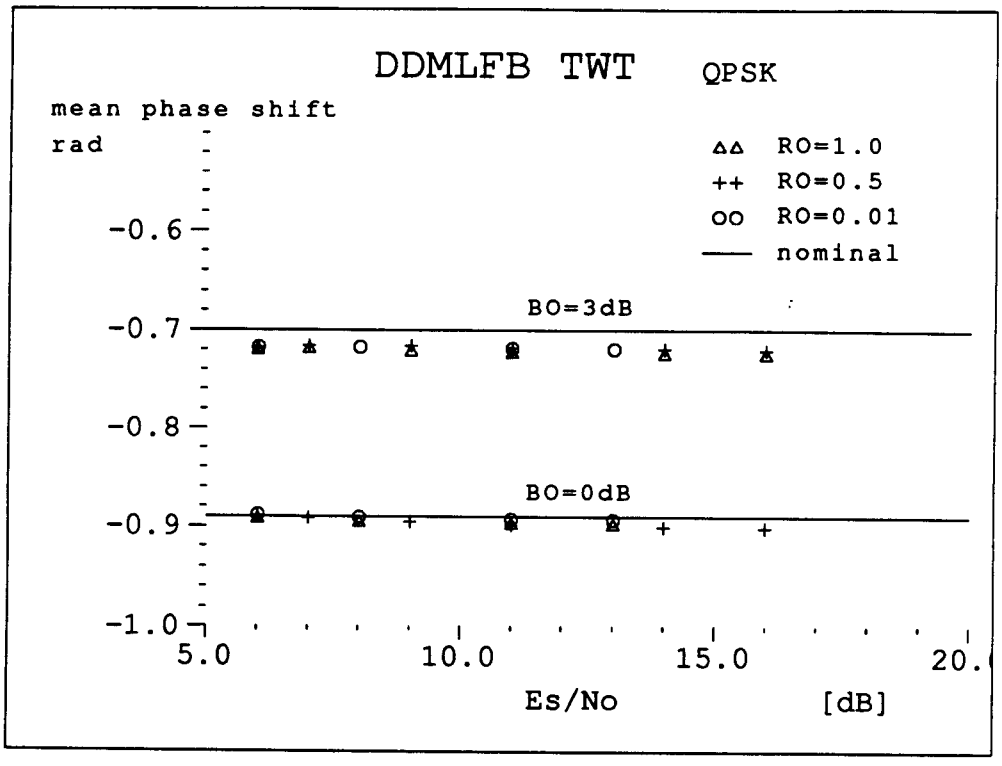


Fig. VII.4-3: Carrier: mean phase shift of DDMLFB for QPSK

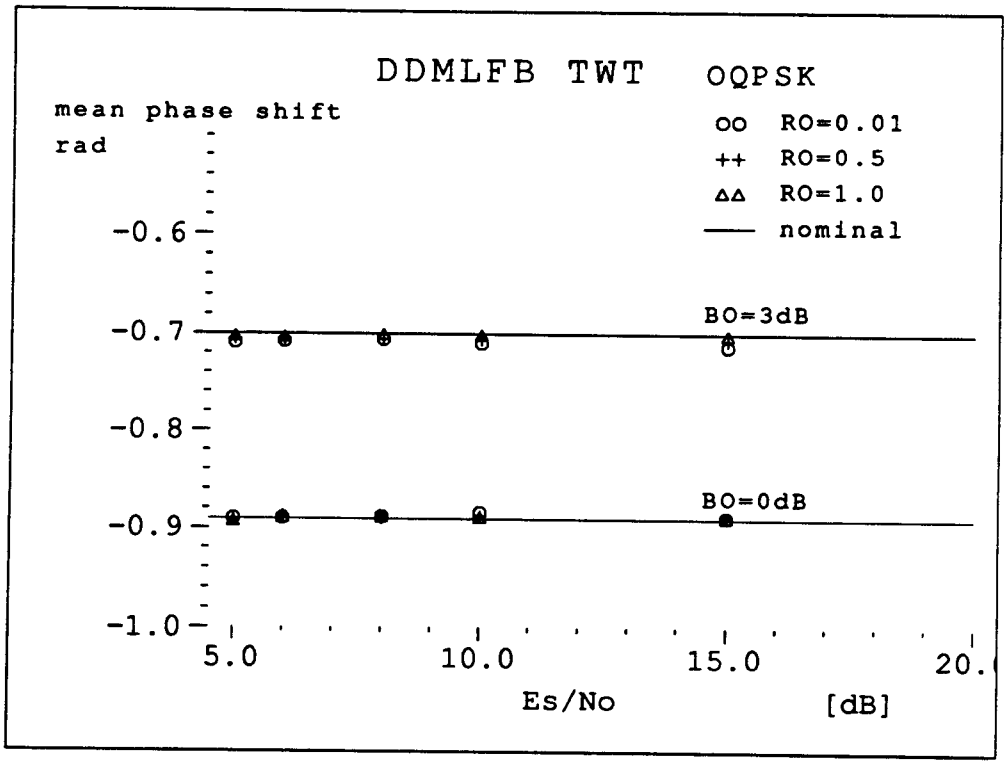


Fig. VII.4-4: Carrier: mean phase shift of DDMLFB for OQPSK

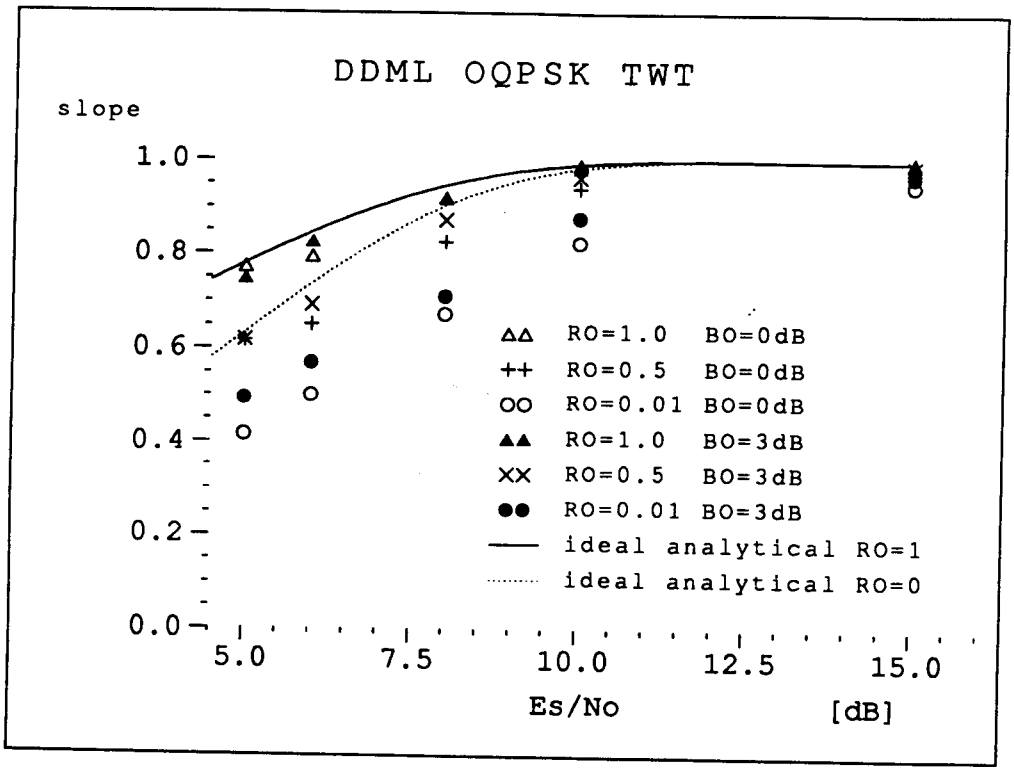


Fig. VII.4-5a: Carrier phase detector slope $K_{\theta\theta}$ of DDMLFB carrier synchronizer for OQPSK with TWT

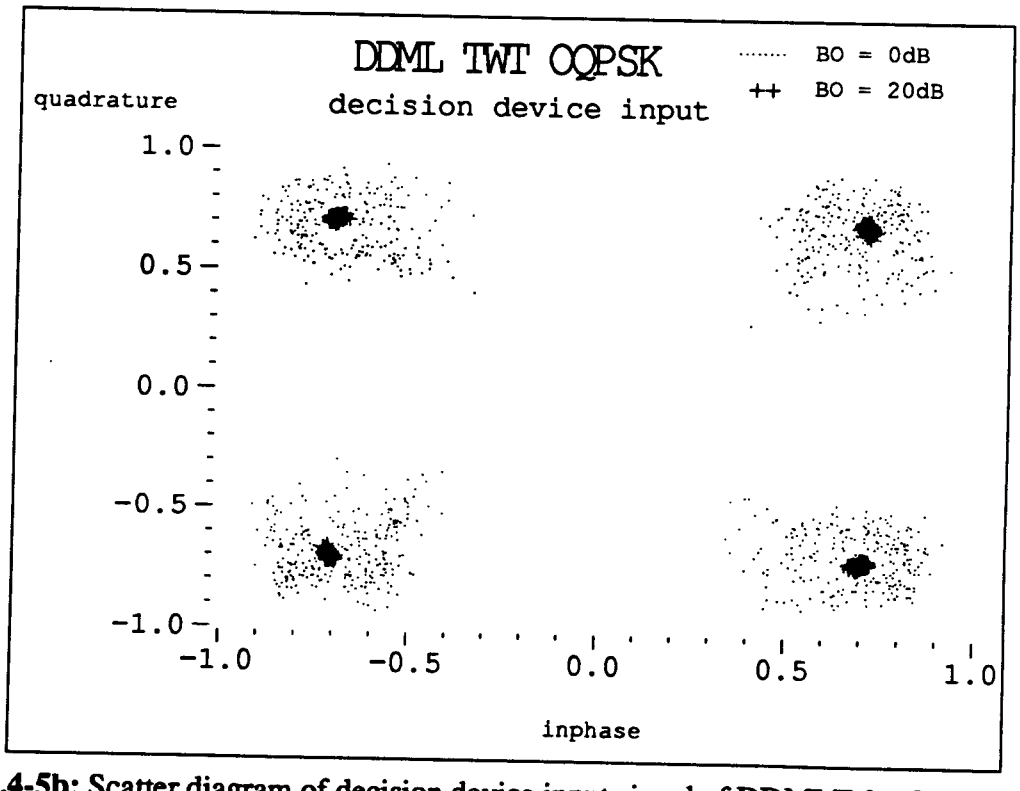


Fig. VII.4-5b: Scatter diagram of decision device input signal of DDMLT for OQPSK with TWT, input backoff=0dB and 20dB, root raised cosine filter, rolloff α=0.1. Note that Im(q) against Re(p) is shown.

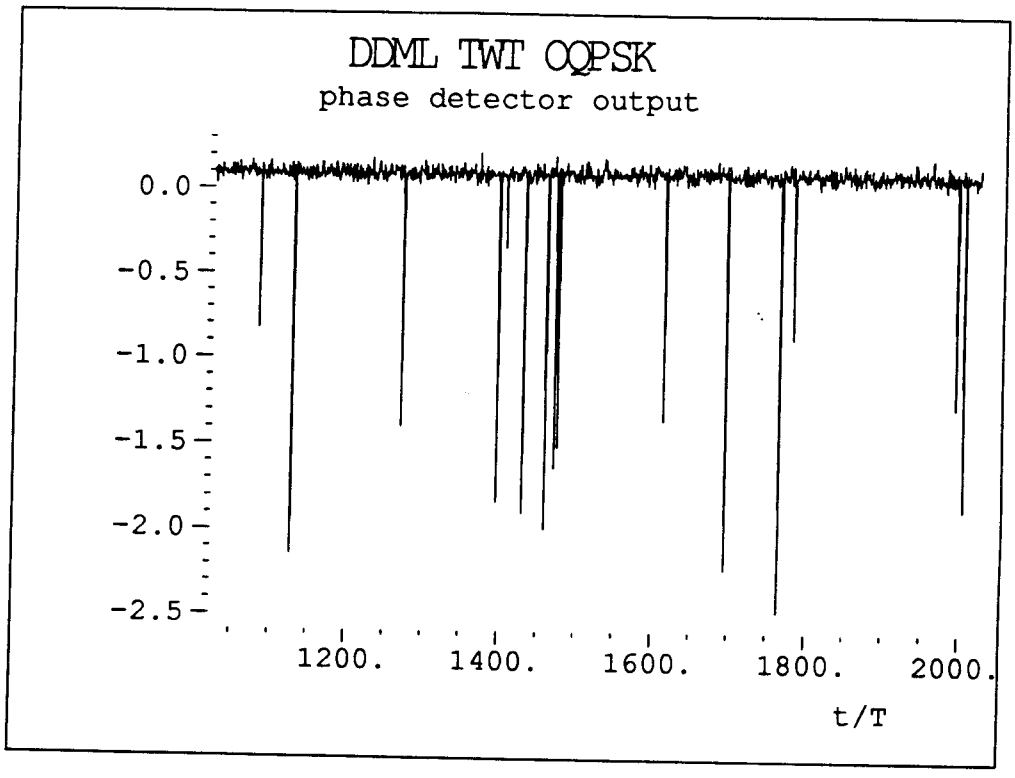


Fig. VII.4-5c: Difference of phase detector output signal for $\Delta\theta = 0$ and $\Delta\theta = 0.1$ rad of DDMLFB, OQPSK, TWT, input backoff 0dB, rolloff α=0.1, E_S/N_0 = 8dB

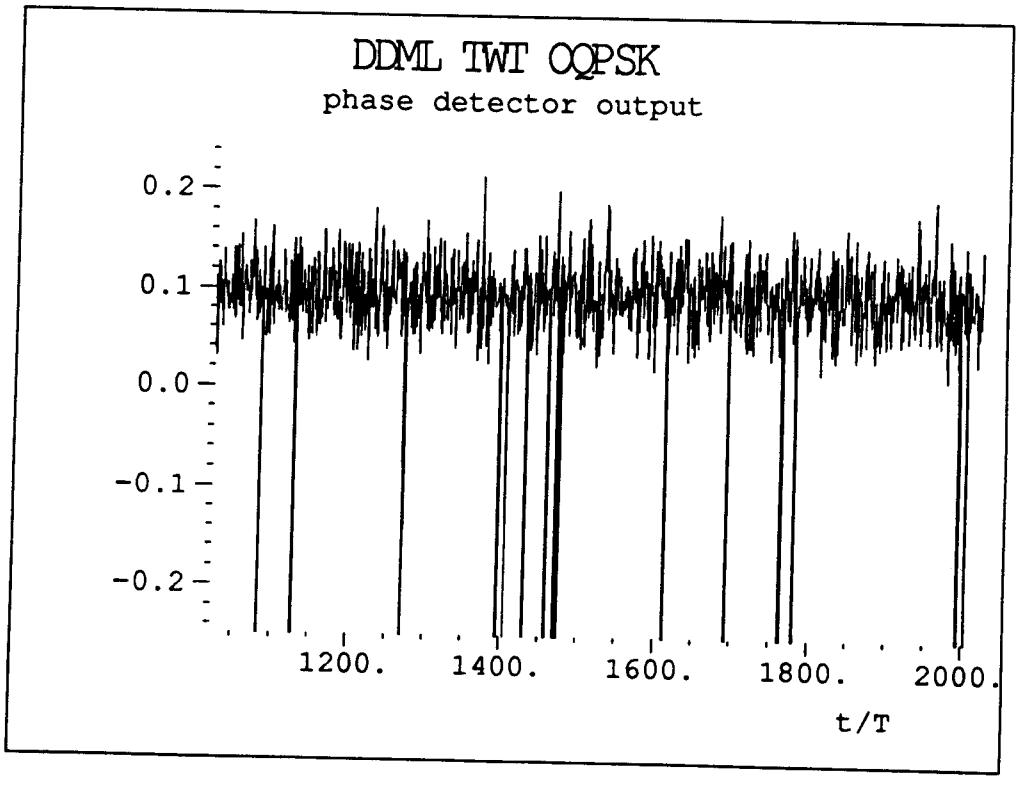


Fig. VII.4-5d: Same data as VII.4-5c, extended scale

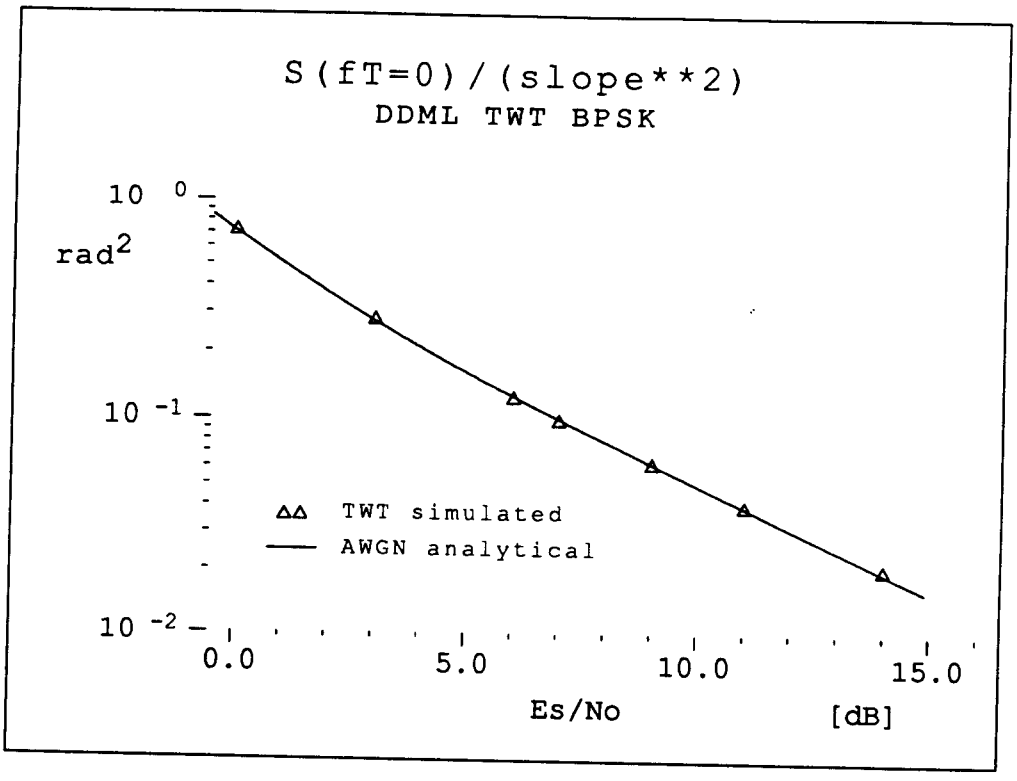


Fig. VII.4-6: Carrier: normalized loop noise power spectral density $S(fT=0)/K^2_{\theta\theta}$ of DDMLFB for BPSK with TWT, 0dB backoff, logarithmic scale.

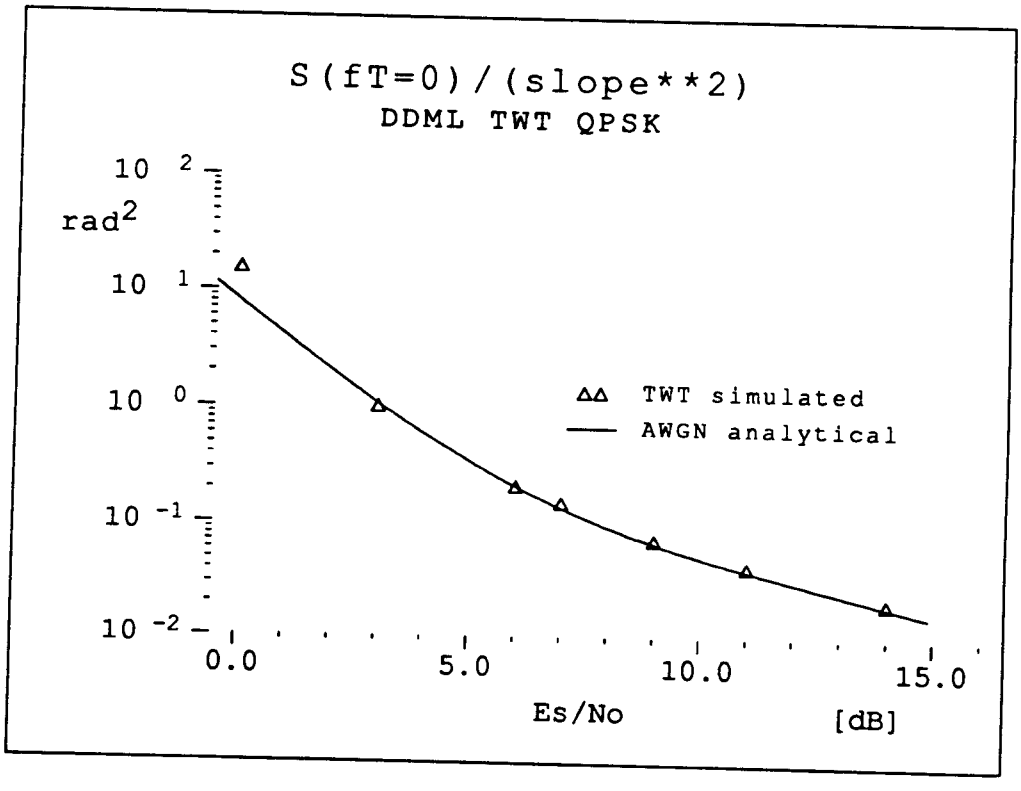


Fig. VII.4-7: Carrier: normalized loop noise power spectral density $S(fT=0)/K^2_{\theta\theta}$ of DDMLFB for QPSK with TWT, 0dB backoff, logarithmic scale.

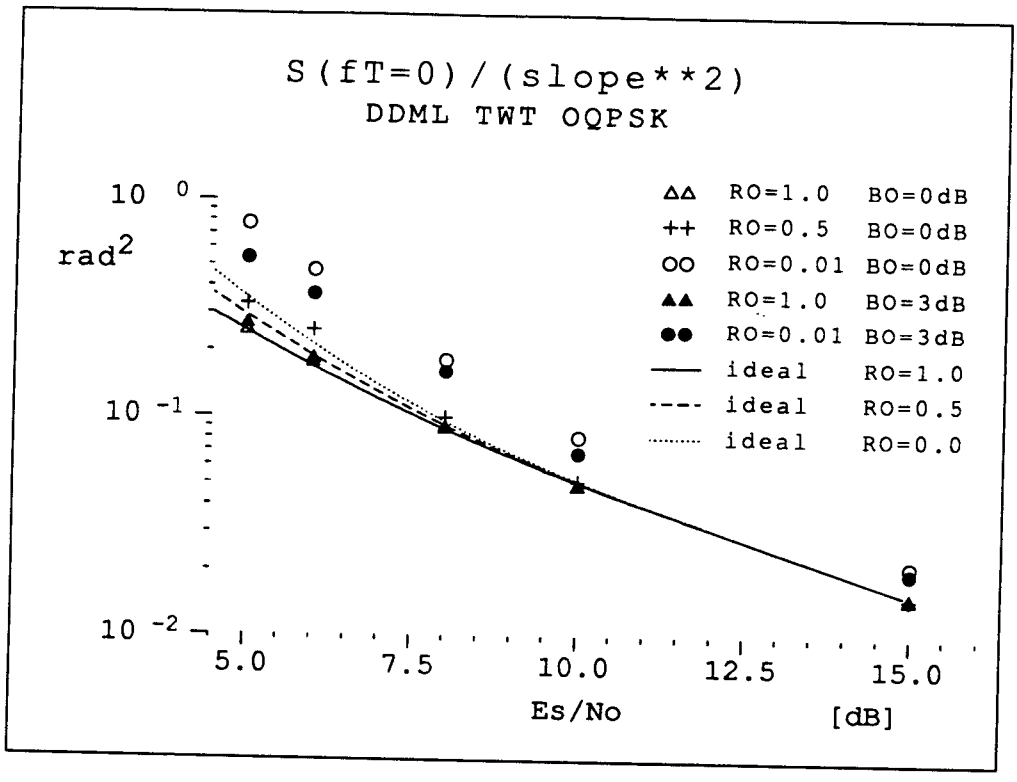


Fig. VII.4-8a: Carrier: normalized loop noise power spectral density $S(fT=0)/K^2_{\theta\theta}$ of DDMLFB for OQPSK with TWT, logarithmic scale.

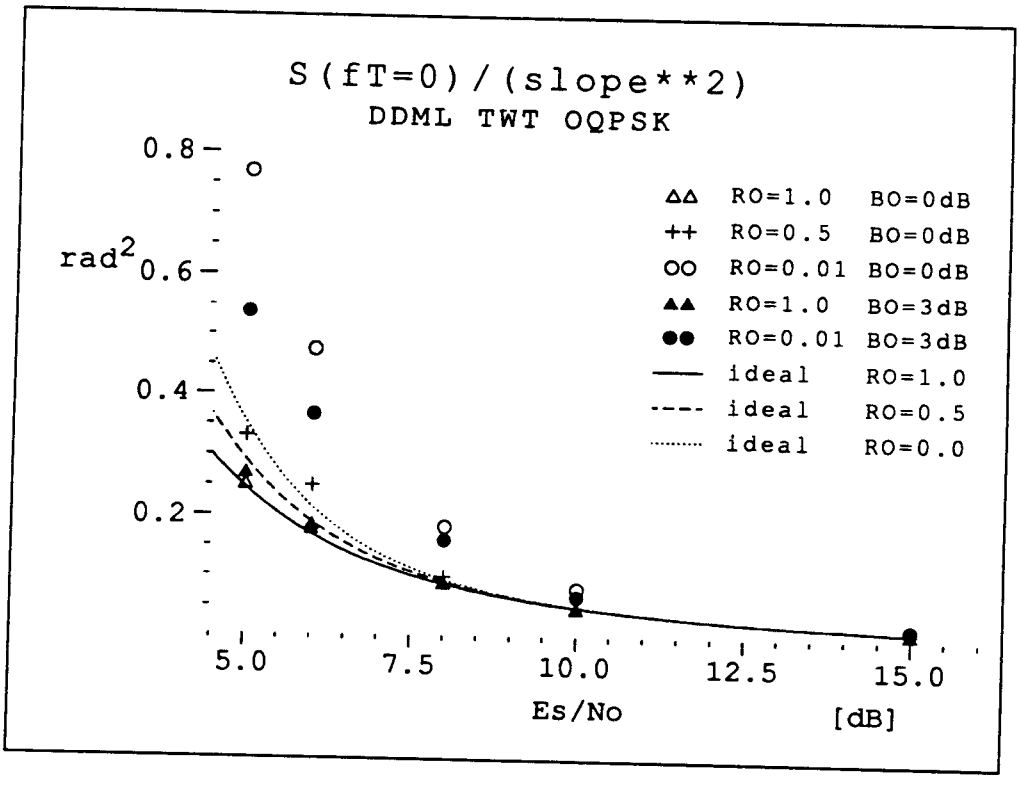


Fig. VII.4-8b: Carrier: normalized loop noise power spectral density $S(fT=0)/K^2_{\theta\theta}$ of DDMLFB for OQPSK with TWT, linear scale.

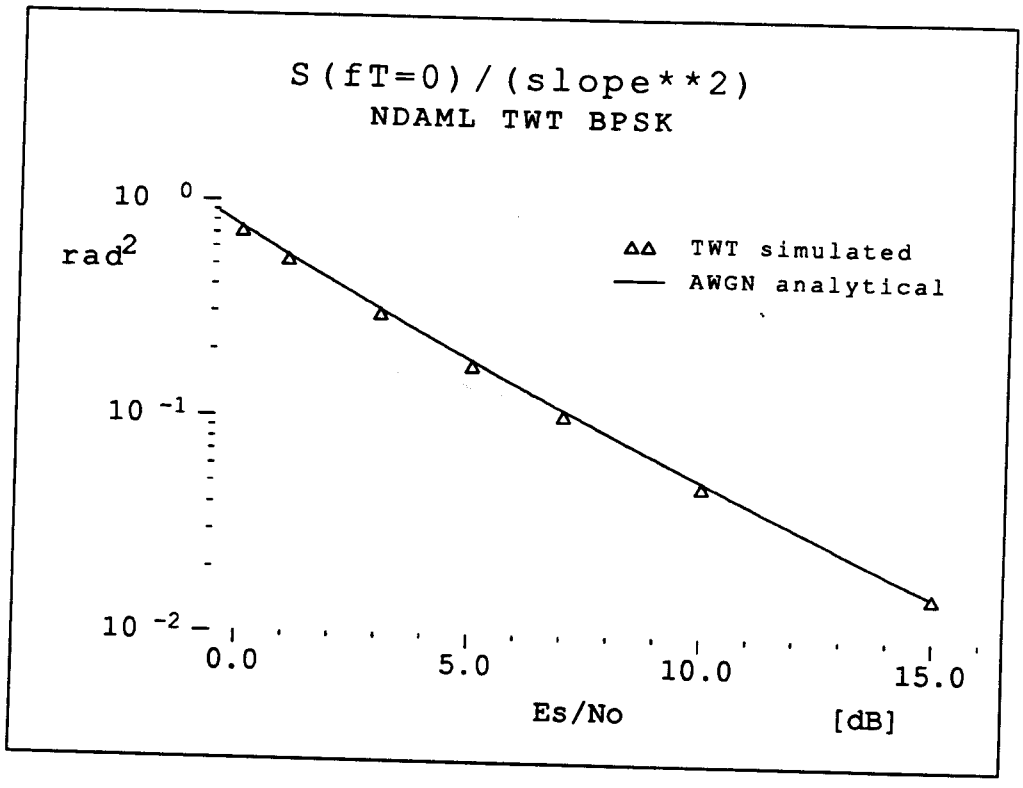


Fig. VII.4-9a: Carrier: normalized loop noise power spectral density $S(fT=0)/K^2_{\theta\theta}$ of NDAMLFB for BPSK with TWT, 0dB backoff, logarithmic scale.

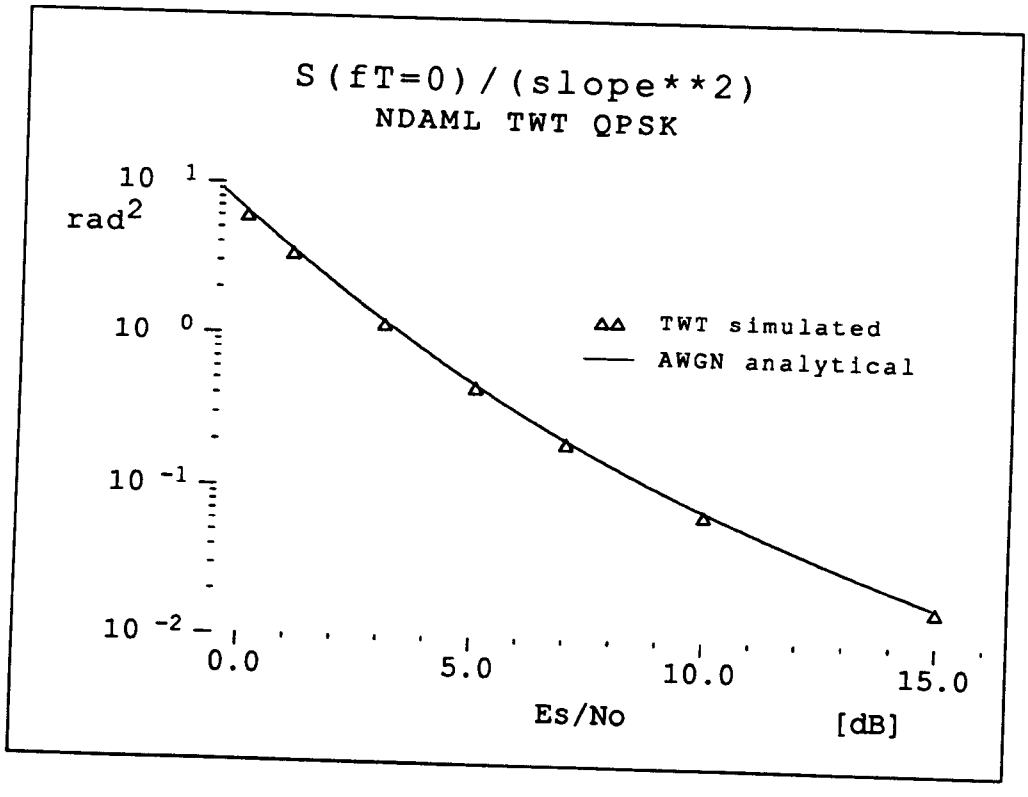


Fig. VII.4-9b: Carrier: normalized loop noise power spectral density $S(fT=0)/K^2_{\theta\theta}$ of NDAMLFB for QPSK with TWT, 0dB backoff, logarithmic scale.

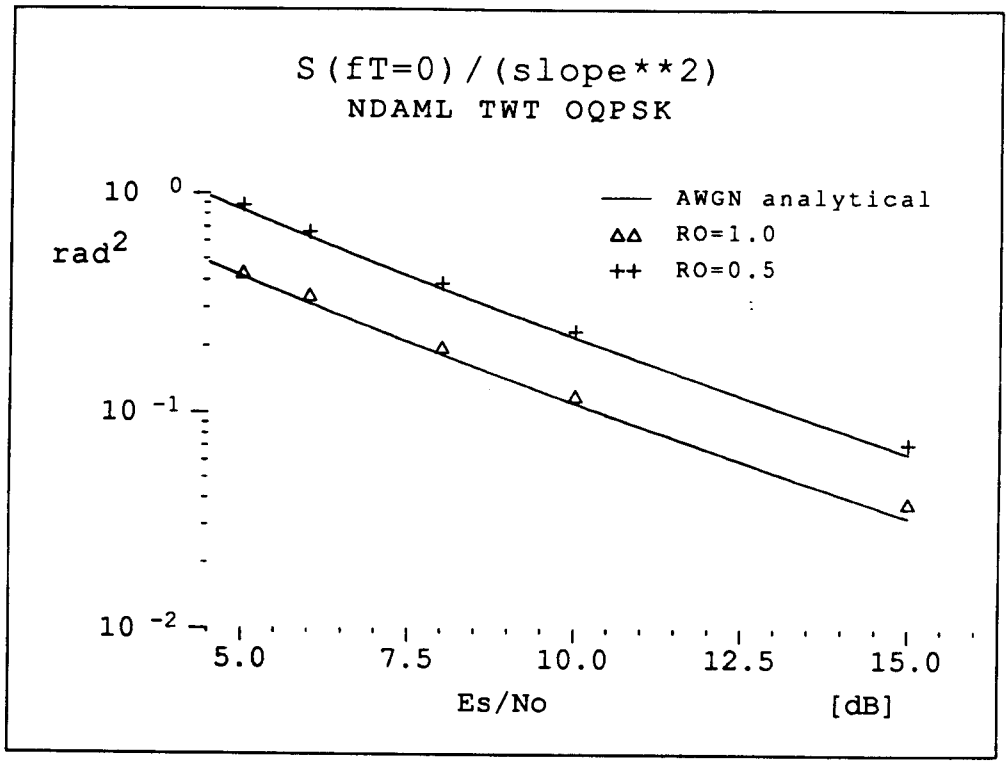


Fig. VII.4-9c: Carrier: normalized loop noise power spectral density $S(fT=0)/K^2_{\theta\theta}$ of NDAMLFB for OQPSK with TWT, 0dB backoff, logarithmic scale.

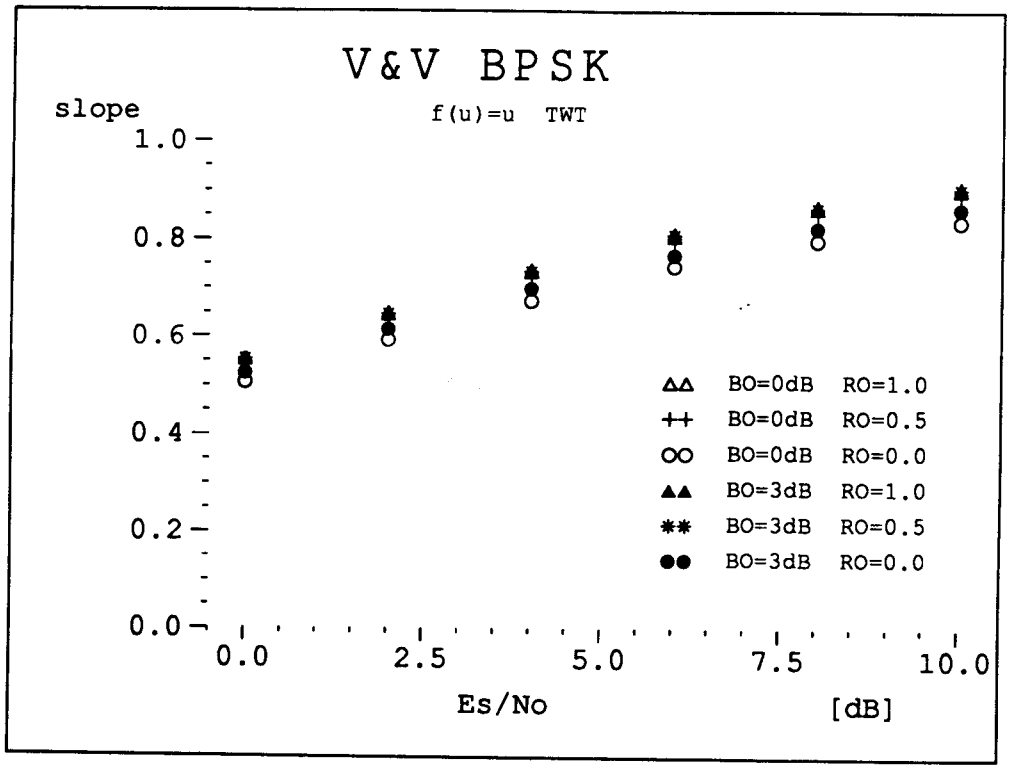


Fig. VII.4-10a: Carrier: V&V phase detector slopes $K_{\theta\theta}$ for BPSK, nonlinearity f(u)=u, with TWT

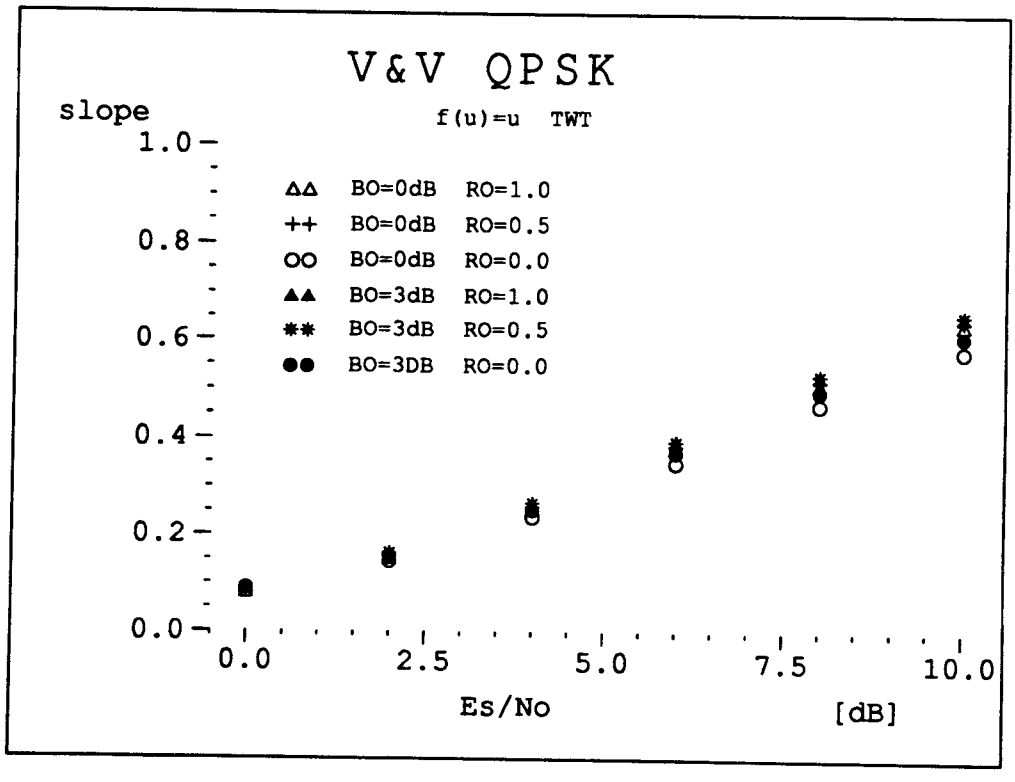


Fig. VII.4-10b: Carrier: V&V phase detector slopes $K_{\theta\theta}$ for QPSK, nonlinearity f(u)=u, with TWT

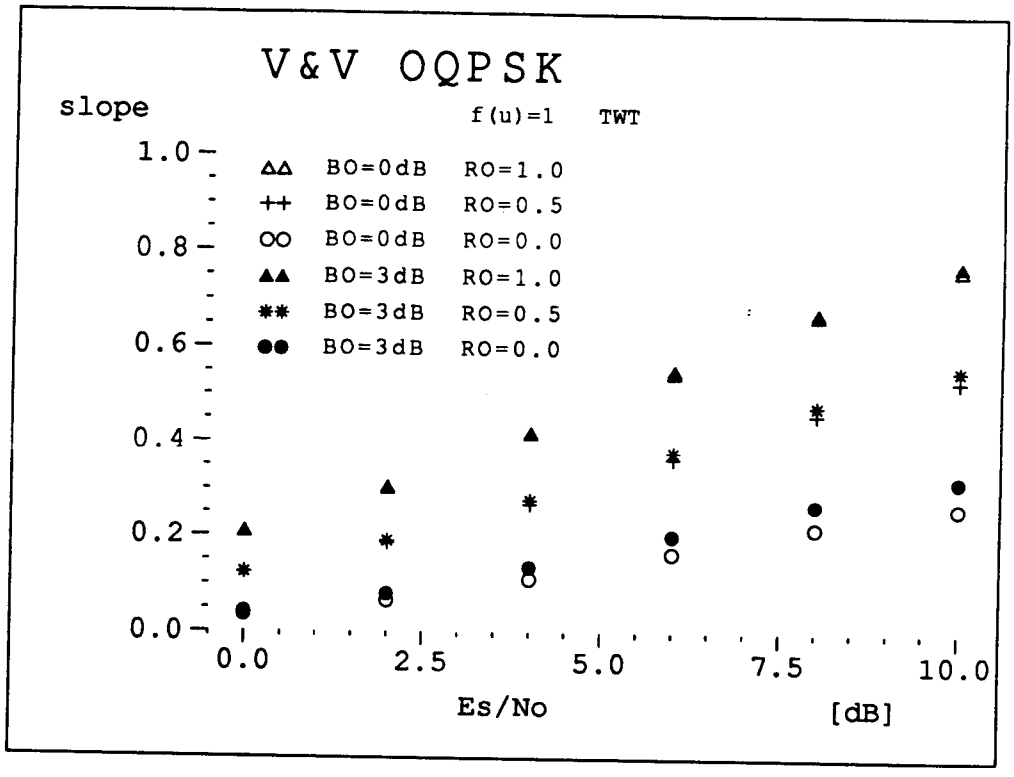


Fig. VII.4-10c: Carrier: V&V phase detector slopes $K_{\theta\theta}$ for OQPSK, nonlinearity f(u)=1, with TWT

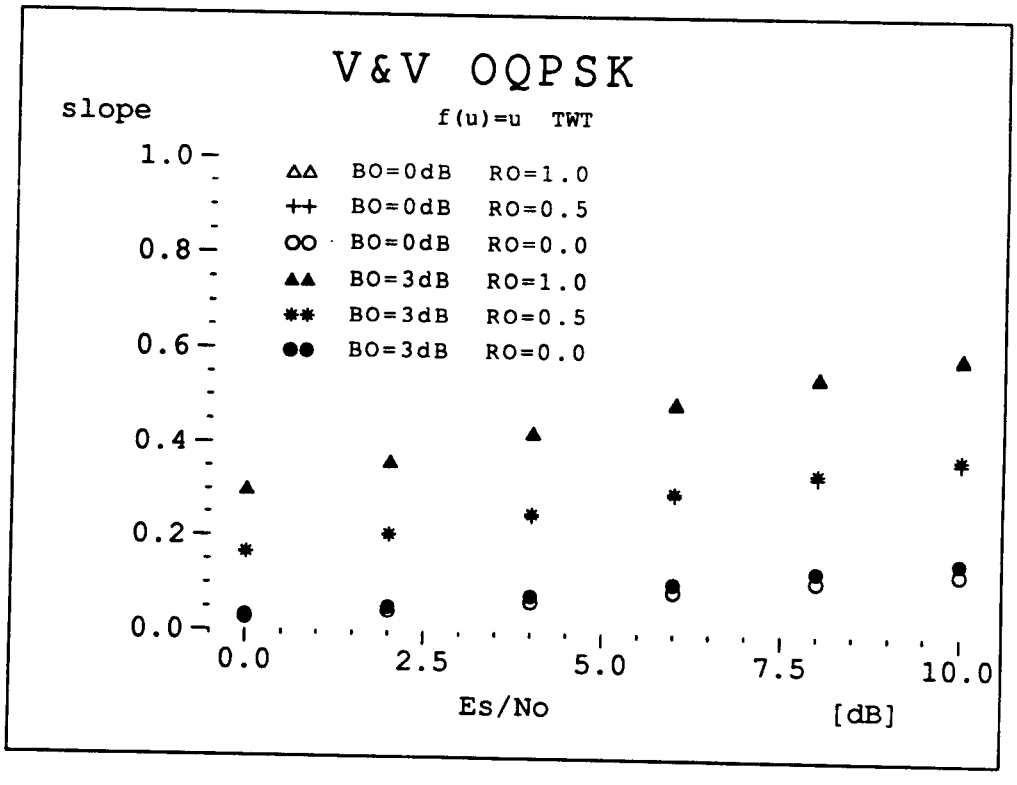


Fig. VII.4-10d: Carrier: V&V phase detector slopes $K_{\theta\theta}$ for OQPSK, nonlinearity f(u)=u, with TWT

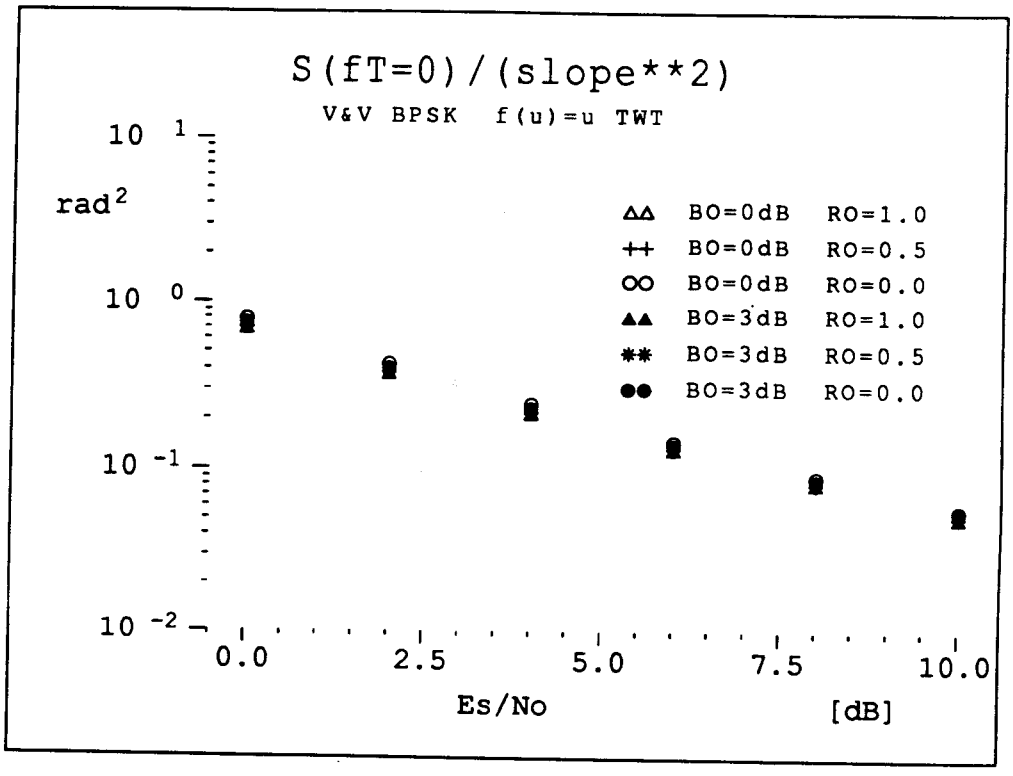


Fig. VII.4-11a: Carrier: normalized power spectral density $S(fT=0)/K_{\theta\theta}$ of V&V for BPSK with TWT, f(u)=u, logarithmic scale

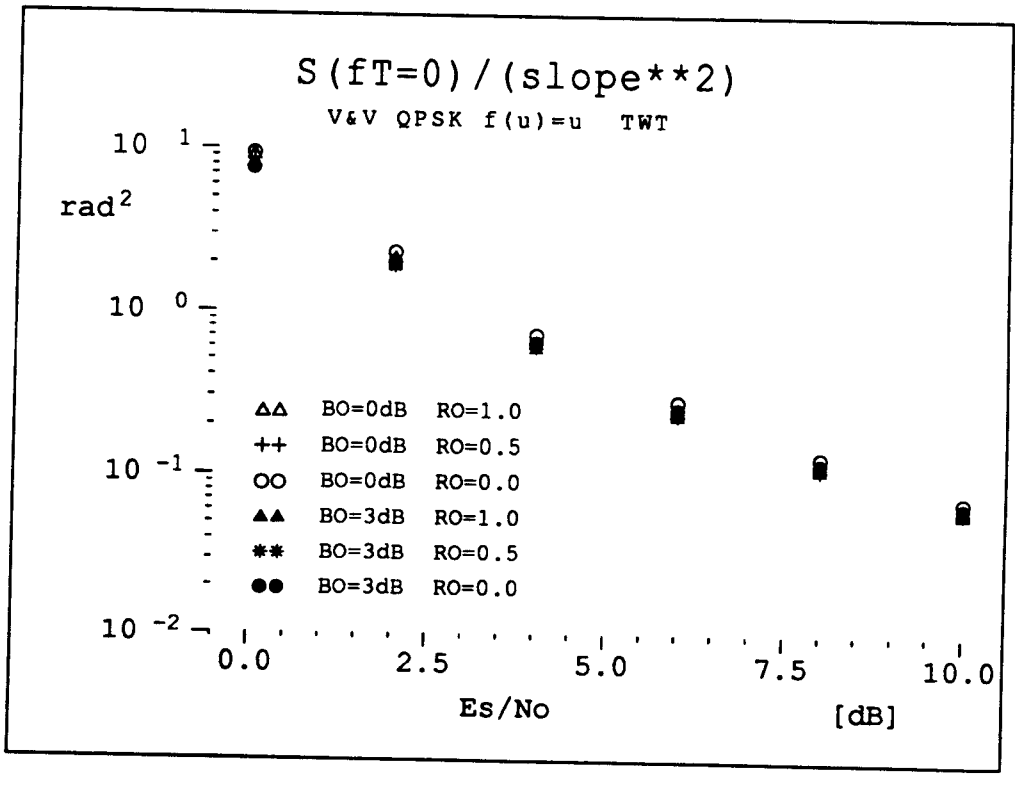


Fig. VII.4-11b: Carrier: normalized power spectral density $S(fT=0)/K_{\theta\theta}$ of V&V for QPSK with TWT, f(u)=u, logarithmic scale

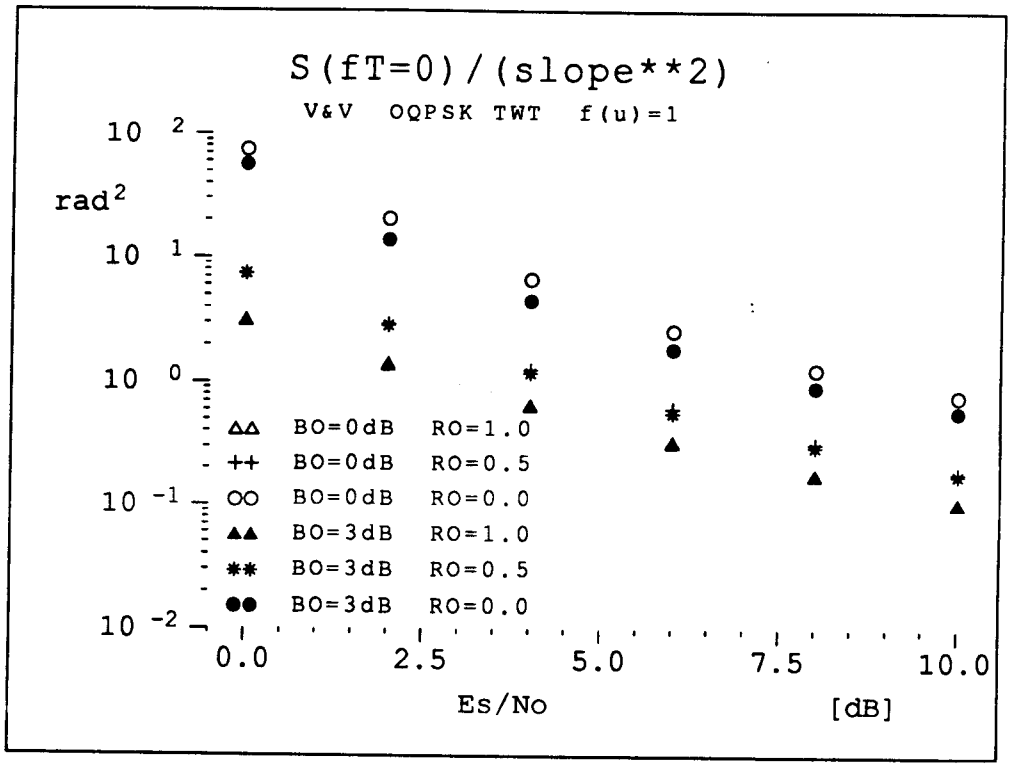


Fig. VII.4-11c: Carrier: normalized power spectral density $S(fT=0)/K_{\theta\theta}$ of V&V for OQPSK with TWT, nonlinearity f(u)=1, logarithmic scale

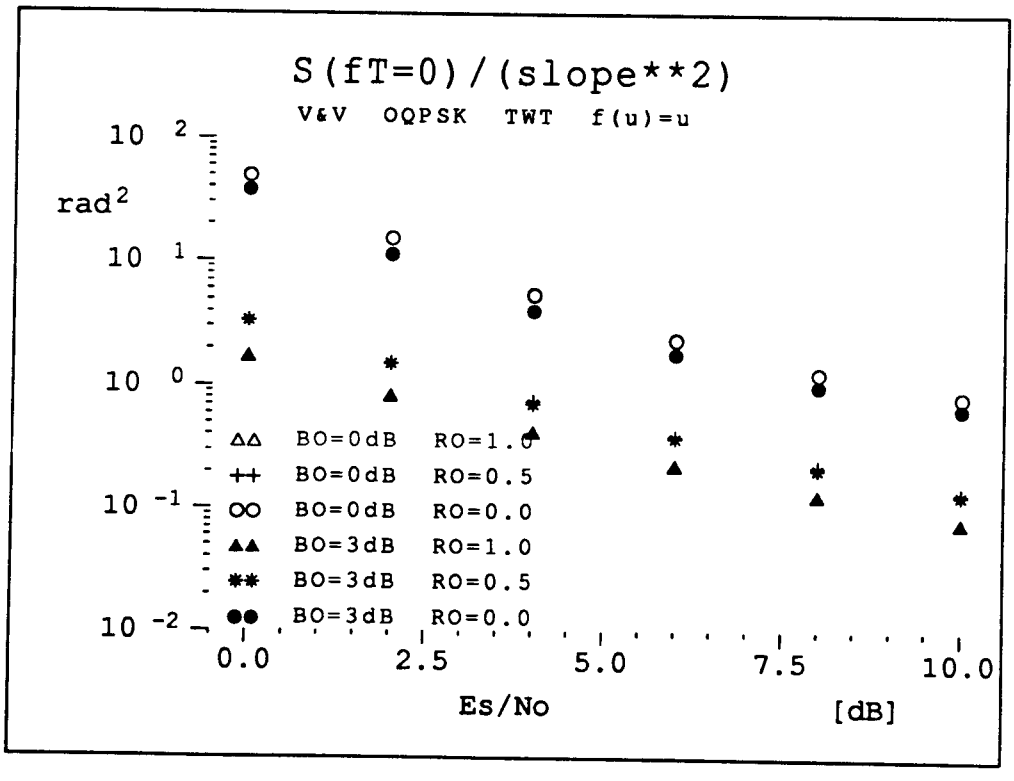


Fig. VII.4-11d: Carrier: normalized power spectral density $S(fT=0)/K_{\theta\theta}$ of V&V for OQPSK with TWT, nonlinearity f(u)=u, logarithmic scale

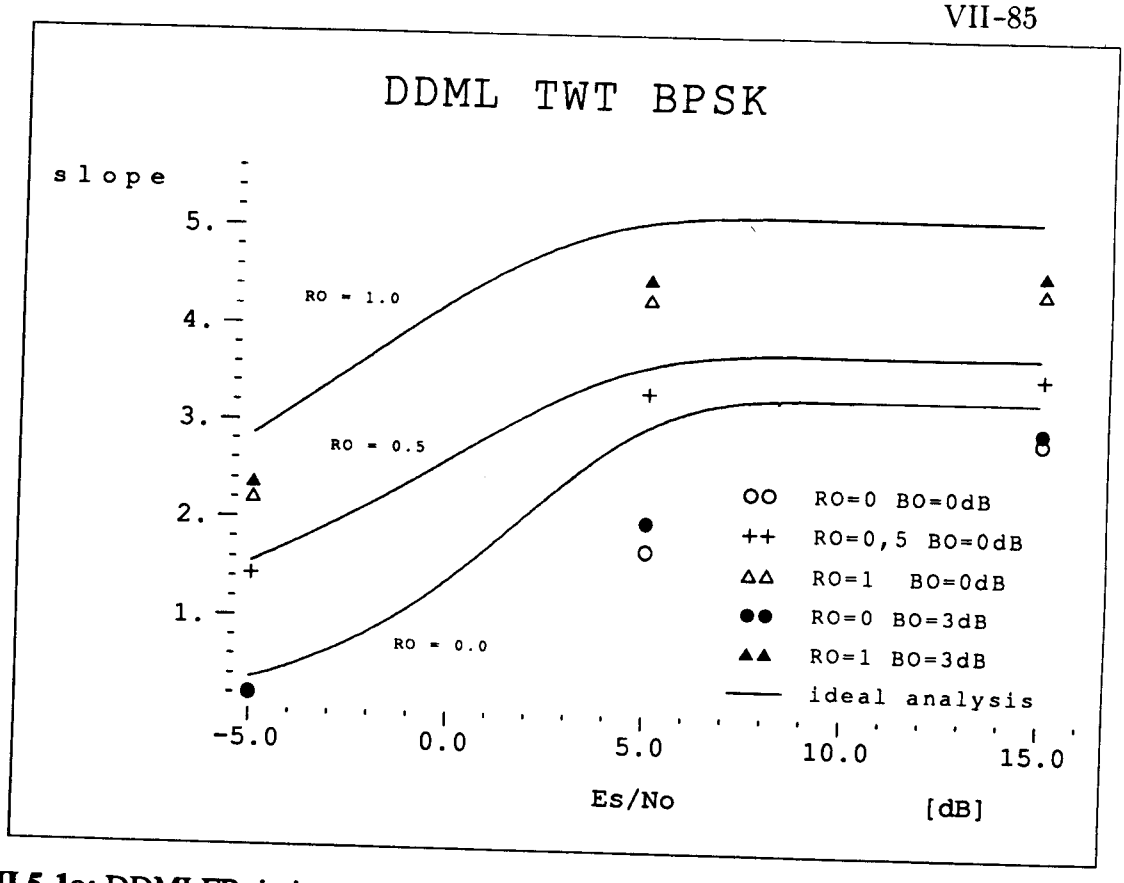


Fig. VII.5-1a: DDMLFB timing error detector slopes $K_{\tau\tau}$ for BPSK with TWT as a function of E_S/N_0

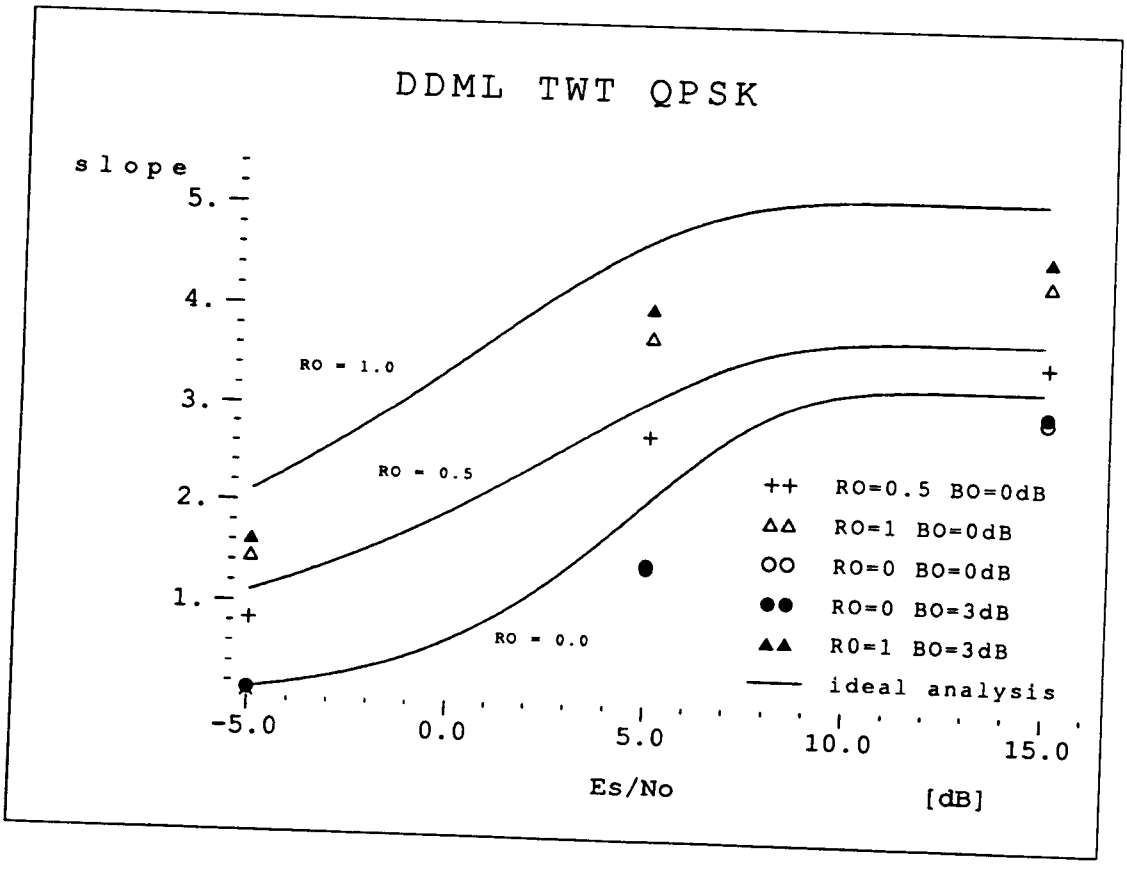


Fig. VII.5-1b: DDMLFB timing error detector slopes $K_{\tau\tau}$ for QPSK with TWT as a function of E_S/N_0

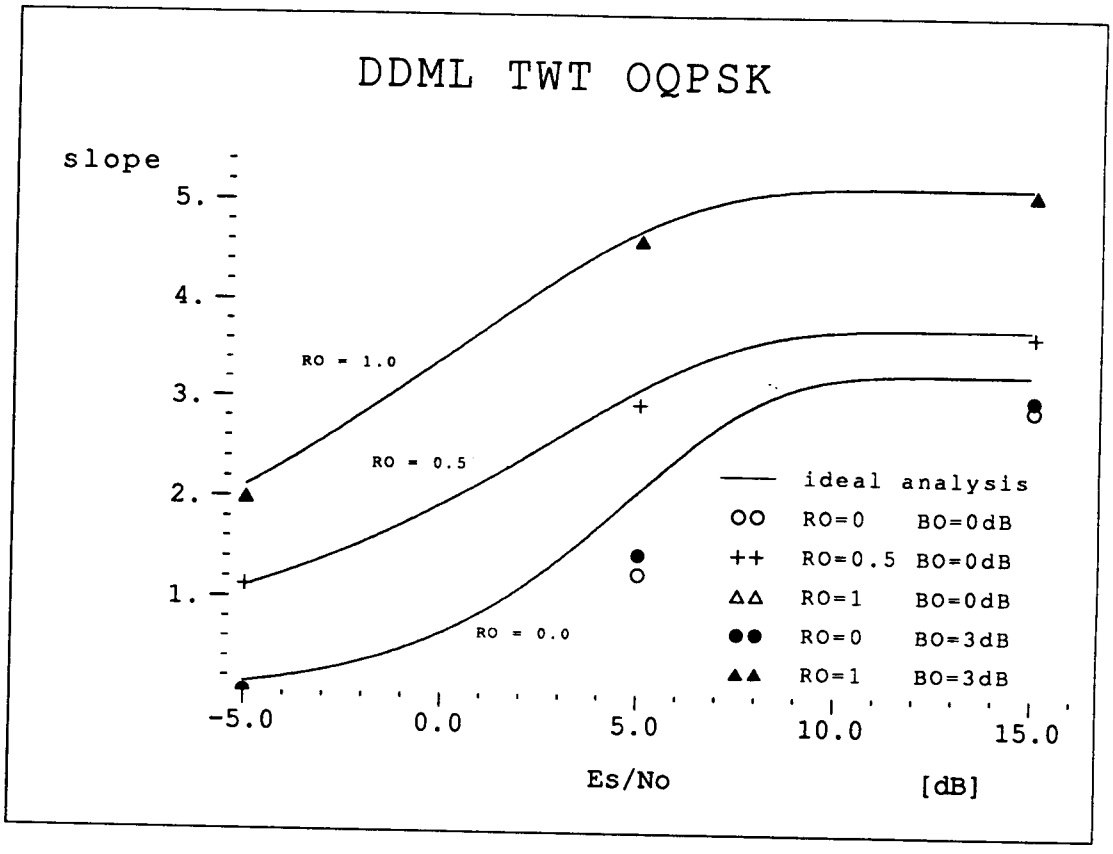


Fig. VII.5-1c: DDMLFB timing error detector slopes $K_{\tau\tau}$ for OQPSK with TWT as a function of E_S/N_0

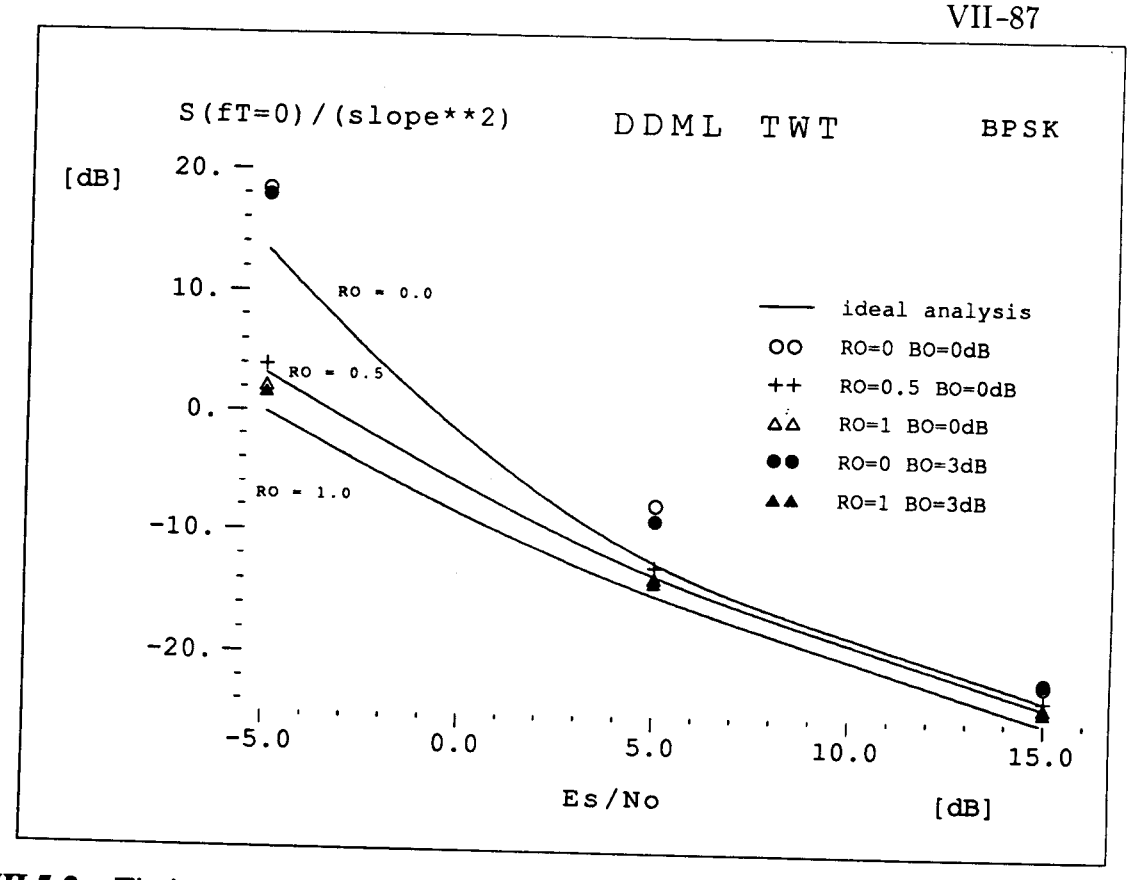


Fig. VII.5-2a: Timing: DDMLFB normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\tau\tau}$) for BPSK with TWT as a function of E_S/N_0

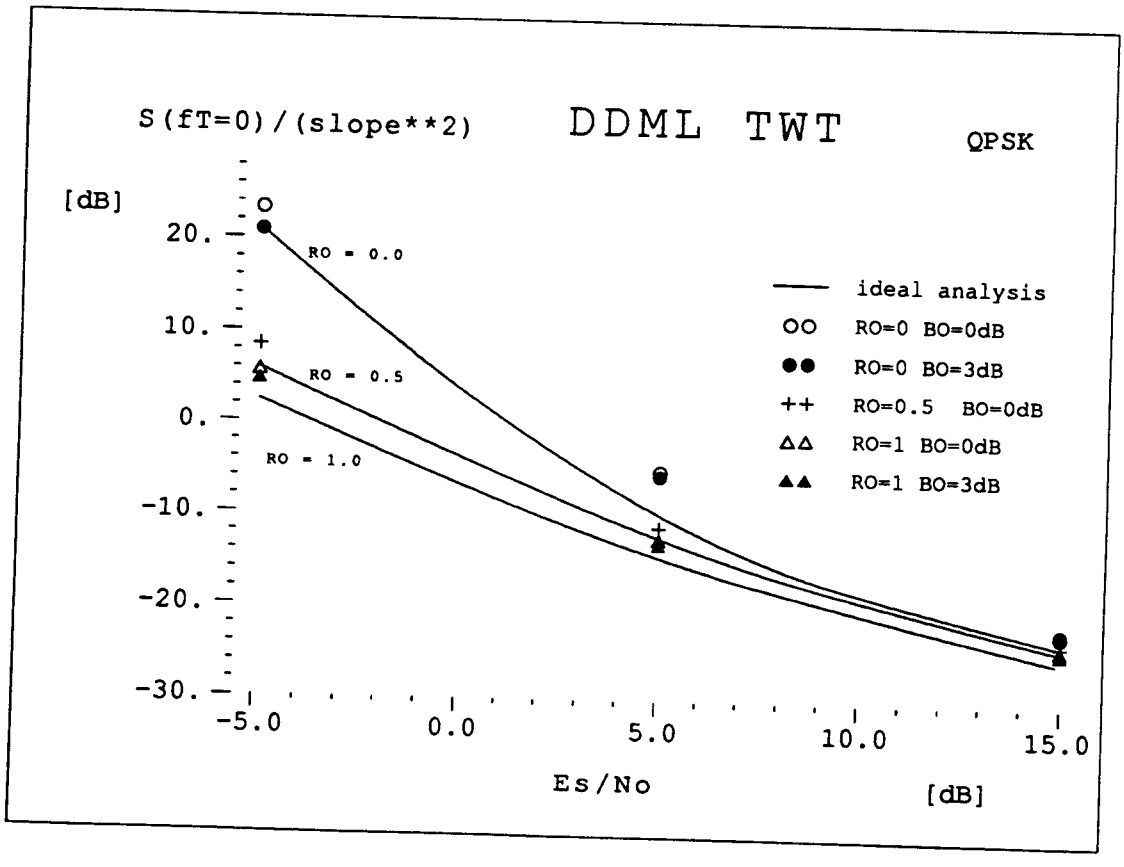


Fig. VII.5-2b: Timing: DDMLFB normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\tau\tau}$) for QPSK with TWT as a function of E_S/N_0

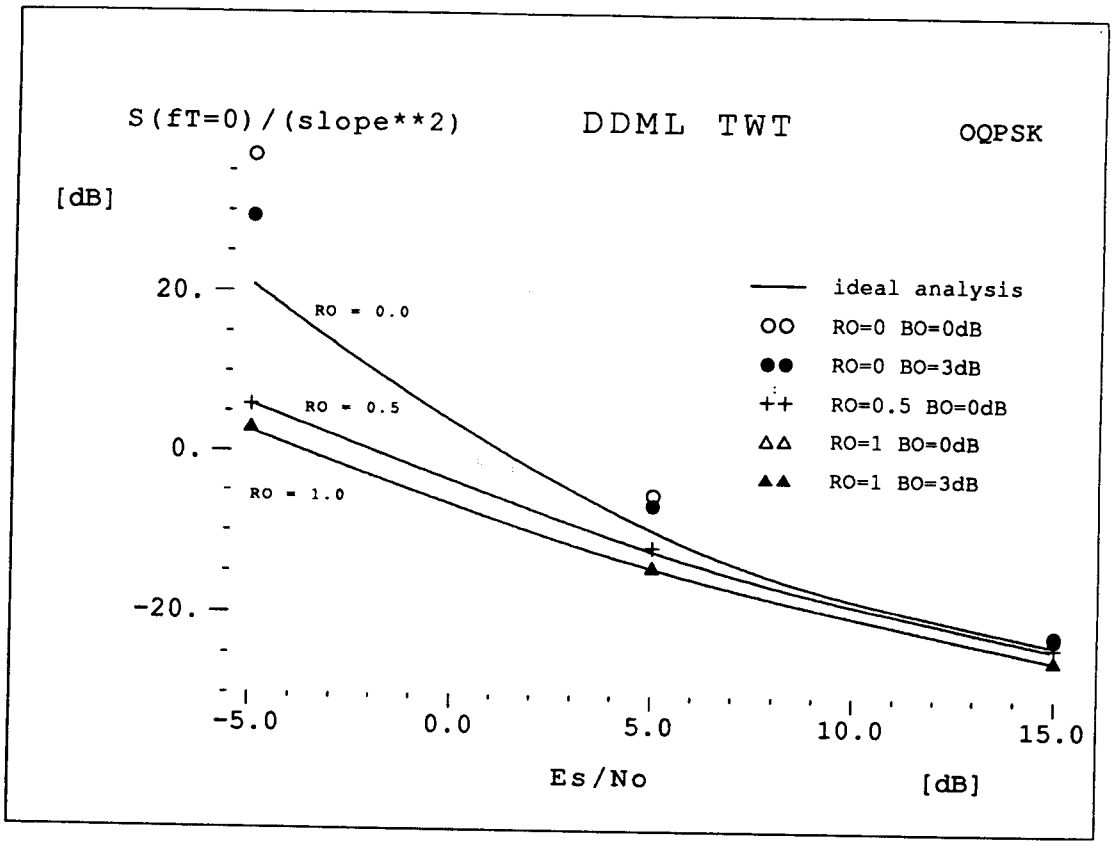


Fig. VII.5-2c: Timing: DDMLFB normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\tau\tau}$) for OQPSK with TWT as a function of E_S/N_0

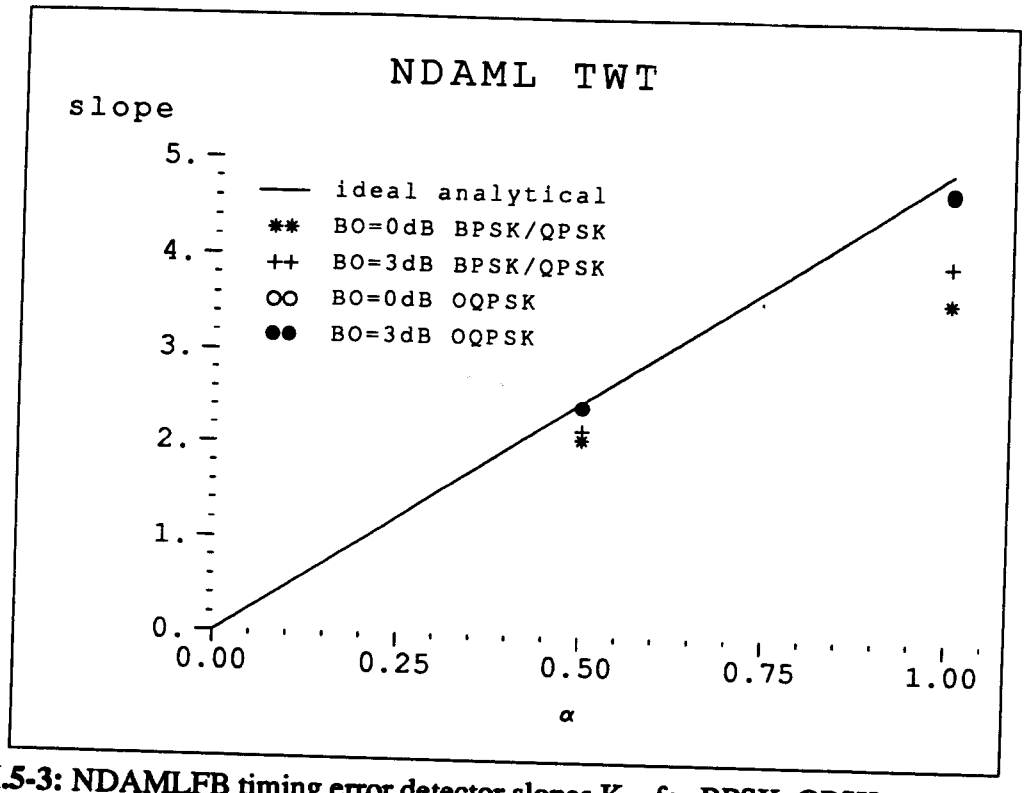


Fig. VII.5-3: NDAMLFB timing error detector slopes $K_{\tau\tau}$ for BPSK, QPSK, and OQPSK with TWT as a function of the rolloff

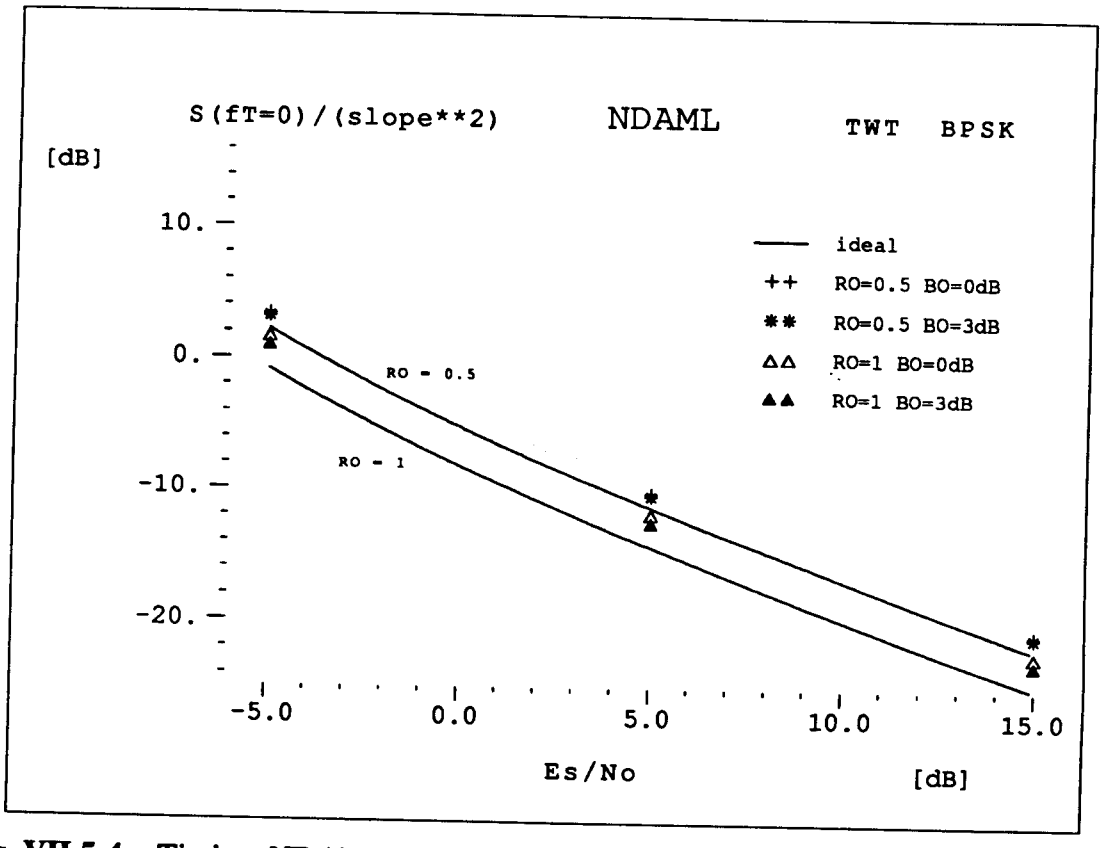


Fig. VII.5-4a: Timing: NDAMLFB normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\tau\tau}$) for BPSK with TWT as a function of E_S/N_0

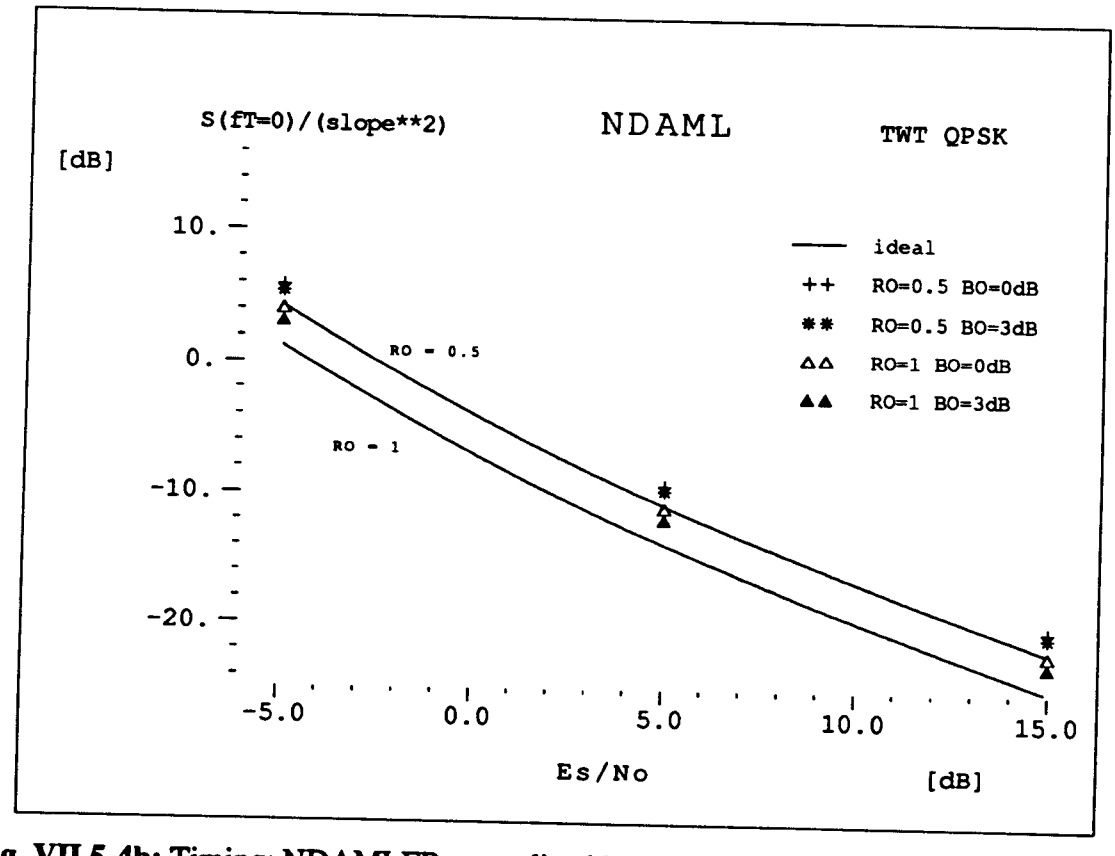


Fig. VII.5-4b: Timing: NDAMLFB normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\tau\tau}$) for QPSK with TWT as a function of E_S/N_0

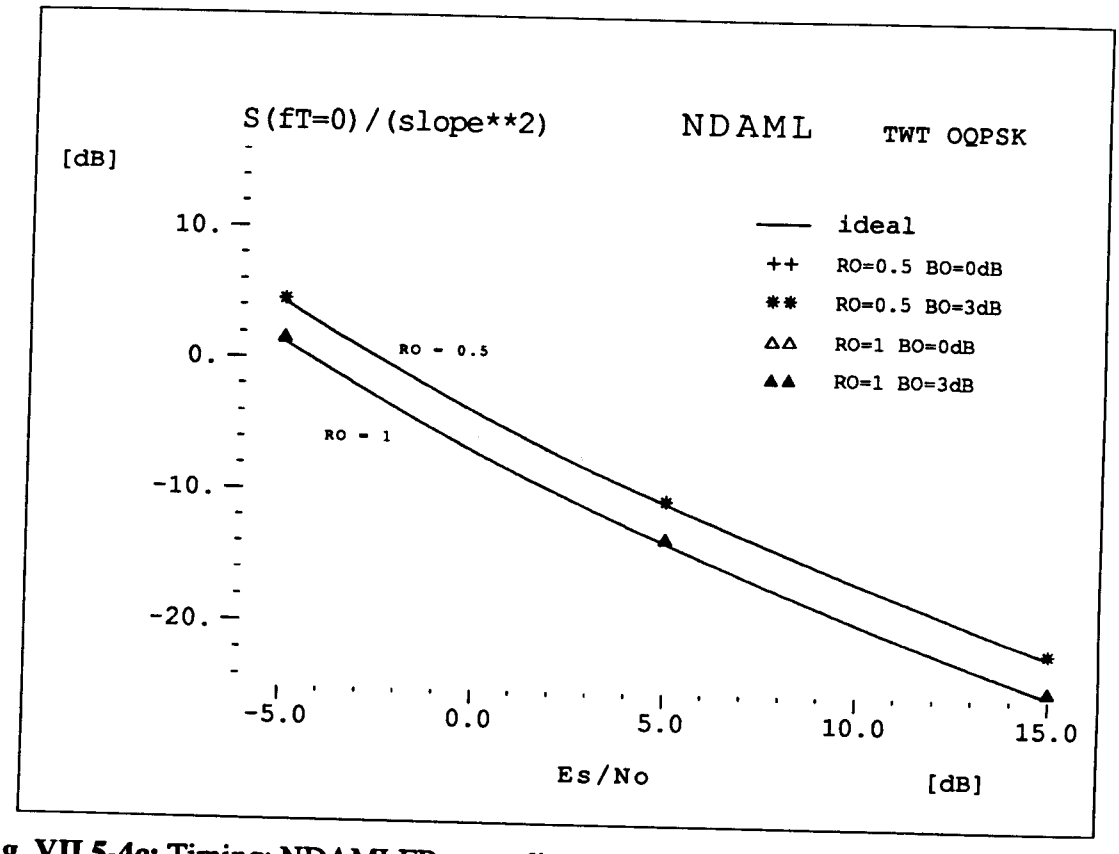


Fig. VII.5-4c: Timing: NDAMLFB normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\tau\tau}$) for OQPSK with TWT as a function of E_S/N_0

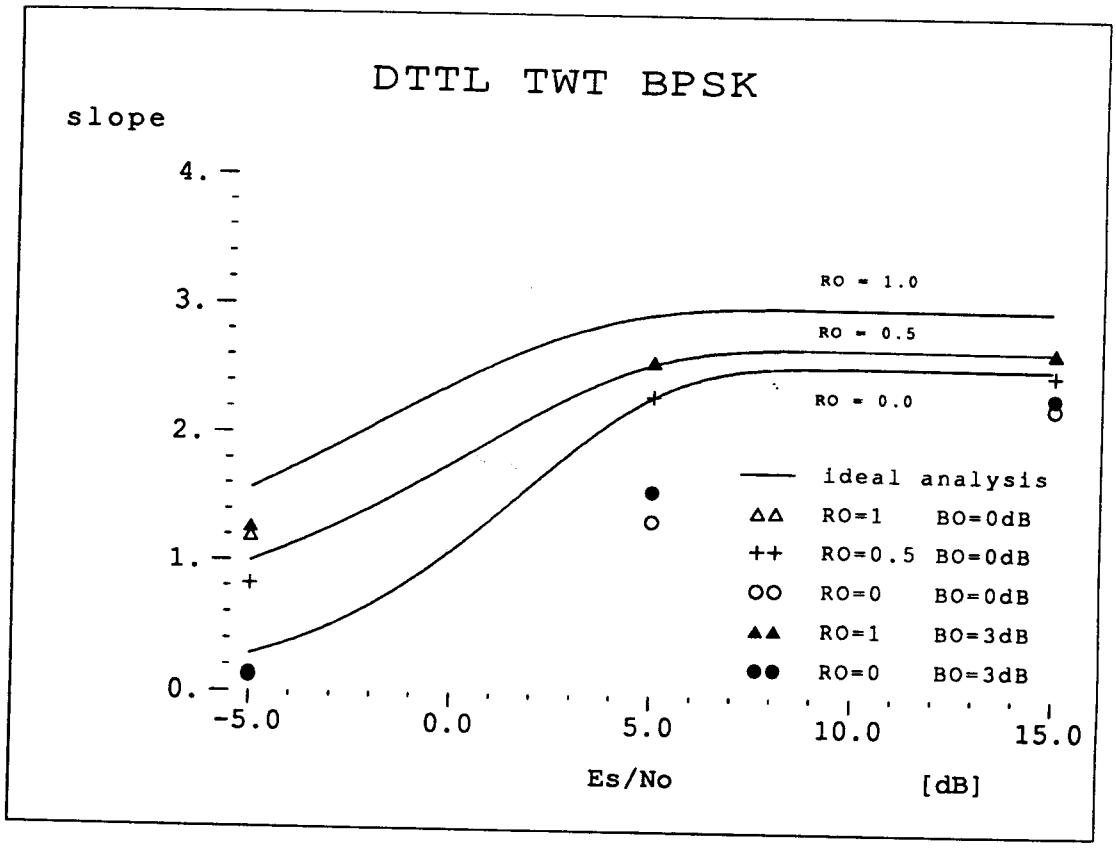


Fig. VII.5-5a: DTTL timing error detector slopes $K_{\tau\tau}$ for BPSK with TWT as a function of E_S/N_0

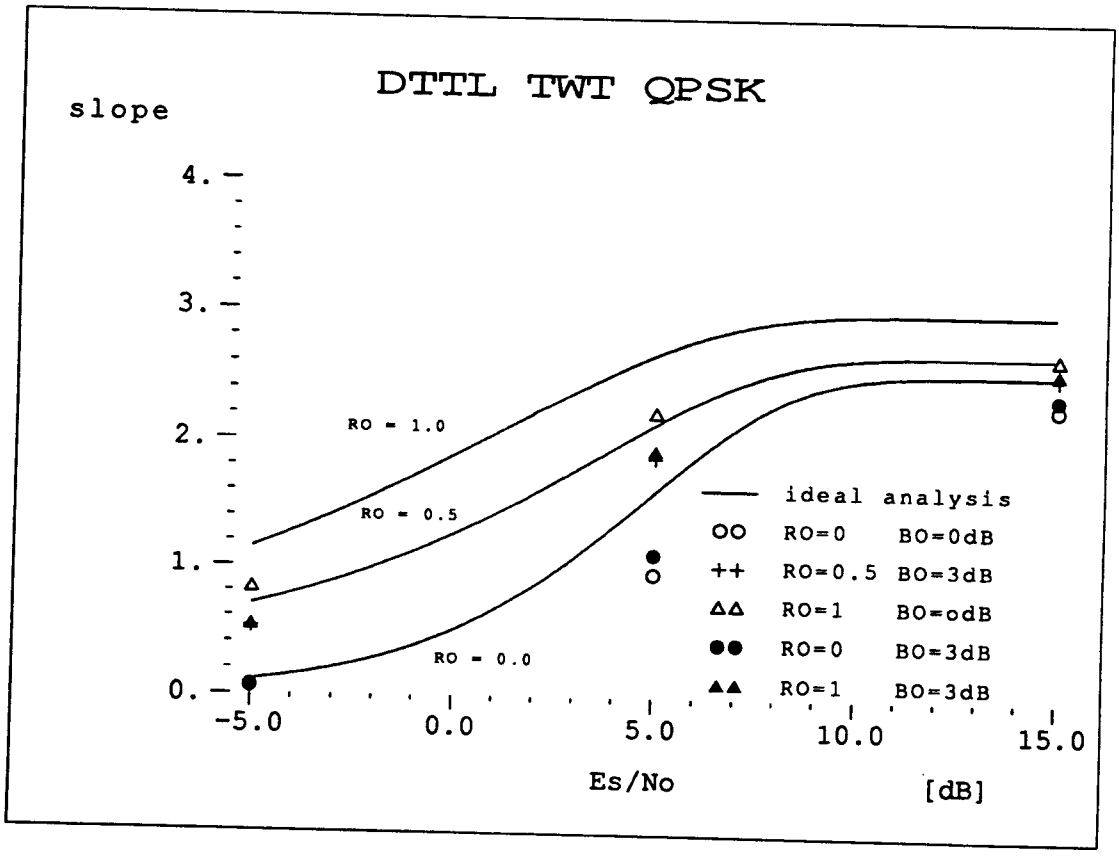


Fig. VII.5-5b: DTTL timing error detector slopes $K_{\tau\tau}$ for QPSK with TWT as a function of E_S/N_0

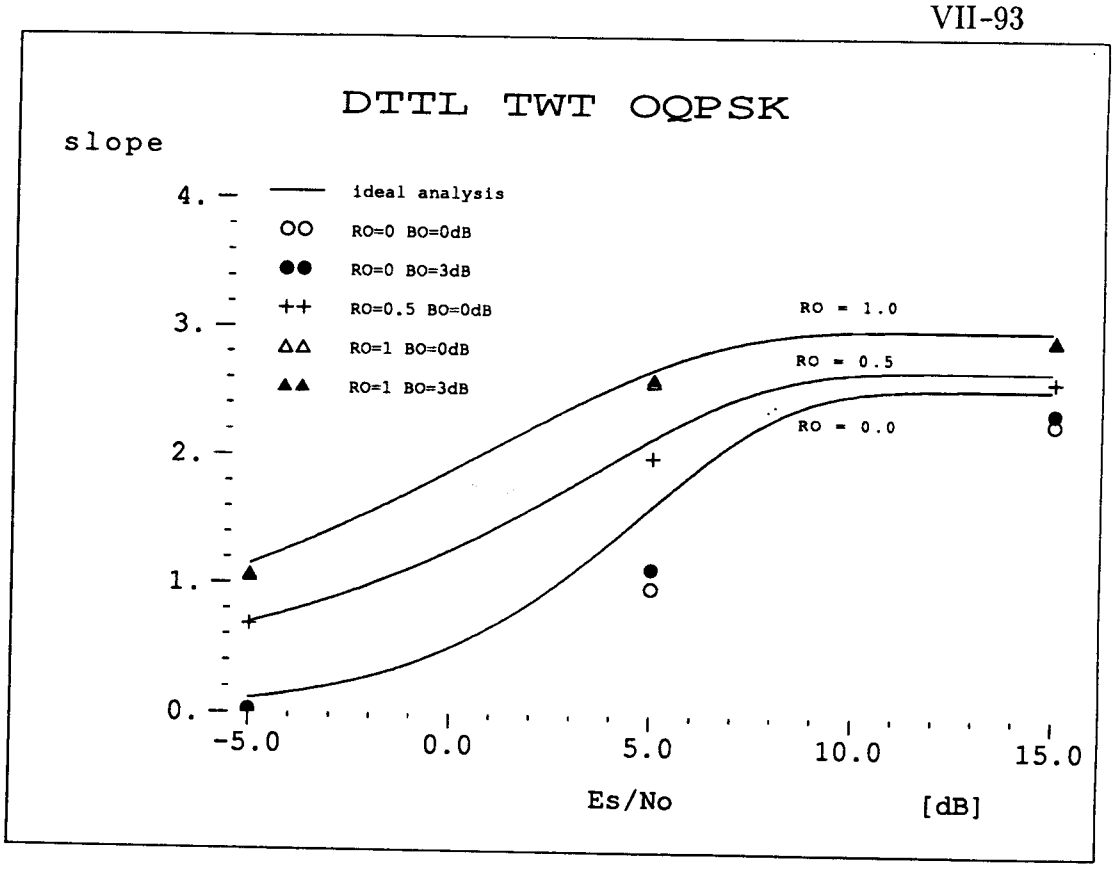


Fig. VII.5-5c: DTTL timing error detector slopes $K_{\tau\tau}$ for OQPSK with TWT as a function of E_S/N_0

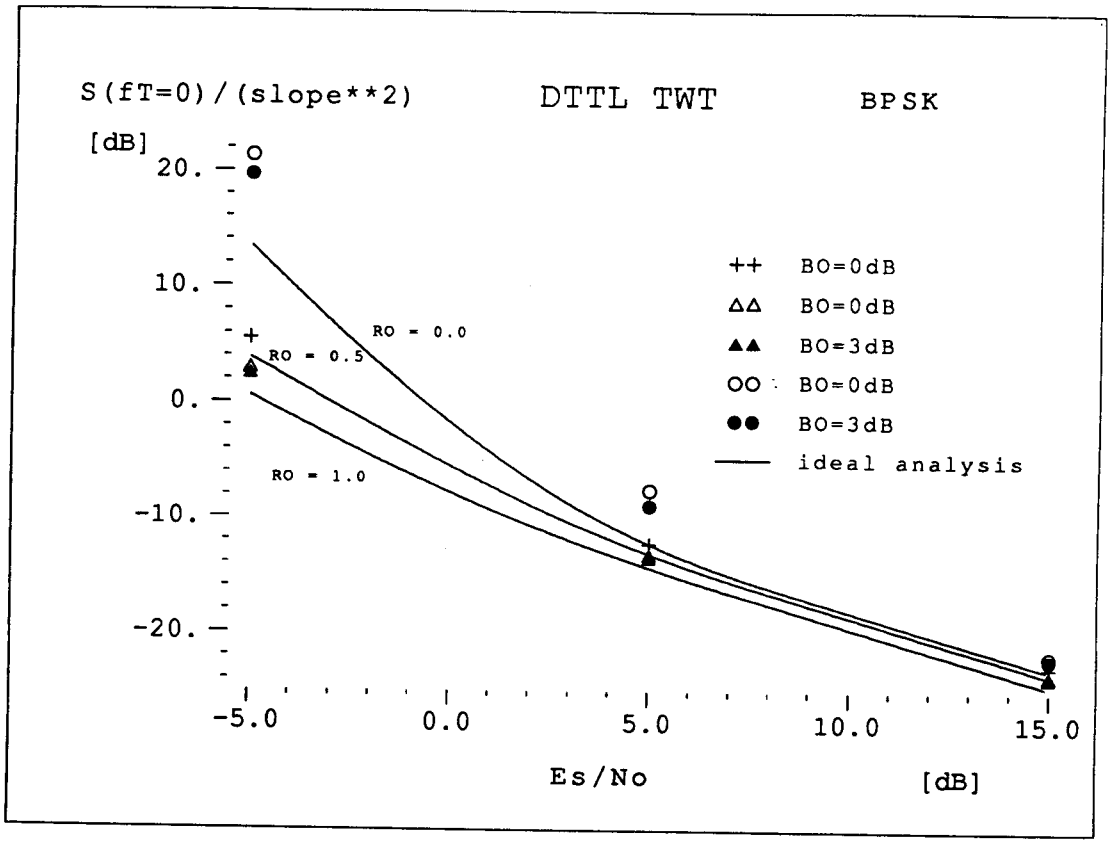


Fig. VII.5-6a: Timing: DTTL normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\tau\tau}$) for BPSK with TWT as a function of E_S/N_0

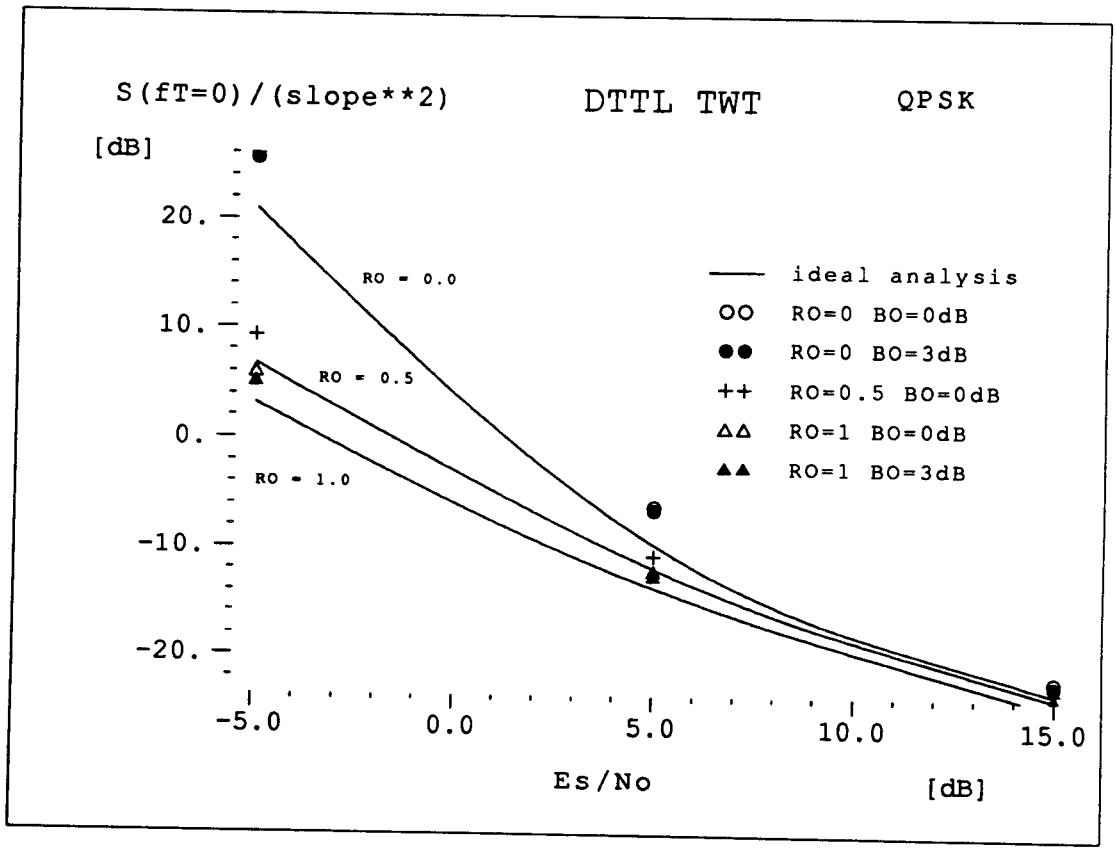


Fig. VII.5-6b: Timing: DTTL normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\tau\tau}$) for QPSK with TWT as a function of E_S/N_0

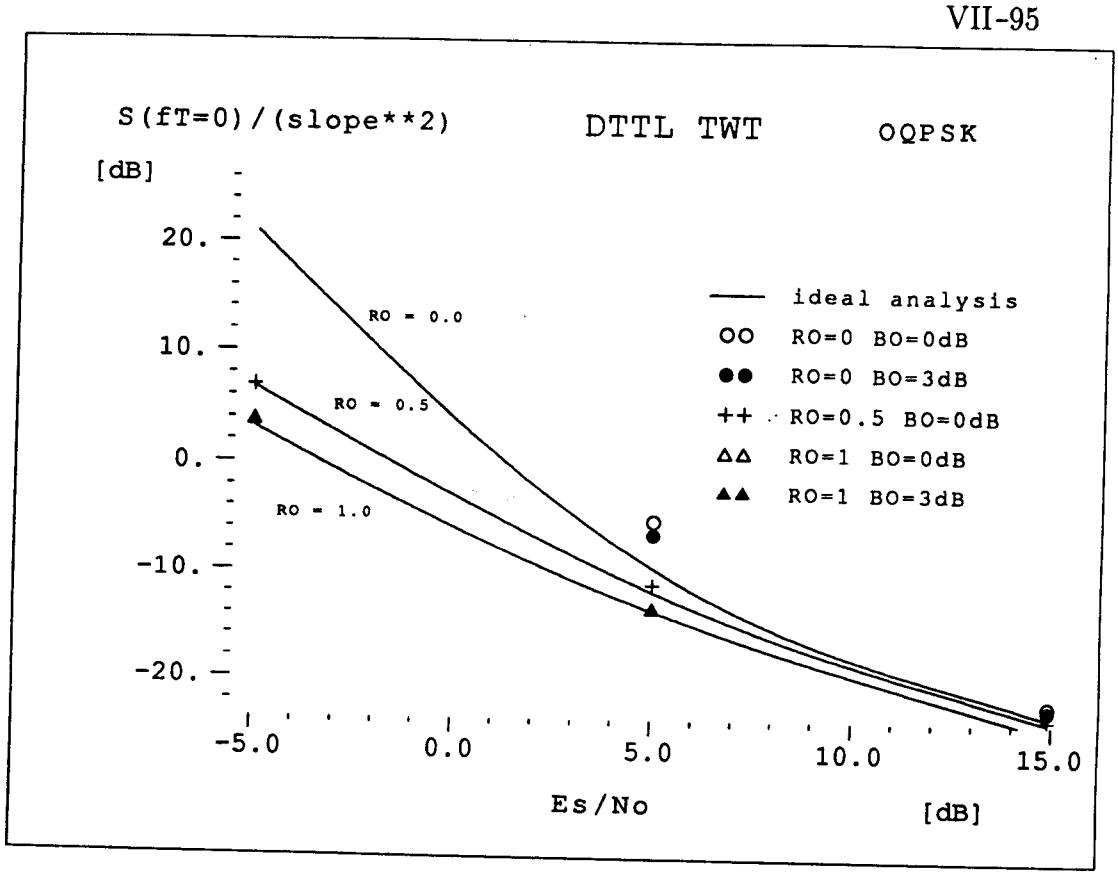


Fig. VII.5-6c: Timing: DTTL normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\tau\tau}$) for OQPSK with TWT as a function of E_S/N_0

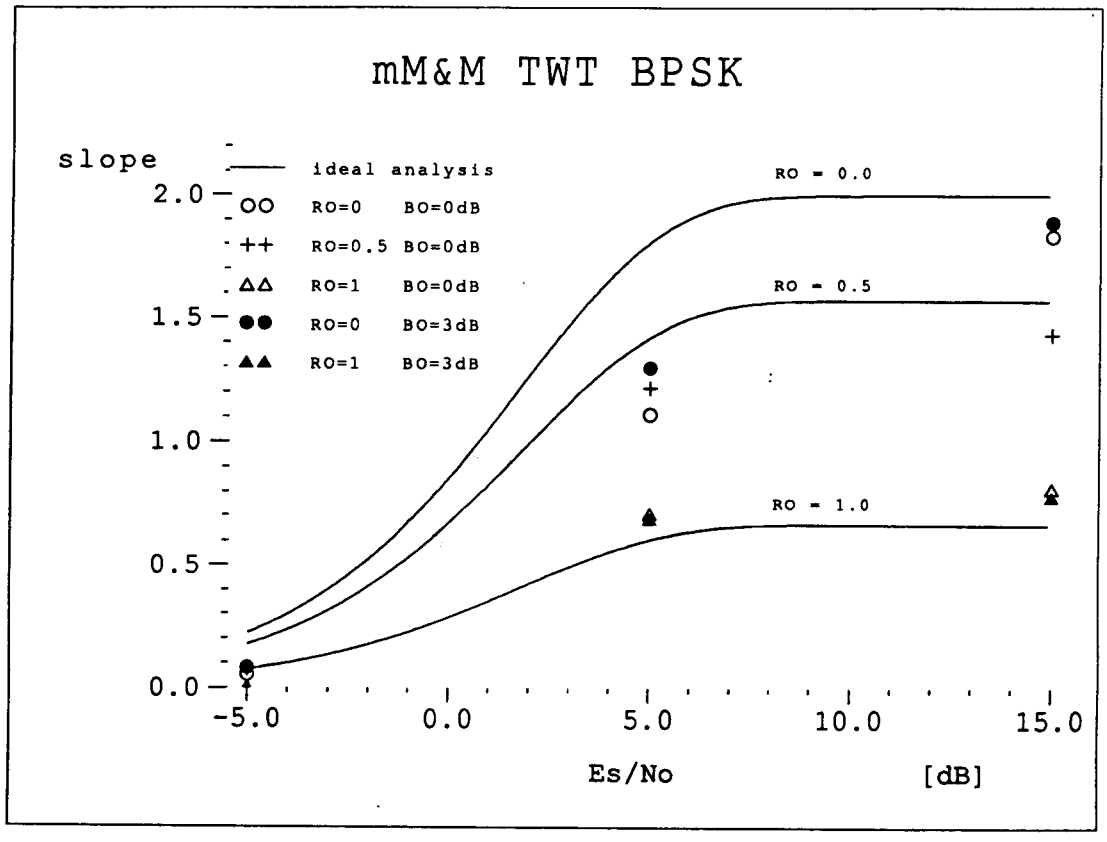


Fig. VII.5-7a: mM&M timing error detector slopes $K_{\tau\tau}$ for BPSK with TWT as a function of E_S/N_0

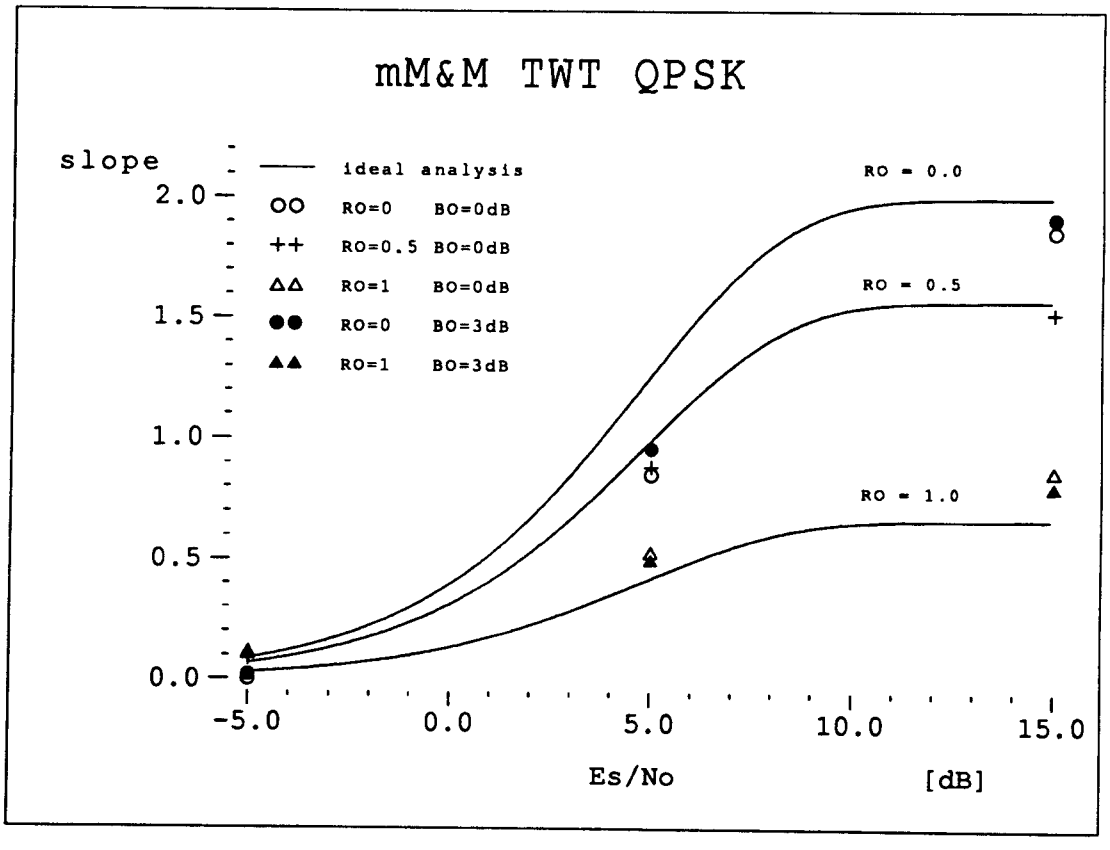


Fig. VII.5-7b: mM&M timing error detector slopes $K_{\tau\tau}$ for QPSK with TWT as a function of E_S/N_0

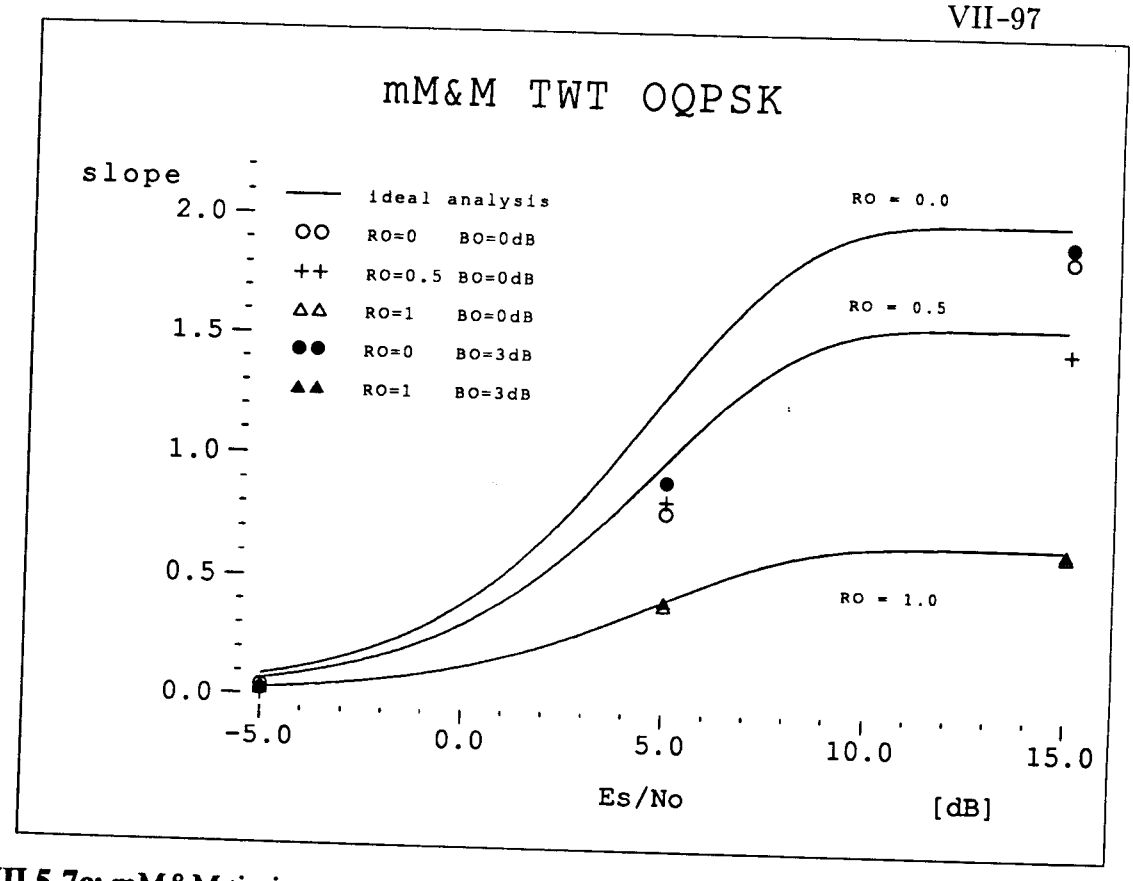


Fig. VII.5-7c: mM&M timing error detector slopes $K_{\tau\tau}$ for BPSK with TWT as a function of E_S/N_0

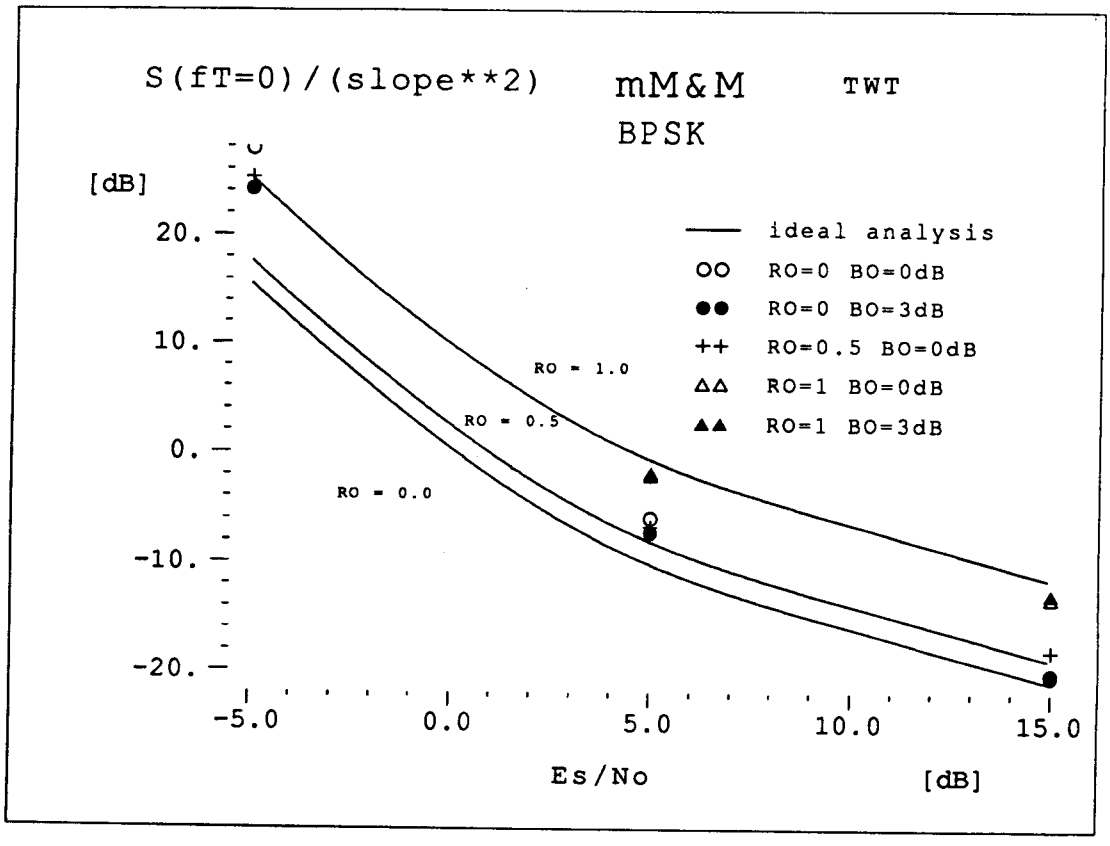


Fig. VII.5-8a: Timing: mM&M normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\tau\tau}$) for BPSK with TWT as a function of E_S/N_0

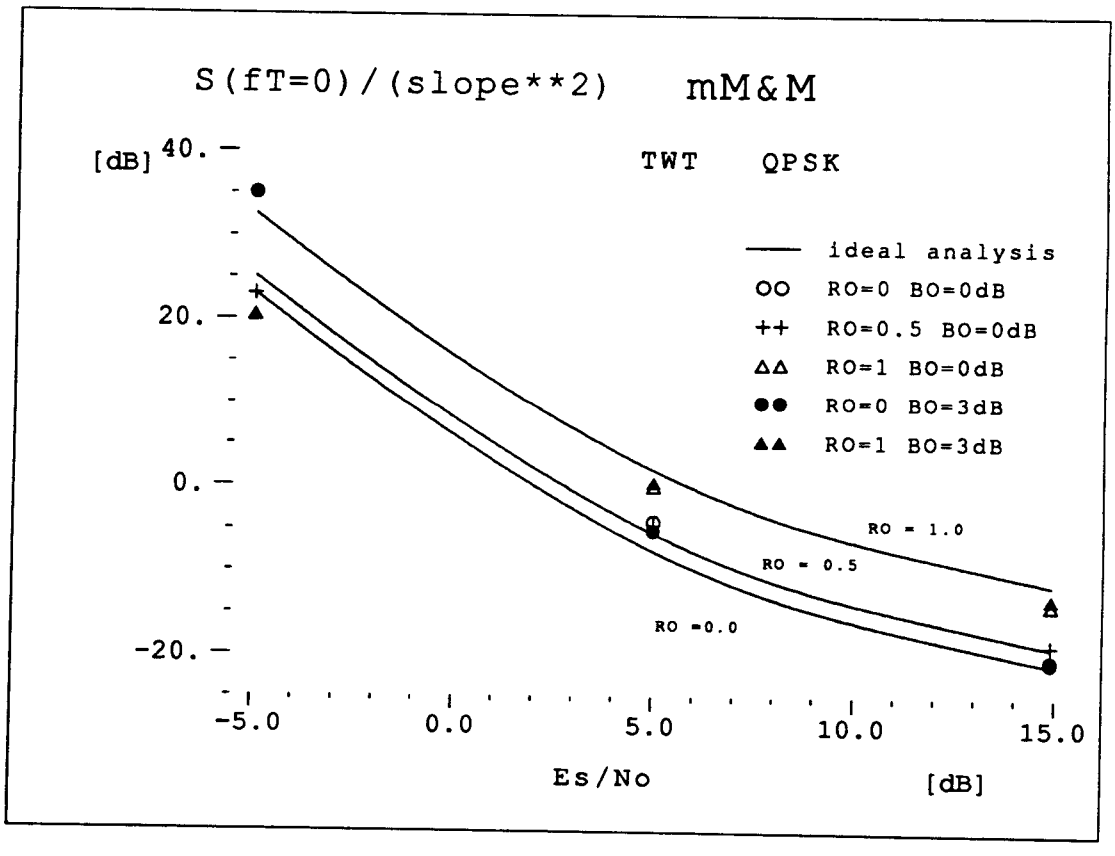


Fig. VII.5-8b: Timing: mM&M normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\tau\tau}$) for QPSK with TWT as a function of E_S/N_0

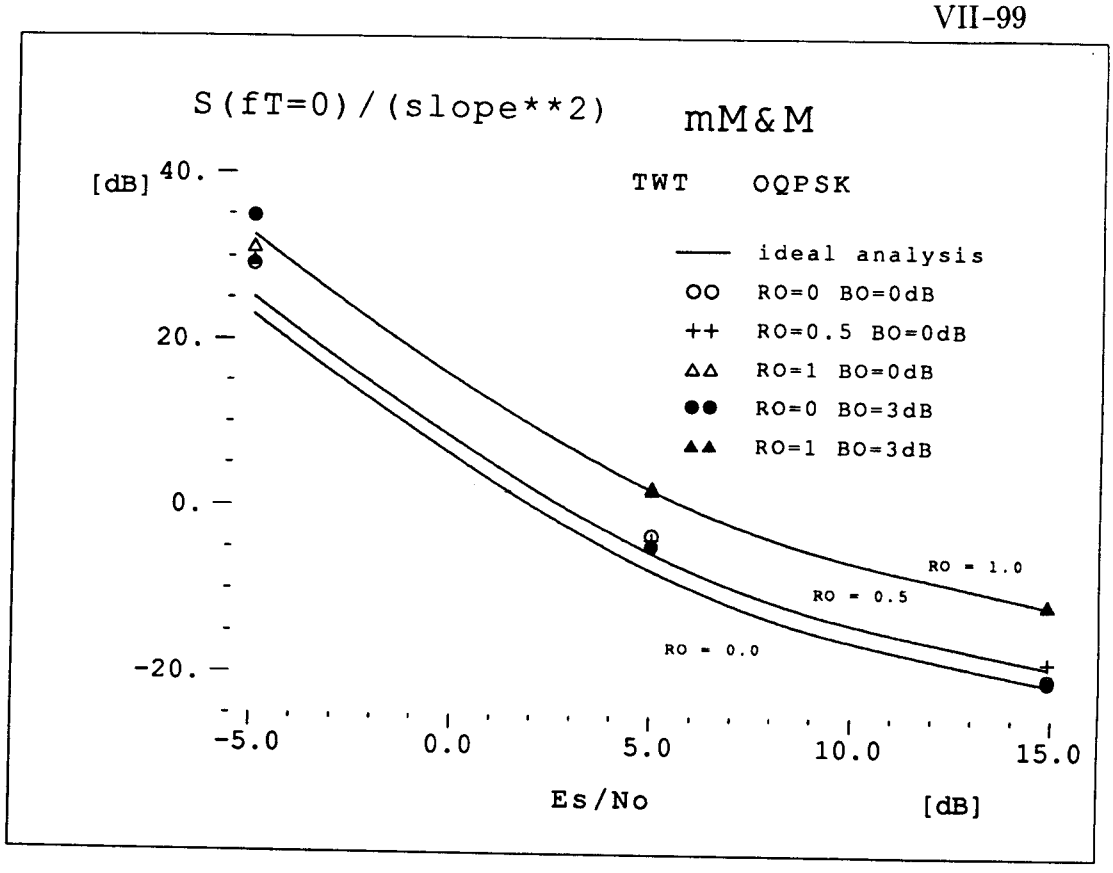


Fig. VII.5-8c: Timing: mM&M normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\tau\tau}$) for OQPSK with TWT as a function of E_S/N_0

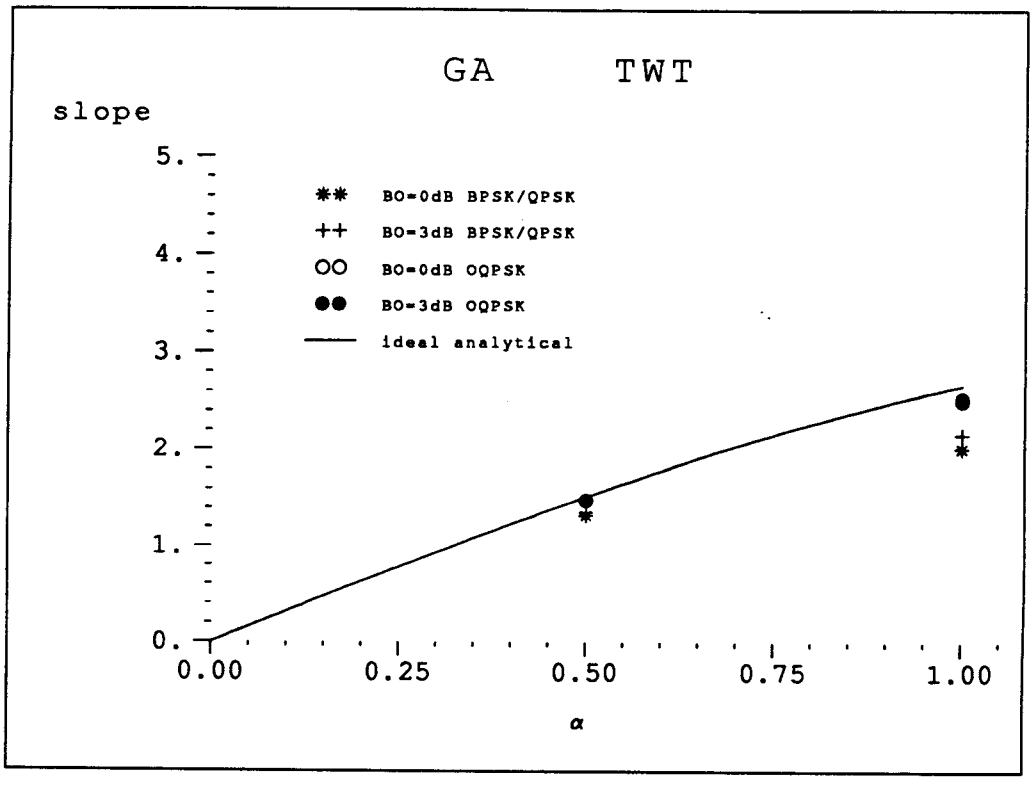


Fig. VII.5-9a: GA timing error detector slopes $K_{\tau\tau}$ for BPSK with TWT as a function of E_S/N_0

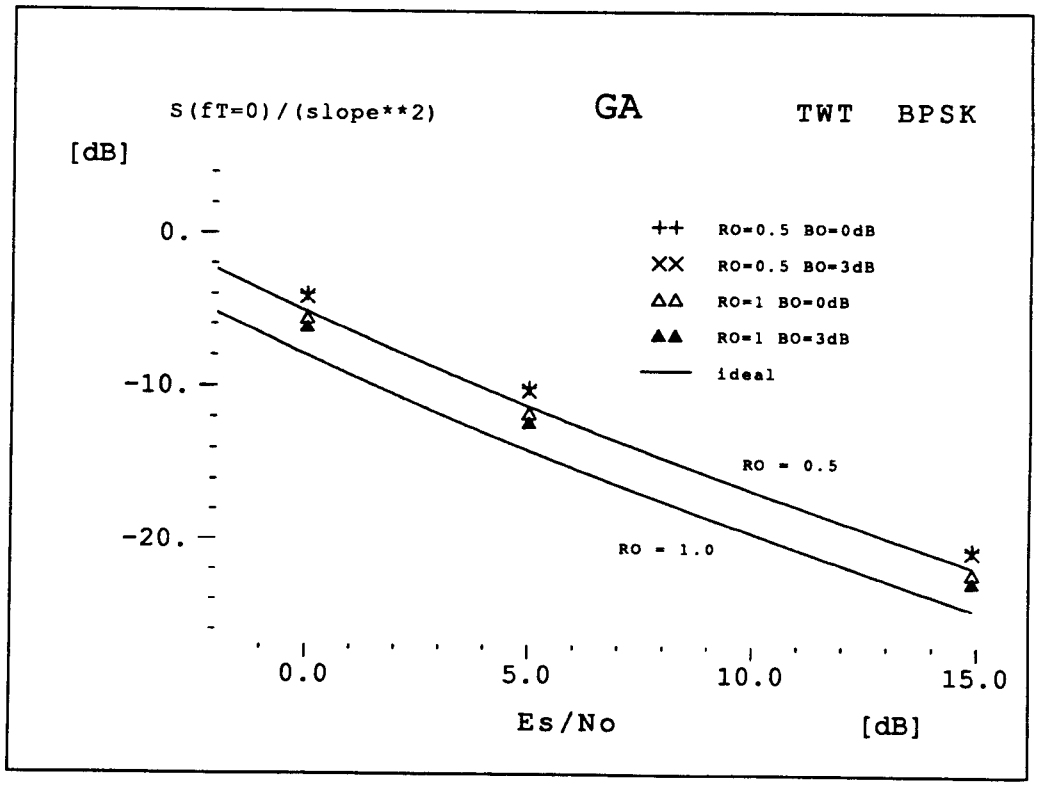


Fig. VII.5-9b: Timing: GA normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\tau\tau}$) for BPSK with TWT as a function of E_S/N_0

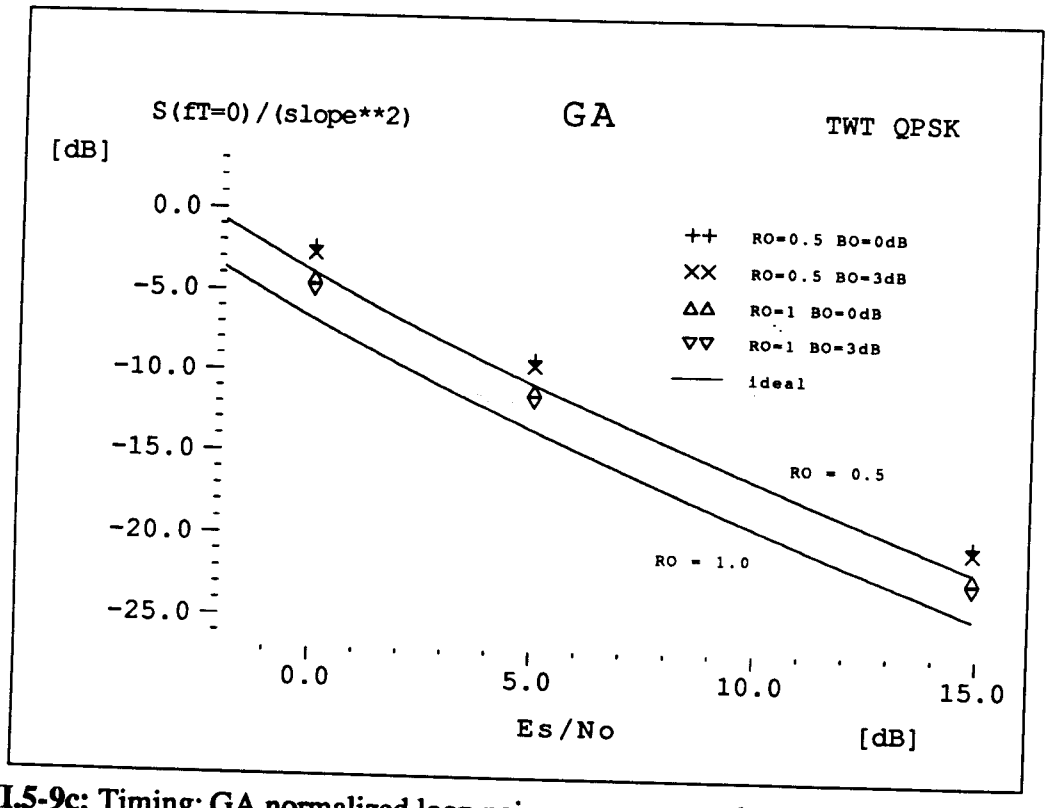


Fig. VII.5-9c: Timing: GA normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\tau\tau}$) for QPSK with TWT as a function of E_S/N_0

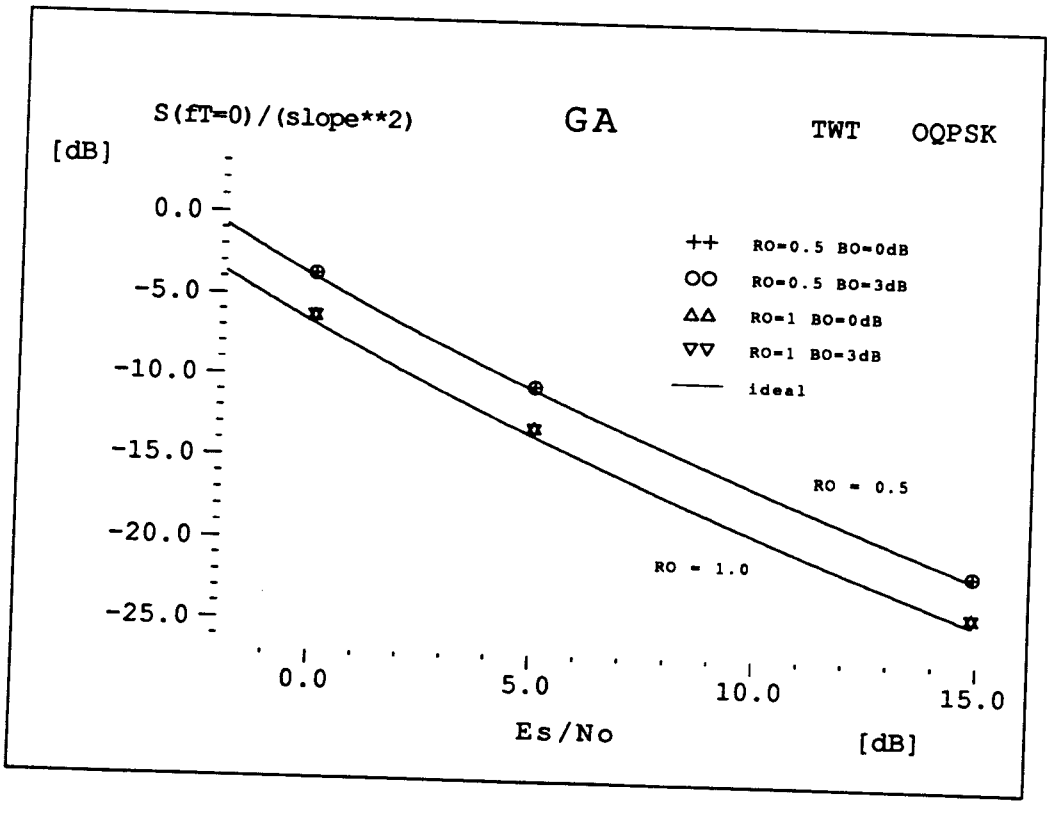


Fig. VII.5-9d: Timing: GA normalized loop noise power spectral density at fT=0 ($S(fT=0)/K^2_{\tau\tau}$) for OQPSK with TWT as a function of E_S/N_0

VIII. INFLUENCE OF DIGITAL SIGNAL PROCESSING ON TRACKING PERFORMANCE

VIII.1 Preliminaries

In this part of the report we will study the influence of implementation aspects on synchronizer performance. Of course, it is not only the synchronizer but also all the preceding components, i.e. HF, IF-filters, amplifiers, down converters etc., that determine the overall receiver performance. On the other hand, from the synchronizers' point of view HF and IF are parts of the transmission channel (except with a long loop which is not considered here) - channel distortions are discussed in part VII. Also, we will neglect hardware imperfections like a d.c. offset in the mixer outputs or the accuracy of the ADC converter levels (integral and differential nonlinearity, etc.).

VIII.1.1 Digital implementation aspects

The basic building blocks of a digital receiver are shown in Fig. VIII.1-1a-d for different synchronizer structures. Although we have quadrature mixers in all versions and thus are in the baseband, most of the results given here also apply for bandpass sampling. A first basic selection we can make is between the structures shown in Fig. VIII.1-1a,b and Fig. VIII.1-1c,d. In the first case we have a feedback synchronizer with analogue (i.e. continuous-time, real valued) carrier phase and timing correction. In the second case we have a feedback or feedforward synchronizer with digital (i.e. sampled, quantized) carrier phase and timing correction. For the digital timning correction we must either have a sufficiently high sampling rate or, better, use an interpolator. Of course combinations like digital carrier phase correction with continuous-time symbol timing correction are possible. A second selection is between a continuous-time matched filter and a digital matched filter, i.e. between the versions Fig. VIII.1-1a,c and Fig. VIII.1-1b,d.

So far, we have seen that feedback and feedforward synchronizers have the same tracking performance under ideal conditions and non-ideal channel conditions. A feedforward synchronizer maximizes a function of the input signal. The corresponding feedback synchronizer adjusts the derivative of this

function (and not of any of the signals) to zero. Therefore, as long as the derivatives with respect to the parameters carrier phase and timing can be linearized around the tracking point, feedback and feedforward performance are identical. Since this is still the case, feedback and feedforward synchronizer performance are identical even with quantization (of the filtered output signal).

But there is a difference when we use the structures in Fig. VIII.1-1a,b or the structures in Fig. VIII.1-1c,d where a digital phase rotator is required to adjust the received signal carrier phase and an interpolator is required to adjust the symbol timing. Therefore the influence of these components has to be studied here as well. The structures in Fig. VIII.1-1a,b are often referred to as hybrid (analog/digital) synchronizers. Although we are mainly interested in a fully digital synchronizer here, we will also refer to the hybrid implementation as reference and as s basis for the study of fully digital synchronizers.

Whether we use a continuous-time or a digital matched filter is not a question of performance since with both solutions we can come arbitrarily close to the optimum. The major advantages of the digital matched filter are that it is not subject to ageing, drift, or tolerances of the components. Also, a digital matched filter can be made programmable so that it is easily adjusted to varying channel conditions or varying data rates, for example. But with a digital matched filter we must have an anti-aliasing filter in front of the analog-to-digital converters (ADC). In some cases, it may be possible to use the intermediate frequency filter as anti-aliasing filter so that in the baseband we only need lowpass filters which supress the mixing frequency and its harmonics. This poses much fewer restrictions on the filter transfer characteristic. The major difference between both solutions is, as far as the implementation aspects studied in this part are concerned, that with a digital matched filter quantization is introduced before the filter and with a continuous-time matched filter quantization is introduced after the matched filter.

In this part we will study the influence of
- quantization before the matched filter,
- quantization after the matched filter,
- matched filter implementation aspects,
- loop filter quantization,
- phase rotator implementation aspects,
- interpolation filter implementation aspects.

A problem is, that analytical performance figures in general cannot be given. Also, when the subsystems are combined to form a synchronizer the overall performance figures cannot be derived by combining the individual performance figures. In particular for coarse quantization the actual performance figures strongly depend on the way the coarse quantized signal is processed. With clever solutions the performance loss can be kept low. Consequently we have not tried to set up 'design figures' from which the performance of any digital synchronizer can be determined by just combining the appropriate performance figures. Rather we
- try to explain basic effects in digital synchronizer components,
- study the sensitivity of the various synchronization algorithms to quantization,
- study the sensitivity of the synchronizer components to quantization,
- show when quantization is critical.

Before we study the influence of digital implementation aspects on carrier and symbol synchronizer tracking performance, in the following subsections we will study common components and their implementation.

VIII.1.2 Digital number representation

The two basic types of number representations are floating point and integer. Since signal processors with floating point arithmetic are available it is possible to implement the algorithms using floating point arithmetic. After AD-conversion the data are converted to floating point representation in the DSP. Since such DSP's mainly use a 32-bit number representation (24 bit mantissa and 8 bit exponent) the floating point performance figures are given by the reference simulations in part II and VII. The computer simulation then can be considered as a digital synchronizer implementation.

A major advantage of floating point representation is the large dynamic range which avoids overflow and also is important to achieve high fidelity, e.g. in audio signal processing, where dynamic ranges of more than 90 dB occur but this is not relevant in the applications considered here. Another advantage which is more relevant is that floating point arithmetic is much less subject to limit cycles.

Since the floating point operations are more complicated, a floating point arithmetic logic unit (ALU) requires a larger chip area on an integrated circuit and/or in general needs more execution time. For the same speed the power consumption in general is higher for the floating point ALU. For instance, the TMS320C25 which is a 16 bit integer DSP has a multiply and accumulate cycle time (MAC) of 100 ns at 0.5 W power consumption. The corresponding 32-bit floating point DSP TMS320C30 has a MAC time of 60 ns at 1 W power consumption.

Advantages of integer DSP's are the availability of the devices in less critical technologies (with respect to e.g. space qualification), a higher maximum MAC speed, and lower minimum power consumption. The advantages of integer signal processing are even more significant if we consider an application specific integrated circuit (ASIC) implementation. In this case the advantages in chip area, power consumption, and maximum speed are very large, in particular, when using very small wordlength. As we will see later in this report, we can make efficient use of small word length integer representations here without significant performance loss.

Of course, there are also different integer number representations. Instead of using one of the different but equivalent representations where, except for the sign digit, each digit corresponds to one power of two, we may use a representation which corresponds to a non-uniform spacing. We will comment on this representation in the next subsection.

VIII.1.3 Analog to digital conversion (ADC)

When studying the influence of the ADC we will neglect any hardware imperfections, i.e. we will assume perfect level accuracy. Here we will only consider the quantization of the ADC. When looking at commercially available analog to digital converters we find that for conversion rates in the

sub-Msamples/s range almost exclusively converters with 8 or more bit are available. But even in the range up to 100 Msamples/s 8 bit converters dominate. Only above 100 Msamples/s more 4 bit converters are found. If we go below 4 bit a discrete implementation becomes feasible, e.g. with a 2 bit quantization we have 3 thresholds. Therefore, a 2 bit AD-converter may be implemented using three comparators. On the other hand, even if we use an 8 bit ADC it may be of interest to process a smaller wordlength, in particular for a hardware solution, e.g. an ASIC, for high data rates. In such a case we could also allow larger power fluctuations of the ADC input signal and insert a digital AGC (automatic gain control) after the ADC which keeps its output signal power constant and appropriately reduces the word length.

When we have selected an ADC with a particular quantization another parameter is left for optimization, the 'linear range' of the ADC. Since all values above the uppermost threshold are mapped to the highest level, we lose all amplitude information for input signal levels above this threshold. Therefore, the corresponding level must not be set too low with respect to the input signal amplitude. On the other hand when we make the linear ADC range too large, higher levels will occur very rarely and, thus, will not be used efficiently.

The first question that arises when we try to determine the optimum range with respect to the input signal is, what is the measure of optimality? In principle, we want to have the best acquisition and tracking performance of carrier and clock synchronizer. But this is not a very useful criterion to optimize the ADC range. Instead, three criteria which are each reasonable from one particular point of view will be presented and discussed.

In the following definitions $x = s + n$ is the input signal of the ADC, where s is the useful signal and n is the additive noise. $p_n(n)$ denotes the probability density function of the noise and $p_s(s)$ the probability density function of the signal. Finally, $Q(x;\xi)$ is the ADC output for an input signal x and a particular parameter set ξ (ξ stands for the parameters which define the quantization characteristic completely, i.e. the quantization levels and the corresponding output levels).

Criterion I: Maximum signal-to-noise ratio (SNR)
Maximizing the signal-to-noise ratio corresponds to minimizing the inverse of the SNR. When we define as useful output signal $s_Q(s)$ the expected value of the ADC output given the input signal s

$$s_Q(s;\xi) = E[Q(s+n;\xi)|s] = \int_{-\infty}^{+\infty} Q(s+n;\xi)\ p_n(n)\ dn$$

the question is what is the ADC output-SNR?
A possible definition is (SNR criterion Ia):

$$Q_{opt}(x) : \min_{\xi} \left\{ \frac{\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} (Q(s+n\,;\,\xi)-s_Q(s;\xi))^2 p_n(n)dn\ p_s(s)ds}{\int_{-\infty}^{+\infty} s_Q^2(s;\xi)\ p_s(s)ds} \right\}$$

(VIII.1.3-1a)

(this expression should be read as follows: $Q_{opt}(x)$ is the quantization characteristic $Q(x;\xi = \xi')$ where ξ' is the value of ξ for which the right hand side takes on its minimum value). In this case we get the quantization characteristic for which the output signal-to-noise ratio is maximized, but we do not know how good a replica of s $s_Q(s)$ is.
Therefore, we may instead minimize (SNR criterion Ib):

$$Q_{opt}(x) : \min_{\xi} \left\{ \frac{\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} (Q(s+n\,;\,\xi)-s)^2 p_n(n)dn\ p_s(s)ds}{\int_{-\infty}^{+\infty} s_Q^2(s;\xi)\ p_s(s)ds} \right\}$$

(VIII.1.3-1b)

Finally, instead of using the power of s_Q as signal power we might use the power of the input signal s (<u>SNR criterion Ic</u>):

$$Q_{opt}(x) : \min_{\xi} \left\{ \frac{\int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} (Q(s+n;\xi)-s)^2 p_n(n)dn \; p_s(s)ds}{\int_{-\infty}^{+\infty} s^2 \; p_s(s)ds} \right\}$$

(VIII.1.3-1c)

<u>Criterion II:</u> Minimum error variance

In this case, we determine the quantization characteristic for which the minimum mean square error between input and output signal is minimized

$$Q_{opt}(x) : \min_{\xi} \left\{ \int_{-\infty}^{+\infty} (Q(x;\xi)-x)^2 \; p_x(x)dx \right\} = \min_{\xi} \sigma_e^2 \qquad \text{(VIII.1.3-2)}$$

i.e. we get the best approximation of signal plus noise.

<u>Criterion III:</u> Minimum signal distortion

Finally, we could minimize the error variance between the expected value of the ADC output $s_Q(s)$ (= useful output signal) and s

$$Q_{opt}(x) : \min_{\xi} \left\{ \int_{-\infty}^{+\infty} (s_Q(s;\xi)-s)^2 \; p_s(s)ds \right\}$$

$$= \min_{\xi} \left\{ \int_{-\infty}^{+\infty} (E[Q(s+n;\xi)|s] - s)^2 \; p_s(s)ds \right\}$$

(VIII.1.3-3)

Note that of course an arbitrary constant gain g_0 in the ADC characteristic does not influence the optimality, i.e. $g_0 Q_{opt}(x)$ also is optimum when $Q_{opt}(x)$ satisfies one of the above optimality criteria.

Discussion

It is interesting to discuss the various criteria in more detail. We will first consider Ia, II and III where we can give an interpretation of the result of the maximization.

Using Ia we obtain the quantization characteristic which yields the maximum output signal-to-noise ratio. As already stated, the difference (e.g. mean square error) between s_Q and s is not minimized in this criterion. Since for further processing s_Q can be considered as the useful signal (and $Q(s+n) - s_Q$ as noise) differences between s_Q and s are waveform distortions which may degrade performance since synchronizer and/or matched filter are optimized for the undistorted waveform.

Criterion III minimizes the mean square error between useful output signal s_Q and input signal s. With this criterion we will obtain the least distorted characteristic $s_Q(s)$ (therefore, this criterion can be of interest for measurement systems). The disadvantage is that the output noise power is not minimized.

By appropriately rewriting Ic it can be shown that this criterion minimizes the sum of ADC output noise power $P_{n,out} = E[(Q(s+n)-s_Q)^2]$ and the mean square error as defined in criterion III. Since the total distortion power, i.e. the sum of noise power and mean square error $E[(s-s_Q)^2]$, is not normalized to the output signal power, we may not obtain the minimum output total-distortion-to-signal-power ratio. It can be easily shown that this minimum ratio is obtained by optimizing criterion Ib, i.e. Ib is the sum of III normalized to the output signal power and Ia. Thus, it can be concluded that Ib is one of the prime candidates for optimization of a synchronizer ADC characteristic.

A second candidate is criterion II. Here we obtain the quantization characteristic for which the mean square error between ADC output and input is minimized. Since the synchronization algorithms have been derived for the ADC input signal plus noise, i.e. the total input signal, it is reasonable to use an ADC characteristic which yields the closest approximation of this signal as output signal.

When we now select a number of $q = 2^B$ ADC output levels, the optimum quantizer can be determined from the appropriate criterion. The resulting quantization characteristic will not be uniform, except for an input signal with uniform distribution.

There are several arguments against non-uniform quantization. Such an optimal quantizer is optimized for a particular case only. When the input signal-to-noise ratio varies a mismatch occurs and, in general, performance will degrade more than with uniform quantization. To give an example in Tab. VIII.1-1 the mean square error between ADC input x and output Q(x)

$$\sigma_e^2 = E[(Q(x)-x)^2]$$

for an input signal with (zero mean) Gaussian distribution is shown (and an ADC characteristic optimized with criterion III). We see that the difference between optimum (non-uniform) and optimum uniform quantization (i.e. when we restrict optimization to uniformly spaced quantization characteristics only) is small. The gain, i.e. the reduction of the quantization noise, increases with the number of bits but then the mean square error itself already is very small. The major advantage of non-uniform spacing is in transmission with high dynamic ranges, e.g. speech transmission. Finally, number representation with non-uniform quantization is more critical and there is a large selection of commercially available uniform quantizing ADCs. Therefore, in this study we only consider uniform quantization. Note that for 1 bit where the ADC corresponds to a hard limiter we cannot distinguish between uniform and non-uniform spacing.

Tab. VIII.1-1: Mean square error σ_e^2 due to quantization for an input signal with zero mean Gaussian distribution

bits	optimum quantizer	optimum uniform quantizer	gain
2	0.1125	0.1188	5.3% (0.2 dB)
3	0.0353	0.0389	9.3% (0.4 dB)
4	0.0095	0.0115	17.0% (0.7 dB)

Before we carry out an optimization we have to discuss the parameters of the quantization characteristics (which so far have been represented by ξ) in more detail. In Fig. VIII.1-2a a quantization characteristic is shown. For a uniform characteristic the step size $\Delta_i = \Delta \neq$ independent of i. When we assume a quantization characteristic continued to $\pm\infty$, we may regard an ADC as a series connection of this infinite quantization characteristic and a soft limiter. The linear range of this soft limiter is $-(V + \Delta)$ to $+(V + \Delta)$, i.e. the soft limiter characteristic is

$$f_{SL}(x) = \begin{cases} V+\Delta & V+\Delta < x \\ x & -(V+\Delta) < x < +(V+\Delta) \\ -(V+\Delta) & x < -(V+\Delta) \end{cases} \qquad \text{(VIII.1.3-4)}$$

Note that $-(V + 3\Delta/2)$ to $+(V + \Delta/2)$ often is referred to as full scale range (FSR). Here, we will refer to V as the overload level and further use the normalized overload level $\rho = V/\sqrt{P_i}$ where $P_i = P_s + P_n$ is the total input signal power. Now, for uniform quantization, given input signal and noise distributions, and q bits the only parameter left for optimization is ρ. A method to determine ρ is for instance described in [MAX'60]

The optimum values of ρ have been determined for a sinusoidal input signal plus additive Gaussian noise. Coarse quantization, i.e. 2, 3, and 4 bit, was studied only.

Criterion Ic always yields very small values for ρ. But then we have a significant compression of large signal amplitudes resulting in strong signal waveform distortions. On the other hand, with criterion III we get very large values of ρ for low signal-to-noise ratios, so that limiting effects occur rarely. This yields the best characteristic $s_Q(s)$ at the cost of a reduced signal-to-noise ratio. At medium or large signal-to-noise ratios (i.e. 10 - 20 dB) the values of ρ for criterion III decrease, thus, approaching the values for criterion II. For criterion Ic we have small values of ρ for low signal-to-noise ratios where the minimization of the noise power dominates. At medium or large signal-to-noise ratios where we have an increasing influence of the mean square signal distortion on optimization, the values of ρ for criterion Ic increase, thus, approaching the values for criterion II.

The values of ρ which optimize Ib or II are always between the extreme cases Ia, Ic and III. Significant differences between the values for Ib and II only occur where the influence of quantization is low, i.e. for 3 or 4 bit quantization and an input SNR below 10 dB, with smaller values of ρ for criterion Ib. Where quantization is critical, i.e. a small number of levels and large SNR, the differences between the optimum ρ values obtained from criteria Ib and II are marginal. To give numerical examples, for 2 bit quantization and 10 dB SNR we have $\rho_{opt,Ib} \approx 0.8$ and $\rho_{opt,II} \approx 0.85$. With 3 bit quantization and 0 dB input SNR the difference is larger. The optimum values are for criterion Ib: $\rho_{opt,Ib} \approx 1.2$ and for criterion II: $\rho_{opt,II} \approx 1.65$. The corresponding values of the output total-distortion-to-signal power ratio as used in (VIII.1.3-1b) are 0.98 for $\rho = 1.2$ and 1.02 for $\rho = 1.65$. This is a ratio (but not necessarily synchronizer performance loss) of approximately 0.2 dB between the two cases. The mean square error σ_e^2 as used in (VIII.1.3-2) is 0.056 for $\rho = 1.2$ and 0.032 for $\rho = 1.65$ which in both cases is small compared to the noise variance (≈ 0.5).

To summarize, both criteria, Ib and II, can be used to obtain good estimates for the optimum ρ (if necessary followed by a 'fine tuning' for optimum synchronizer performance). Here we have used criterion II. Although, no rigorous comparative study was carried out, it was verified by variation of quantization characteristics that with this criterion the performance was always in the vicinity of the optimum.

As reference in Fig. VIII.1-2b the optimum ρ as a function of the sample signal-to-noise ratio is shown for 2, 3 and 4 bit quantization. As input signal an unmodulated carrier with a frequency deviation is assumed, i.e. we have a sinusoidal signal in inphase and quadrature channel. In Fig. VIII.1-2b we observe that for low number of bits the variation of ρ with the signal-to-noise ratio (SNR) is small. Thus, with an automatic level control device which keeps its output signal power constant (noncoherent AGC with squaring detector) we always operate close to the optimum even with varying input signal-to-noise ratio.

It is difficult to determine the influence of the quantization on the synchronizer performance. But there are some important and useful rules of thumb. In Fig. VIII.1-2c the mean square error (VIII.1.3-2) between input

and output signal assuming the optimum values shown in Fig. VIII.1-2b is shown. For low signal to noise ratios and not too coarse quantization, the influence of the quantization can be considered as an additive uniformly distributed noise with variance

$$\sigma_e^2 = \frac{\Delta^2}{12}$$

(which is the variance of a random variable uniformly distributed in $[-\Delta/2,+\Delta/2]$). Also, when consecutive noise samples are uncorrelated the quantization noise is approximately uncorrelated. In this case quantization may be considered as an additive white noise with power spectral density

$$S_{n_q}(f) = \frac{\Delta^2}{12} T_s$$

Using the normalized frequency variable fT we have

$$S_{n_q}(fT) = \frac{\Delta^2}{12} (T/T_s)^{-1}$$

where T/T_s is the number of samples per symbol. For high signal-to-noise ratios this is no longer true. The correlation of the quantization noise depends on the signal correlation and also the noise is no longer uniformly distributed.

In addition to quantization, we have to take into account the clipping effect, which may be modelled as a soft limiter (see (VIII.1.3-4)). Like a soft limiter this causes signal supression. When we have a large signal plus noise, the mean of the ADC output will be smaller than the input signal.

In Fig. VIII.1-3a,b,c the mean of the ADC output, i.e. the useful signal $s_Q = E_n[Q(s+n)]$, is shown as a function of s for 2, 3, and 4 bit ADC. The noise variance (power) is 0.5, 0.0909, and 0.0099. Assuming that the nominal signal power is such that the total power (noise plus signal power) is unity, this corresponds to SNR = 0 dB, 10dB, and 20dB. This is readily verified from

$$SNR = \frac{E[s^2]}{E[n^2]} = \frac{P_s}{P_n} = \frac{1 - P_n}{P_n} \quad => \quad P_n = \frac{1}{1 + SNR}$$

Thus, we have $P_n(SNR=0dB) = 0.5$, $P_n(SNR=10dB) = 0.0909$, and $P_n(SNR=20dB) = 0.0099$. The values of ρ corresponding to these SNR's were taken from Fig. VIII.1-2b, i.e. according to criterion II. We find that the signal distortion is small for a 3 bit ADC when SNR $\leq$ 10 dB and for a 4 bit ADC. With a 2 bit ADC signal distortions are small for very low SNR only.

In addition, in Fig. VIII.1-4a,b,c the ADC output noise variance ($P_{n,out}$) is shown for the same conditions as in Fig. VIII.1-3a,b,c. Note that now in a figure the curves for 2 bit, 3 bit, and 4 bit quantization are shown for one input SNR. For 2 bit quantization the variance is increased significantly except for SNR = 0 dB (P_n= 0.5) where we only have a slight increase. For SNR = 20 dB 3 bit quantization becomes critical as well while 4 bit quantization is uncritical for all SNR shown.

It looks somewhat surprising that the output noise variance becomes smaller than the input noise variance for large signal amplitudes (note that in these cases even the output SNR is larger than the input SNR). This "noise supression" is not in contradiction to theory as can be demonstrated easily: assume that the ADC always outputs a value s_1. Then, the output noise variance is zero for an arbitrary input noise variance when the input signal has the amplitude s_1.

From previous discussions we may draw some conclusions on the interaction of quantization and sampling rate. Assume that the sampling rate and the antialiasing filter bandwidth (and, thus, the noise bandwidth) are increased. For a given E_s/N_0 value the sample signal-to-noise ratio then decreases inversely proportional to the sampling rate. This reduces the influence of the quantization noise and the E_s/N_0 value, for which the quantization error variance influences the performance significantly, is shifted towards larger values (3 dB per doubling of the sampling rate). In addition, the quantization noise power now is spread over a wider frequency band according to the increased sampling rate so that the power spectral density also is reduced. Assuming an approximately white quantization noise spectrum the power spectral density is reduced by 3 dB per doubling of the

sampling rate. Therefore, in a given synchronizer loop noise bandwidth we now have less quantization noise power.

In the E_s/N_0 region where the quantization effects influence the performance significantly an increasing correlation between quantization errors occurs, the power spectral density then deviates from a white spectrum. Thus, the achievable gain is reduced because a larger portion of the power spectrum falls into the loop noise bandwidth.

VIII.1.4 Digital filters

In this section we discuss digital matched filters, i.e. we have the situation shown in Fig. VIII.1-1b,d. The two ADC's are preceeded by antialiasing filters, the matched filters are implemented using digital signal processing after the A/D conversion. In contrast to the loop filters the digital matched filters replace filters defined in the continuous-time domain, i.e. the digital matched filter output should be (at least approximately) identical to the sampled and quantized ideal continuous-time matched filter output.

Two effects are important in such digital filters, a limited impulse response length when we use an FIR-implementation and quantization of signal and coefficients. Since infinite impulse response (IIR) filters are more critical with respect to quantization effects (limit cycles) often a finite impulse response (FIR) filter implementation is preferred. Since the number of FIR taps which determines the length of the impulse response also determines the implementation complexity, short filter lengths are preferred. This also means, that the desired characteristic is met only approximately. For the digital matched filter the mismatch effect corresponds to a a linear distortion on the channel. Therefore, we will not consider this effect here explicitly, but rather refer the reader to part VII on linear distortions.

The second source of distortions is the quantization. This is a very important topic in filter design. Of course, it is possible to simulate for instance a filter with quantized coefficients. But this would give a wrong impression of the performance capabilities of a digital matched filter. There are various efficient methods to reduce coefficient quantization effects without getting a high computational complexity. This will be demonstrated by an example. Assume that we use 8 bit coefficients where at most 3 bits

are allowed to be 1. Then, a multiplication of the signal with a coefficient might be implemented by two adders as shown in Fig. VIII.1-5. A multiplication by K = 133 corresponds to adding the input data to a version of the input data shifted by two bit (multiplication by $2^2 = 4$) and by seven bit (multiplication by $2^7 = 128$) as shown on the left-hand side in Fig. VIII.1-5. On the right-hand side in Fig. VIII.1-5 the general case of a multiplication by an arbitrary constant K is shown which can be represented by a digital number differing from 0 in 3 bit positions only. When we have an 8 bit coefficient representation, where only 2 coefficients are allowed to be one:

$$K = c_1 2^{n_1} + c_0 2^{n_0} \quad ; \quad c_1 \in [0,1] \quad ; \quad n_0 < n_1 < 8$$

36 coefficient values are possible and multiplication corresponds to a single addition (similarly we have 92 coefficient values when at most 3 of 8 bits are allowed to be one).

The digital matched filter design is out of the scope of this study, therefore we will not discuss it here In the simulation results given in this part of the report we have used the same matched filters as in parts II and VII. For quantization after the matched filter the simulation setup is obvious. For quantization before the matched filter we have converted the integer output of the ADC to floating point representation and then again used the same matched filters as in the previous parts. Thus, quantization and truncation in the matched filter is excluded from consideration. We purely observe the quantization effects due to the ADC.

VIII.1.5 Loop filter, NCO

In contrast to the digital filters discussed in the previous section the loop filters should not be designed to approximate the behaviour of a particular continuous-time filter. For a digital implementation very efficient filtering algorithms can be derived in feedback and feedforward synchronizer structures, which yield better performance under nonlinear operating conditions (i.e. during acquisition) than common analogue loop filters. Here we are concerned with the tracking mode, i.e. linearization applies. In this situation we can restrict our attention to a few basic building blocks from

which most of the appropriate digital loop filters can be built. Note that there is no principle difference in the derivation of filters for carrier and timing synchronizers. To avoid confusion by neutral or double indices we only use carrier synchronization as reference in the discussions.

Feedforward synchronization

Here we may have two different situations. When the system is operating in burst mode and carrier frequency is very stable, we can assume that we have a small frequency error in the complex baseband signal only. Then, it may be sufficient to estimate the carrier phase by an accumulator filter

$$y(k) = \frac{1}{M} \sum_{m=k-M+1}^{k} x(m)$$

A method to compensate a frequency deviation in continuous time operation (that can also be applied to burst mode when the frequency fluctuations are slow compared to the burst repetition rate) is, to predict the carrier phase based on a frequency estimate and use the prediction error for updating the frequency estimate. Using the approach discussed in appendix IV.A.1 we have the general filter recursion

$$\tilde{x}_{k+1|k} = \tilde{x}_{k|k-1} + K_k \, [\hat{\theta}(k) - H^T \tilde{x}_{k+1|k}]$$

and in the second order case disussed in section IV.2.1 we have

$$\tilde{\theta}(k+1|k) = \tilde{\theta}(k|k-1) + \tilde{\Omega}(k|k-1) + K_1(k) \, [\hat{\theta}(k) - \tilde{\theta}(k|k-1)]$$

$$\tilde{\Omega}(k+1|k) = \tilde{\Omega}(k|k-1) + K_2(k) \, [\hat{\theta}(k) - \tilde{\theta}(k|k-1)] \qquad \text{(VIII.1.5-1a)}$$

with $K_1(k)$, $K_2(k)$ being the time varying filter coefficients.

Feedback

In equation (VIII.1.5-1a) we update a phase and frequency estimates using a prediction error $\hat{\theta}(k) - \tilde{\theta}(k|k-1)$. In a feedback system this error signal is readily available as the phase detector output. To be precise, we only have

a function of the phase error available but in the tracking mode, when linearization applies, this phase detector output signal is approximately proportional to the phase error. The corresponding feedback loop filter equation therefore is given by (see also section IV.2.1, eq. IV.2-11)

$$\tilde{x}_{k+1|k} = \tilde{x}_{k|k-1} + K_k \, u_\theta(k;\tilde{\theta}(k|k-1))$$

For the second oder case we have

$$\tilde{\theta}(k+1|k) = \tilde{\theta}(k|k-1) + \tilde{\Omega}(k|k-1) + K_1(k) \, u_\theta(k;\tilde{\theta}(k|k-1))$$

$$\tilde{\Omega}(k+1|k) = \tilde{\Omega}(k|k-1) + K_2(k) \, u_\theta(k;\tilde{\theta}(k|k-1)) \qquad \text{(VIII.1.5-1b)}$$

It can be easily shown that this filter can be implemented by a filter of the type

$$F_\theta(z) = \sum_{i=0}^{N} C_i \left[\frac{1}{z-1} \right]^i \qquad \text{(VIII.1.5-2)}$$

(the structure is shown in Fig. VIII.1-6) followed by a filter of the type

$$F_{VCO}(z) = \frac{1}{z-1} \qquad \text{(VIII.1.5-3)}$$

Note that for first and second order loops $C_i = K_i$ (where K_i are the elements of K), for higher order loops the relation is more complicated. We could have incorporated filter equation (VIII.1.5-3) directly in (VIII.1.5-2) by letting $C_0 = 0$. The reason for separating this part is that it implements the discrete time equivalent of an integrator. In a hybrid synchronizer we have a DAC controlled VCO or a NCO. For both devices, the input signal controls the frequency (or equivalently the phase increment from a sample to the next sample $\Delta\tilde{\theta}(k) = \tilde{\theta}(k+1|k) - \tilde{\theta}(k|k-1)$) so that VCO or NCO correspond to an integrator with respect to the phase. Thus, in a hybrid synchronizer the VCO or NCO operation corresponds to (VIII.1.5-3) and in a synchronizer with digital carrier (or timing) error correction, this integrator is added to the loop filter.

For quantization effects, we can restrict our considerations to the accumulator filter, to the filter (VIII.1.5-2) and the integrator equivalent (VIII.1.5-3). Although integrator equivalents (VIII.1.5-3) are also basic building blocks of the filter (VIII.1.5-2) some additional comments will be needed.

Now in the filter algorithms presented so far, we have 5 quantization effects:

1. the wordlength of the input signal, i.e. the error detector signal
2. the wordlength of the filter coefficients
3. the wordlength of the registers (the blocks denoted z^{-1} in Fig. VIII.1-6)
4. intermediate output wordlength, i.e. the wordlength passed to the subsequent integrator block
5. output wordlength, i.e. the wordlength used for control of VCO, NCO, phase rotator or interpolator.

For the input signal wordlength we may either use the wordlength as provided by the corresponding error detector or we use the wordlength according to the rule that the quantization noise still is negligible or at least acceptably small. The latter is of interest for feedback synchronizers where processing may increase the wordlength significantly (e.g. NDAML carrier synchronizer for QPSK) and for feedforward synchronizers.

For the coefficients we may use a similar strategy as described in the previous section, see for instance Fig. VIII.1-5. The effect of quantization can be described readily. The actual loop filter transfer function is the one obtained when we insert the quantized coefficients C_{iQ} instead of the ideal coefficients C_i in eq. (VIII.1-2). This means, we do not have a real quantization error we rather have a limited set of transfer functions to select from.

Register wordlength truncation corresponds to limitation of the maximum and minimum value that can be represented in the filter. When we add a new input signal to a register value, the least significant bit (LSB) of the input value should match the LSB of the register value. When the LSB of the input value corresponds to a higher bit in the register wordlength the lower bits of the register word will never change which is not reasonable.

When the LSB of the register corresponds to a higher bit of the input word we have the same effect as with a truncation of the input signal before accumulation. When we have overflow (note that we assume that overflow is handled such that the maximum value is processed further and underflow such that the minimum value is processed further) this corresponds to a limitation effect. This limit in general should be set such that it is not exceeded during normal operation or that its influence does not degrade performance. For instance, in a second order loop the register length in the loop filter should be such that the maximum frequency deviation the system is supposed to follow can be represented.

For the intermediate output wordlength, i.e. the wordlength that is used as input to the next register or to the VCO, NCO, or filter (VIII.1.5-3), the analysis is more difficult. Of course, the only reasonable wordlength limitation is truncation or rounding of lower bits since otherwise we could have limited the previous register wordlength already. The quantization noise still can be assumed to be uniformly distributed but consecutive quantization noise samples no longer are uncorrelated but this is not helpful for analysis.

More important is that this truncation or rounding will give rise to limit cycles. The feedback loop filter itself does not exhibit limit cycles when overflow is detected and handled appropriately. The reason is that we have a very simple type of feedback in the filter, but we have an error feedback via the correction unit, e.g. phase rotator, VCO, or NCO. As an example we discuss the effect of truncation or rounding of the loop filter output. All correcting units are controlled by a digitally represented phase increment (i.e. a frequency). For a VCO this is the DAC input word which determines the frequency of the VCO. The NCO is controlled directly by a digitally represented phase increment. Phase rotator or interpolator are preceeded by a filter of the type (VIII.1.5-3) as discussed above, which holds the phase (or timing) increment. When the input signal frequency now is such that the corresponding phase increment between two consecutive samples is between two possible values for the digital phase increment we will have oscillations of the digital phase increment around this value. In the noise free case we will have periodic oscillations with a basic frequency that depends on the difference between the input phase increment and the next representable digital phase increment. It is well-known that it is hard to tackle limit cycles analytically. We may give a rule of thumb when we use a closed loop transfer function without excessive peaking (which may amplify limit cycles

at the peak frequency). When the phase increment quantization is such that within the closed loop time constant the accumulated phase error is small compared to the noise induced phase error variance, the periodic oscillations will be covered by random fluctuations. A more thorough analysis is required when the spectral purity of the recovered carrier is of importance, e.g. for retransmission. Note that the situation is similar for the feedforward synchronizer filter (VIII.1.5-1a) where we feed back the prediction of the phase multiplied by small constants K_i. Here the phase increment is the sum of the phase prediction error and the frequency prediction ($\tilde{\Omega}(k|k-1) = \tilde{\omega}(k|k-1)T_s$).

For the accumulator filter the situation is much easier. When the input to the accumulator filter can be x_{max} at most then the value of the accumulator after adding M input values can be at most $M \cdot x_{max}$. Thus, the accumulator wordlength must be at most input wordlength + ld(M) (ld: logarithm to the base 2). For the output wordlength the same design guide line as for the feedback loop filter applies. Since we have a pure feedforward structure we do not have limit cycles. Using the design guide line given above for the accumulator wordlength even overflow control is not necessary.

VIII.1.6 Simulation Study

In sections VIII.2 and VIII.3 we analyse the influence of quantization on carrier and clock synchronization algorithms. We have seen in the section on A/D conversion (VIII.1.3) that we have not only an increase of the variance due to the quantization, but also a signal distortion, i.e. the ADC characteristic $s_Q(s)$ deviates from a linear function. When the quantization is sufficiently fine for a given E_S/N_0, e.g. 3 bit for $E_S/N_0 \leq 10$ dB and 4 bit for E_S/N_0 up to 20 dB and above, the quantization effect can be modelled as an additive white noise with variance $\sigma_q^2 = \Delta^2/12$, where Δ is the ADC step size. For coarse quantization we will have an influence of the signal distortion due to the ADC on the synchronization algorithm performance.

We have to consider two different cases. With a continuous time matched filter the ADC is located after the matched filter as shown in Fig. VIII.1-1c. With a digital matched filter the ADC is located before the

matched filter as shown in Fig. VIII.1-1d. In both cases we may have additional processing between the ADC and phase/timing synchronization. For instance, when we want to study the influence of A/D conversion on the carrier phase synchronizers which assume timing to be known we in addition have an interpolator as shown in Fig. VIII.1-7a and Fig. VIII.1-7b. The dashed lines show the alternatives for feedback and feedforward synchronization.

Since performance figures cannot be given analytically we have to study the influence of A/D conversion on synchronization algorithms by simulation. To separate the influence of A/D conversion from any quantization effect in the matched filter and/or the interpolator, floating point arithmetic has been used after the ADC.

In sections VIII.2 and VIII.3 we will present simulation results for the case where the ADC is after the matched filter and for the case that the ADC is before the matched filter. Since we study the effect of quantization on synchronization algorithms separately from the influence of the other components we may treat the latter as ideal. In this case the performance measure of interest is, as in part II and VII, the normalized loop noise power spectral density at $f = 0$. The slopes of the synchronizer curves are not given separately because they are misleading with quantization. The reason is, that we have used the optimized level setting for each E_S/N_0. In an actual system the ADC levels are usually optimized for the most critical case. With an AGC which keeps the total output power constant, independent of the input signal power, we do not have an optimized level setting for signal-to-noise ratios above the most critical (lowest) E_S/N_0.

Matched filter output quantization

The normalized loop noise power spectral density at $f = 0$ has been determined by simulation for selected carrier and clock synchronization algorithms with an ADC after the matched filter. The ADC output is connected directly to the synchronizer input, i.e. without interpolator or phase rotator in between. Also, synchronized sampling ($\epsilon = 0$) and zero carrier phase ($\theta = 0$) have been assumed. Since we put the signal in a 'favourable' position with respect to synchronization and detection, the corresponding results are too optimistic, giving the best case. But they serve as a lower bound for digital carrier phase and timing correction. The results

can be applied directly to a hybrid synchronizer with feedback timing and phase correction as shown in Fig. VIII.1-1a,b. Also, the figures give the approximate performance loss for coarsely quantized arithmetic in feedback synchronizers (i.e. introducing wordlength truncation after the phase rotator and the interpolator, respectively).

Quantization before the matched filter

The normalized loop noise power spectral density at $f = 0$ has been determined assuming an ADC at the matched filter input as shown in Fig. VIII.1-7b in principle. To separate the influence of AD-conversion from any quantization effect in the further processing, floating point arithmetic is used after the ADC. Again we have assumed synchronized sampling ($\epsilon = 0$) and zero phase ($\theta = 0$). Also, the matched filter output has been connected directly to the synchronizer under evaluation, i.e. a phase rotator or an interpolator where not included in the simulated systems. Neglecting the influence of interpolator and phase rotator is in line with the approach of studying the influence of the individual components separately assuming ideal performance for the remaining components.

Impact of synchronized sampling (ϵ=0) and zero phase (θ=0)

For a complete performance analysis it would be of interest to determine the performance as a function of phase θ and timing error ϵ. The difficulty is, that we already have various parameters like rolloff, different modulation schemes for which the performance of the various synchronizers is given as a function of E_S/N_0. With this we run into the problem of combinatorial explosion. Therefore we have to reduce the number of cases to be analyzed. An interesting solution would have been to provide best and worst cases. The problem is to find the corresponding phase and timing errors. They even might depend on the signal-to-noise ratio. Finally, we will have frequency differences rather than fixed phase and timing error. Then, the synchronizer will tend to average the noise over a cycle of the corresponding input parameter.

With quantization before the matched filter we have at least 2 samples per symbol. Therefore, we not only have samples in a preferred position (the optimum position for decision) but also in between. This significantly reduces

the influence of a timing error ϵ and a phase θ at the ADC input on performance degradation.

VIII.2 TRACKING PERFORMANCE RESULTS FOR CARRIER SYNCHRONIZATION

VIII.2.1 Preliminaries

We are primarily studying fully digital synchronizers where correction of carrier phase and timing error are implemented digitally. With a hybrid synchronizer as shown in Fig. VIII.1-1a or Fig. VIII.1-1b the signal is rotated into a particular position before AD conversion. In a fully digital synchronizer the signal will rotate with a possible frequency difference between carrier frequency and local reference before correction in the phase rotator (and, thus, also at the ADC). Therefore, we will have varying conditions in the inphase and quadrature channels. For instance, with BPSK the signal may either be completely in the inphase or completely in the quadrature phase channel, or anywhere between.

For practical purposes we have assumed zero phase (θ=0) in the simulations for the synchronizer performance figures which are given in the following subsections (see section VIII.1.6). For a digital synchronizer we have to take into account the impact of the rotating phase on these performance figures. Although synchronizer performance does not depend on signal mean and variance only, the mean ADC characteristic and the ADC output noise can serve as good indicators for the influence of the ADC on performance.

Let us first assume that we have an unmodulated carrier. With a phase error ϕ we have $\cos\phi$ in the inphase channel and $\sin\phi$ in the quadrature channel. With a 2 bit ADC the mean of the output is not proportional to the mean of the input signal for high signal-to-noise ratios since the ADC characteristic shown in VIII.1-3a significantly deviates from the straight line. As $\sin\phi$ and $\cos\phi$ in general have different amplitudes, the distortion will be different in the inphase and quadrature channels. Therefore, we cannot expect that the mean phase of the output signal equals the phase of the input signal. The difference between mean output signal phase and input signal phase as a function of the input signal phase is shown in Fig. VIII.2-1 for 2 bit and 3 bit quantization, and various sample signal-to-noise ratios (ADC levels set according to criterion II, eq. VIII.1.3-2). For 2 bit quantization the phase error is significant for high signal-to-noise ratios, e.g. for 20 dB the error is up to 10 degrees. For

lower signal-to-noise ratios the situation is better as can be seen in Fig. VIII.2-1b, which is also expected from the ADC characteristics of Fig. VIII.1-3a. From these figures we can conclude that 2 bit quantization is unacceptable with digital carrier phase correction at moderate to high signal-to-noise ratios.

A second problem is in the shape of the mean ADC output signal itself. With digital carrier phase correction the sampled matched filter output for an unmodulated carrier is

$$p_Q(kT) = Q(\cos\theta + n_I(kT)) + jQ(\sin\theta + n_Q(kT))$$

With a feedback synchronizer a phase detector output may be defined as

$$u(kT) = \mathrm{Im}\left\{ p_Q(kT)\ \exp(-j\hat{\theta})\right\}$$

Note that this is also the phase detector output for the DDML when the influence of decision errors is negligible. Let us determine the detector gain for small variations of θ. This gain is the mean of the derivative of u with respect to the phase θ

$$\left.\frac{\partial E[u]}{\partial\theta}\right|_{\hat{\theta}=\theta} = \left.\frac{\partial}{\partial\theta}\, s_Q(\sin\theta)\ \cos\hat{\theta} - \frac{\partial}{\partial\theta}\, s_Q(\cos\theta)\ \sin\hat{\theta}\right|_{\hat{\theta}=\theta}$$

Note that the above expression is not the 'slope' which determines the synchronizer dynamics - the slope is the mean of the derivative of u with respect to $\hat{\theta}$. Now assume a 2 bit ADC and a SNR of 20 dB. For θ close to zero this gain is stronger with quantization than without quantization. This will to a large extent compensate the higher variance with quantization (see Fig. VIII.1-4c). For a phase θ in the range $8^o < \theta < 25^o$ the ADC input signal in the quadrature phase channel, $\sqrt{2}\ \sin\theta$, is in the range $0.2 < \sqrt{2}\ \sin\theta < 0.6$. In this range $s_Q(\sin\theta)$ is approximately constant (Fig. VIII.1-3a) and the derivative with respect to θ is very small. Since $\cos\theta$ also is approximately constant the gain is much smaller than without quantization. In addition the ADC output noise variance is larger than the input noise variance. Therefore, we will have a significant performance degradation in this range of ‡. With a hybrid feedback carrier phase

correction as in Fig. VIII.1-1a, b the signal will always be rotated such that $\theta = 0$, i.e. in a 'favourable' (optimum) position. Therefore, with a digital carrier phase correction the sensitivity to deviations of the ADC characteristic $s_q(s)$ from a linear function is higher.

With an interpolator between ADC and carrier synchronizer or with the carrier phase estimation algorithms which do not require knowledge of the symbol timing, we have unsynchronized sampling at a rate above 1 sample/symbol. Due to the modulation we will therefore have varying signal levels at the ADC input even with a constant phase θ. This reduces the dependency of the performance on the absolute values of timing error ϵ and phase θ. In all cases where we have more than 1 sample/symbol performance depends much less on ϵ and θ.

To summarize, with hybrid feedback synchronization the signal is rotated before AD conversion and/or sampled synchronously. Therefore, carrier phase error and/or symbol timing error are close to zero before AD conversion. With digital correction carrier phase and symbol timing vary with time. As discussed above, the influence of the AD converter (e.g.bias, quantization noise) depends on the input signal level and thus on the values of $\epsilon-\hat{\epsilon}$ and $\theta-\hat{\theta}$. With hybrid feedback, i.e. $\epsilon-\hat{\epsilon}=0$ and/or $\theta-\hat{\theta}=0$, we may find less influence of the ADC (i.e. we have what was called above a 'favourable' position) then with digital correction where ϵ and θ take on arbitrary values, including values where performance degradation by the ADC is stronger than for the favourable position(s). Therefore, digital carrier phase and symbol timing correction in general requires finer ADC quantization than hybrid feedback We will resume this discussion in the following sections.

VIII.2.2 Phase rotator

In the phase rotator the input signal is rotated by the estimated carrier phase. Again the performance also depends on the way the phase rotator is actually implemented, in particular, for coarse quantization. Nevertheless we can provide good estimates for the achievable minimum performance degradation due to quantization in the phase rotator.

Neglecting the detailed implementation we may consider the phase rotator as a black box with 2 input signals, the estimated carrier phase θ and the noisy input signal. Output signal is the phase rotated input signal.

The sample signal-to-noise ratio at the phase rotator input is E_s/N_0. Since we have no filtering in the phase rotator the output sample signal-to-noise ratio is E_s/N_0 plus quantization noise. For the carrier phase estimate the situation is more critical. Because of the filtering in the loop (or in the feedforward synchronizer filter) the carrier phase error variance (i.e. the inverse of the loop signal-to-noise ratio) is much smaller than the normalized input sample variance (i.e. the inverse of the sample signal-to-noise ratio). Therefore, we need a larger wordlength for the carrier phase. The effects of carrier phase input quantization will be discussed with a numerical example which is in the region of interest for many applications. When we assume 6 bit quantization of the carrier phase the spacing is approx. 5.6 (= ±2.8) degrees. Assuming a uniform distribution of the quantization error this corresponds to a variance of

$$\sigma_\phi^2 = \frac{\Delta^2}{12} = \frac{(5.6)^2}{12}\ \mathrm{deg}^2 = 2.6\ \mathrm{deg}^2 \approx (0.03\ \mathrm{rad})^2 \qquad \text{(VIII.2.2-1)}$$

or, equivalently, a loop SNR of 20 log(1/0.03) = 30 dB (note that in the analysis of conventional synchronizers the inverse of the phase error variance measured in rad^2 often is refered to as the loop signal-to-noise ratio). When we implement the phase rotator with a complex multiplier, we need a sine/cosine table. In this case the wordlength used in the table (or equivalently) the wordlengths used at the sine/cosine input of the multiplier also affect performance. With the finite wordlength for sine and cosine we introduce an additional phase error. This is the same effect as discussed in the previous section but now for an infinite signal-to-noise ratio since we have an undistorted sine/cosine.

Fig. VIII.2-1c shows the 6 bit output of a sine/cosine table with 6 bit phase input (the unnormalized amplitude is 31). We see that deviations from the sinusoidal form are very small. The phase error contribution from the sine/cosine table output quantization is below 1 degree (note that the phase error is shown 10 times magnified, i.e. a value of 10 in Fig. VIII.2-1c corresponds to an error of 1 degree). The 6 bit input phase quantization corresponds to a maximum error of ± 2.8 degree. With a smaller wordlength

the increased mean phase error and the increased quantization error variance reduce performance significantly. Note that a 6 bit sine/cosine table corresponds to a memory with only 16 x 5 bits when we make efficient use of signal symmetries.

When a loop SNR > 30 dB is required the wordlength must be increased until the quantization noise, due to the phase rotator (as calculated in VIII.2.2-1), is sufficiently small.

VIII.2.3 Carrier Synchronization: Quantization at the matched filter output

With the carrier synchronization algorithms which require timing to be known we either have synchronized sampling or we must have an interpolator between ADC and carrier synchronizer. In the second case we have the problem that we have to take the interpolator implementation into account. Therefore, it only makes sense to study the influence on further processing (carrier synchronization and demodulation) of quantization after the matched filter together with the particular interpolator implementation. On the other hand, processing in the interpolator increases the wordlength (note that we have several samples per symbol at the interpolator input). In this case, truncation of the wordlength at the interpolator output can be expected to have similar effects as an ADC at the matched filter output with synchronized sampling.

As discussed in section VIII.1.6 in the simulation study of quantization after the matched filter we have not only assumed synchronized sampling, but also a phase $\theta = 0$. We will discuss the impact of $\theta \neq 0$ later.

When we assume that quantization effects in the remaining part of the carrier synchronizer (loop filter, VCO/NCO, phase rotator) are negligible, we can determine the carrier phase error variance, as under ideal conditions, from the normalized loop noise power spectral density at $f = 0$. Therefore, we have determined this quantity by simulation for the different carrier synchronization algorithms.

DDML

In Fig. VIII.2-2a-c the normalized loop noise power spectral densities are shown for BPSK, QPSK and OQPSK. With 4 bit quantization the performance is virtually identical to the performance without quantization for all cases. With 2 bit quantization the situation is different. While for BPSK and QPSK we only have a slight deviation from the ideal curve we have a significant degradation for OQPSK and rolloff 0. In the DDML for OQPSK the real part of q and the imaginary part p (which are not used for decision) are also processed. This explains why quantization effects depend on the rolloff in this case.

NDAML

The normalized loop noise power spectral density at f = 0 is shown in Fig. VIII.2-4a-c for BPSK, QPSK, and OQPSK. We have very similar results as for the DDML. In particular with BPSK and QPSK we observe a very similar influence of quantization. While for 4 bit we have virtually ideal performance, for 2 bit we have a slight degradation for high E_S/N_0.

Again, the synchronizer for OQPSK shows a stronger influence of a 2 bit quantization. For rolloff $\alpha = 0.5$ and 2 bit quantization we observe a very strong degradation due to quantization. Again this can be explained by the fact that we also process samples taken at 'unfavorable' positions (i.e. the real part of q and the imaginary part of p). On the other hand, we will have less influence of timing error ϵ and phase θ in this case.

V&V

In Fig. VIII.2-6a - Fig. VIII.2-6d the normalized loop noise power spectral density at f = 0 is shown for BPSK, QPSK, and OQPK. For BPSK and QPSK we find in Fig. VIII.2-6a and b similar effects as with the NDAML which is reasonable when we recall that the NDAML is a special case of the V&V. While for 4 bit we have virtually ideal performance for 2 bit we have a slight degradation for high E_S/N_0.

Again, the synchronizer for OQPSK shows a stronger influence of a 2 bit quantization. For OQPSK and rolloff $\alpha = 1.0$ we have a similar result as in the case of NDAML. For rolloff $\alpha = 0.5$ the situation is different. For the V&V the performance depends on the nonlinearity used. For f(u) = 1

performance degradations are smaller than for f(u) = u. In both cases they are smaller than for the NDAML ($f(u) = u^2$). When we take a closer look at Fig. VIII.2-6c and d we find that the performance for f(u) = u with 2 bit quantization is very close to the performance with f(u) = 1 with 2 bit quantization, i.e. the performance advantage that we have with f(u) = u under ideal conditions vanishes for coarse quantization.

Summary

For synchronized sampling and input phase $\theta = 0$ we find that with 4 bit quantization the performance degradation is negligible for the E_S/N_0 range considered. With BPSK and QPSK the performance degradation with 2 bit quantization is negligible for low signal-to-noise ratios and small for $E_S/N_0 = 10$ dB. The synchronizers for OQPSK are more sensitive to coarse quantizations in particular for small rolloff.

Applicability to synchronizers with digital carrier phase correction

As discussed in section VIII.2.1 we have a different situation with digital carrier phase correction (i.e. a phase rotator). Then, the carrier phase (error) at the ADC is no longer close to zero (it is corrected later!). Because of the effects discussed in section VIII2.2 we need a finer quantization (about 1 bit more, in general).

- Feedback algorithms which require knowledge of symbol timing

When the influence of quantization before a digital phase rotator is negligible, the influence of a wordlength reduction at the phase rotator output is also described by the simulation models used for those carrier synchronizers which assume timing to be known. Therefore, in this case the simulation results fully apply.

When we have digital symbol timing correction we have more than one sample/symbol (for timing synchronization) at the matched filter output. Thus we have not only a 'favourable' signal position with AD conversion after the matched filter. Contrary to the case where the ADC is before the matched filter consecutive noise samples no longer are uncorrelated. But the quantization noise still is approximately uncorrelated when we assume

independent data symbols. Therefore, we still can assume that the influence of ϵ and θ on performance is small.

- Feedback algorithms which do not require knowledge of symbol timing

Algorithms which do not require timing in most cases operate on more than one sample per symbol and with $\epsilon \neq 0$. Therefore, we do not only have samples taken at 'favourable' positions. This means that we have a a higher performance loss than for the special case $\epsilon=0$, $\theta=0$. With more than one sample per symbol we have a slightly reduced sensitivity to ϵ and θ. For 3 bit and 4 bit we can use the ADC performance figures (Fig. VIII.1-3 and Fig. VIII.1.4) to estimate the performance loss over the ideal case results. In the operating range the ADC characteristic $s_Q(s)$ should be close to linear, otherwise the slope of the synchronizer may vary with ϵ and θ. Also, the ADC output noise variance is approximately independent of s in this case. Then, quantization can be modelled as additive noise with variance $\Delta^2/12$ where Δ is the ADC level spacing (or, equivalently read off from Fig. VIII.1-4a,b,c).

- Feedforward algorithms

With feedforward synchronizers we have an arbitrary phase θ. This means we again have not only 'favourable' signal positions and thus we also need more than 2 bit quantization and the above arguments apply.

VIII.2.4 Carrier Synchronization: Quantization before the matched filter

Again we have used the special case $\epsilon=0$, $\theta=0$ in the simulations. The impact of this assumption on performance figures will be discussed at the end of this section.

DDML

The effects of quantization before the matched filter are shown in Fig. VIII.2-3a,b,c. With 1 bit at 4 samples per symbol we observe a factor of 2 increase for BPSK and approximately one order of magnitude with QPSK. With 2 and 4 bit at 2 samples per symbol the performance in both cases is close to ideal performance. Therefore, we can expect that with 2 bit

and 4 samples per symbol we practically have ideal performance, as we do with 4 bit.

With OQPSK 1 bit quantization is very critical in particular for small rolloff. For 2 and 4 bit we have similar results as with QPSK.

NDAML

The degradations are very similar to those for DDML except for OQPSK where the NDAML does not operate for rolloff $\alpha = 0$ and, therefore, a comparison is not possible (Fig. VIII.2-5).

V&V

In Fig. VIII.2-7a-d the normalized loop noise power spectral density at f=0 is shown for BPSK, QPSK, and OQPSK with f(u)=1 and f(u)=u.

Independent of the nonlinearity the degradations are similar to those for DDML and NDAML. For 4 bit degradations are negligible, for 2 bit with 2 samples/symbol slight degradations occur for E_S/N_0->10 dB. For 1 bit quantization and 8 samples/symbol significant degradations occur which are independent of E_S/N_0 and small for BPSK, strong for QPSK, and in between for OQPSK. In contrast to the results for quantization after the matched filter now the performance advantage for f(u) = u is maintained even for coarse quantization in all cases (BPSK, QPSK, and OQPSK).

Impact of $\epsilon \neq 0$, $\theta \neq 0$

Since we have at least two samples/symbol even with synchronized sampling and phase $\theta=0$ we have at least one sample in a 'non-favourable' position where we do not have particular signal levels (as for the sample at the decision instant).

It has been checked by example that the performance figures for arbitrary input signal synchronization parameters ϵ, θ do not differ strongly from the performance figures for the special values $\epsilon=0$, $\theta=0$ when we have at least 3 bit quantization. For 2 bit quantization performance degradation strongly depends on ϵ and θ.

- Algorithms which do not require knowledge of symbol timing

For algorithms which do not require timing we have the same situation at the ADC except that the minimum number of samples/symbols may be higher than for known timing. Therefore, we will have similar degradations or, with a larger number of samples/symbol, even less degradation as compared to the algorithms which require timing to be known.

- Feedback/feedforward carrier synchronisation

Note that for this case the performance figures for feedback carrier synchronizers are identical to those for feedforward synchronizers.

VIII.2.5 Carrier Synchronization: Conclusions

First we will discuss the figures for quantization after the matched filter. We find that the degradation due to quantization is practically independent of the type of synchronizer we use. When we operate on synchronized samples with small phase errors performance degradation is small for 2 bit quantization and E_s/N_0 up to 10 dB. That means the feedback synchronization algorithms are very insensitive to wordlength truncation. With the feedforward synchronizers we may have an arbitrary θ. In this case performance degradation due to quantization can be significantly larger for 'unfavourable' values of θ. Therefore, a larger wordlength is required. From Fig. VIII.1-3, Fig. VIII.1-4 we can determine the required number of bits and the approximate corresponding performance degradation. For feedforward synchronizers a 3 to 4 bit ADC should be used after the matched filter or a 3 to 4 bit processing wordlength should be used in the carrier phase estimation algorithm. Also the ADC characteristic $s_Q(s)$ should be close to linear, otherwise the slope of the synchronizer may vary with θ (and ϵ). Also, the ADC output noise variance is approximately independent of the instantaneous signal amplitude s in this case. Then, quantization can be modelled as additive noise with variance $\Delta^2/12$ where Δ is the ADC level spacing (or, equivalently read off from Fig. VIII.1-4a,b,c).

For OQPSK where we have two samples/symbol in the processing, we directly observe a stronger performance degradation for 2 bit which comes

from the fact that every second sample is taken at a nonoptimum instant. It was found that introducing a phase shift of $\theta=45^{o}$ yields approximately the same performance figures. This means for OQPSK the performance figures apply for feedforward and feedback synchronizers.

With quantization before the matched filter 2 to 3 bit quantization yields small performance loss for $E_S/N_0 < 10$ dB with 2 samples/symbol. The performance loss can be reduced further by using more than 2 samples/symbol. For 1 bit quantization (hardlimiter) we have a remaining performance loss even for very low SNR which also is not reduced further for a larger number of samples per symbol.

With a 4 bit ADC we will have a neglible performance loss for all synchronizers and E_S/N_0 up to 20 dB. For high signal to noise ratios the loop noise will be dominated by the quantization noise but then we are in a range where synchronization is no longer critical for the applications considered here.

For quantization before the matched filter the performance loss was found to be practically identical for the different synchronization algorithms. The performance loss only depends on E_S/N_0, the number of samples/symbol, and the number of bits of the ADC.

VIII.3 TRACKING PERFORMANCE RESULTS FOR SYMBOL SYNCHRONIZATION

VIII.3.1 Preliminaries

When we want to study the influence of quantization on the performance of the various timing synchronization algorithms we again have the problem that in a fully digital receiver we have a rotating phase and unsynchronized sampling. In particular for coarse quantization the influence of the quantization may strongly depend on the actual phase θ and timing error ϵ. As discussed in section VIII.1.6 we again have assumed zero phase error ($\theta = 0$) and synchronized sampling ($\epsilon = 0$) in the simulation study for practical purposes. The impact of non-synchronized sampling and non-zero phase error will be discussed later. That there is no basic rule from which worst case and best case positions (i.e. particular values for ϵ, θ) can be derived will be demonstrated by two examples.

Let us assume continuous-time filters (quantization after the matched filter). When we use DDMLFB timing synchronization

$$u_\tau(k;\hat{\theta};\hat{\epsilon}) = \hat{a}_k p_R'(k;\hat{\theta};\hat{\epsilon}) + \hat{b}_k p_I'(k;\hat{\theta};\hat{\epsilon})$$

we have a decision part ($\hat{a}_k$, $\hat{b}_k$) and the derivative matched filter output. We do not have an influence of quantization on the decisions when we have synchronized sampling and zero phase. The reason is that we always have a threshold at zero which is the optimum decision threshold. The second signal we need in a DDMLFB is the derivative matched filter output. With synchronized sampling at 1/T this has a zero-crossing at the sampling instant which is a 'favourable' situation for quantization with an ADC which has a threshold at zero. Thus we can expect a low implementation loss.

On the other hand a counter example is the mM&M algorithm

$$u_\tau(k;\hat{\theta};\hat{\epsilon}) = (\hat{a}_{k-1} - \hat{a}_{k+1})p_R(k;\hat{\theta};\hat{\epsilon}) + (\hat{b}_{k-1} - \hat{b}_{k+1})p_I(k;\hat{\theta};\hat{\epsilon})$$

For the decisions we still have a 'favourable' position as with the DDML. But the information on a possible timing error is in the amplitude of $p_{I/R}$.

Now assume we have a high E_S/N_0, e.g. 20 dB, and 2 bit quantization. Then, in the correct position we have an amplitude of 1. When we look at the ADC characteristic in Fig. VIII.1-3a we find that the slope at s=1 which determines how strong the change of the output signal is for a given change of the input signal is close to zero. On the other hand, for high E_S/N_0 the noise variance becomes proportional to p^2 (or for the quantized signal proportional to the corresponding quantization level). Therefore, for high SNR the noise variance will approach a constant value while the slope decreases, i.e. performance will degrade with increasing SNR. On the other hand, when the signal is shifted in time, the level of p may frequently be in the vicinty of a threshold of the ADC where the sensitivity to small variations is much higher. Therefore, we can assume that there are better cases than the synchronized sampling with zero phase for the mM&M algorithm.

The figures for quantization after the matched filter give the correct performance figures for a hybrid synchronizer with feedback carrier synchronization and synchronized sampling. Also, they are a good approximation for the influence of wordlength truncation after the interpolator in feedback timing synchronizers. As discussed in section VIII.1.6 with quantization before the matched filter the influence of timing error ϵ and phase θ at the ADC input on performance degradation is less significant than for quantization after the matched filter. The performance degradations will be approximately identical to the degradations for $\epsilon = 0$ and $\theta = 0$ given in the figures with quantization before the matched filter.

VIII.3.2 Interpolator

There are various ways to implement an interpolation filter, which makes it difficult to discuss quantization effects in general. As with any digital filter which is supposed to approximate a function defined in the continuous-time domain (see section VIII.1.4) there are a variety of solutions to reduce quantization influence for coarse quantization.

The interaction of coarse quantization, sampling rate (samples/symbol) and approximation of the interpolation function is an interesting topic which deserves a study of its own but it is out of the scope of this study.

VIII.3.3 Timing Synchronization: Quantization at the matched filter output

With the timing synchronizers which require carrier synchronization we must have a phase error correction before the synchronizer. With digital correction the phase rotator is between ADC and timing synchronizer. In this case we also have to take the phase rotator implementation into account. On the other hand, a larger wordlength might be used there to keep the decision loss in the demodulator small. Then, the influence of the phase rotator will be negligible. When we have feedback timing synchronisation we, in addition, have an interpolator before the timing error detector. Processing in the interpolator might further increase the wordlength. In this case, truncation of the wordlength at the interpolator output can be expected to have similar effects as an ADC at the matched filter output with synchronized sampling and zero phase.

When we assume that quantization effects in the remaining part of the timing synchronizer (loop filter, VCO/NCO, interpolator) are negligible, we can determine the timing error variance, as under ideal conditions, from the normalized loop noise power spectral density at $f = 0$. Therefore, we have determined this quantity by simulation for the different timing synchronization algorithms.

Finally, note that as a representative of DDEL(λ=1/2), DTTL the DTTL has been simulated, as a representative of DDEL(λ=1), M&M, mM&M the mM&M has been simulated, and as a representative of NDAEL(λ=1), GA the GA has been simulated.

<u>DDML</u>

In Fig. VIII.3-1a,b the normalized loop noise power spectral density at f=0 of DDML is shown for BPSK and (O)QPSK. With 4 bit quantization the performance degradation is negligible for all E_S/N_0 shown. For 2 bit quantization performance loss is negligible below approximately 5 dB, and strong around 15 dB. With (O)QPSK the loss is smaller which can be explained from the fact that the sample signal-to-noise ratio at the inphase ADC is 3 dB higher for BPSK (the degradation increases with the SNR).

NDAML

In Fig. VIII.3-3a,b the normalized loop noise power spectral density at f=0 of NDAML for BPSK and (O)QPSK is shown. As with DDML we observe that with 4 or 8 bit quantization the loss is negligible in the total E_s/N_0 range simulated. For 2 bit quantization we have a significant performance degradation, which is larger for (O)QPSK than for BPSK now. This probably is caused by the fact that the matched filter output level for (O)QPSK ($\approx \pm 0.7$) is closer to an ADC threshold than for BPSK ($\approx \pm 1$). Therefore, we get more quantization noise from this path (this is not relevant for the DDML where a decision with threshold zero follows in this signal path). But we must be aware that this effect strongly depends on the signal level which may vary in a practical system. We could improve O(Q)PSK performance by scaling the input by 1/0.7 when we have analogue carrier phase correction. For digital correction the phase error may take on arbitrary values in the ADC input signal (since it is corrected after AD conversion). In this case performance will be degraded for phase error values different from 0. Note that the ADC levels used in the simulation were optimized for the case of uncorrected phase error in the ADC input signal.

DTTL

In Fig. VIII.3-5a,b the normalized loop noise power spectral density at f=0 of DTTL for BPSK and (O)QPSK is shown. With 4 bit quantization the performance degradation is negligible for all E_S/N_0 shown. For 2 bit quantization performance loss is negligible below approximately 5 dB, and significant around 15 dB. The results for the DTTL are almost the same as for the DDML synchronizer. In contrast to the DDML, 2 bit quantization loss for BPSK now is approximately the same as for (O)QPSK at high E_s/N_0.

mM&M

In Fig. VIII.3-7a,b the normalized loop noise power spectral density at f=0 of mM&M for BPSK and (O)QPSK is shown. Again we do not have a performance degradation with 4 bit quantization in the total range studied by simulation. For 2 bit quantization we have a tremendous degradation for BPSK and rolloff $\alpha \rightarrow 1$. The explanation is given in the example in section VIII.3.1. For BPSK the level of synchronized samples with zero phase is ± 1 . When we look at Fig. VIII.1-3a we see that the slope is

almost zero for E_S/N_0= 20 dB. With (O)QPSK we are at $\pm\sqrt{2}$ where the slope is larger. Therefore, the performance degradation for QPSK is much smaller (see section VIII.3.1). The performance loss for BPSK is also smaller for rolloff α -> 0. The reason here is that the strong intersymbol interference which is present for small offset $\epsilon \neq 0$ causes varying signal levels sometimes close to the ADC threshold.

As a conclusion the mM&M is very critical at high SNR, in particular, for large rolloff when we have a coarse quantization.

GA

In Fig. VIII.3-9a,b the normalized loop noise power spectral density at f=0 of GA for BPSK and (O)QPSK is shown. With 4 bit quantization the performance degradation is negligible for all E_S/N_0 shown. For 2 bit quantization performance loss is negligible below approximately 5 dB for BPSK. With (O)QPSK it is small for rolloff 1.0 and constantly about 2dB below $E_S/N_0 \approx 5$ dB. For higher E_S/N_0 the performance degradation is strong for (O)QPSK only.

Summary

For synchronized sampling and input phase $\theta = 0$ we find that with 4 bit quantization the performance degradation is negligible for the E_S/N_0 range considered. The performance degradation with 2 bit quantization is negligible for low signal-to-noise ratios, small for E_S/N_0 = 10 dB, and significant to strong for E_S/N_0 -> 15 dB. The algorithms using a difference approximation (DTTL, GA) partly are less sensitive to coarse quantization than their 'ideal' counterparts (DDML, NDAML). Only the mM&M was found to be very sensitive to coarse quantization at high E_S/N_0.

Impact of $\theta \neq 0$, $\epsilon \neq 0$

When we interpret the results as the influence of a wordlength truncation at the interpolator output the signal is actually in the position assumed in the simulations for those timing synchronizers which require carrier synchronization. Therefore, in this case the simulation results fully apply. When we apply the results to AD conversion at the matched filter output with non-synchronized sampling we have more than one sample/symbol. Then we have not only samples taken at 'special' instants. Contrary to the case where the ADC is before the matched filter consecutive noise samples are not uncorrelated. But the quantization noise still is approximately uncorrelated when we assume independent data symbols. For 3 bit and 4 bit quantization we can use the ADC performance figures (Fig. VIII.1-3 and Fig. VIII.1-4) to estimate the performance loss over the ideal case results. In the operating range the ADC characteristic $s_Q(s)$ should be close to linear, otherwise the slope of the synchronizer may vary with ϵ and θ. Also, the ADC output noise variance is approximately independent of s in this case. Then, quantization can be modelled as additive noise with variance $\Delta^2/12$ where Δ is the ADC level spacing (or, equivalently read off from Fig. VIII.1-4a,b,c).

- Algorithms which do not require carrier synchronization

With rotating θ and $\epsilon \neq 0$ we do not have samples taken at a special position. This means that we will have a significant performance loss with 2 bit quantization. For 3 bit and 4 bit we can use the ADC performance figures (Fig. VIII.1-3 and Fig. VIII.1.4) to estimate the performance loss over the ideal case results with the same design guidelines as discussed above.

VIII.3.4 Timing Synchronization: Quantization before the matched filter

DDML

In Fig. VIII.3-2a,b the normalized loop noise power spectral density at f=0 of DDML for BPSK and (O)QPSK is shown with quantization introduced before the matched filter. For 1 bit quantization we have a loss already for very low signal-to-noise ratios while for 2 bit quantization a significant loss occurs for 15 dB only. As for quantization after the matched filter we observe that BPSK seems to be more sensitive to quantization than QPSK. As discussed previously we have a more unbalanced situation with BPSK.

NDAML

In Fig. VIII.3-4a,b the normalized loop noise power spectral density at f=0 of NDAML for BPSK and (O)QPSK is shown with quantization introduced before the matched filter. Again, for 1 bit quantization we have a loss already for very low signal-to-noise ratios while for 2 bit quantization only a slight loss occurs for 15 dB. Performance degradation is similar for BPSK and (O)QPSK.

DTTL

In Fig. VIII.3-6a,b the normalized loop noise power spectral density at f=0 of DTTL for BPSK and (O)QPSK is shown with quantization introduced before the matched filter. We find a result which is almost identical to that of the DDML sychronizer.

mM&M

In Fig. VIII.3-8a,b the normalized loop noise power spectral density at f=0 of mM&M for BPSK and (O)QPSK is shown with quantization introduced before the matched filter. We have a very low performance loss up to 15 dB E_S/N_O in all cases considered except for rolloff $\alpha = 1$ and BPSK where we have a significant loss. This indicates the sensitivity of mM&M against coarse quantization even when the quantization is introduced at the matched filter input. This in particular applies for rolloff $\alpha \rightarrow 1$.

GA

In Fig. VIII.3-10a,b the normalized loop noise power spectral density at f=0 of GA for BPSK and (O)QPSK is shown with quantization introduced before the matched filter. Again, for 1 bit quantization we have a loss already for very low signal-to-noise ratios while for 2 bit quantization only a slight loss occurs for 15 dB. The influence of quantization is virtually identical to that for the NDAML.

Impact of $\theta \neq 0$ and $\epsilon \neq 0$

In the simulations with four samples per symbol even with synchronized sampling and phase error zero (i.e. synchronized carrier phase) we have at least two samples in 'non-favourable' positions where we do not have particular signal levels (as for the sample at the decision instant and possibly for the samples midway between the optimum decision instants). With digital correction the phase may take on arbitrary values in the ADC input signal since it is corrected after AD conversion. With the same reasoning as above we can expect to have only small variations of the performance with θ. It has been checked by example that the performance figures for arbitrary input signal synchronization parameters ϵ, θ do not differ strongly from the performance figures for the special values $\epsilon=0$, $\theta=0$.

For 2 bit quantization and 2 samples/symbol (a case not shown in the figures) we have less influence of ϵ and θ on performance with quantization before the matched filter than with quantization after the matched filter. The reason is that the signal is more random at the sampling instants because of the intersymbol interference. On the other hand this also means that we can take less advantage of 'favourable' positions and therefore have stronger degradations than for quantization before the matched filter.

VIII.3.5 Feedforward timing synchronizers

With feedforward synchronizers we do not have synchronized sampling and the interpolator comes after the synchronizer. For quantization before the matched filter the same arguments apply as for the feedback synchronizers, i.e. the same design rules for the ADC can be used. For quantization after the matched filter we cannot use 2 bit quantization since the significant distortion of the ADC characteristic $s_Q(s)$ and the ADC output noise dependence on the signal level s will cause unacceptable effects. We rather might use a quantization such that $s_Q(s)$ is close to a linear function and the ADC output noise variance is approximately independent of s. Then, quantization can be modelled as additive noise with variance $\Delta^2/12$ where Δ is the ADC level spacing (or, equivalently read off from Fig. VIII.1-4a,b,c).

With the DDMLFF and the NDAMLFF theoretically we have to determine the maximum of a function of the matched filter output for values of the trial parameter $\tilde{\tau}$ in the range $-T/2 < \tilde{\tau} \leq +T/2$. Practically, we can only determine this function for a finite number of values of the trial parameter $\tilde{\tau}$. The spacing between the trial parameter values can be determined easily. Here again the well known rule applies that when $(\Delta\tau)^2/12$ is much smaller than the variance $\sigma^2_{\hat{\tau}} = E[\hat{\tau}^2] - E^2[\hat{\tau}]$, then the parameter quantization is negligible.

For these two feedforward synchronizers a problem occurs which is not present in any of the other synchronizers. When the correct timing value τ_0 is not in the center of the range of the trial parameter $\tilde{\tau}$ we may get a bias in the estimate. Assume that the range is $-T/2 < \tilde{\tau} \leq +T/2$ and $\tau_0 = +T/4$. Now there are more possible estimates smaller than τ_0 than there are estimates larger than τ_0. When the error variance $\sigma^2_{\hat{\tau}}$ is not much smaller than $(T/4)^2$ this will cause a bias in the example. We run into this problem anyway when τ_0 approaches T/2. This can only be avoided by either extending the trial parameter range to $\pm T$ or by centering the $\pm T/2$ range arround a prediction from a feedforward loop filter.

VIII.3.6 Symbol Timing Synchronization: Conclusions

First we will discuss the figures for quantization after the matched filter. We find that when we operate on synchronized samples with small phase errors, performance degradation is small for 2 bit quantization and E_s/N_0 up to 10 dB. That means the feedback synchronization algorithms are, with the exception of the mM&M group, insensitive to wordlength truncation.

With the feedforward synchronizers we may have an arbitrary ϵ. In this case performance degradation can be significantly larger for 'unfavourable' values of ϵ. Therefore, a larger wordlength is required. From Fig. VIII.1-3, Fig. VIII.1-4 we can determine the required number of bits and the approximate corresponding performance degradation. The most important rule is that in the operating range the ADC characteristic $s_Q(s)$ should be close to linear, otherwise the slope of the synchronizer may vary with ϵ and θ. Also, the ADC output noise variance is approximately independent of s in this case. Then, quantization can be modelled as additive noise with variance $\Delta^2/12$ where Δ is the ADC level spacing (or, equivalently read off from Fig. VIII.1-4a,b,c). .

With quantization before the matched filter 2 to 3 bit quantization yields small performance loss for $E_S/N_0 < 10$ dB with 4 samples/symbol. Symbol synchronizers seem to be more sensitive to quantization than carrier synchronizers which may be explained by the fact that they extract the information more from the waveform. For 1 bit quantization (hardlimiter) we have a remaining performance loss even for very low SNR which also is not reduced further for a larger number of samples per symbol.

With a 4 bit ADC we will have a neglible performance loss for all synchronizers and E_S/N_0 up to 20 dB. For high signal to noise ratios the loop noise will be dominated by the quantization noise but then we are in a range where synchronization is no longer critical for the applications considered here.

VIII.4 Conclusions

An engineer designing a digital synchronizer would of course prefer to have formulas or figures from which he can directly determine the required ADC levels and quantization. But it is not possible to provide such results when we use coarse quantization (i.e. a few bit only). Although, we have given a lot of simulation results about the influence of quantization on synchronizer performance, many problems and questions concerning the influence of structures and quantization on performance have been discussed only qualitatively. It is neither possible nor reasonable to provide a set of formulas or figures that answer all the questions. Rather it is required that for a particular application the complete synchronizer is simulated under the given operating conditions to finally determine the required wordlengths, in particular when we want to go down to the most coarse quantization that still yields acceptable performance. The results given here allow complexity estimates and first selections to have a good starting point for the final design.

Whenever we have the signal in a particular position, i.e. with a hybrid synchronizer or after correction by phase rotator and interpolator, we can use wordlengths as low as 2 bits with few exceptions (e.g. mM&M). We have a similar performance degradation behaviour in all cases which starts around $E_S/N_0 \approx 5$ dB and becomes stronger when we go to higher E_S/N_0. On the other hand for high E_S/N_0 values synchronization is less critical, in particular, when we have a stable carrier as is usually the case in satellite communications.

With a fully digital synchronizer with digital timing and phase correction we cannot use 2 bit quantization after the matched filter. The reason is that with 2 bit (and also for $E_S/N_0 \geq 10$ dB with 3 bit) we have a significant degradation of the ADC characteristic $s_Q(s)$. These distortions significantly influence synchronization algorithm performance. This yields slopes and noise performance figures which strongly depend on phase θ and timing error ϵ. In normal operation this is not acceptable. Therefore, the quantization should be selected such that we have a characteristic $s_Q(s)$ which is close to a linear function and that the ADC output noise is approximately independent of the level s (see Fig. VIII.1-3, Fig. VIII.1-4). Then, quantization effects can be modelled by increasing the noise variance by $\Delta^2/12$ where Δ is the

ADC level spacing (or, equivalently read off the total noise from Fig. VIII.1-4a,b,c).

For quantization before the matched filter we mainly have a signal distortion whose influence is to a large extent independent of the synchronizer type. For E_S/N_0 values up to 15 dB performance loss is small even with 2 bit at 4 samples per symbol. Then, in the range considered here, i.e. $E_S/N_0 < 15$ dB, the performance loss over the ideal case is small. For higher signal to noise ratios the performance loss increases until finally the quantization noise determines system performance. In this situation, carrier and timing synchronization is in general not critical. When we use the above discussed values for quantization the performance loss is similar for any synchronizer combination. Therefore, selection can be based on other criteria than quantization. For instance, it does not seem reasonable to use a DDMLFB timing synchronizer where we need a derivative matched filter, but rather use one of the corresponding synchronizers with a difference approximation of the derivative which are better suited for digital implementation.

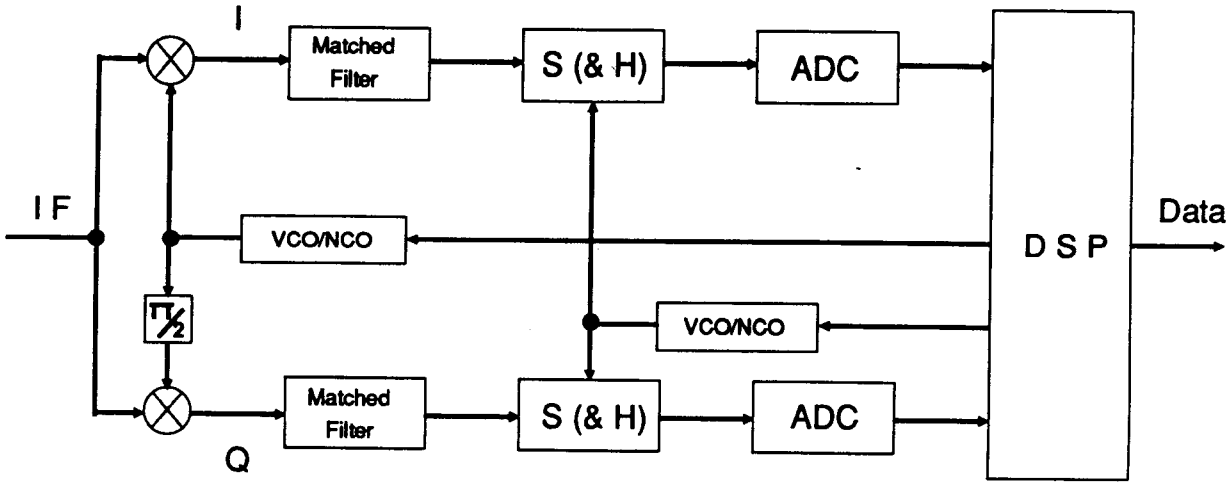


Fig. VIII.1-1a: Feedback synchronizer with continuous-time matched filter

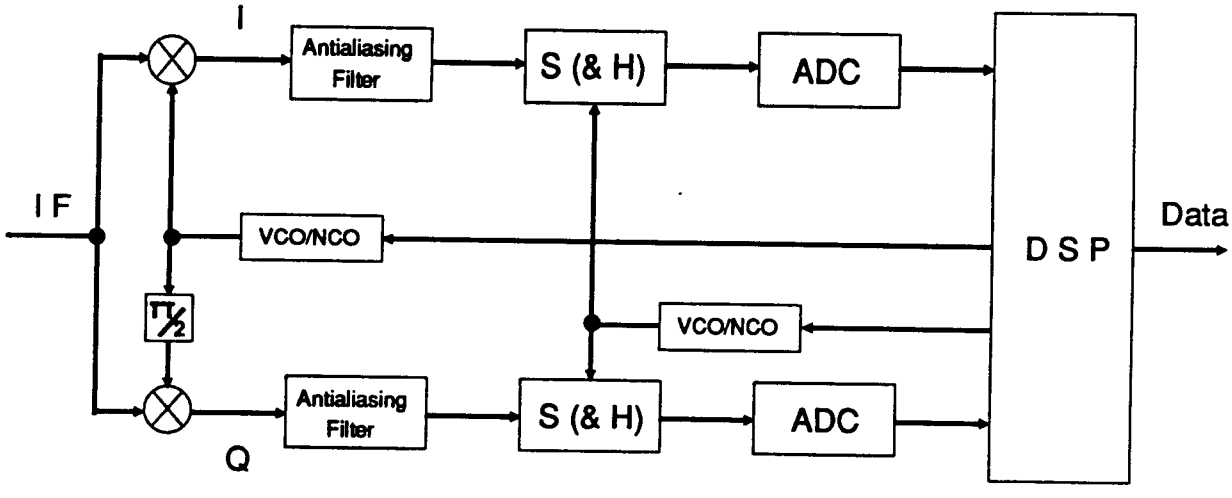


Fig. VIII.1-1b: Feedback synchronizer with digital matched filter

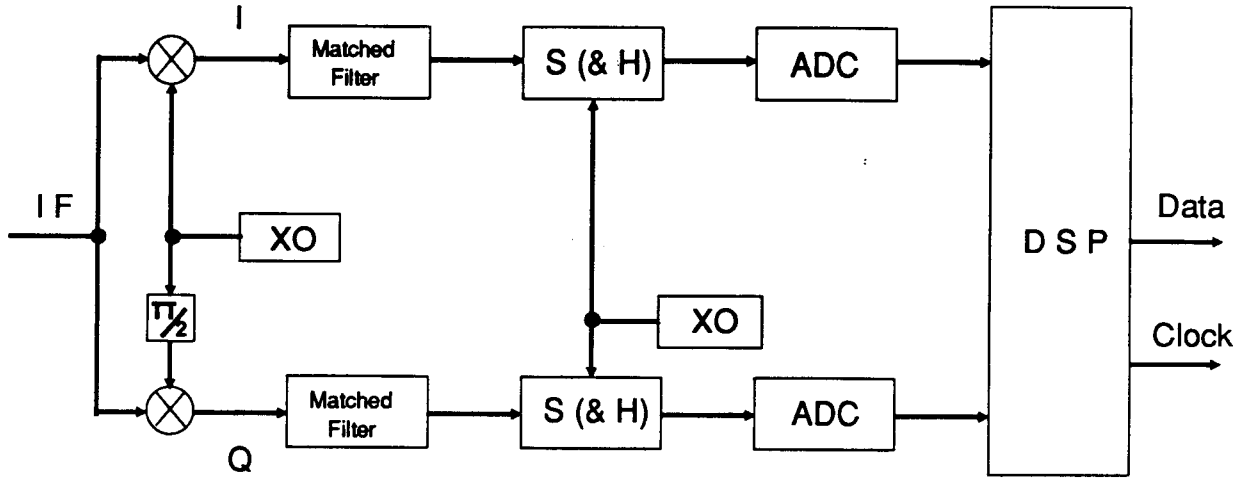


Fig. VIII.1-1c: Synchronizer with continuous-time matched filter and digital feedback or feedforward carrier phase and timing correction

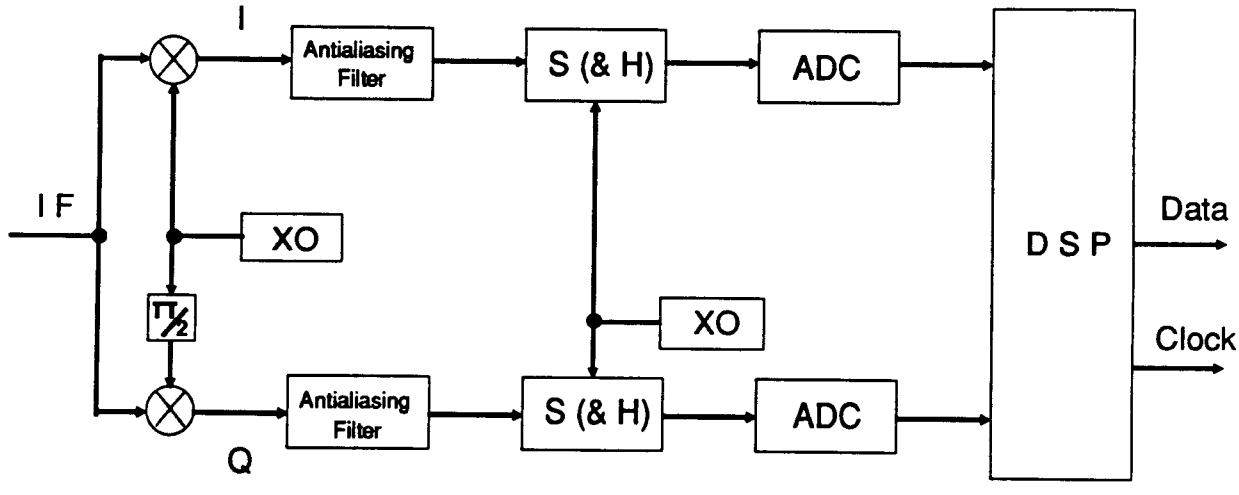


Fig. VIII.1-1d: Synchronizer with digital matched filter and digital feedback or feedforward carrier phase and timing correction

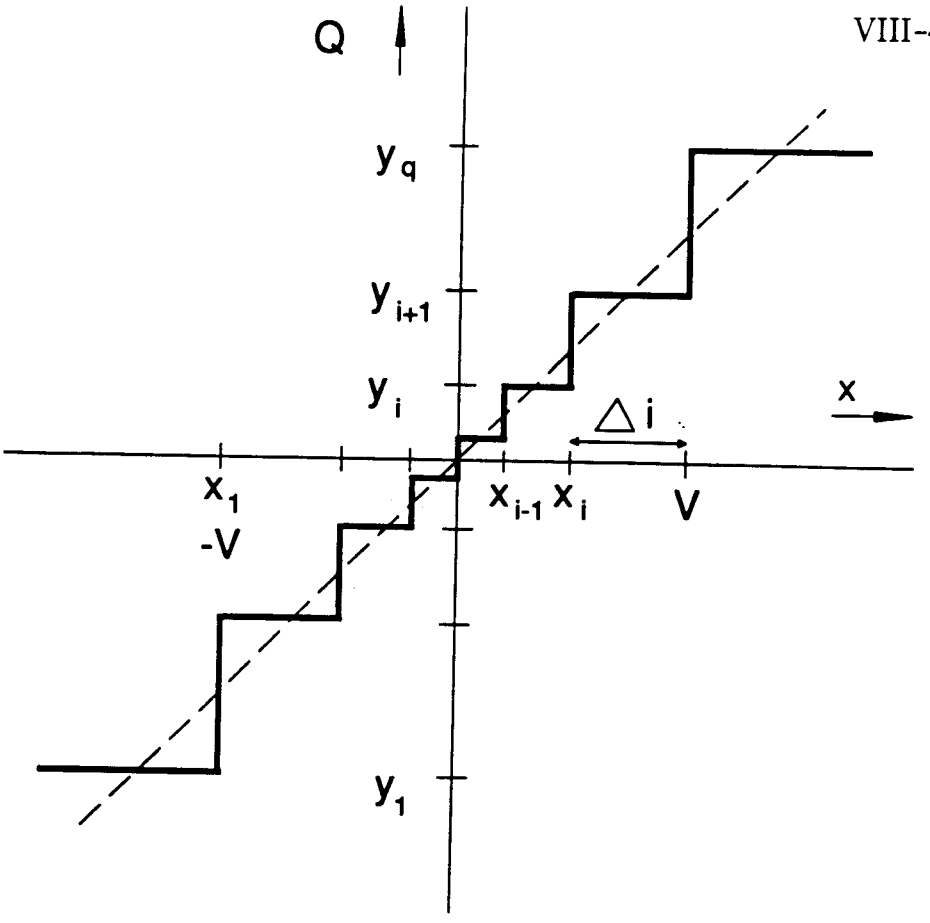


Fig. VIII.1-2a: Quantization characteristic, for uniformly quantizing analog-to digital converters (ADC) Δ_i is independent of i characteristic

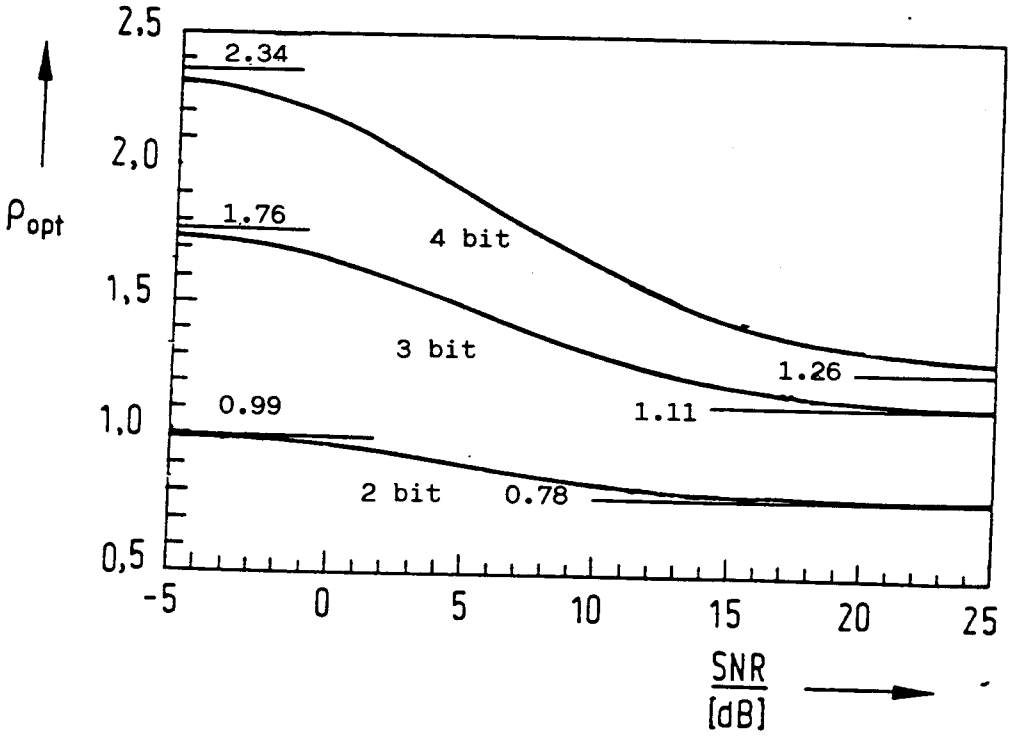


Fig. VIII.1-2b: Optimum normalized overload level ρ (sinusoid plus Gaussian noise), criterion II

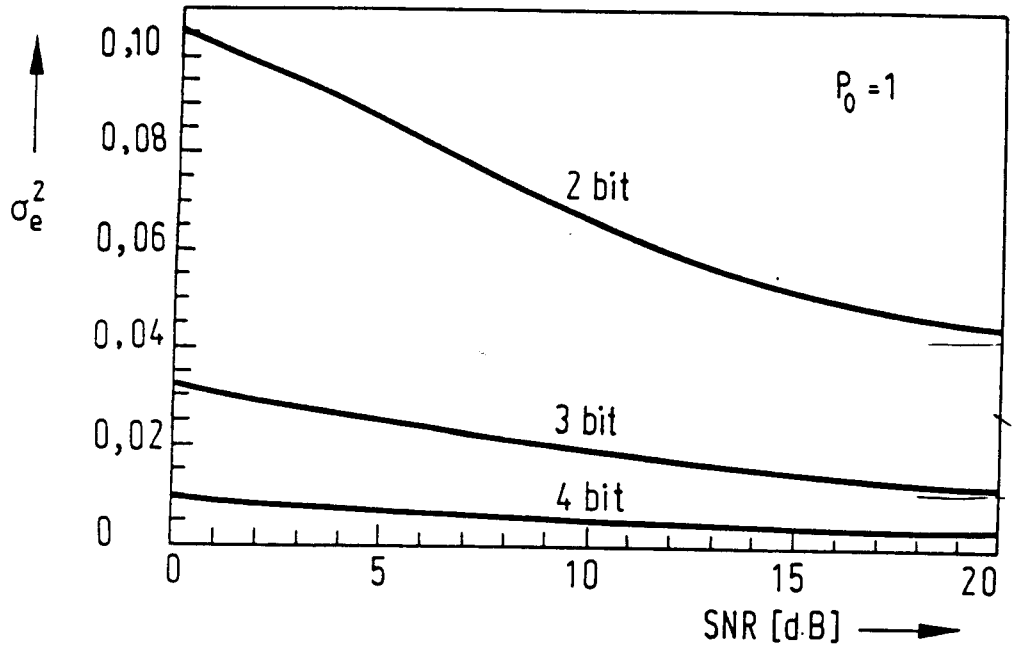


Fig. VIII.1-2c: Quantization error variance for optimum ρ (sinusoid plus gaussian noise), criterion II

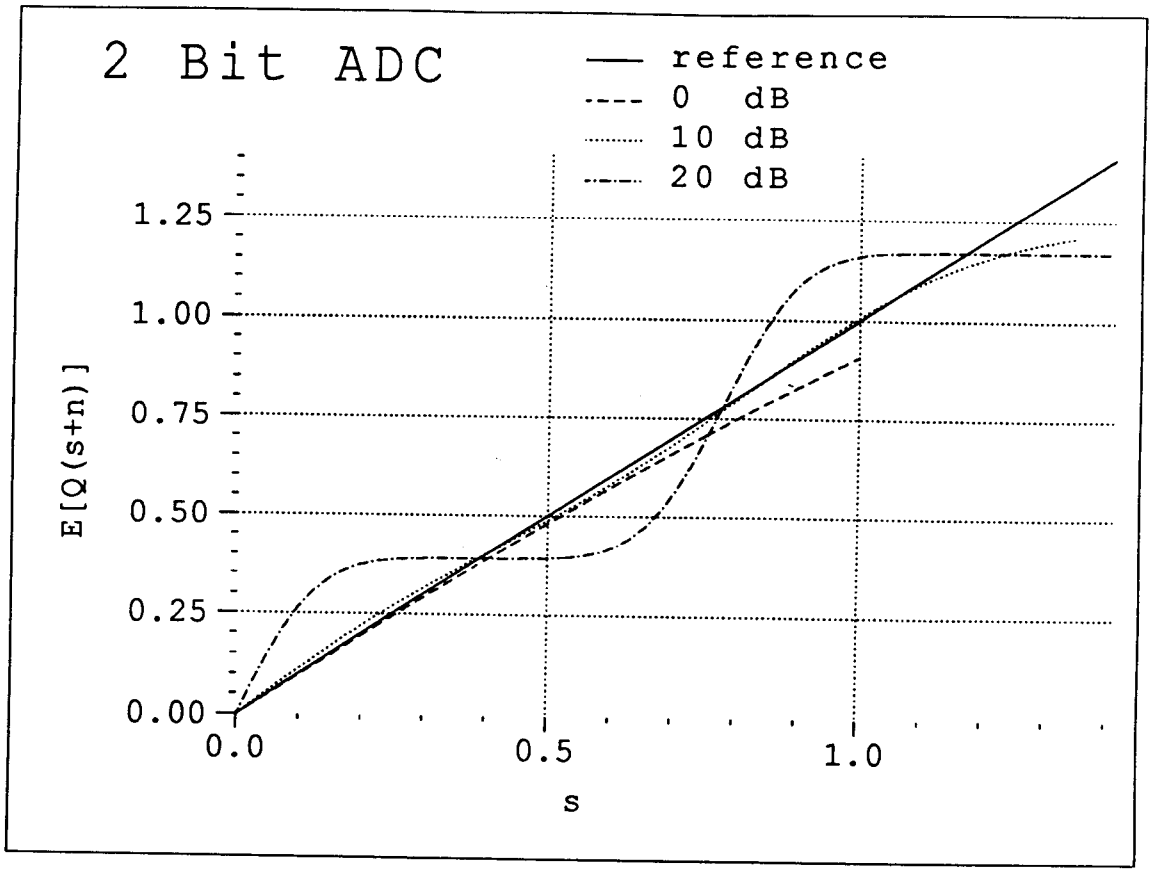


Fig. VIII.1-3a: Useful ADC output signal $s_Q(s)$ as a function of s for 2 bit quantization and input noise power $P_{n,in}$=0.5 , 0.0909 , 0.0099 (SNR=0, 10, 20dB)

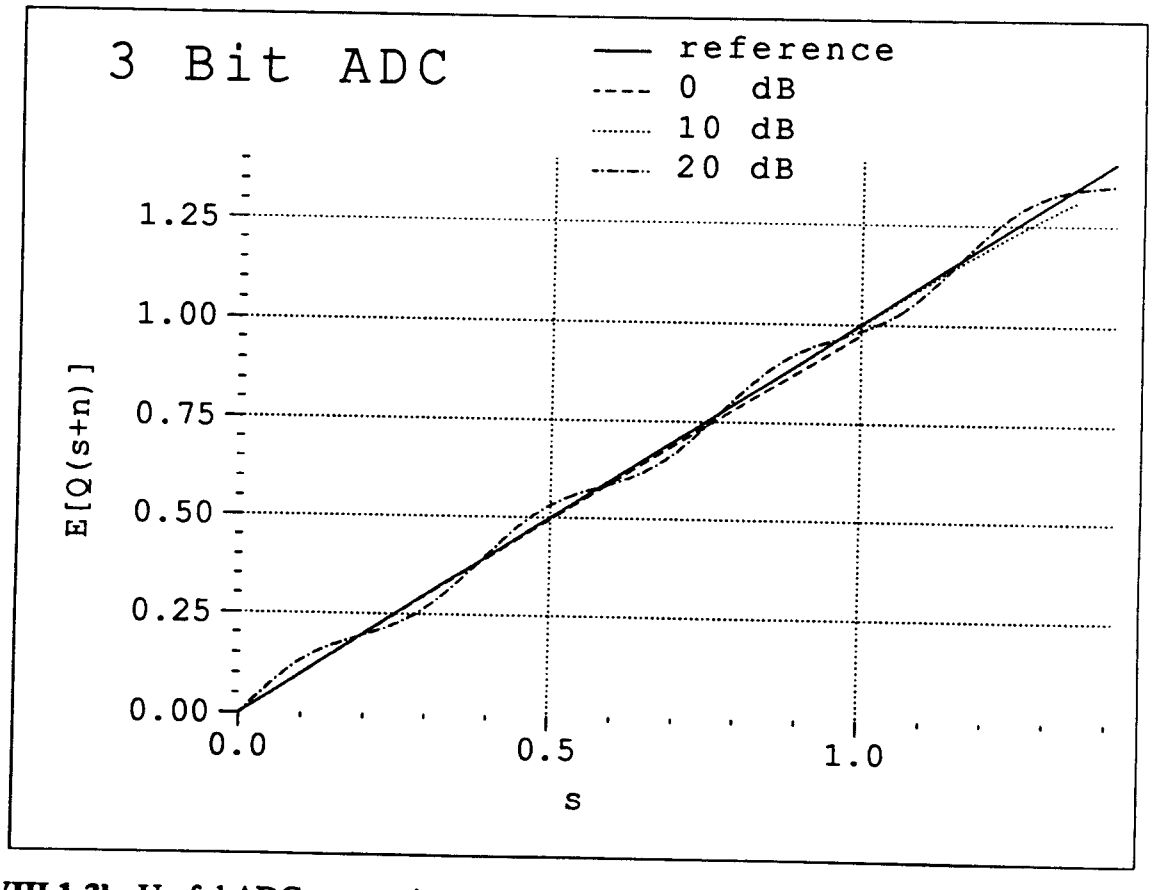


Fig. VIII.1-3b: Useful ADC output signal $s_Q(s)$ as a function of s for 3 bit quantization and input noise power $P_{n,in}$=0.5 , 0.0909 , 0.0099 (SNR=0, 10, 20dB)

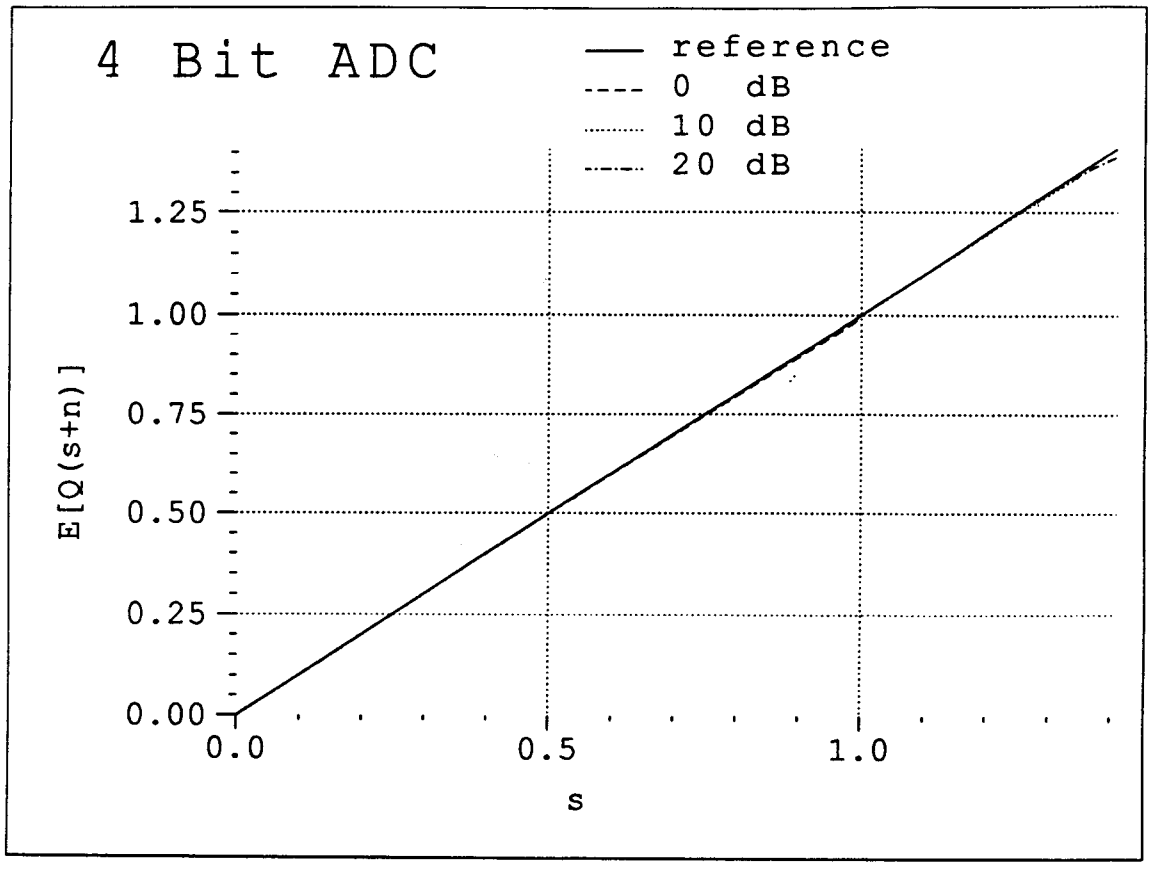


Fig. VIII.1-3c: Useful ADC output signal $s_Q(s)$ as a function of s for 4 bit quantization and input noise power $P_{n,in}$=0.5 , 0.0909 , 0.0099 (SNR=0, 10, 20dB)

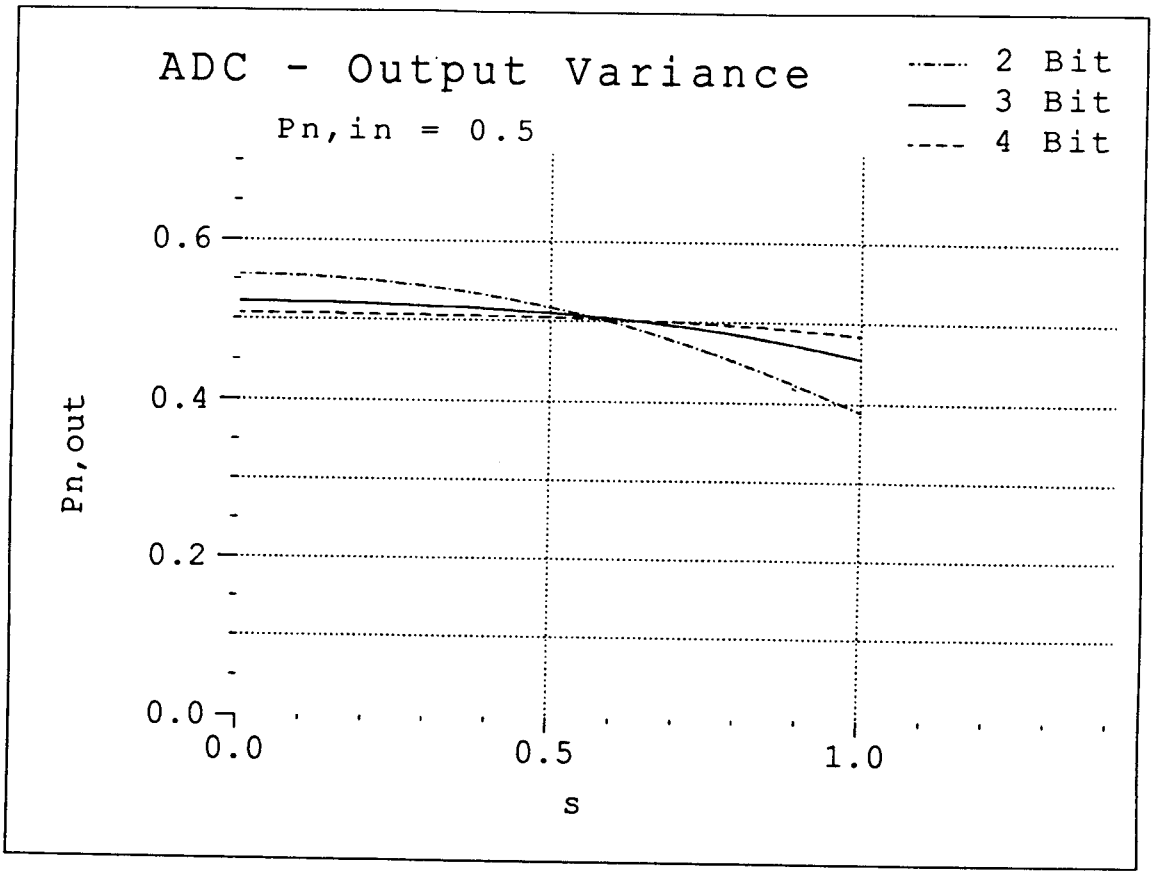


Fig. VIII.1-4a: ADC output noise power $P_{n,out} = E[Q^2(s+n)]-E^2[Q(s+n)]$ as a function of s for 2, 3, and 4 bit quantization and an input noise power $P_{n,in}$=0.5 (SNR=0dB)

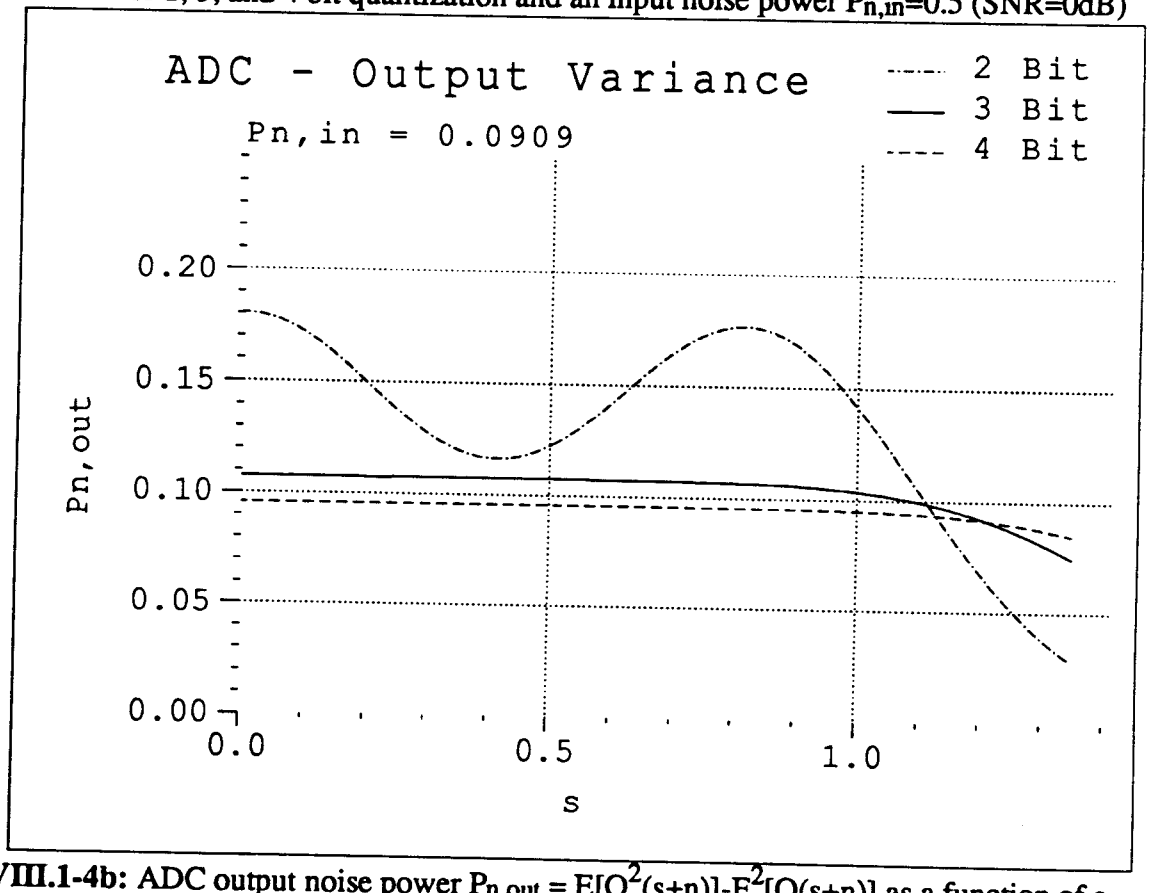


Fig. VIII.1-4b: ADC output noise power $P_{n,out} = E[Q^2(s+n)]-E^2[Q(s+n)]$ as a function of s for 2, 3 and 4 bit quantization and an input noise power $P_{n,in}$=0.0909 (SNR=10dB)

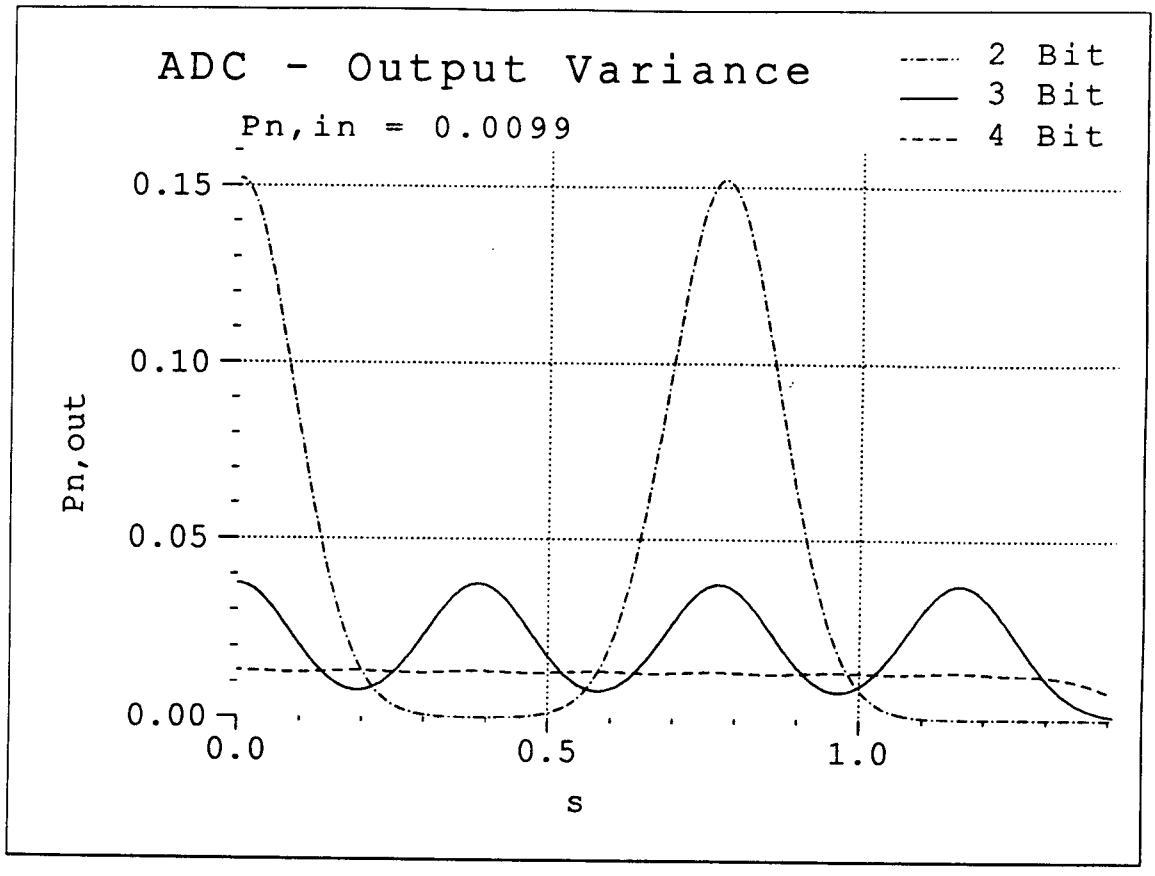


Fig. VIII.1-4c: ADC output noise power $P_{n,out} = E[Q^2(s+n)]-E^2[Q(s+n)]$ as a function of s for 2, 3, and 4 bit quantization and an input noise power $P_{n,in}$=0.0099 (SNR=20dB)

Output = k *Input

Example

$K = 133 = 1*2^7 + 1*2^2 + 1*2^0$

MSB LSB

Input Data

+

Input Data | 0 0

+

Input Data | 0 0 0 0 0 0 0

= Output Data

General

$K = 1*2^{n2} + 1*2^{n1} + 1*2^{n0}$

Input Data

+

Input Data | 0 . . . 0 ($n_1 - n_0$)

+

Input Data | 0 . . . . . .0 ($n_2 - n_0$)

= Output Data | 0 . . . 0 (n_0)

Fig. VIII.1-5: Efficient coefficient multiplication method for small number of 1's in coefficient

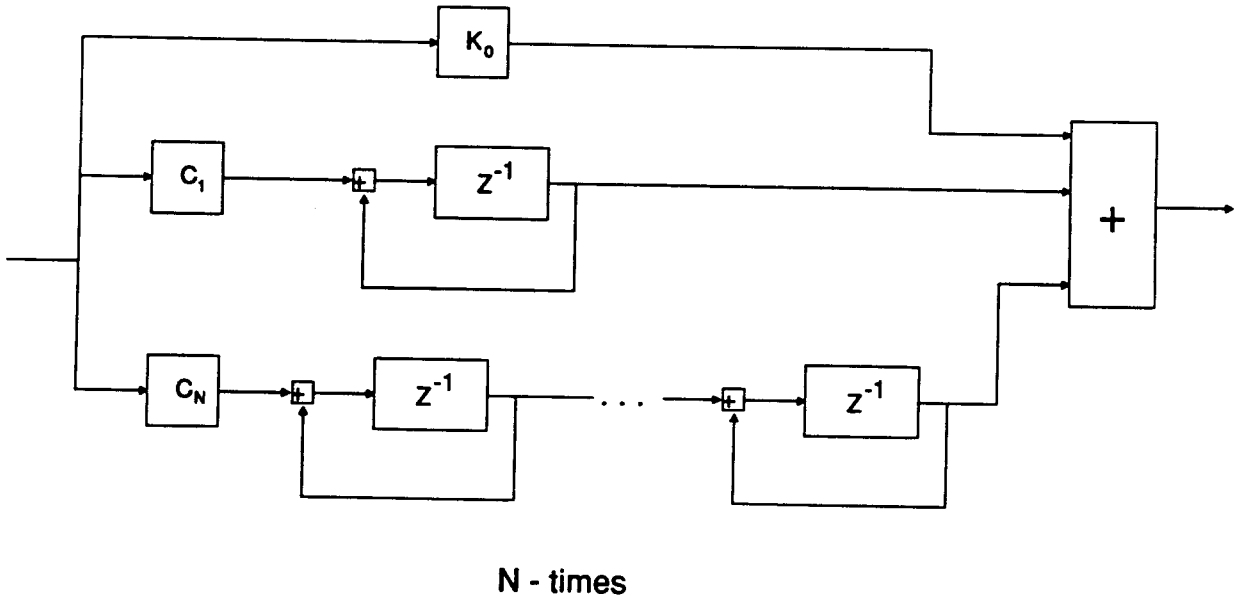


Fig. VIII.1-6: Structure of feedback loop filter

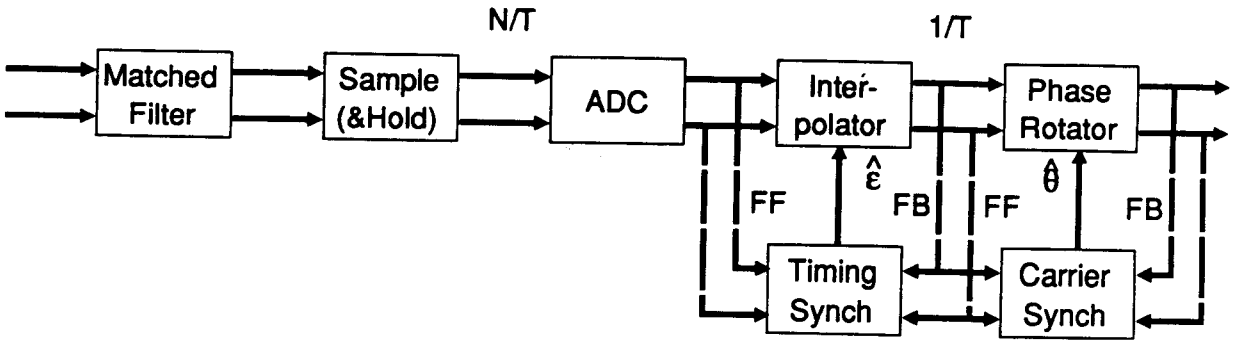


Fig. VIII.1-7a: Quantization scenarios studied by simulation: quantization **after** the matched filter

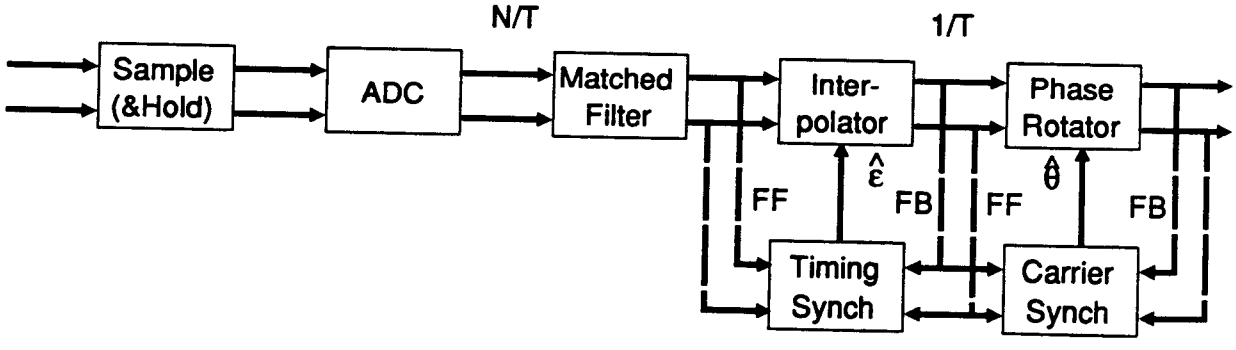


Fig. VIII.1-7b: Quantization scenarios studied by simulation: quantization **before** the matched filter

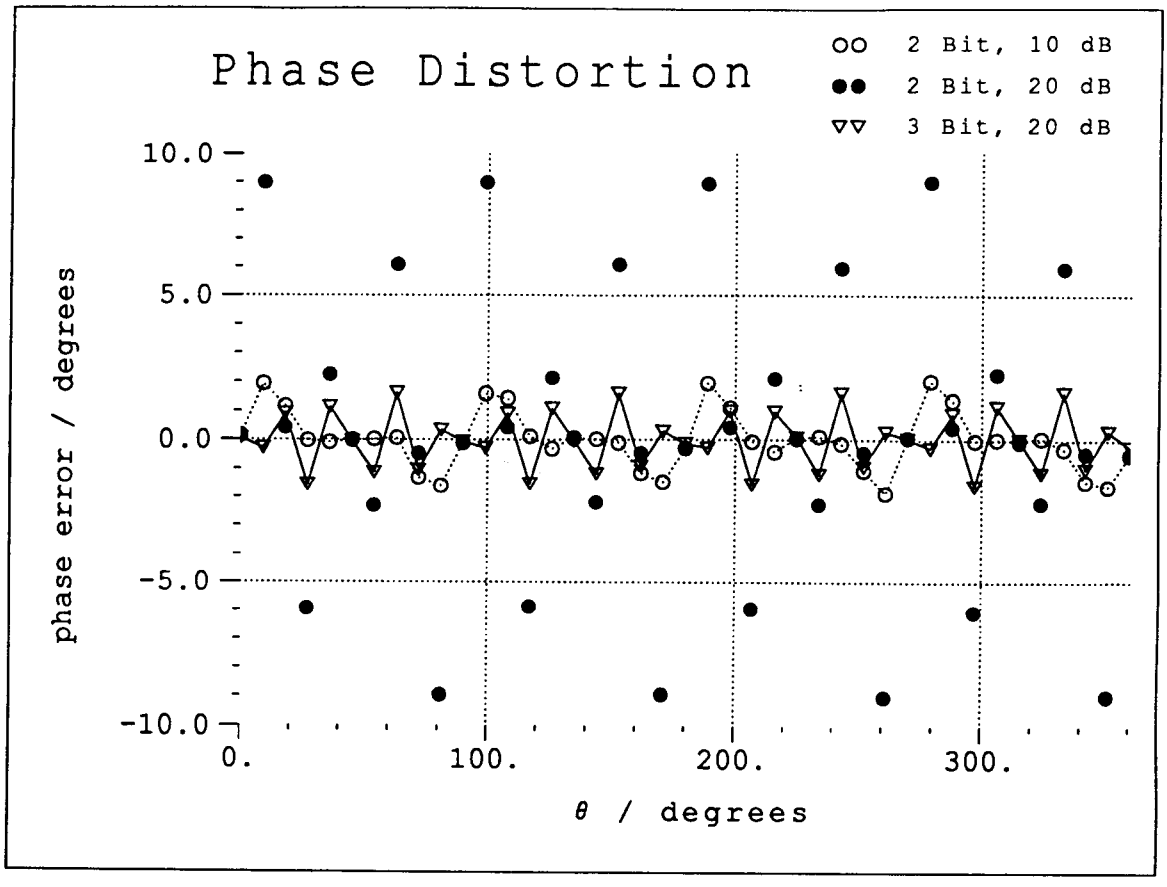


Fig. VIII.2-1a: Mean phase error due to quantization for sinusoidal input

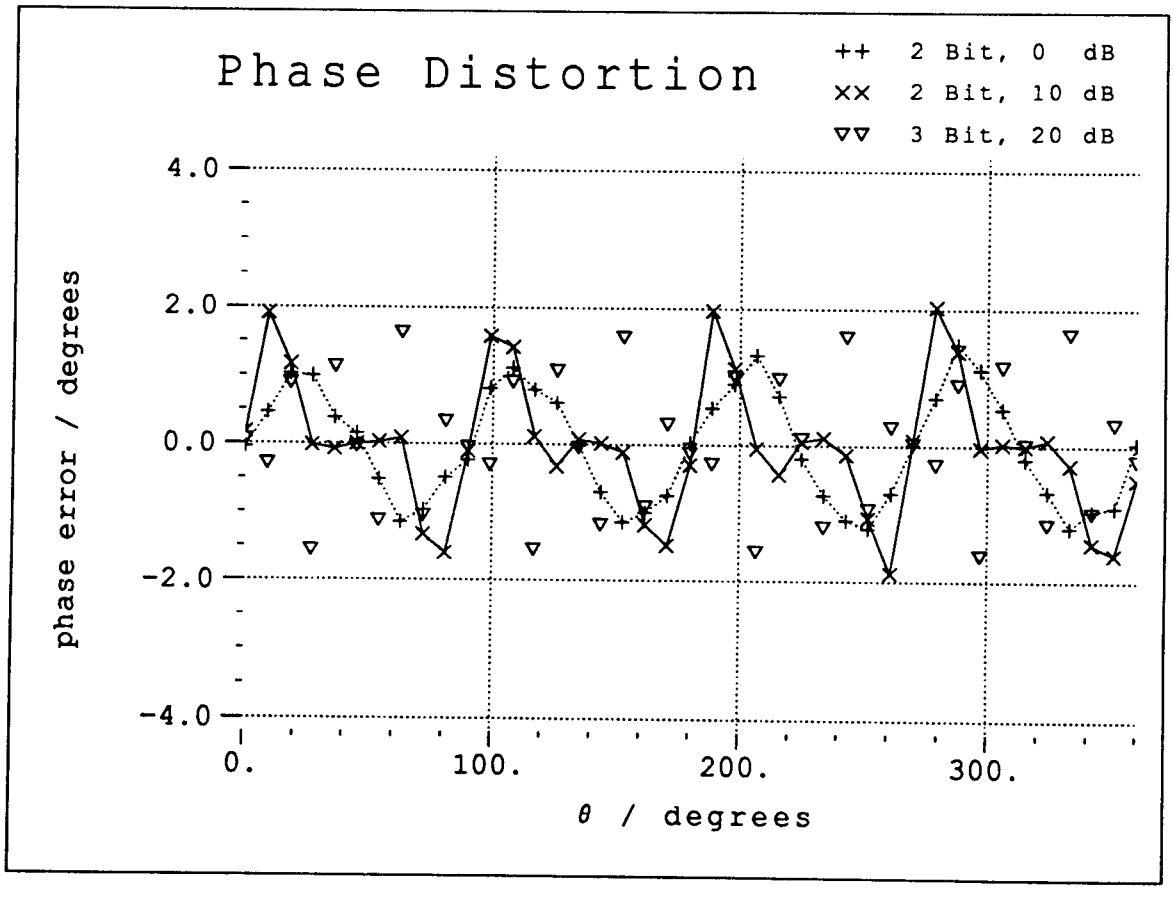


Fig. VIII.2-1b: Mean phase error due to quantization for sinusoidal input

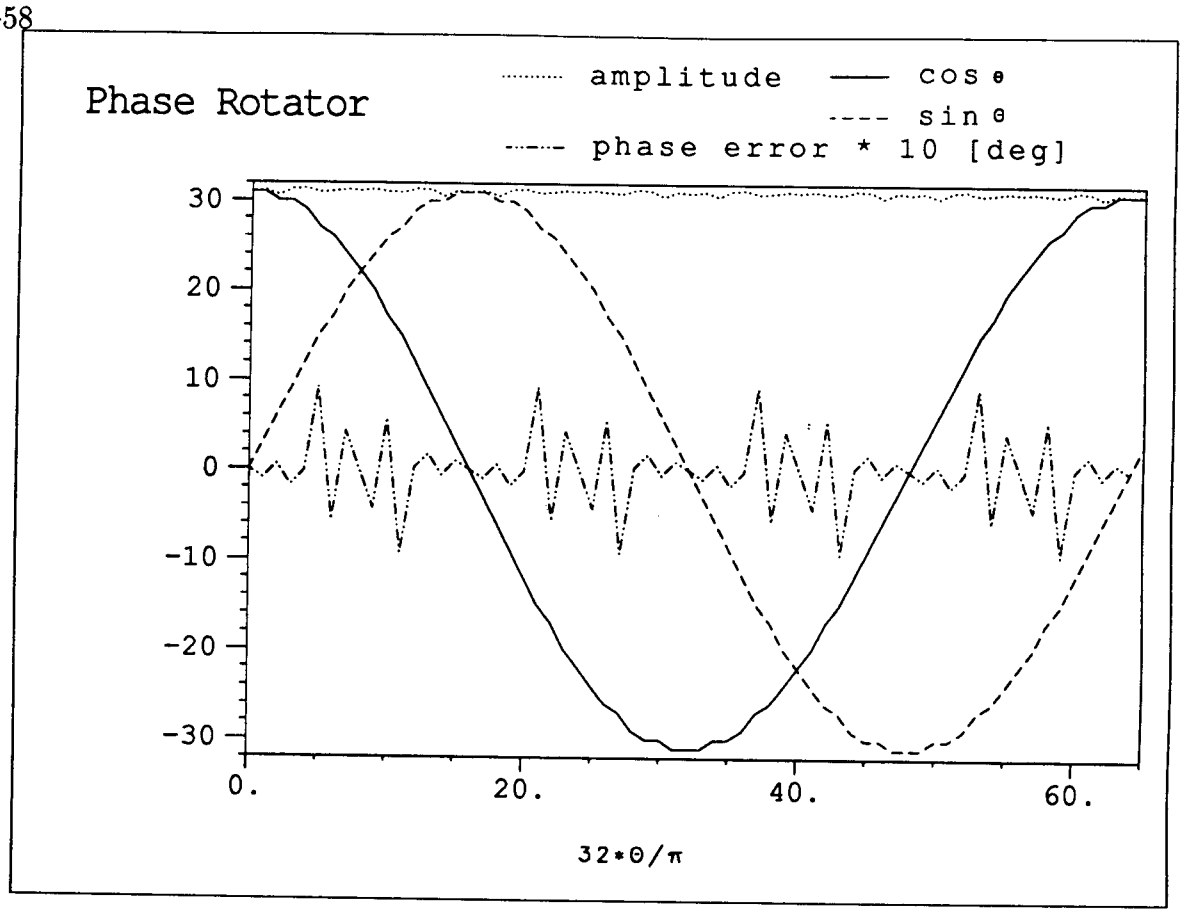


Fig. VIII.2-1c: Output of a 6 bit sine/cosine table and corresponding phase error (10 times magnified) in degrees

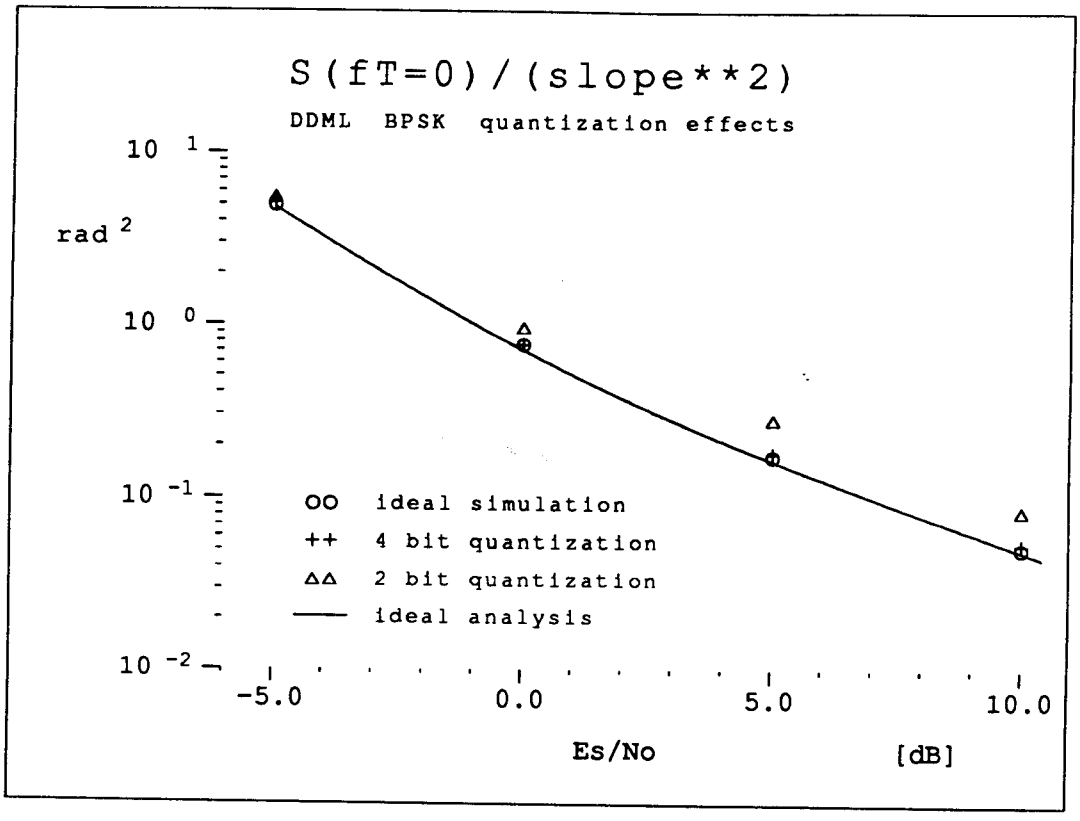


Fig. VIII.2-2a: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ of DDML for BPSK, matched filter output quantized, logarithmic scale

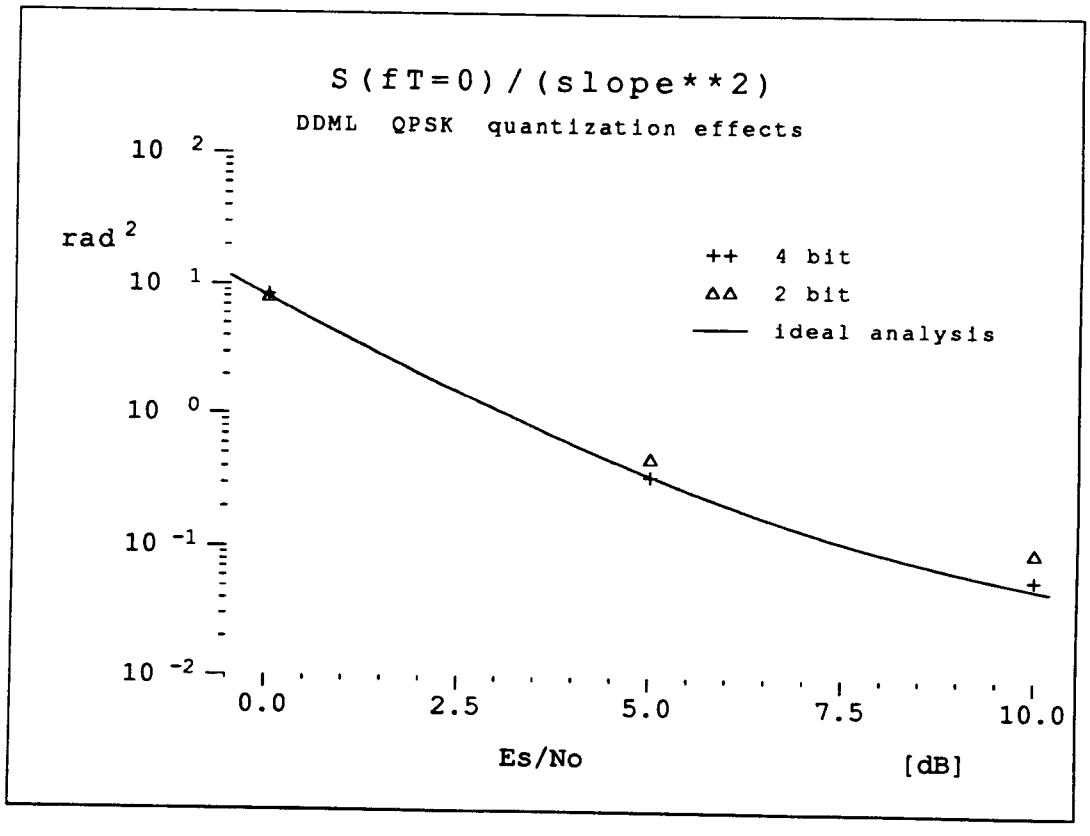


Fig. VIII.2-2b: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ of DDML for QPSK, matched filter output quantized, logarithmic scale

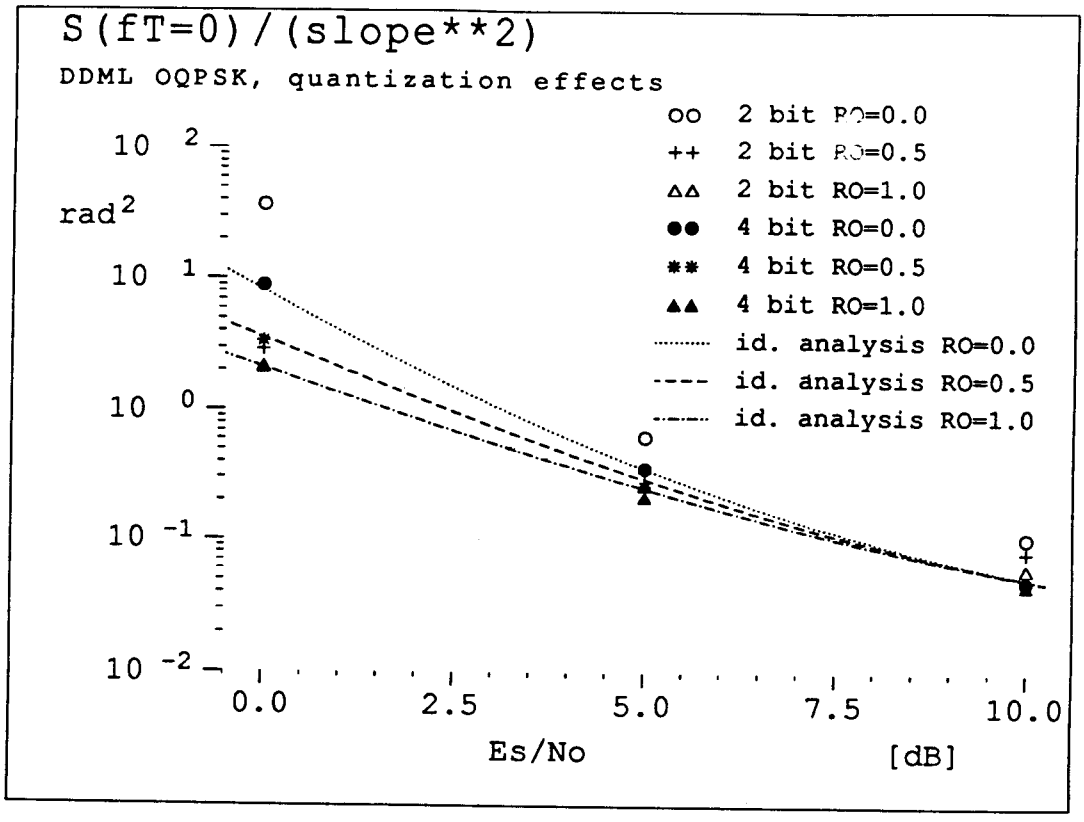


Fig. VIII.2-2c: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ of DDML for OQPSK, matched filter output quantized, logarithmic scale

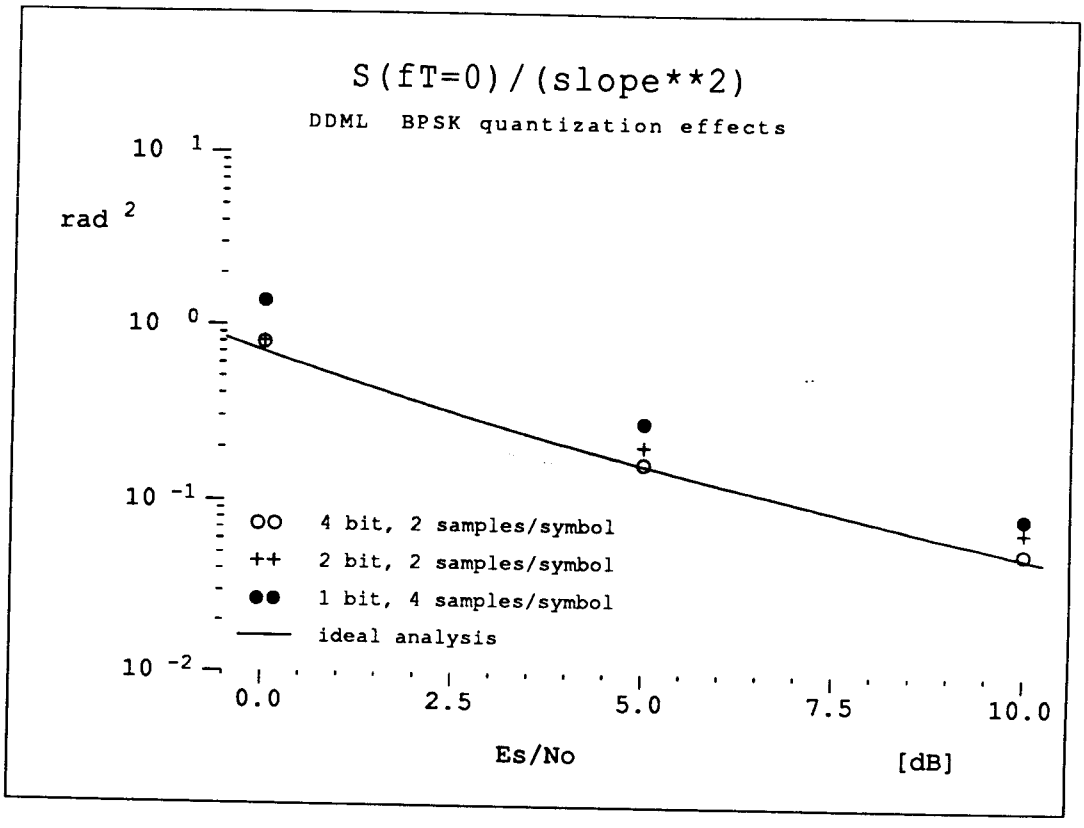


Fig. VIII.2-3a: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ of DDML for BPSK, quantization before matched filter, logarithmic scale

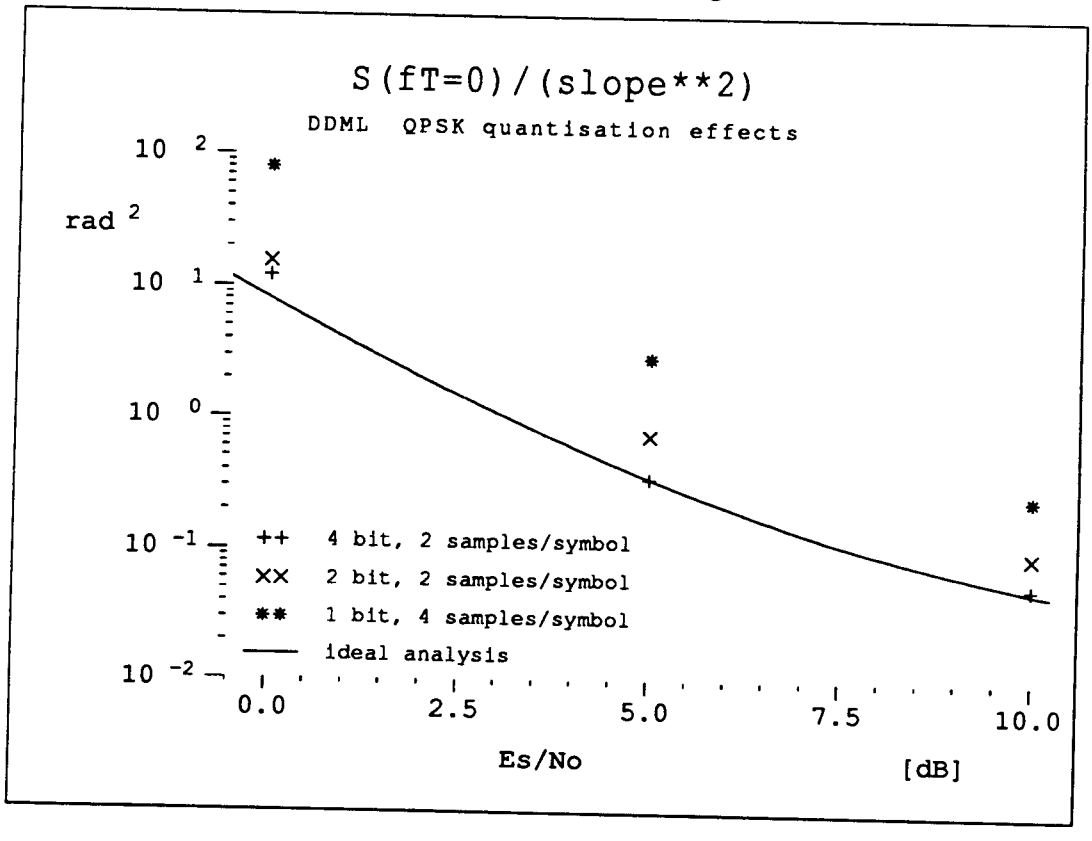


Fig. VIII.2-3b: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ of DDML for QPSK, quantization before matched filter, logarithmic scale

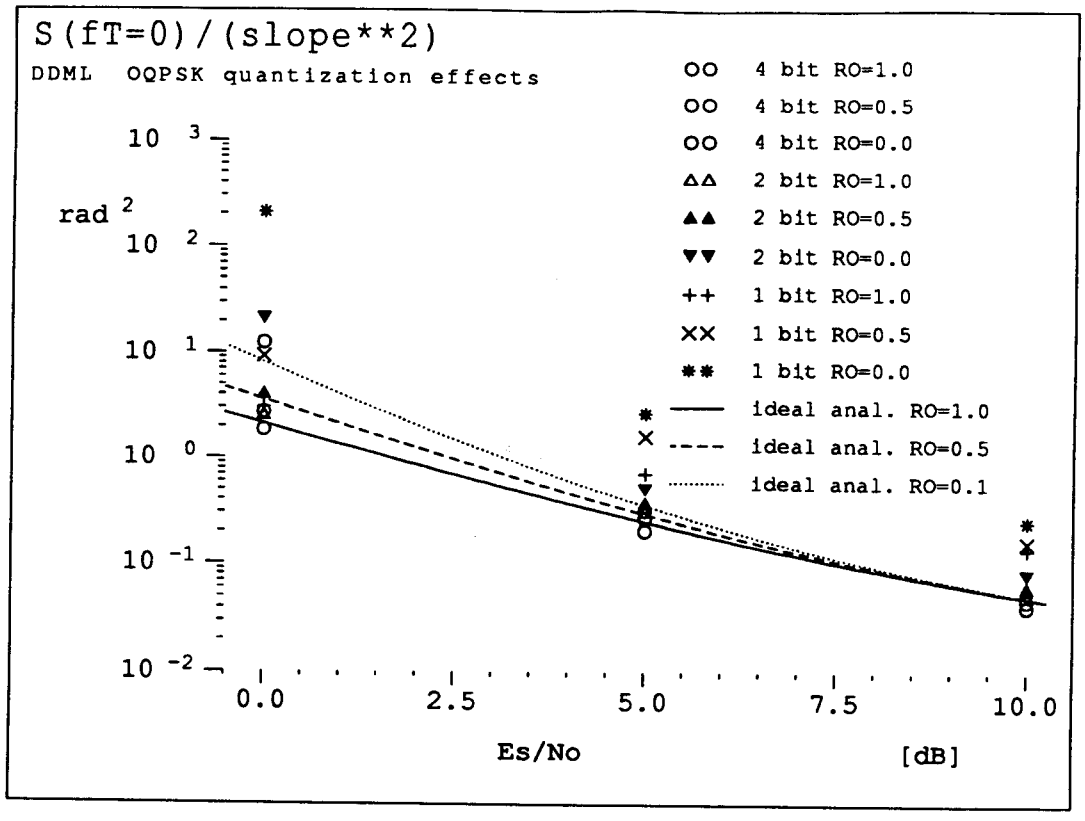


Fig. VIII.2-3c: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ of DDML for OQPSK, quantization before matched filter, logarithmic scale, for 4 bit / 2 samples per symbol, 2 bit / 2 samples per symbol, and 1 bit / 4 samples per symbol

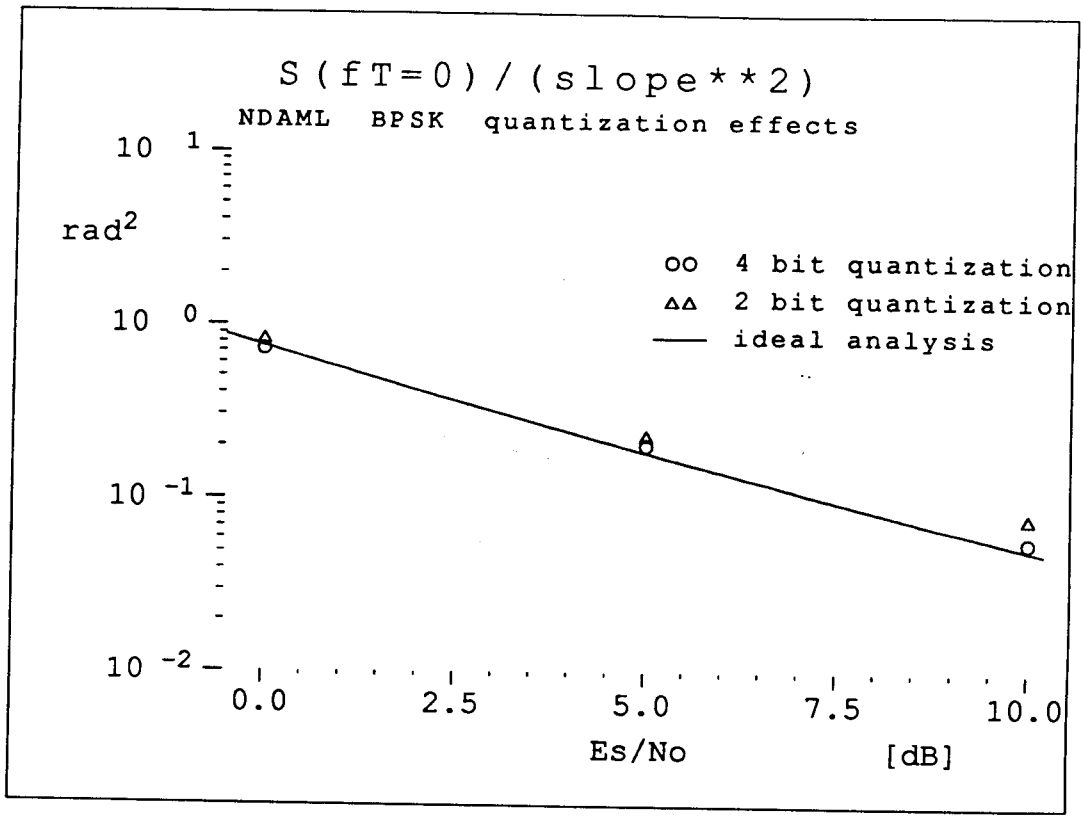


Fig. VIII.2-4a: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ of NDAML for BPSK, matched filter output quantized, logarithmic scale

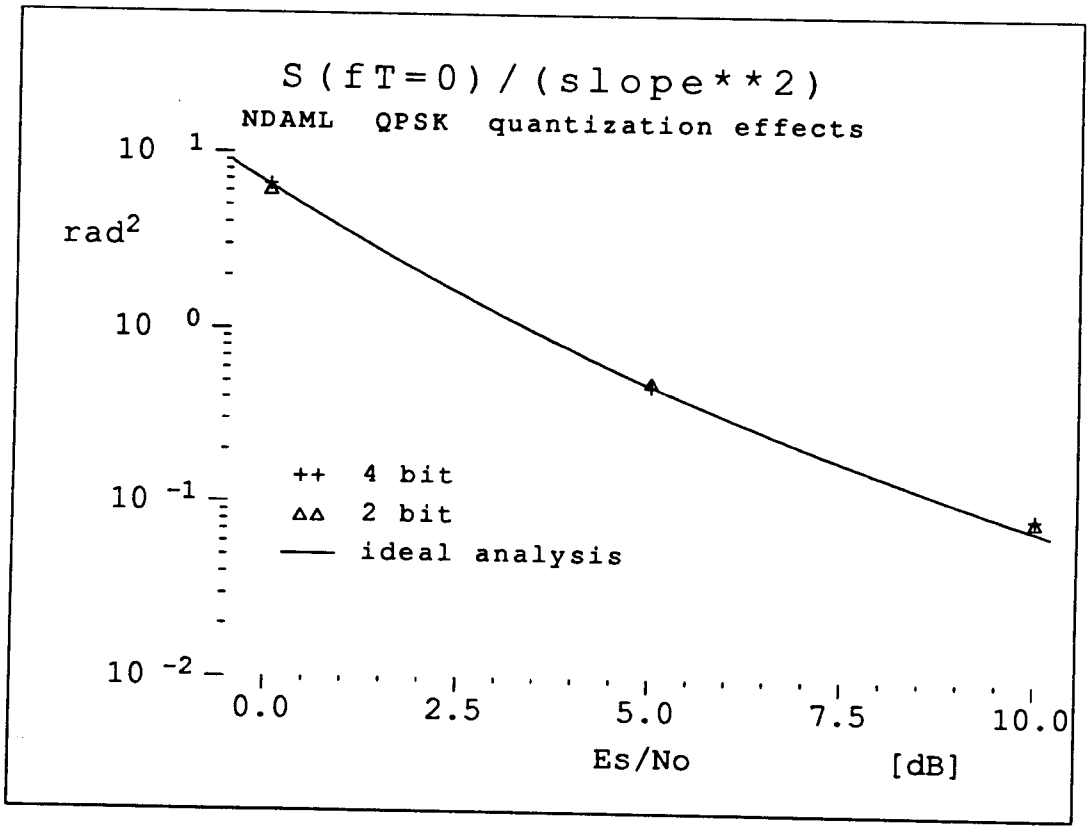


Fig. VIII.2-4b: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ of NDAML for QPSK, matched filter output quantized, logarithmic scale

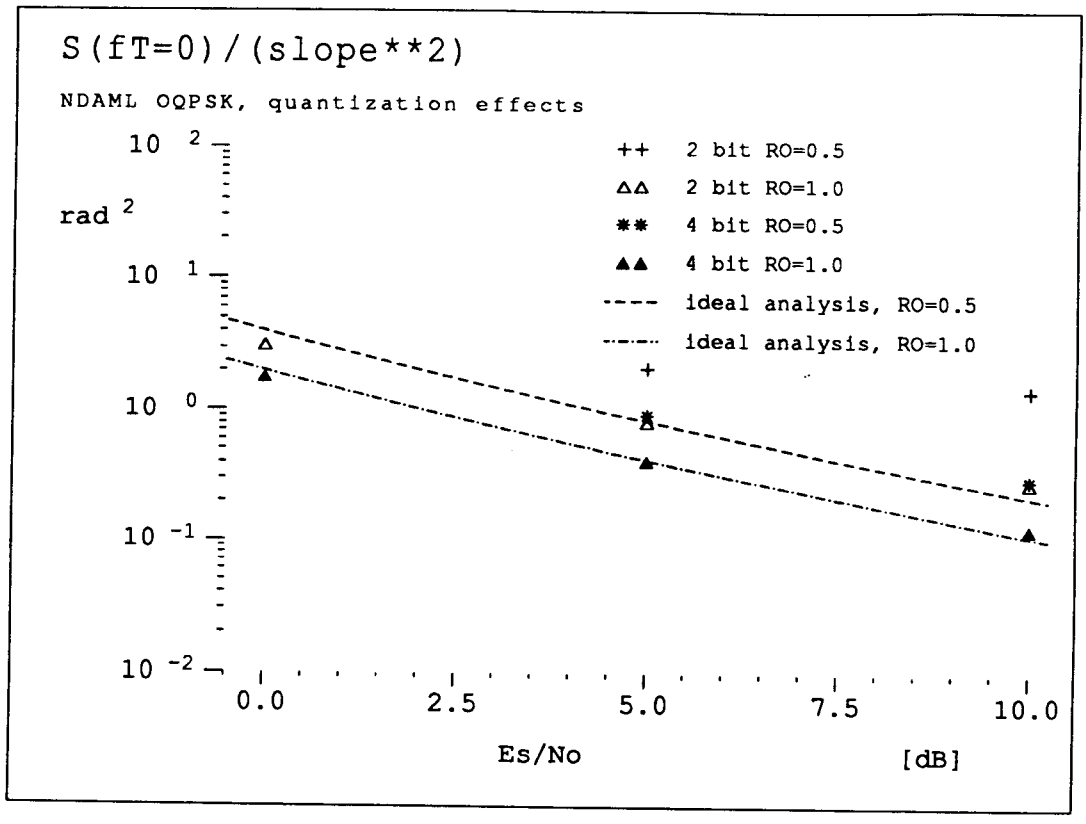


Fig. VIII.2-4c: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ of NDAML for OQPSK, matched filter output quantized, logarithmic scale

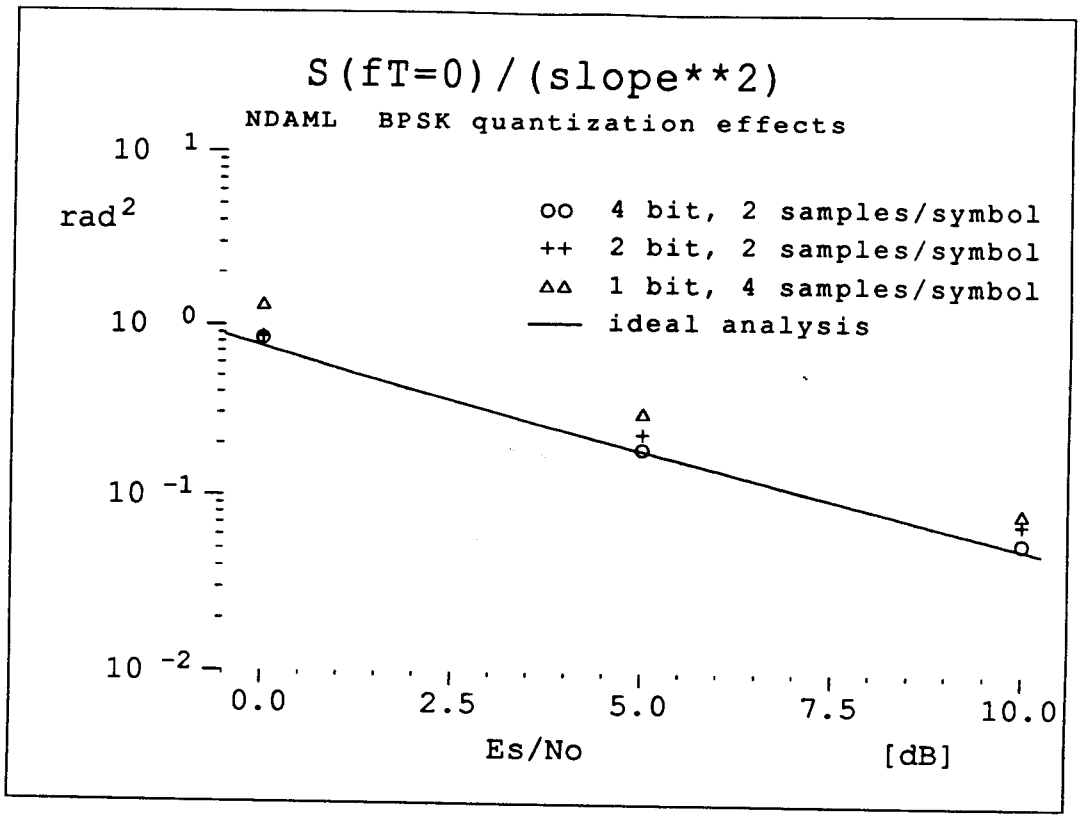


Fig. VIII.2-5a: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ of NDAML for BPSK, quantization before matched filter, logarithmic scale

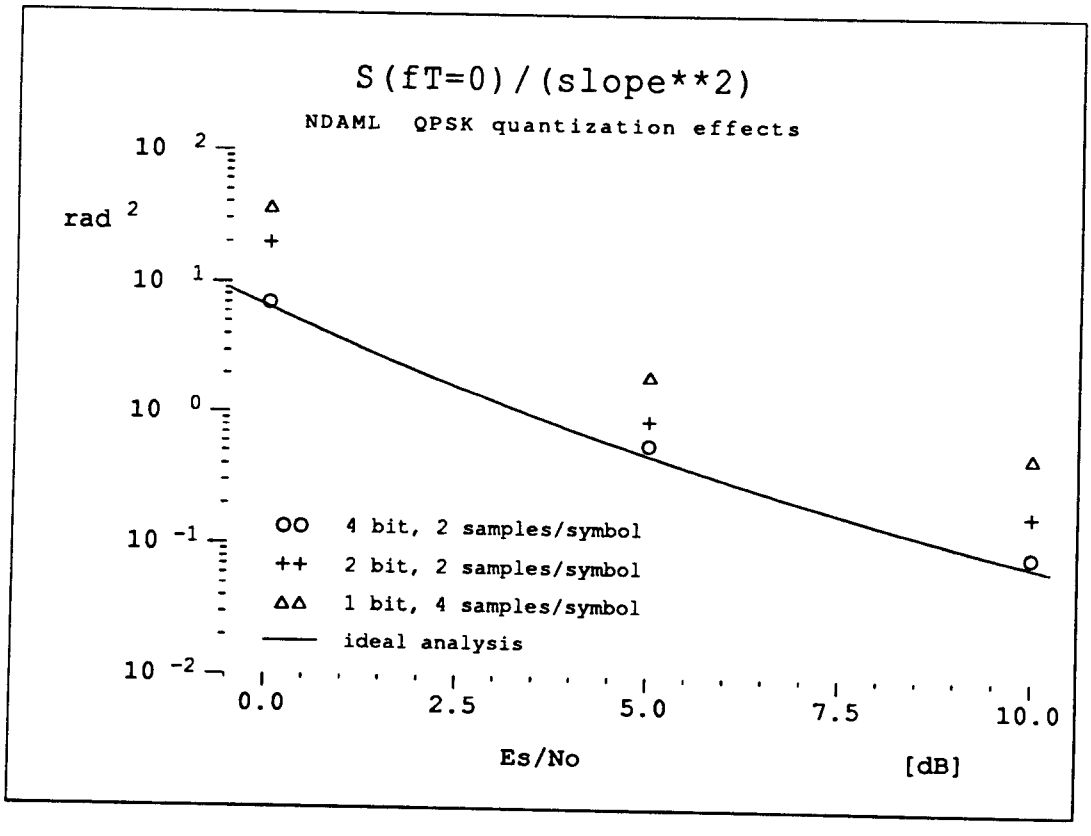


Fig. VIII.2-5b: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ of NDAML for QPSK, quantization before matched filter, logarithmic scale

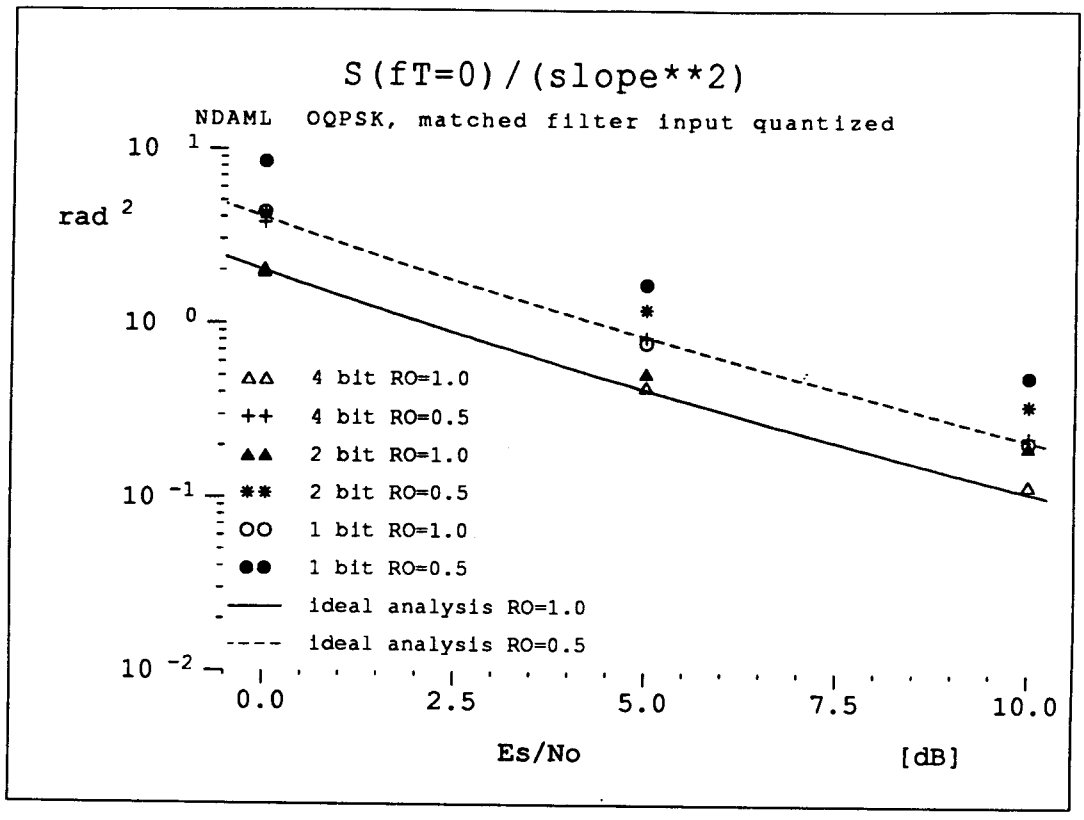


Fig. VIII.2-5c: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ of NDAML for OQPSK, quantization before matched filter, logarithmic scale, for 4 bit / 2 samples per symbol and 2 bit / 2 samples per symbol

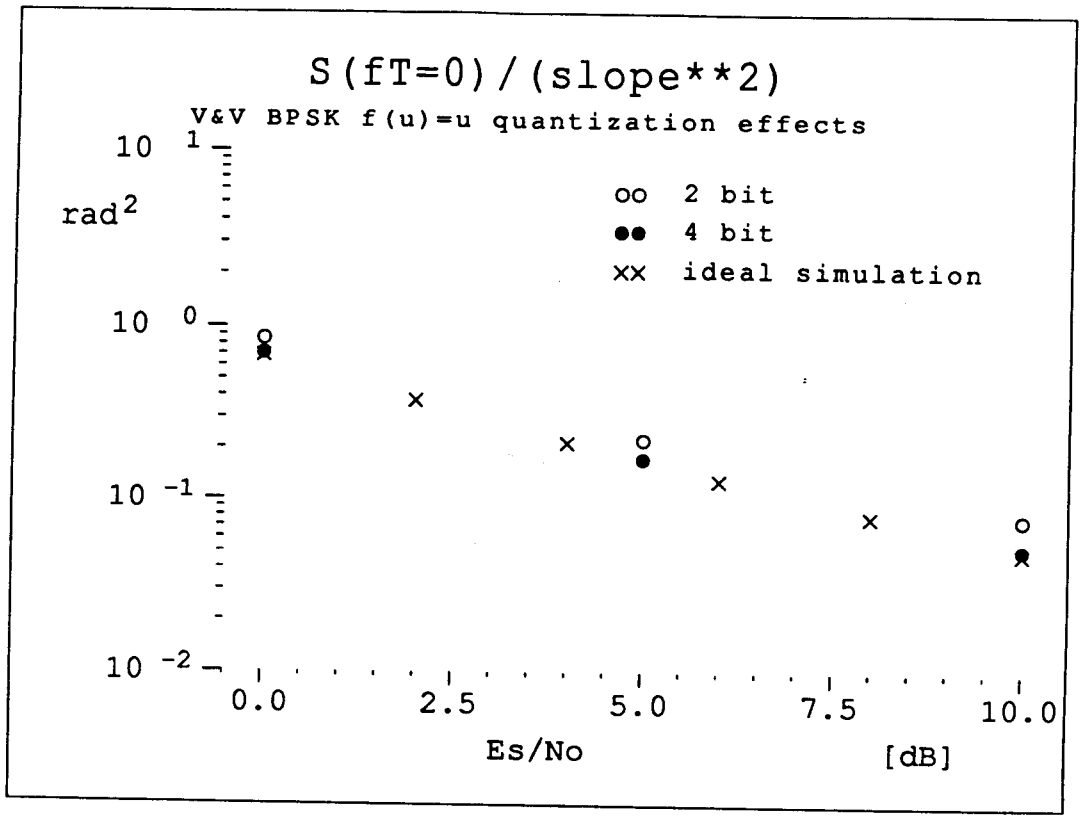


Fig. VIII.2-6a: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ of V&V for BPSK, f(u)=u, matched filter output quantized, logarithmic scale

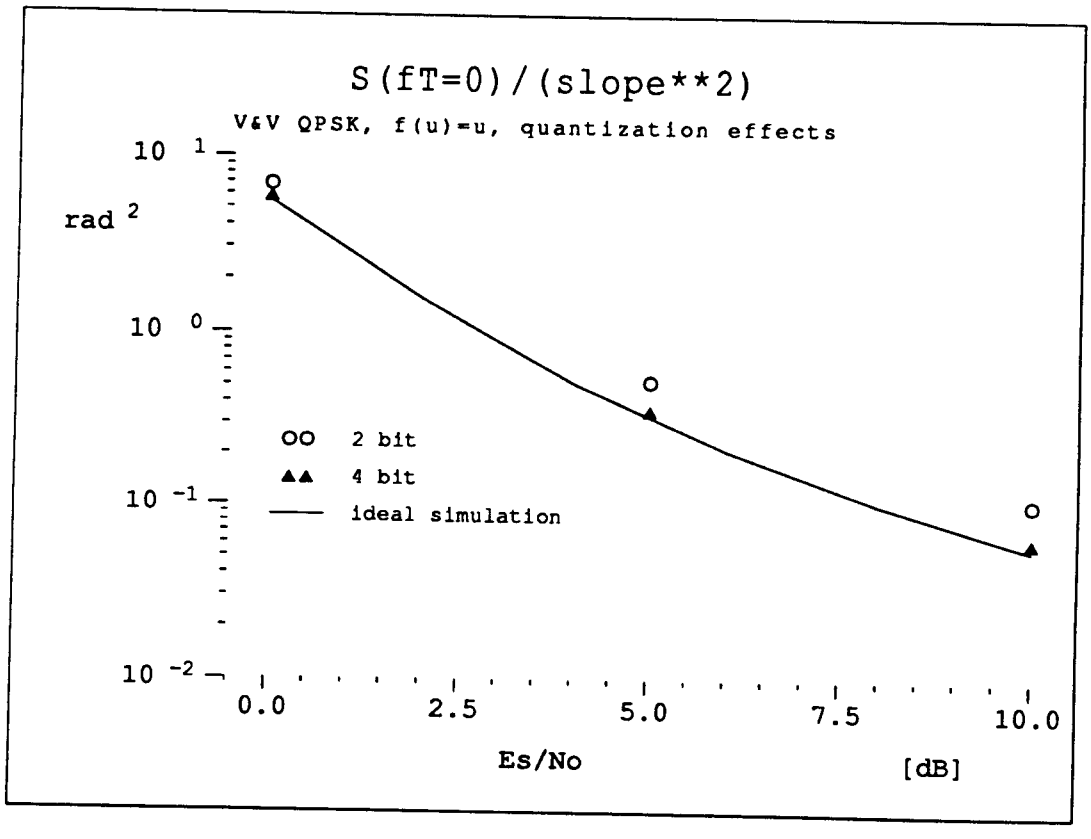


Fig. VIII.2-6b: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ of V&V for QPSK, f(u)=u, matched filter output quantized, logarithmic scale

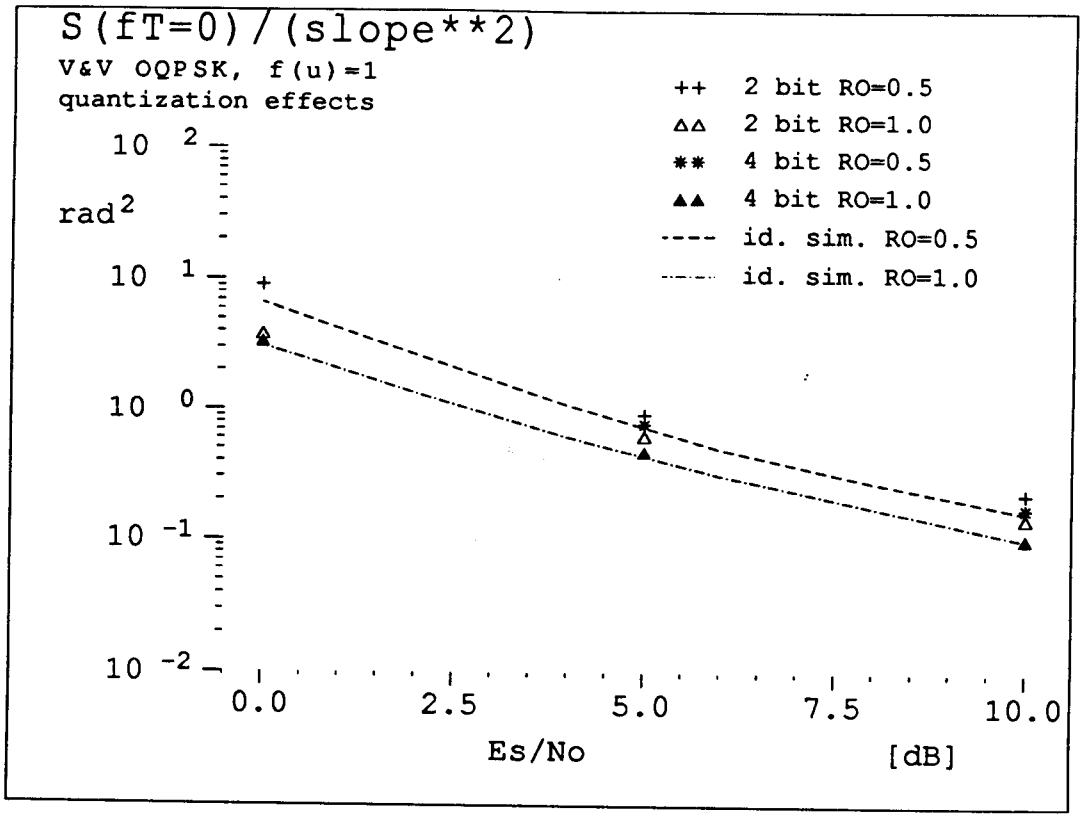


Fig. VIII.2-6c: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ of V&V for OQPSK, f(u)=1, matched filter output quantized, logarithmic scale

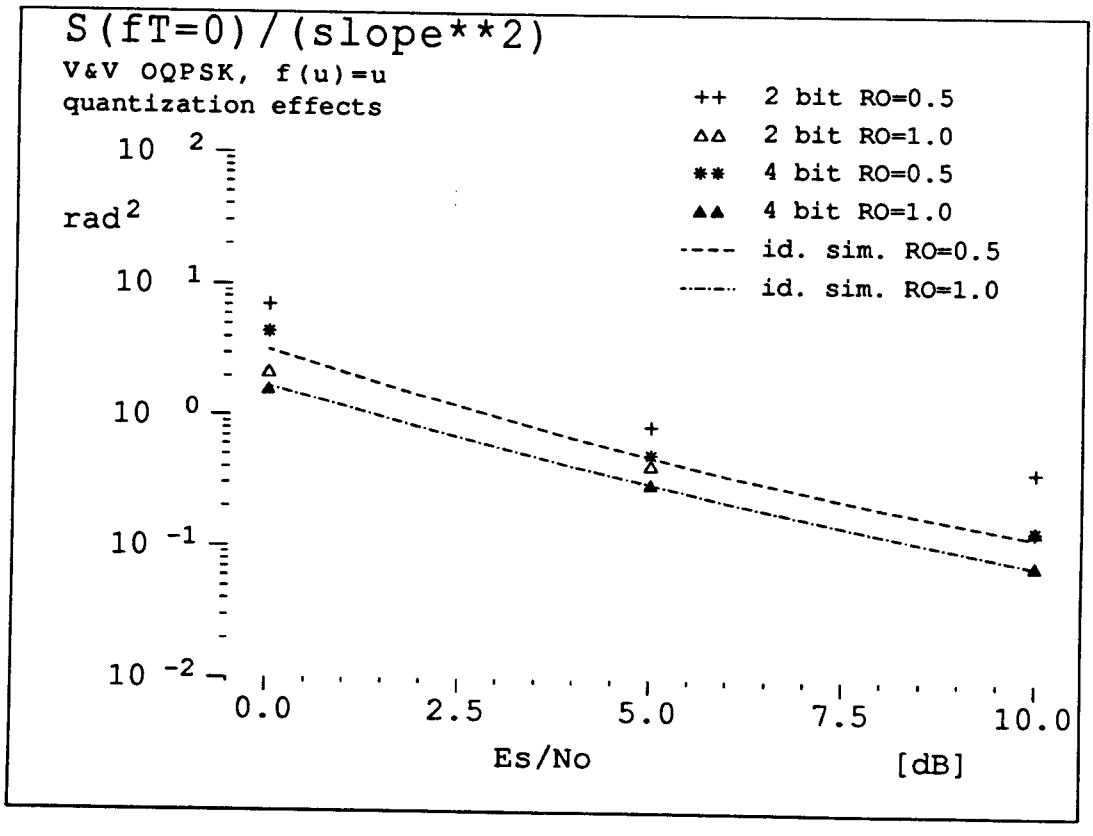


Fig. VIII.2-6d: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ of V&V for OQPSK, f(u)=u, matched filter output quantized, logarithmic scale

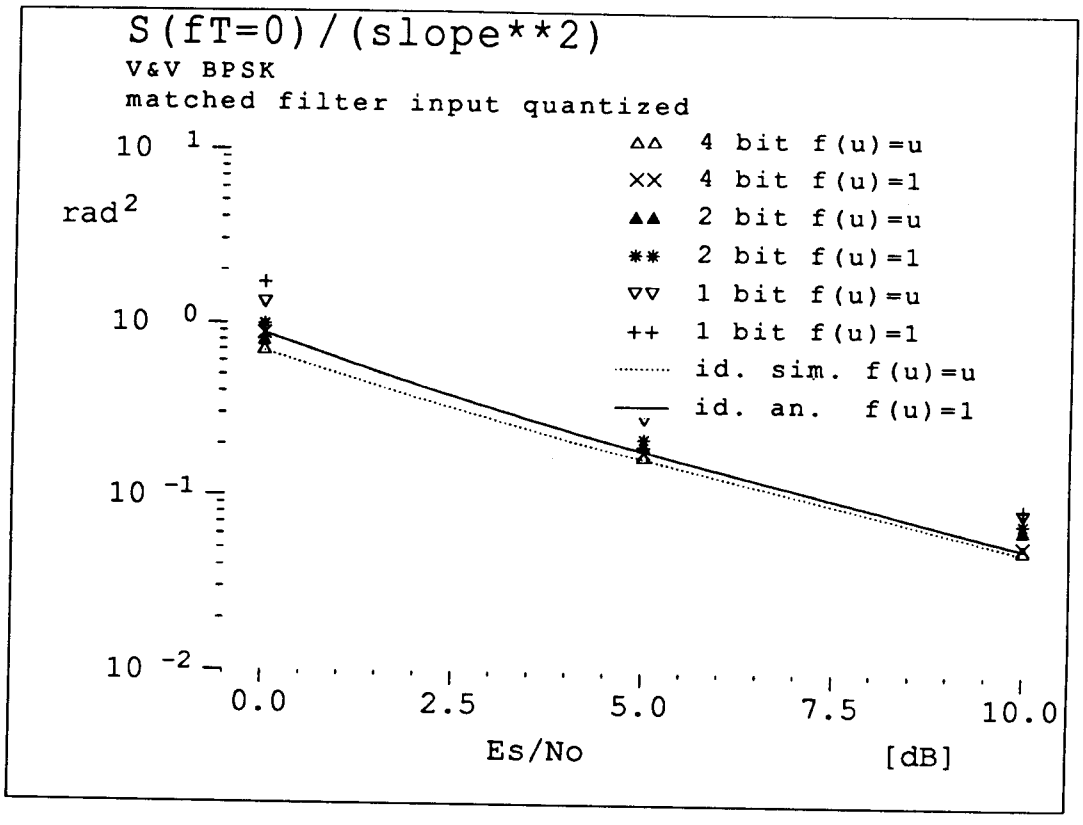


Fig. VIII.2-7a: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ of V&V for BPSK, f(u)=u, quantization before matched filter, logarithmic scale, for 4 bit / 2 samples/symbol and 2 bit / 2 samples/symbol, and 1 bit / 8 samples/symbol

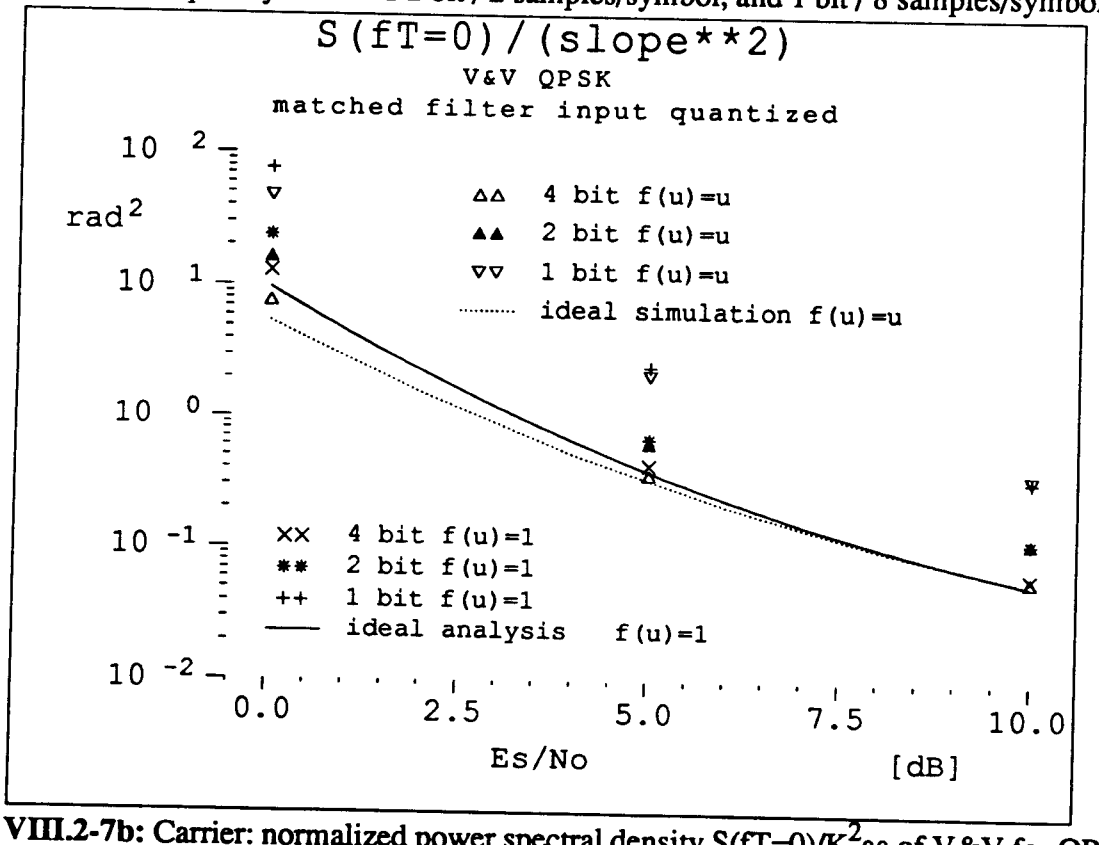


Fig. VIII.2-7b: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ of V&V for QPSK, f(u)=u, quantization before matched filter, logarithmic scale, for 4 bit / 2 samples/symbol and 2 bit / 2 samples/symbol, and 1 bit / 8 samples/symbol

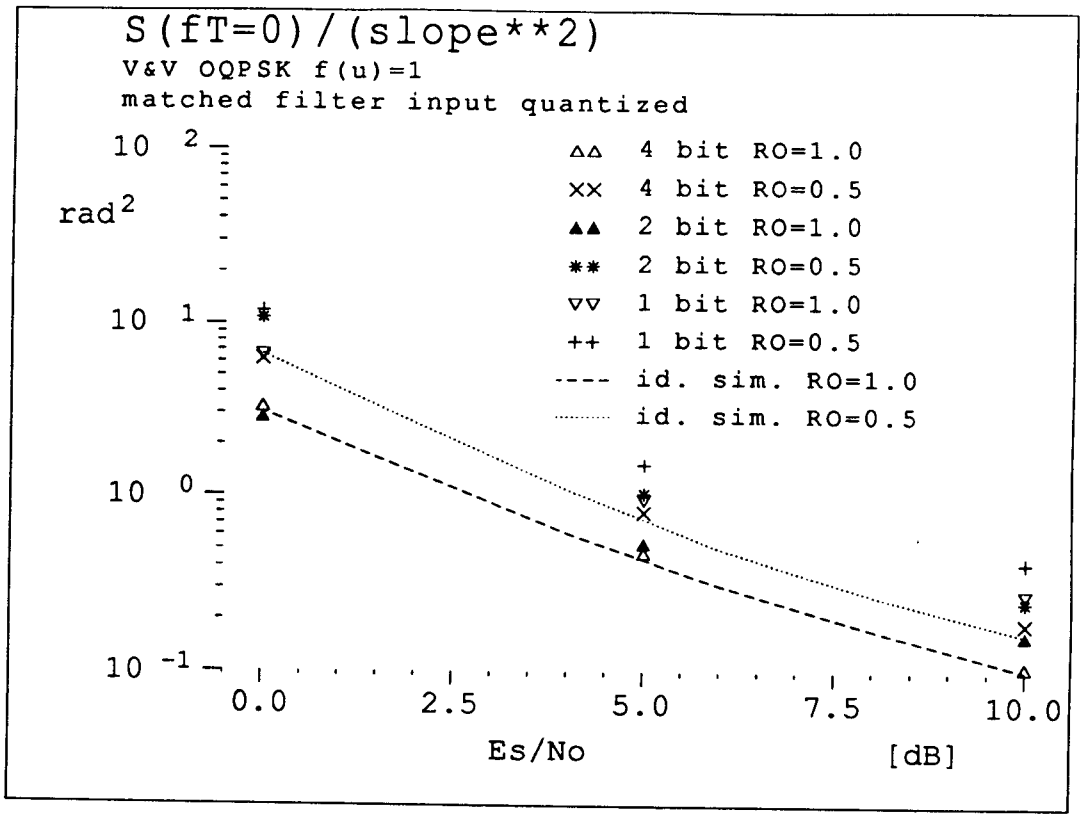


Fig. VIII.2-7c: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ of V&V for OQPSK, f(u)=1, quantization before matched filter, logarithmic scale, for 4 bit / 2 samples per symbol and 2 bit / 2 samples per symbol

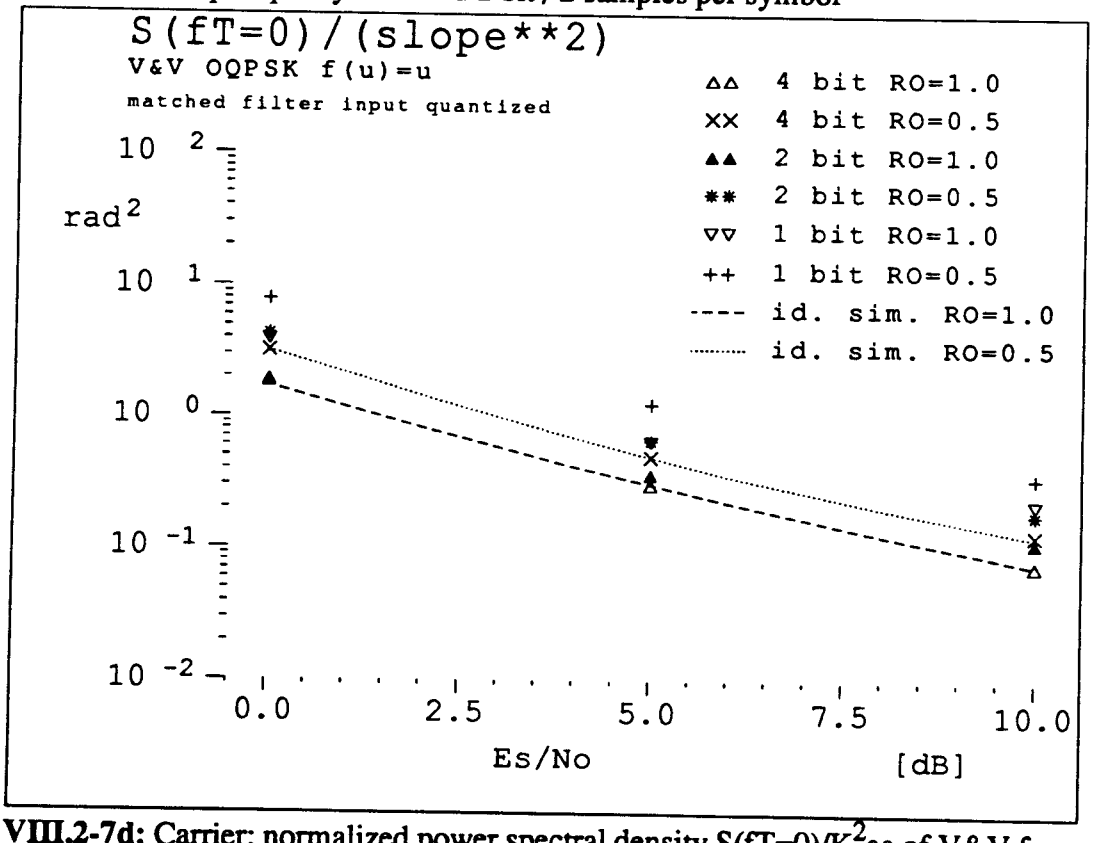


Fig. VIII.2-7d: Carrier: normalized power spectral density $S(fT=0)/K^2_{\theta\theta}$ of V&V for OQPSK, f(u)=u, quantization before matched filter, logarithmic scale, for 4 bit / 2 samples per symbol and 2 bit / 2 samples per symbol

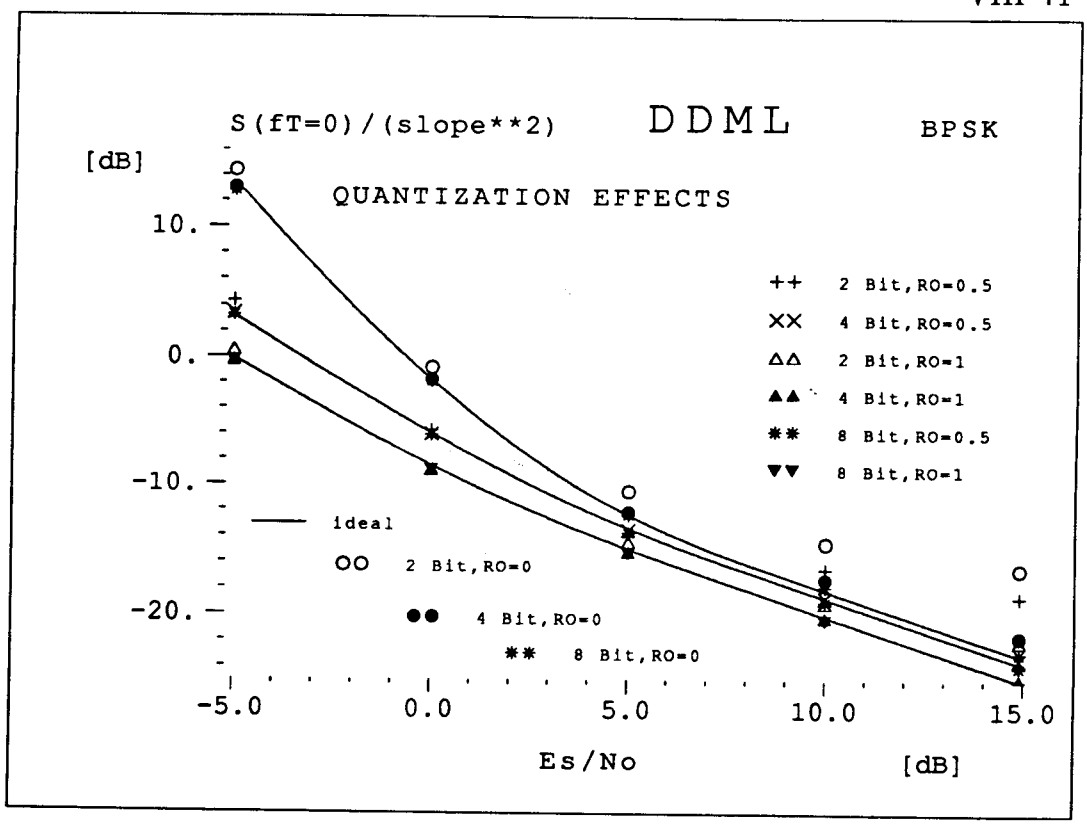


Fig. VIII.3-1a: Timing: normalized power spectral density $S(fT=0)/K^2_{\tau\tau}$ of DDML for BPSK, matched filter output quantized, logarithmic scale

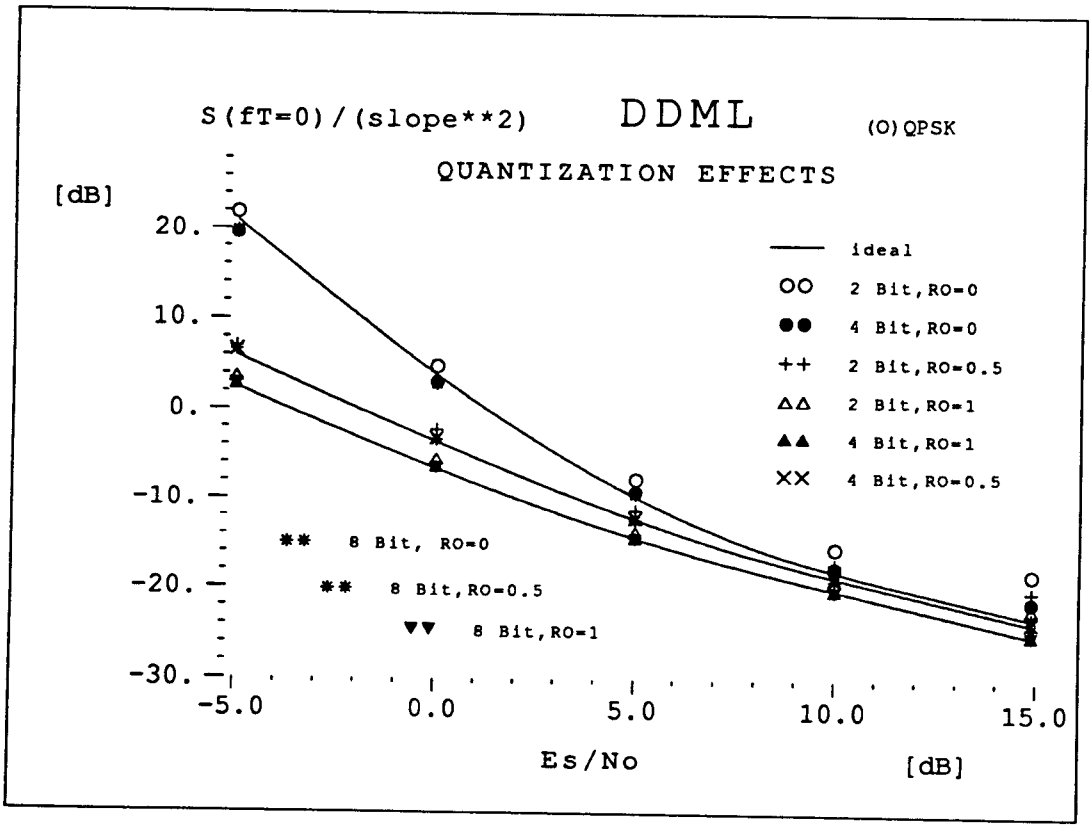


Fig. VIII.3-1b: Timing: normalized power spectral density $S(fT=0)/K^2_{\tau\tau}$ of DDML for (O)QPSK, matched filter output quantized, logarithmic scale

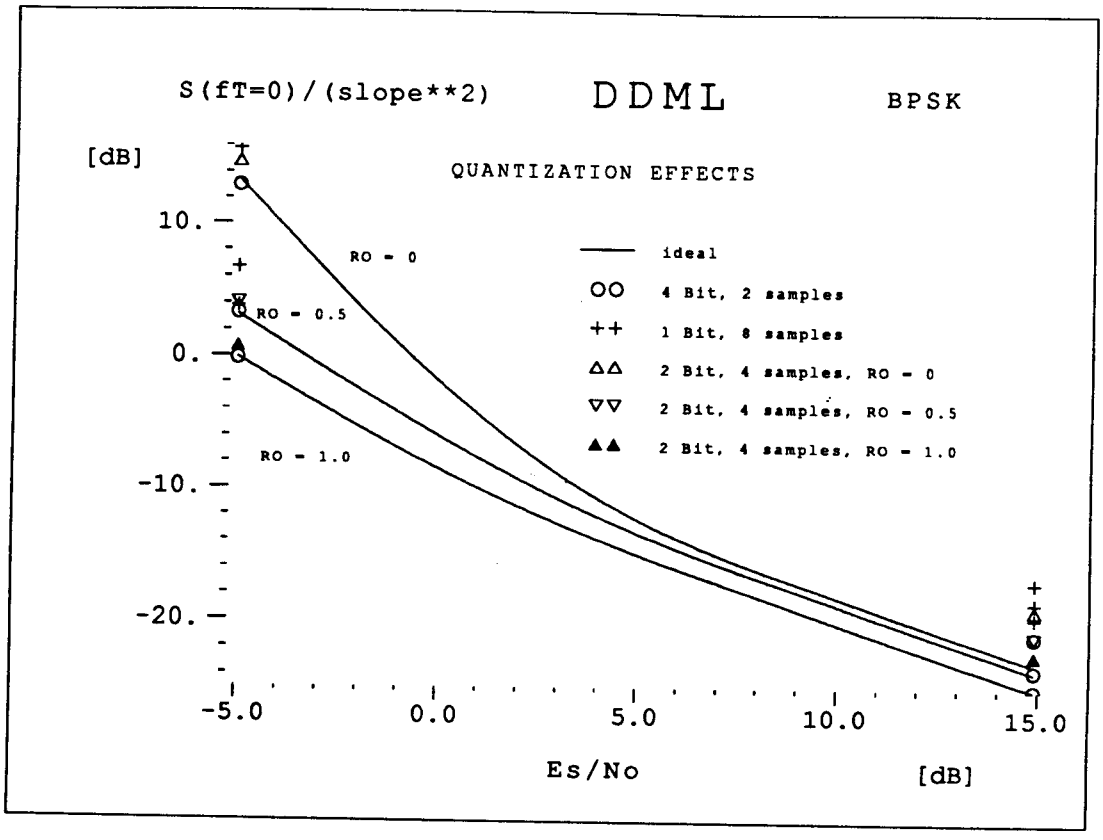


Fig. VIII.3-2a: Timing: normalized power spectral density $S(fT=0)/K^2_{\tau\tau}$ of DDML for BPSK, quantization before matched filter, logarithmic scale, for 2, 4, and 8 samples/symbol and 1, 2, and 4 bit quantization

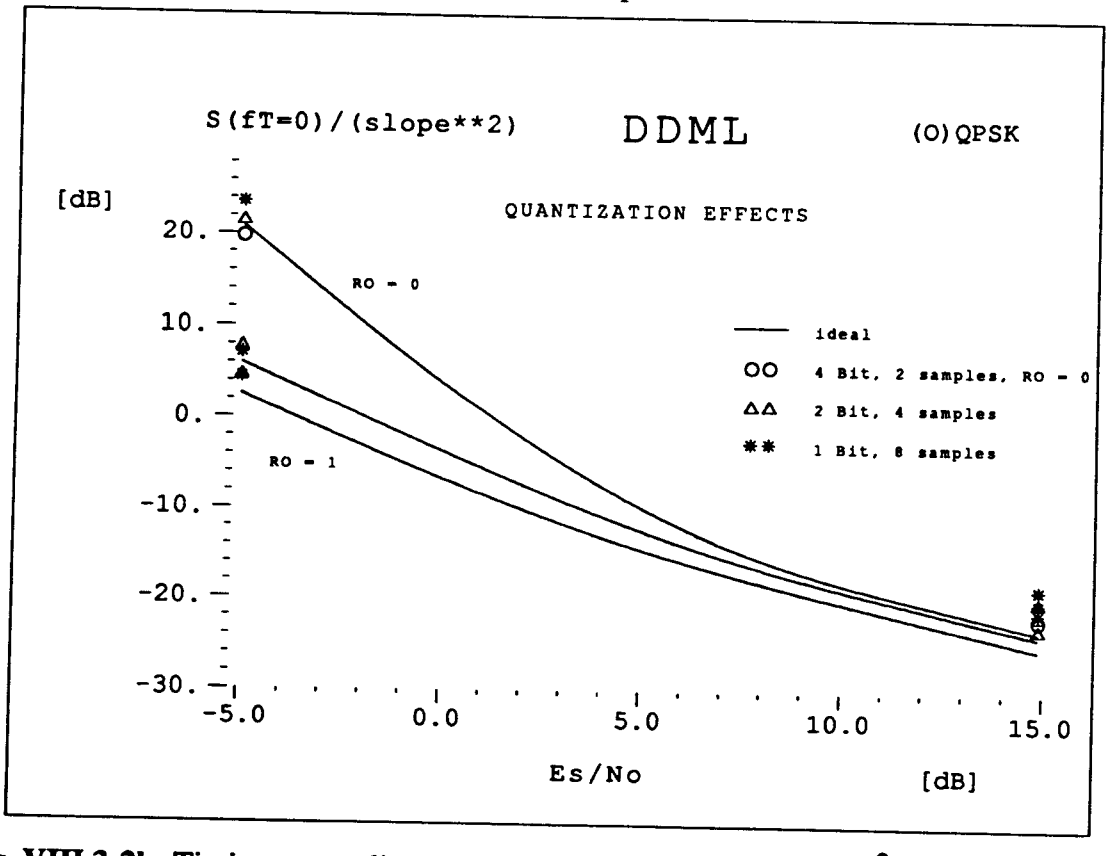


Fig. VIII.3-2b: Timing: normalized power spectral density $S(fT=0)/K^2_{\tau\tau}$ of DDML for (O)QPSK, quantization before matched filter, logarithmic scale, for 2, 4, and 8 samples/symbol and 1, 2, and 4 bit quantization

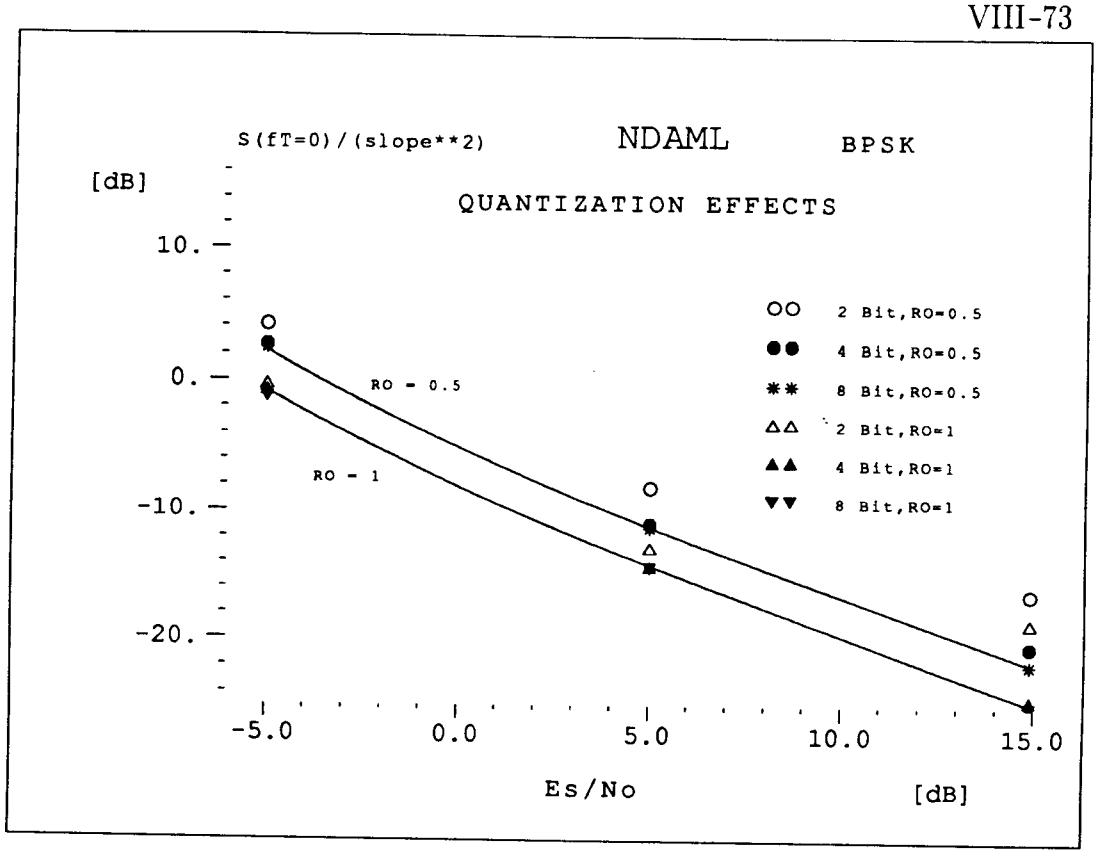


Fig. VIII.3-3a: Timing: normalized power spectral density $S(fT=0)/K^2_{\tau\tau}$ of NDAML for BPSK, matched filter output quantized, logarithmic scale

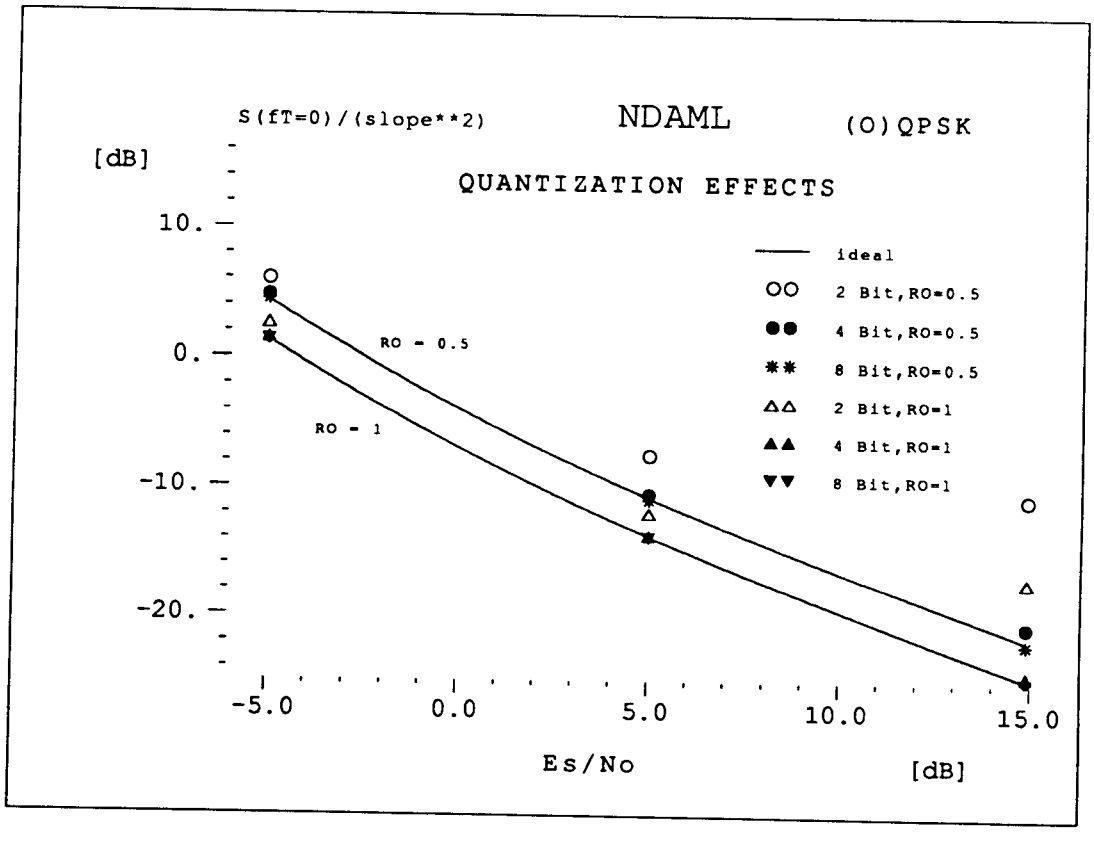


Fig. VIII.3-3b: Timing: normalized power spectral density $S(fT=0)/K^2_{\tau\tau}$ of NDAML for (O)QPSK, matched filter output quantized, logarithmic scale

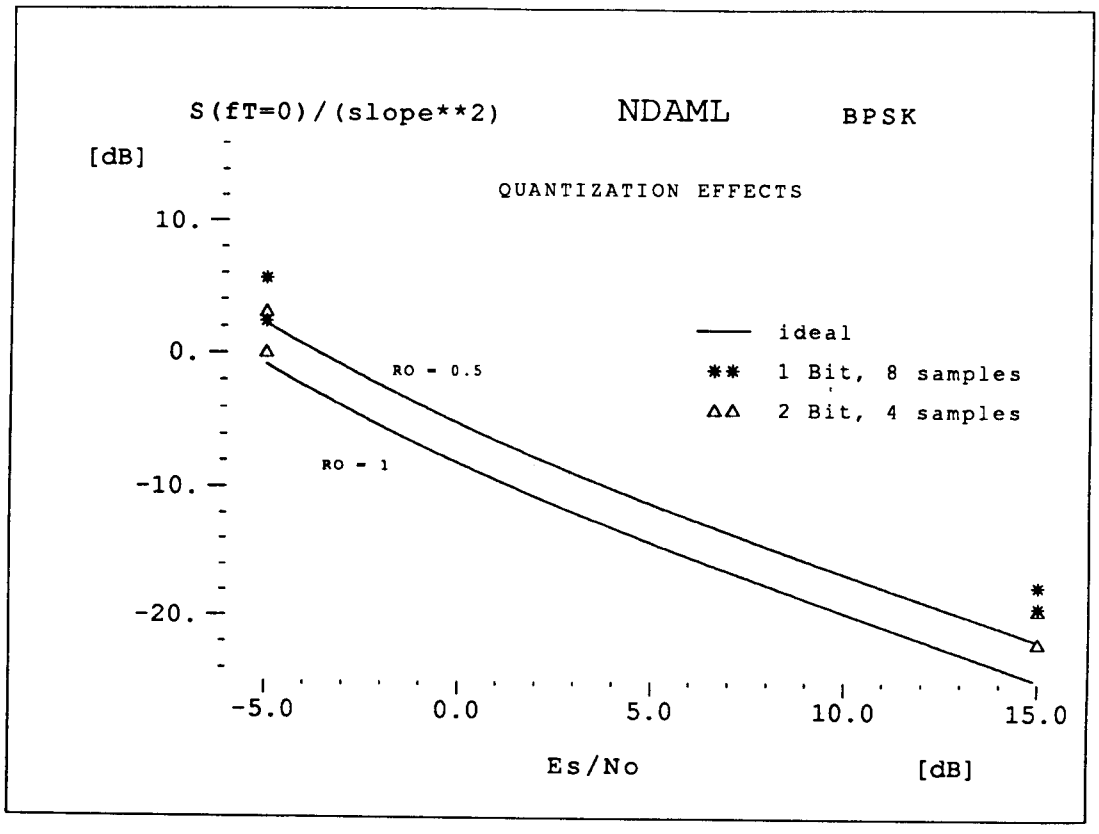


Fig. VIII.3-4a: Timing: normalized power spectral density $S(fT=0)/K^2_{\tau\tau}$ of NDAML for BPSK, quantization before matched filter, logarithmic scale, for 4 and 8 samples/symbol with 1 and 2 bit quantization

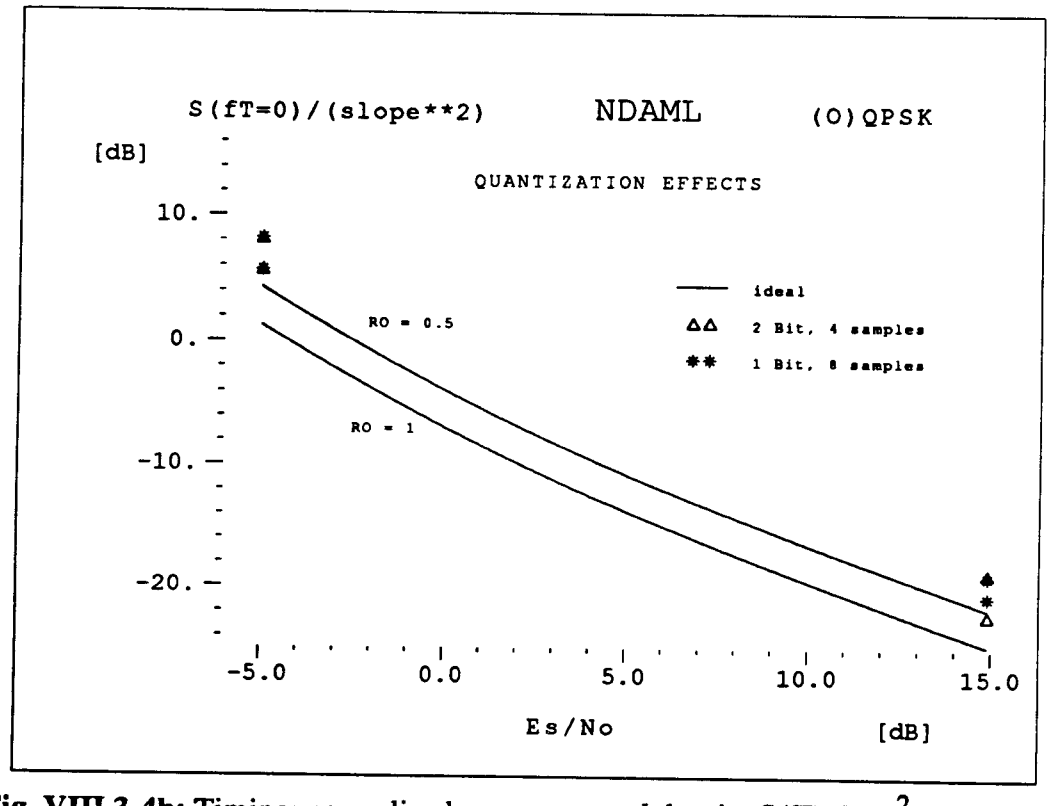


Fig. VIII.3-4b: Timing: normalized power spectral density $S(fT=0)/K^2_{\tau\tau}$ of NDAML for (O)QPSK, quantization before matched filter, logarithmic scale, for 4 and 8 samples/symbol with 1 and 2 bit quantization

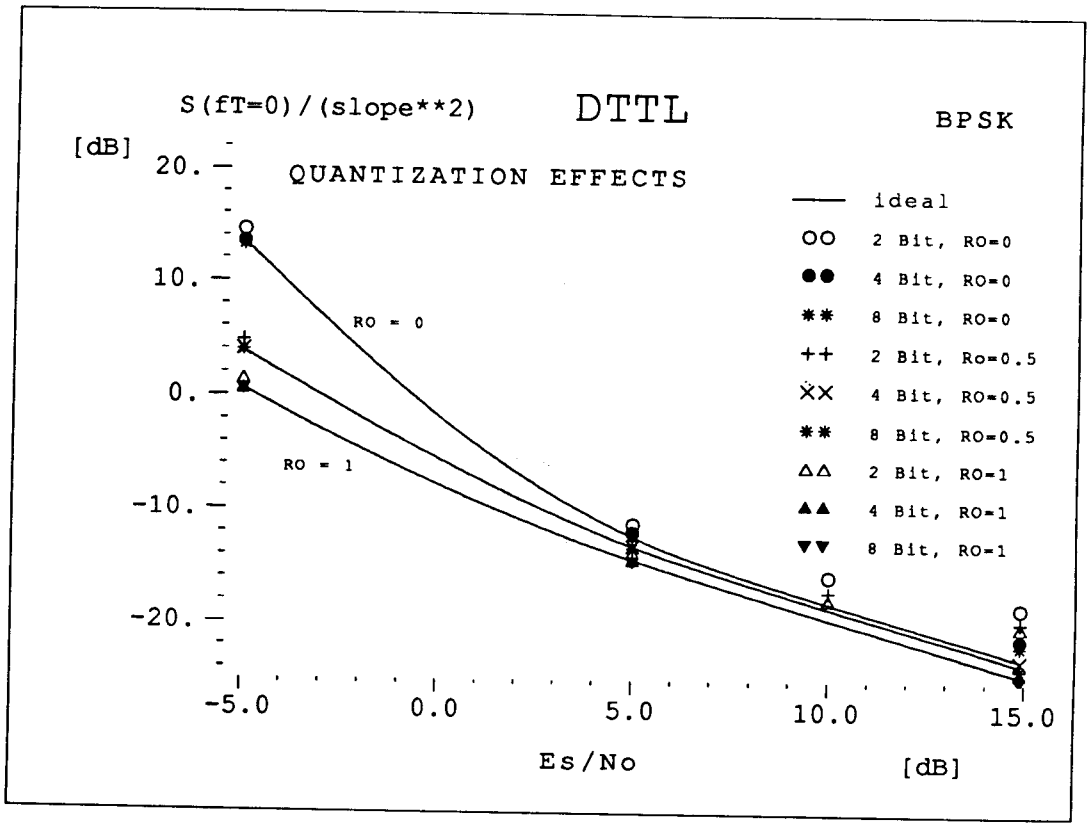


Fig. VIII.3-5a: Timing: normalized power spectral density $S(fT=0)/K^2_{\tau\tau}$ of DTTL for BPSK, matched filter output quantized, logarithmic scale

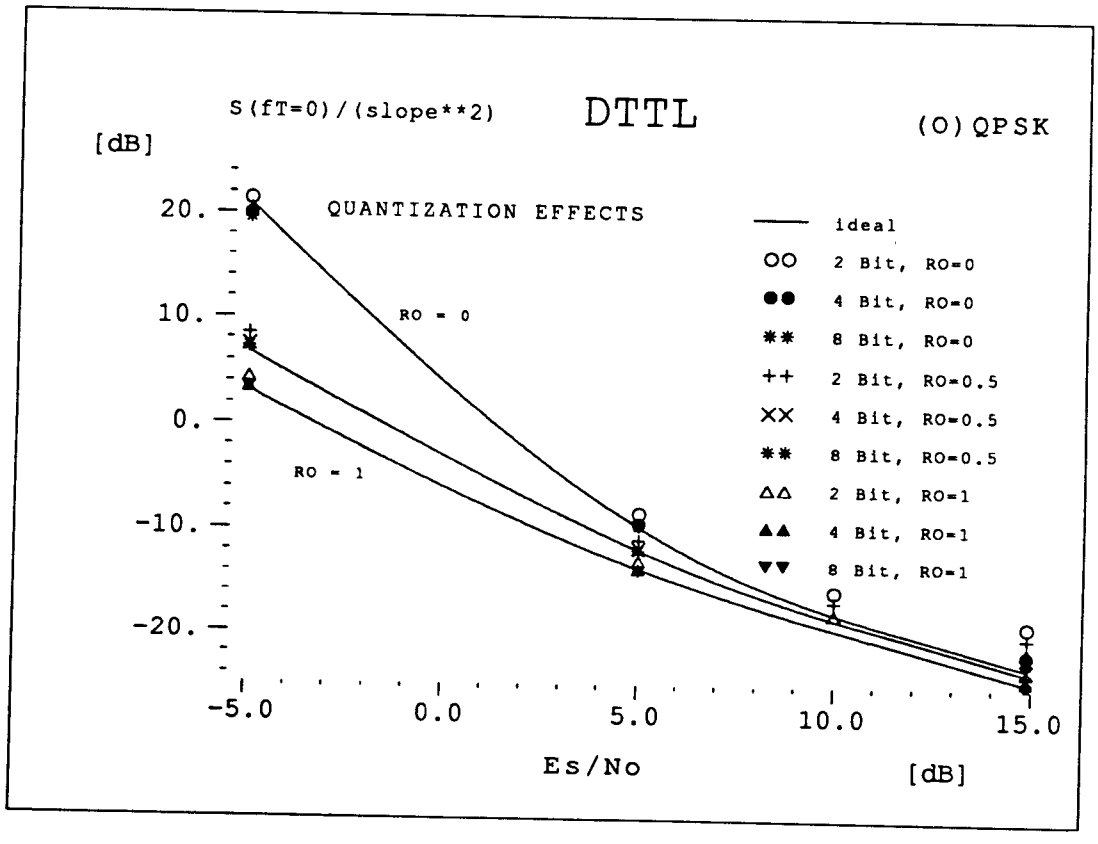


Fig. VIII.3-5b: Timing: normalized power spectral density $S(fT=0)/K^2_{\tau\tau}$ of DTTL for (O)QPSK, matched filter output quantized, logarithmic scale

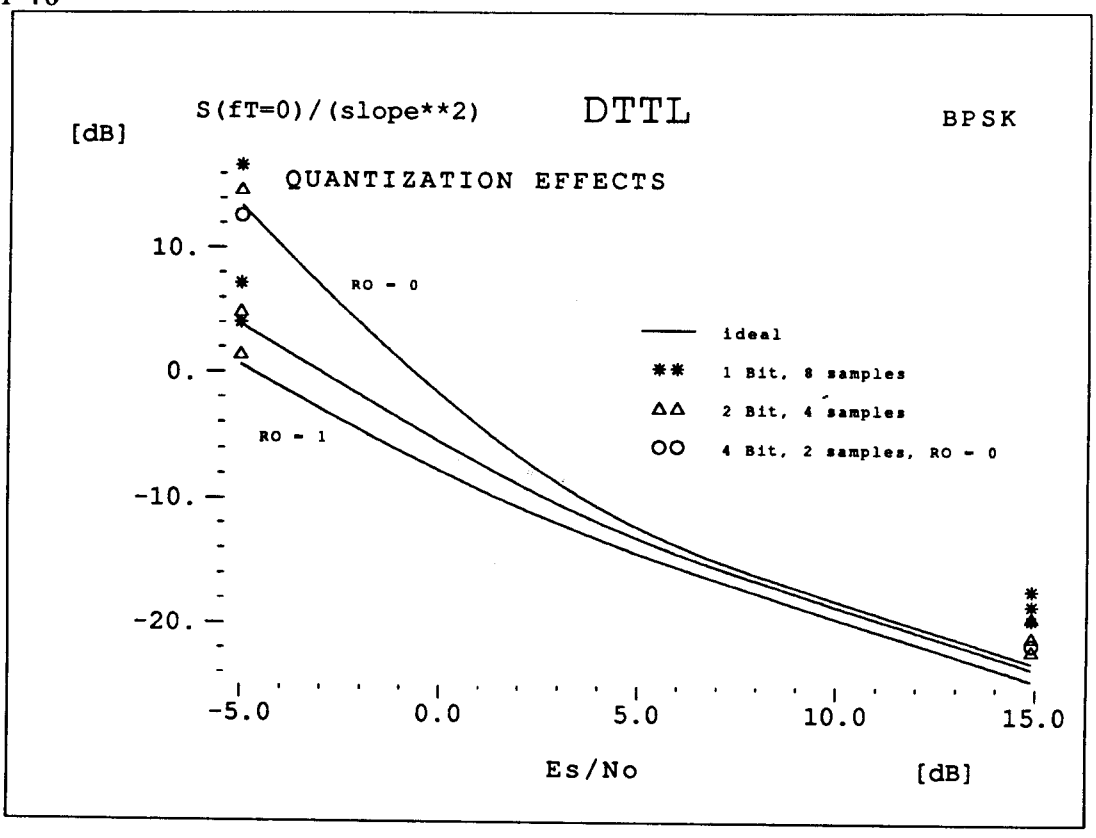


Fig. VIII.3-6a: Timing: normalized power spectral density $S(fT=0)/K^2_{\tau\tau}$ of DTTL for BPSK, quantization before matched filter, logarithmic scale, for 2, 4, and 8 samples/symbol and 1, 2, and 4 bit quantization

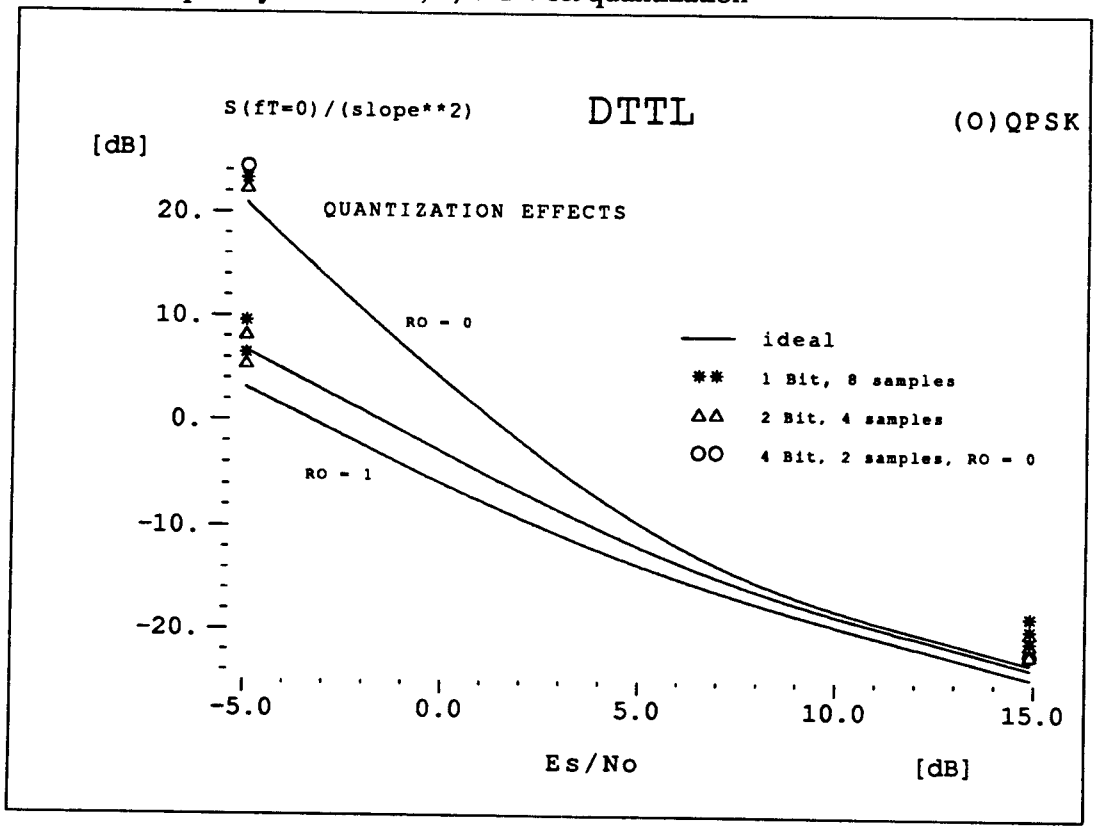


Fig. VIII.3-6b: Timing: normalized power spectral density $S(fT=0)/K^2_{\tau\tau}$ of DTTL for (O)QPSK, quantization before matched filter, logarithmic scale, for 2, 4, and 8 samples/symbol and 1, 2, and 4 bit quantization

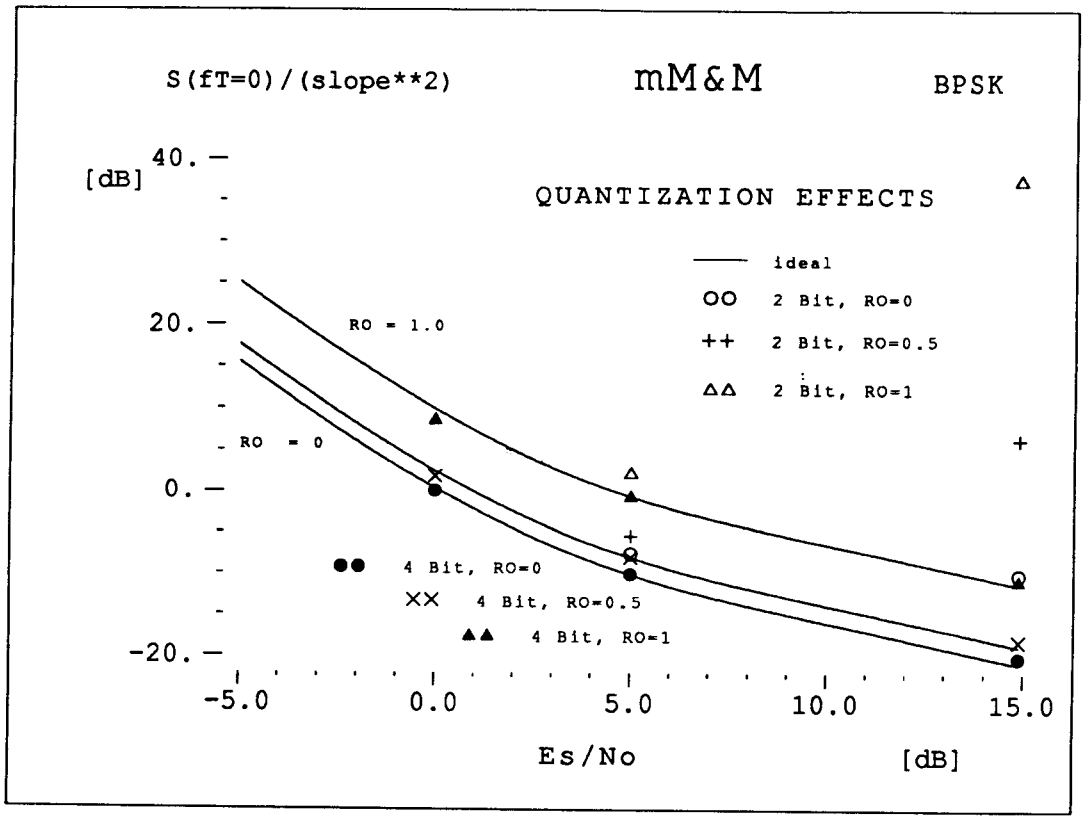


Fig. VIII.3-7a: Timing: normalized power spectral density $S(fT=0)/K^2_{\tau\tau}$ of mM&M for BPSK, matched filter output quantized, logarithmic scale

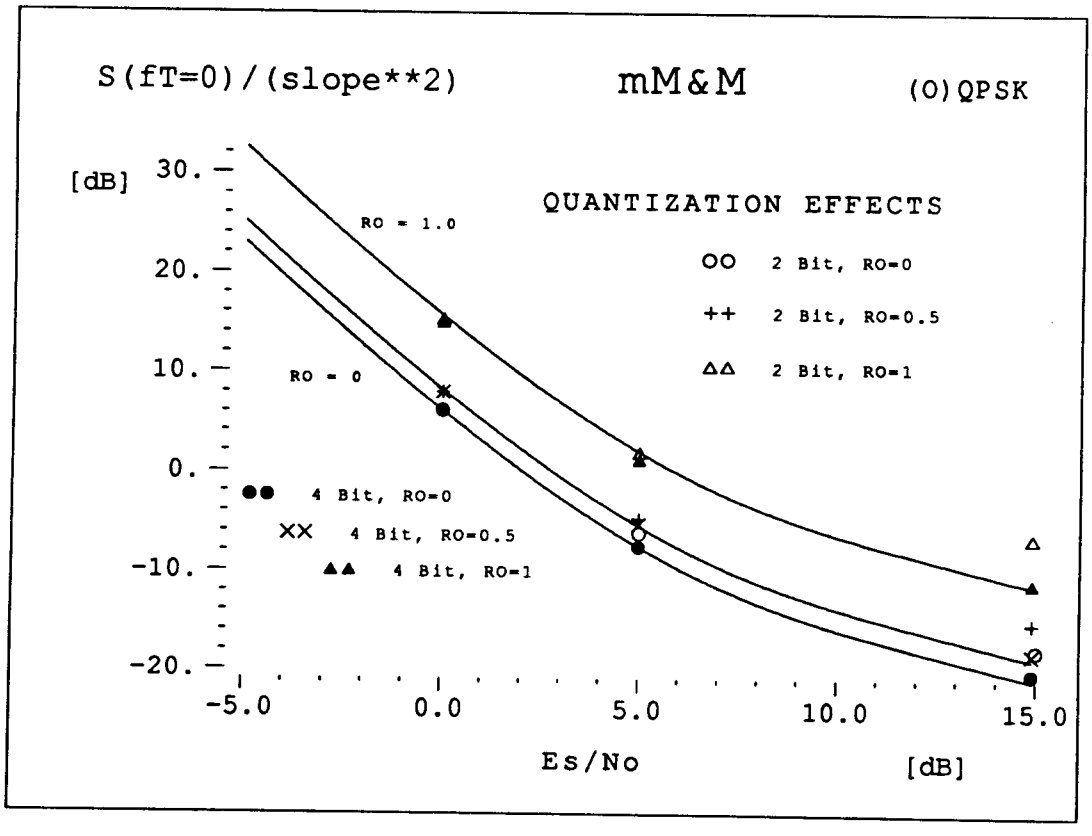


Fig. VIII.3-7b: Timing: normalized power spectral density $S(fT=0)/K^2_{\tau\tau}$ of mM&M for (O)QPSK, matched filter output quantized, logarithmic scale

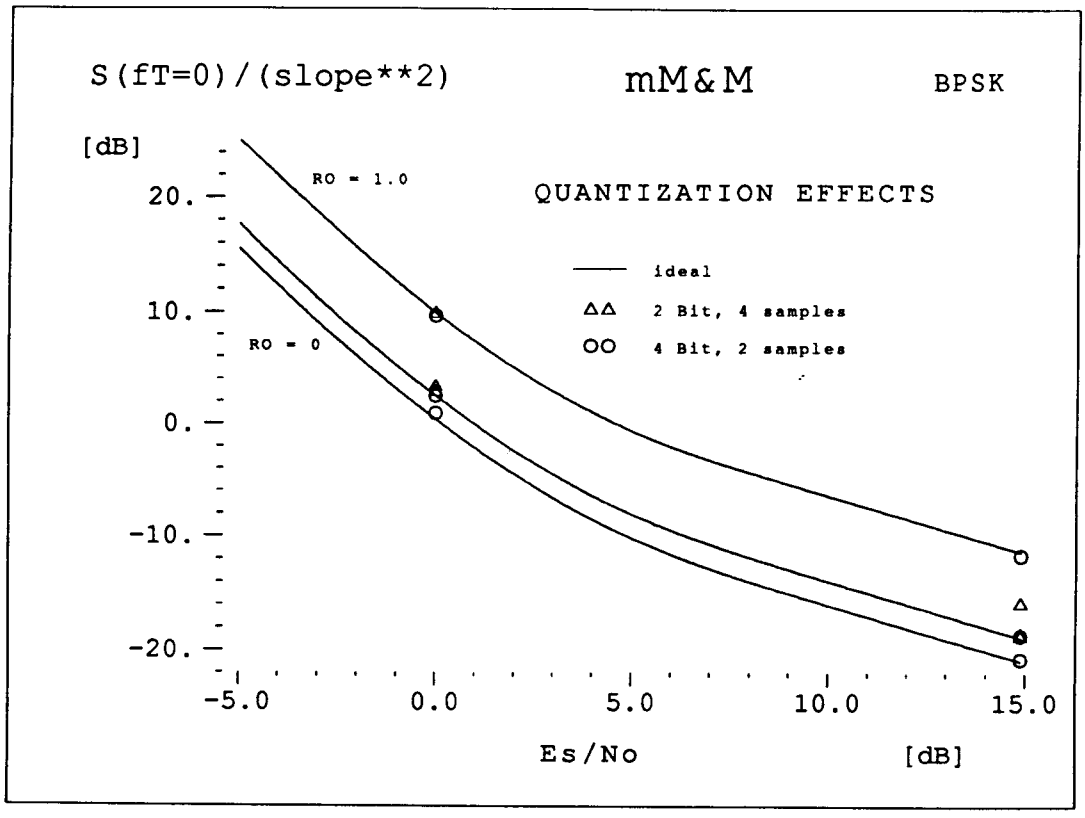


Fig. VIII.3-8a: Timing: normalized power spectral density $S(fT=0)/K^2_{\tau\tau}$ of mM&M for BPSK, quantization before matched filter, logarithmic scale, for 2 and 4 samples/symbol with 2 and 4 bit quantization

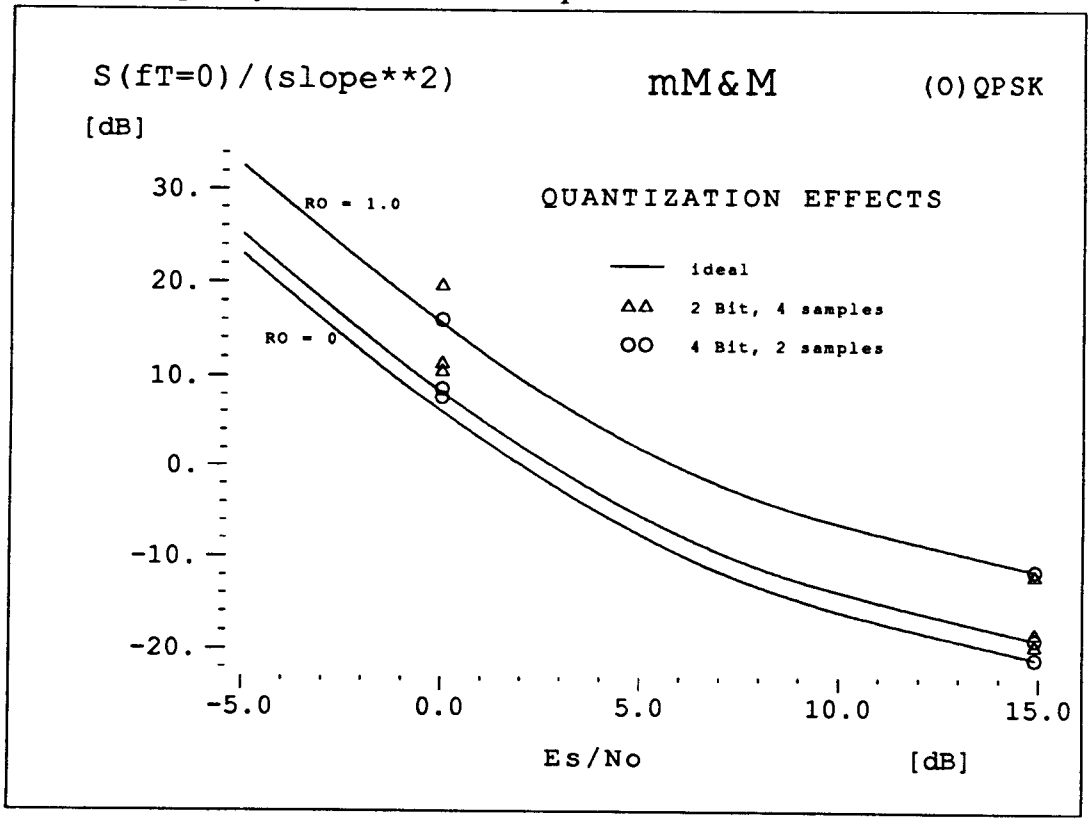


Fig. VIII.3-8b: Timing: normalized power spectral density $S(fT=0)/K^2_{\tau\tau}$ of modified M&M for (O)QPSK, quantization before matched filter, logarithmic scale, for 2 and 4 samples/symbol with 2 and 4 bit quantization

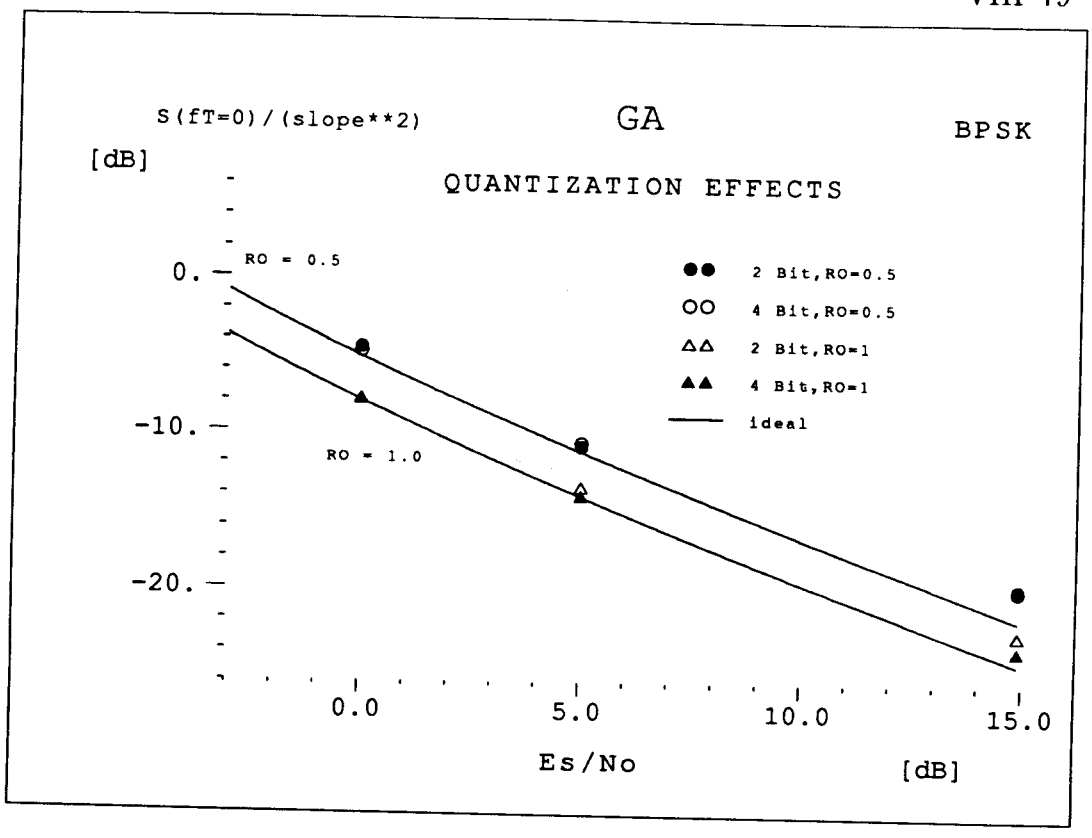


Fig. VIII.3-9a: Timing: normalized power spectral density $S(fT=0)/K^2_{\tau\tau}$ of GA for BPSK, matched filter output quantized, logarithmic scale

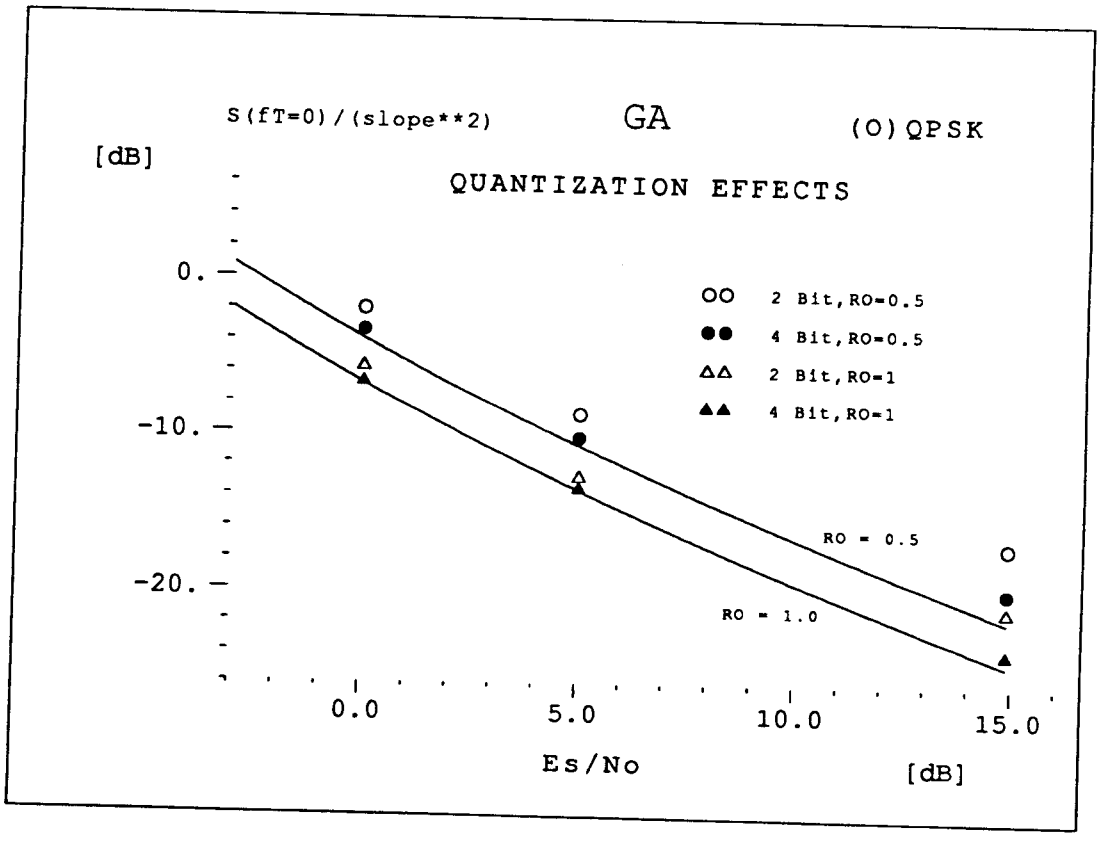


Fig. VIII.3-9b: Timing: normalized power spectral density $S(fT=0)/K^2_{\tau\tau}$ of GA for (O)QPSK, matched filter output quantized, logarithmic scale

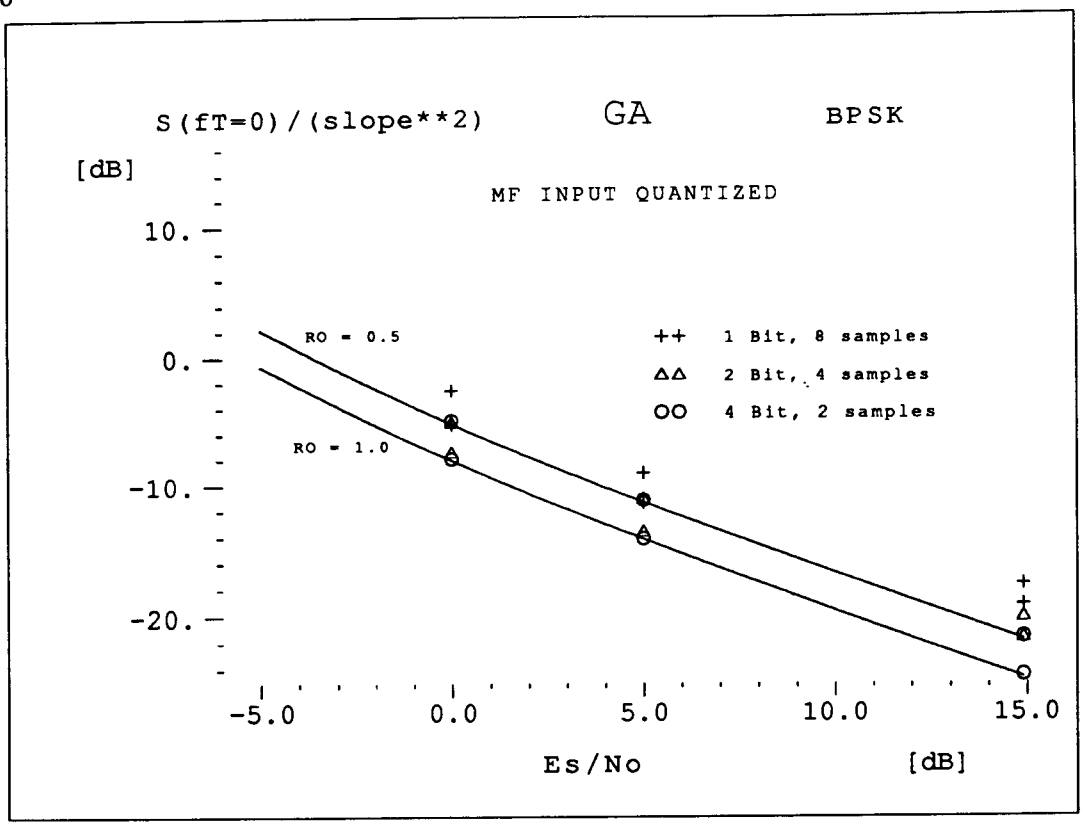


Fig. VIII.3-10a: Timing: normalized power spectral density $S(fT=0)/K^2_{\tau\tau}$ of GA for BPSK, quantization before matched filter, logarithmic scale, for 2, 4, and 8 samples/symbol and 1, 2, and 4 bit quantization

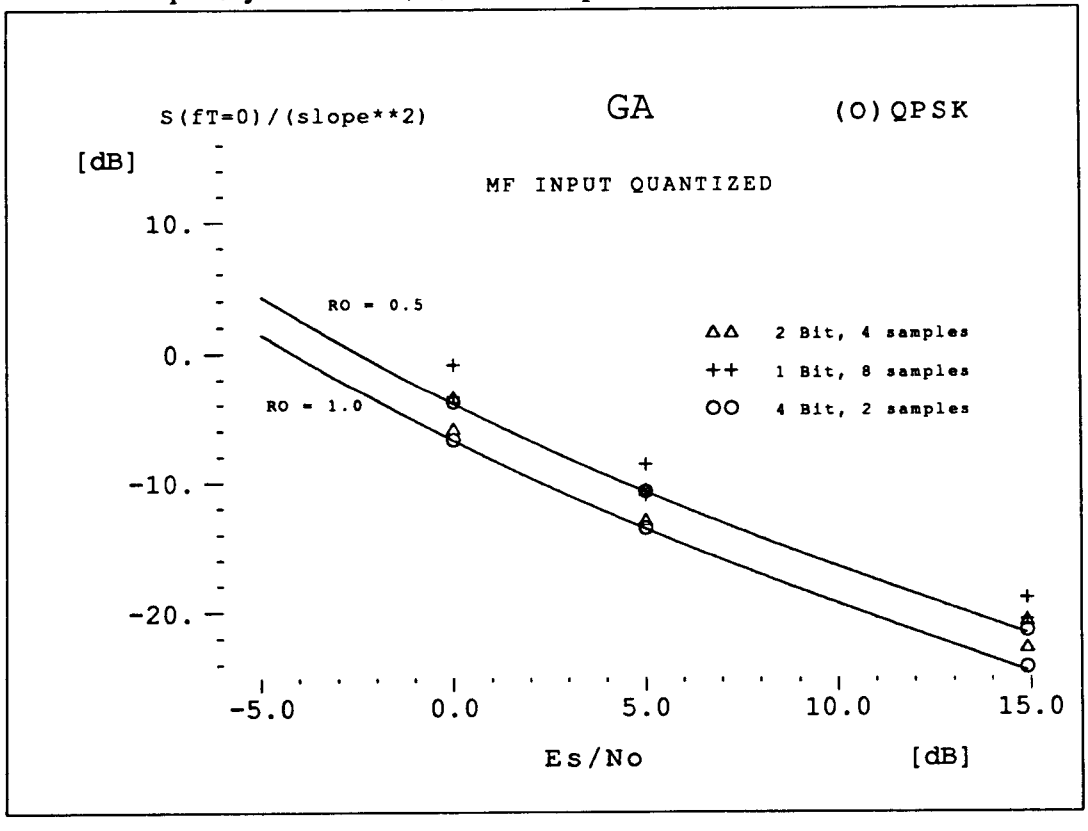


Fig. VIII.3-10b: Timing: normalized power spectral density $S(fT=0)/K^2_{\tau\tau}$ of GA for (O)QPSK, quantization before matched filter, logarithmic scale, for 2, 4, and 8 samples/symbol and 1, 2, and 4 bit quantization

TABLES

T.1 MISCELLANEOUS DEFINITIONS

Representation of received signal

$$x_L(t) = \left[\sum_m a_m\, h_I(t-mT-\tau) \;+\; j \sum_m b_m\, h_Q(t-mT-\tau) \right] e^{j\theta} \;+\; n(t)$$

where a_m and $b_m \in \{-1,+1\}$, and $n(t)$ is white Gaussian noise whose real and imaginary parts have a two–sided power spectral density of $N_o/(2E_b)$.

– BPSK : $h_I(t) = h(t)$, $h_Q(t) = 0$.

$$x_L(t) = \sum_m a_m\, h(t-mT-\tau)\, e^{j\theta} \;+\; n(t)$$

– QPSK : $h_I(t) = h_Q(t) = h(t)$.

$$x_L(t) = \sum_m (a_m + jb_m)\, h(t-mT-\tau)\, e^{j\theta} \;+\; n(t)$$

– OQPSK : $h_I(t) = h(t)$, $h_Q(t) = h(t - T/2)$

$$x_L(t) = \left[\sum_m a_m\, h(t-mT-\tau) \;+\; j \sum_m b_m\, h(t-mT-T/2-\tau) \right] e^{j\theta} \;+\; n(t)$$

The baseband pulse $h(t)$ has unit energy.

Matched filter output pulse

$$g(t) = \int_{-\infty}^{+\infty} h(t+u)\, h(u)\, du = h(t) \odot h(-t)$$

Log–likelihood functions

$$L_{DD}(\tilde{\theta},\tilde{\tau}) = (2E_b/N_o) \left[\sum_m \hat{a}_m \operatorname{Re}[p(m;\tilde{\tau})\, e^{-j\tilde{\theta}}] + \sum_m \hat{b}_m \operatorname{Im}[q(m;\tilde{\tau})\, e^{-j\tilde{\theta}}] \right]$$

$$L_{NDA}(\tilde{\theta},\tilde{\tau}) = \sum_m \ln\cosh \left[(2E_b/N_o) \operatorname{Re}[p(m;\tilde{\tau})\, e^{-j\tilde{\theta}}] \right] + \sum_m \ln\cosh \left[(2E_b/N_o) \operatorname{Im}[q(m;\tilde{\tau})\, e^{-j\tilde{\theta}}] \right]$$

$$L_{NDA}(\tilde{\theta},\tilde{\tau}) \cong \tfrac{1}{2} (2E_b/N_o)^2 \sum_m \left[(\operatorname{Re}[p(m;\tilde{\tau})\, e^{-j\tilde{\theta}}])^2 + (\operatorname{Im}[q(m;\tilde{\tau})\, e^{-j\tilde{\theta}}])^2 \right] - \tfrac{1}{12} (2E_b/N_o)^4 \sum_m \left[(\operatorname{Re}[p(m;\tilde{\tau})\, e^{-j\tilde{\theta}}])^4 + (\operatorname{Im}[q(m;\tilde{\tau})\, e^{-j\tilde{\theta}}])^4 \right]$$

Matched filter output samples

$$p(m;\tilde{\tau}) = [x_L(t) \odot h_I(-t)]\big|_{t=mT+\tilde{\tau}}$$

$$q(m;\tilde{\tau}) = [x_L(t) \odot h_Q(-t)]\big|_{t=mT+\tilde{\tau}}$$

– BPSK : $q(m;\tilde{\tau}) = 0$

– QPSK : $q(m;\tilde{\tau}) = p(m;\tilde{\tau})$

– OQPSK : $q(m;\tilde{\tau}) = p(m;\tilde{\tau} + T/2) = p(m + 1/2;\tilde{\tau})$

Receiver decisions

$$\hat{a}_m = \mathrm{sgn}(\mathrm{Re}[p(m;\tilde{\tau})\, e^{-j\tilde{\theta}}])$$

$$\hat{b}_m = \mathrm{sgn}(\mathrm{Im}[q(m;\tilde{\tau})\, e^{-j\tilde{\theta}}])$$

Derivative matched filter output samples

$$p'(k;\tilde{\tau}) = [x_L(t) \odot h_I'(-t)]\big|_{t=kT+\tilde{\tau}}$$

$$q'(k;\tilde{\tau}) = [x_L(t) \odot h_Q'(-t)]\big|_{t=kT+\tilde{\tau}}$$

– BPSK : $h_I'(-t) = h'(-t)$; $h_Q'(-t) = 0$

– QPSK : $h_I'(-t) = h_Q'(-t) = h'(-t)$

– OQPSK : $h_I'(-t) = h'(-t)$; $h_Q'(-t) = h'(-t-T/2)$

Prefilter output signal

$$y_{LI}(t) = x_L(t) \odot h_{oI}(t)$$

$$y_{LQ}(t) = x_L(t) \odot h_{oQ}(t)$$

$$y_L(t) = x_L(t) \odot h_o(t)$$

– BPSK : $h_{oI}(t) = h_o(t)$; $h_{oQ}(t) = 0$

– QPSK : $h_{oI}(t) = h_{oQ}(t) = h_o(t)$

– OQPSK : $h_{oI}(t) = h_o(t)$; $h_{oQ}(t) = h_o(t-T/2)$

$h_o(t)$ is not necessarily matched to $h(t)$.

Samples used for carrier–dependent symbol synchronization

$$p_R(k;\tilde{\tau},\hat{\theta}) = \mathrm{Re}[p(k;\tilde{\tau})\, e^{-j\hat{\theta}}] = \mathrm{Re}[x_L(t)\, e^{-j\hat{\theta}} \odot h_I(-t)]\big|_{t=kT+\tilde{\tau}}$$

$$q_I(k;\tilde{\tau},\hat{\theta}) = \mathrm{Im}[q(k;\tilde{\tau})\, e^{-j\hat{\theta}}] = \mathrm{Im}[x_L(t)\, e^{-j\hat{\theta}} \odot h_Q(-t)]\big|_{t=kT+\tilde{\tau}}$$

$$p_R'(k;\tilde{\tau},\hat{\theta}) = \mathrm{Re}[p'(k;\tilde{\tau})\, e^{-j\hat{\theta}}] = \mathrm{Re}[x_L(t)\, e^{-j\hat{\theta}} \odot h_I'(-t)]\big|_{t=kT+\tilde{\tau}}$$

$$q_I'(k;\tilde{\tau},\hat{\theta}) = \mathrm{Im}[q'(k;\tilde{\tau})\, e^{-j\hat{\theta}}] = \mathrm{Im}[x_L(t)\, e^{-j\hat{\theta}} \odot h_Q'(-t)]\big|_{t=kT+\tilde{\tau}}$$

T.2 SAMPLES NEEDED FOR VARIOUS RECEIVER OPERATIONS

Samples needed for decisions	
BPSK :	$\mathrm{Re}[p(k;\hat{\tau})\exp(-j\hat{\theta})]$
QPSK :	$\mathrm{Re}[p(k;\hat{\tau})\exp(-j\hat{\theta})]$ and $\mathrm{Im}[p(k;\hat{\tau})\exp(-j\hat{\theta})]$
OQPSK :	$\mathrm{Re}[p(k;\hat{\tau})\exp(-j\hat{\theta})]$ and $\mathrm{Im}[p(k+1/2;\hat{\tau})\exp(-j\hat{\theta})]$

Samples needed for timing–dependent carrier synchronization	
BPSK :	$p(k;\hat{\tau})$ or $p(k;\hat{\tau})\exp(-j\hat{\theta})$
QPSK :	$p(k;\hat{\tau})$ or $p(k;\hat{\tau})\exp(-j\hat{\theta})$
OQPSK :	$p(k;\hat{\tau})$ and $p(k+1/2;\hat{\tau})$, or $p(k;\hat{\tau})\exp(-j\hat{\theta})$ and $p(k+1/2;\hat{\tau})\exp(-j\hat{\theta})$

Samples needed for FB symbol synchronization	
A) Samples at decision instants	
BPSK :	$\mathrm{Re}[p(k;\hat{\tau})\exp(-j\hat{\theta})]$
QPSK :	$\mathrm{Re}[p(k;\hat{\tau})\exp(-j\hat{\theta})]$ and $\mathrm{Im}[p(k;\hat{\tau})\exp(-j\hat{\theta})]$
OQPSK :	$\mathrm{Re}[p(k;\hat{\tau})\exp(-j\hat{\theta})]$ and $\mathrm{Im}[p(k+1/2;\hat{\tau})\exp(-j\hat{\theta})]$
B) Additional samples halfway between decision instants	
BPSK :	$\mathrm{Re}[p(k+1/2;\hat{\tau})\exp(-j\hat{\theta})]$
QPSK :	$\mathrm{Re}[p(k+1/2;\hat{\tau})\exp(-j\hat{\theta})]$ and $\mathrm{Im}[p(k+1/2;\hat{\tau})\exp(-j\hat{\theta})]$
OQPSK :	$\mathrm{Re}[p(k+1/2;\hat{\tau})\exp(-j\hat{\theta})]$ and $\mathrm{Im}[p(k+1;\hat{\tau})\exp(-j\hat{\theta})]$

T.3 INTERPOLATOR OUTPUT SAMPLING RATES

Interpolator output sampling rates (in real–valued samples per symbol)						
<u>Carrier synchronization</u>						
– timing–independent	x x		x x	x x		
– timing–dependent		x x			x x	x x
<u>Symbol synchronization</u>						
– FF	x x	x x				
– FB (at decision instants only)			x x		x x	
– FB (at and halfway between decision instants)				x x		x x
<u>Phase rotator</u>						
– follows interpolator	x	x	x	x	x	x
– precedes interpolator	x	x	x	x	x	x
<u>Interpolator output rate</u>						
– BPSK :	2 1	2 2	2 1	4 2	2 2	4 3
– QPSK :	2 2	2 2	2 2	4 4	2 2	4 4
– OQPSK :	4 2	4 4	4 2	4 4	4 4	4 4

T.4 CARRIER SYNCHRONIZATION ALGORITHMS

DDMLFB

$$u_\theta(k;\hat{\theta},\hat{\tau}) = \mathrm{Im}[(\hat{a}_k p(k;\hat{\tau}) - j\hat{b}_k q(k;\hat{\tau}))\, e^{-j\hat{\theta}}] \qquad \text{(BPSK, (O)QPSK)}$$

DDMLFF

$$\hat{\theta} = \arg\left[\sum_{k=1}^{K} (\hat{a}_k p(k;\hat{\tau}) - j\hat{b}_k q(k;\hat{\tau}))\right] \qquad \text{(BPSK, (O)QPSK)}$$

$$\hat{C} + j\hat{S} = \frac{1}{K}\sum_{k=1}^{K} (\hat{a}_k p(k;\hat{\tau}) - j\hat{b}_k q(k;\hat{\tau})) \qquad \text{(BPSK, (O)QPSK)}$$

NDAMLFBB/O, NDAMLFBQ

$$u_\theta(k;\hat{\theta},\hat{\tau}) = \frac{1}{2}\,\mathrm{Im}[(p^2(k;\hat{\tau}) - q^2(k;\hat{\tau}))\, e^{-j2\hat{\theta}}] \qquad \text{(BPSK, OQPSK)}$$

or

$$u_\theta(k;\hat{\theta},\hat{\tau}) = \frac{1}{4}\,\mathrm{Im}[-p^4(k;\hat{\tau})\, e^{-j4\hat{\theta}}] \qquad \text{(QPSK)}$$

NDAMLFFB/O, NDAMLFFQ

$$\hat{\theta} = \frac{1}{2}\arg\left[\sum_{k=1}^{K} (p^2(k;\hat{\tau}) - q^2(k;\hat{\tau}))\right] \qquad \text{(BPSK, OQPSK)}$$

or

$$\hat{\theta} = \frac{1}{4}\arg\left[-\sum_{k=1}^{K} p^4(k;\hat{\tau})\right] \qquad \text{(QPSK)}$$

V&VFFB/Q, V&VFFO

$$\hat{\theta} = \frac{1}{M}\arg\left[\sum_{k=1}^{K} F(|p(k;\hat{\tau})|)\exp(jM\arg(p(k;\hat{\tau})))\right] \qquad \text{(BPSK, QPSK)}$$

or

$$\hat{\theta} = \frac{1}{2}\arg\left[\sum_{k=1}^{K} F(|p(k;\hat{\tau})|)\exp(2j\arg(p(k;\hat{\tau}))) - \sum_{k=1}^{K} F(|q(k;\hat{\tau})|)\exp(2j\arg(q(k;\hat{\tau})))\right] \qquad \text{(OQPSK)}$$

V&VFBB/Q, V&VFBO

$$u_\theta(k;\hat{\theta},\hat{\tau}) = \frac{1}{M} F(|p(k;\hat{\tau})|)\sin(M\arg(p(k;\hat{\tau})) - M\hat{\theta}) \qquad \text{(BPSK, QPSK)}$$

or

$$u_\theta(k;\hat{\theta},\hat{\tau}) = \frac{1}{2} F(|p(k;\hat{\tau})|) \sin(2 \arg(p(k;\hat{\tau})) - 2\hat{\theta})$$

$$- \frac{1}{2} F(|q(k;\hat{\tau})|) \sin(2 \arg(q(k;\hat{\tau})) - 2\hat{\theta}) \qquad \text{(OQPSK)}$$

<u>DDMLFBT̄</u>

$$u_\theta(t;\hat{\theta}) = \mathrm{Im}[(m_I(t) - jm_Q(t))\, y_L(t)\, e^{-j\hat{\theta}}] \qquad \text{(BPSK, (O)QPSK)}$$

where

$$m_I(t) = \mathrm{sgn}(\mathrm{Re}[y_L(t)\, e^{-j\hat{\theta}}])$$

$$m_Q(t) = \begin{cases} \mathrm{sgn}(\mathrm{Im}[y_L(t)\, e^{-j\hat{\theta}}]) & \text{((O)QPSK)} \\ 0 & \text{(BPSK)} \end{cases}$$

<u>DDMLFFT̄</u>

$$\hat{\theta} = \arg\left[\int_0^{KT} (m_I(t) - jm_Q(t))\, y_L(t)\, dt \right] \qquad \text{(BPSK,(O)QPSK)}$$

$$\hat{C} + j\hat{S} = \frac{1}{KT} \int_0^{KT} (m_I(t) - jm_Q(t))\, y_L(t)\, dt \qquad \text{(BPSK,(O)QPSK)}$$

<u>NDAMLFBT̄B, NDAMLFBT̄Q/O</u>

$$u_\theta(t;\hat{\theta}) = \frac{1}{2} \mathrm{Im}[y_L^2(t)\, e^{-2j\hat{\theta}}] \qquad \text{(BPSK)}$$

or

$$u_\theta(t;\hat{\theta}) = \frac{1}{4} \mathrm{Im}[y_L^4(t)\, e^{-4j\hat{\theta}}] \qquad \text{((O)QPSK)}$$

<u>NDAMLFFT̄B, NDAMLFFT̄Q/O</u>

$$\hat{\theta} = \frac{1}{2} \arg\left[\int_0^{KT} y_L^2(t)\, dt \right] \qquad \text{(BPSK)}$$

or

$$\hat{\theta} = \frac{1}{4} \arg\left[\int_0^{KT} y_L^4(t)\, dt \right] \qquad \text{((O)QPSK)}$$

NDASQFFTO

$$\hat{\theta} = \frac{1}{4}\arg\left[\int_0^{KT} y_L^2(t)\, e^{j2\pi t/T}\, dt\right] + \frac{1}{4}\arg\left[\int_0^{KT} y_L^2(t)\, e^{-j2\pi t/T}\, dt\right] \qquad \text{(OQPSK)}$$

T.5 SYMBOL SYNCHRONIZATION ALGORITHMS

DDMLFF

$$\hat{\tau}: \max_{\tilde{\tau}} \left[\sum_{k=1}^{K} (\hat{a}_k \, p_R(k;\tilde{\tau},\hat{\theta}) + \hat{b}_k \, q_I(k;\tilde{\tau},\hat{\theta})) \right]$$

DDMLFB

$$u_\tau(k;\hat{\theta},\hat{\tau}) = \hat{a}_k \, p_R'(k;\hat{\tau},\hat{\theta}) + \hat{b}_k \, q_I'(k;\hat{\tau},\hat{\theta})$$

NDAMLFF

$$\hat{\tau}: \max_{\tilde{\tau}} \left[\sum_{k=1}^{K} (p_R^2(k;\tilde{\tau},\hat{\theta}) + q_I^2(k;\tilde{\tau},\hat{\theta})) \right]$$

NDAMLFB

$$u_\tau(k;\hat{\theta},\hat{\tau}) = p_R(k;\hat{\tau},\hat{\theta}) \, p_R'(k;\hat{\tau},\hat{\theta}) + q_I(k;\hat{\tau},\hat{\theta}) \, q_I'(k;\hat{\tau},\hat{\theta})$$

DDMLFB$\bar{S}\bar{N}\bar{F}\bar{B}$

$$u_\tau(k;\hat{\theta},\hat{\tau}) = \hat{a}_k \, p_R'(k;\hat{\tau},\hat{\theta}) + \hat{b}_k \, q_I'(k;\hat{\tau},\hat{\theta}) - \sum_m (\hat{a}_k \hat{a}_m + \hat{b}_k \hat{b}_m) \, g'(kT-mT)$$

DDMLFB$\bar{S}\bar{N}\bar{F}\bar{F}$, NDAMLFB$\bar{S}\bar{N}\bar{F}\bar{F}$

Replace derivative matched filter in DDMLFB or NDAMLFB algorithm by a filter with transfer function $H_{mod}(f)$, given by

$$H_{mod}(f) = H^*(f) \left[j2\pi f - \sum_{m=-\infty}^{+\infty} \frac{g'(mT)}{g(0)} \exp(-j2\pi f m T) \right]$$

DDMMSEFB

$$u_\tau(k;\hat{\theta},\hat{\tau}) = (p_R(k;\hat{\tau},\hat{\theta}) - \hat{a}_k) \, p_R'(k;\hat{\tau},\hat{\theta}) + (q_I(k;\hat{\tau},\hat{\theta}) - \hat{b}_k) \, q_I'(k;\hat{\tau},\hat{\theta})$$

DDEL(λ=1/2), DTTL

$$u_\tau(k;\hat{\theta},\hat{\tau}) = \hat{a}_k \, (p_R(k+\tfrac{1}{2};\hat{\tau},\hat{\theta}) - p_R(k-\tfrac{1}{2};\hat{\tau},\hat{\theta})) + \hat{b}_k \, (q_I(k+\tfrac{1}{2};\hat{\tau},\hat{\theta}) - q_I(k-\tfrac{1}{2};\hat{\tau},\hat{\theta})) \qquad (\mathrm{DDEL}(\lambda=1/2))$$

$$u_\tau(k;\hat{\theta};\hat{\tau}) = (\hat{a}_k - \hat{a}_{k+1})\, p_R(k+\tfrac{1}{2};\hat{\tau},\hat{\theta}) + (\hat{b}_k - \hat{b}_{k+1})\, q_I(k+\tfrac{1}{2};\hat{\tau},\hat{\theta}) \qquad \text{(DTTL)}$$

<u>NDAEL(λ=1/2), GA</u>

$$u_\tau(k;\hat{\theta},\hat{\tau}) = p_R(k;\hat{\tau},\hat{\theta})\,(p_R(k+\tfrac{1}{2};\hat{\tau},\hat{\theta}) - p_R(k-\tfrac{1}{2};\hat{\tau},\hat{\theta})) + q_I(k;\hat{\tau},\hat{\theta})\,(q_I(k+\tfrac{1}{2};\hat{\tau},\hat{\theta}) - q_I(k-\tfrac{1}{2};\hat{\tau},\hat{\theta})) \qquad \text{(NDAEL}(\lambda=1/2))$$

$$u_\tau(k;\hat{\theta},\hat{\tau}) = p_R(k+\tfrac{1}{2};\hat{\tau},\hat{\theta})\,(p_R(k;\hat{\tau},\hat{\theta}) - p_R(k+1;\hat{\tau},\hat{\theta})) + q_I(k+\tfrac{1}{2};\hat{\tau},\hat{\theta})\,(q_I(k;\hat{\tau},\hat{\theta}) - q_I(k+1;\hat{\tau},\hat{\theta})) \qquad \text{(GA)}$$

<u>DDEL(λ=1), M&M, mM&M</u>

$$u_\tau(k;\hat{\theta},\hat{\tau}) = \hat{a}_k\,(p_R(k+1;\hat{\tau},\hat{\theta}) - p_R(k-1;\hat{\tau},\hat{\theta})) + \hat{b}_k\,(q_I(k+1;\hat{\tau},\hat{\theta}) - q_I(k-1;\hat{\tau},\hat{\theta})) \qquad \text{(DDEL}(\lambda=1))$$

$$u_\tau(k;\hat{\theta},\hat{\tau}) = \hat{a}_k\, p_R(k+1;\hat{\tau},\hat{\theta}) - \hat{a}_{k+1}\, p_R(k;\hat{\tau},\hat{\theta}) + \hat{b}_k\, q_I(k+1;\hat{\tau},\hat{\theta}) - \hat{b}_{k+1}\, q_I(k;\hat{\tau},\hat{\theta}) \qquad \text{(M\&M)}$$

$$u_\tau(k;\hat{\theta},\hat{\tau}) = (\hat{a}_{k-1} - \hat{a}_{k+1})\, p_R(k;\hat{\tau},\hat{\theta}) + (\hat{b}_{k-1} - \hat{b}_{k+1})\, q_I(k;\hat{\tau},\hat{\theta}) \qquad \text{(mM\&M)}$$

<u>NDANLFF</u>

$$\hat{\tau} = \frac{T}{2\pi} \arg\left[\int_0^{KT} \left[NL(Re[y_{LI}(t)e^{-j\hat{\theta}}]) + NL(Im[y_{LQ}(t)e^{-j\hat{\theta}}] \right] e^{j2\pi t/T}\, dt \right]$$

Special case : NDASQFF, where NL(.) is a squarer.

<u>NDAMLFF$\bar{C}$B/Q</u>

$$\hat{\tau} : \max_{\tilde{\tau}} \left[\sum_{k=1}^{K} |p(k;\tilde{\tau})|^2 \right] \qquad \text{(BPSK, QPSK)}$$

<u>NDAMLFB$\bar{C}$B/Q</u>

$$u_\tau(k;\hat{\tau}) = Re[p^*(k;\hat{\tau})\, p'(k;\hat{\tau})] \qquad \text{(BPSK, QPSK)}$$

NDAMLFBS̄Ñ F̄F̄C̄B/Q

Replace derivative matched filter in DDMLFB or NDAMLFB algorithm by a filter with transfer function $H_{mod}(f)$, given by

$$H_{mod}(f) = H^*(f) \left[j2\pi f - \sum_{m=-\infty}^{+\infty} \frac{g'(mT)}{g(0)} \exp(-j2\pi f m T) \right]$$

NDAEL(λ=1/2)C̄B/Q

$$u_\tau(k;\hat{\tau}) = \mathrm{Re}[p^*(k;\hat{\tau})\,(p(k+\tfrac{1}{2};\hat{\tau}) - p(k-\tfrac{1}{2};\hat{\tau}))] \qquad \text{(BPSK, QPSK)}$$

GAC̄B/Q

$$u_\tau(k;\hat{\tau}) = \mathrm{Re}[p^*(k+1/2;\hat{\tau})\,(p(k;\hat{\tau}) - p(k+1;\hat{\tau}))] \qquad \text{(BPSK, QPSK)}$$

NDASQFFC̄B/Q, NDASQFFC̄O

$$\hat{\tau} = \frac{T}{2\pi} \arg \left[\int_0^{KT} |y_L(t)|^2 \, e^{j2\pi t/T} \, dt \right] \qquad \text{(BPSK, QPSK)}$$

or

$$\hat{\tau} = \frac{T}{4\pi} \arg \left[\int_0^{KT} y_L^2(t) \, e^{j2\pi t/T} \, dt \right] - \frac{T}{4\pi} \arg \left[\int_0^{KT} y_L^2(t) \, e^{-j2\pi t/T} \, dt \right] \qquad \text{(OQPSK)}$$

NDANLFBC̄B/Q

$$u_\tau(k;\hat{\tau}) = (F(|p(k;\hat{\tau})|^2) - F(|p(k+1;\hat{\tau})|^2)\,\mathrm{Re}[p(k;\hat{\tau})p^*(k+1;\hat{\tau})] \qquad \text{(BPSK,QPSK)}$$

T.6 JOINT CARRIER AND SYMBOL SYNCHRONIZATION ALGORITHMS

JDDMLFF

$$\hat{\tau}: \max_{\tilde{\tau}} \left[\left| \sum_{k=1}^{K} (\hat{a}_k \, p(k;\tilde{\tau}) - j\hat{b}_k \, q(k;\tilde{\tau})) \right| \right] \qquad \text{(BPSK,(O)QPSK)}$$

$$\hat{\theta} = \arg \left[\sum_{k=1}^{K} (\hat{a}_k \, p(k;\hat{\tau}) - j\hat{b}_k \, q(k;\hat{\tau})) \right] \qquad \text{(BPSK,(O)QPSK)}$$

JNDAMLFFB/O, JNDAMLFFQ

$$\hat{\tau}: \max_{\tilde{\tau}} \left[\sum_{k=1}^{K} (|p(k;\tilde{\tau})|^2 + |q(k;\tilde{\tau})|^2) + \left| \sum_{k=1}^{K} (p^2(k;\tilde{\tau}) - q^2(k;\tilde{\tau})) \right| \right] \qquad \text{(BPSK,OQPSK)}$$

$$\hat{\theta} = \frac{1}{2} \arg \left[\sum_{k=1}^{K} (p^2(k;\hat{\tau}) - q^2(k;\hat{\tau})) \right] \qquad \text{(BPSK,OQPSK)}$$

or

$$\hat{\tau}: \max_{\tilde{\tau}} \left[\sum_{k=1}^{K} |p(k;\tilde{\tau})|^2 \right] \qquad \text{(QPSK)}$$

$$\hat{\theta} = \frac{1}{4} \arg \left[- \sum_{k=1}^{K} p^4(k;\hat{\tau}) \right] \qquad \text{(QPSK)}$$

T.7 TRACKING ERROR VARIANCE AT MODERATE E_s/N_o

T.7.1 CARRIER SYNCHRONIZATION

TRACKING ERROR VARIANCE AT MODERATE E_s/N_o

$$\mathrm{Var} \cong (2B_L T)\, A \frac{N_o}{2E_s}$$

CARRIER SYNCHRONIZATION

		Degradation [dB] from CRB
DDML, NDAMLB/Q, V&VB/Q (same as CRB)		
$\alpha = 0.$	$A = 1.$	0.
$\alpha = 0.5$	$A = 1.$	0.
$\alpha = 1.$	$A = 1.$	0.
NDAMLO, NDASQFF$\bar{T}$O		
$\alpha = 0.$	$A = \infty$	∞
$\alpha = 0.5$	$A = 4.$	6.
$\alpha = 1.$	$A = 2.$	3.
NDAML$\bar{T}$B (continuous–time operation)		
$\alpha = 0.$	$A = 1.$	0.
$\alpha = 0.5$	$A = 1.061$	0.26
$\alpha = 1.$	$A = 1.111$	0.45

T.7.2 SYMBOL SYNCHRONIZATION

TRACKING ERROR VARIANCE AT MODERATE E_s/N_o

$$\mathrm{Var} \cong (2B_L T)\, A \frac{N_o}{2E_s}$$

SYMBOL SYNCHRONIZATION

		Degradation [dB] from CRB
DDML (same as CRB)		
$\alpha = 0.$	A = 0.304	0.
$\alpha = 0.5$	A = 0.266	0.
$\alpha = 1.$	A = 0.194	0.
DDEL(λ=1/2), DTTL		
$\alpha = 0.$	A = 0.308	0.05
$\alpha = 0.5$	A = 0.277	0.17
$\alpha = 1.$	A = 0.222	0.58
NDAML, NDASQFF, NDAML$\bar{C}$B/Q, NDASQFF$\bar{C}$B/Q, NDASQFF$\bar{C}$O		
$\alpha = 0.$	A = ∞	∞
$\alpha = 0.5$	A = 0.405	1.84
$\alpha = 1.$	A = 0.203	0.20
NDAEL(λ=1/2), GA, NDAEL(λ=1/2)$\bar{C}$B/Q, GA$\bar{C}$B/Q		
$\alpha = 0.$	A = ∞	∞
$\alpha = 0.5$	A = 0.406	1.84
$\alpha = 1.$	A = 0.211	0.36
DDEL(λ=1), M&M, mM&M		
$\alpha = 0.$	A = 0.5	2.16
$\alpha = 0.5$	A = 0.811	4.84
$\alpha = 1.$	A = 4.5	13.66
NDANLFB$\bar{C}$B		
$\alpha = 0.$	A = 0.5	2.16
$\alpha = 0.5$	A = 0.811	4.84
$\alpha = 1.$	A = 4.5	13.66
NDANLFB$\bar{C}$Q		
$\alpha = 0.$	A = 1.	5.16
$\alpha = 0.5$	A = 1.621	7.84
$\alpha = 1.$	A = 9.	16.66

REFERENCES

[Ana'86] F. Ananasso, L. Cellai and G. Gallinaro, "Study of reference recovery for serial demodulation of offset–binary modulation schemes," Final report to ESA contract No. 6253/85/NL/DG, Nov. 1986.

[And'79] B.D.O. Anderson and J.B. Moore, Optimal filtering. Englewood–Cliffs, NJ : Prentice–Hall, 1979.

[Asc'82] G. Ascheid and H. Meyr, "Cycle slips in phase–locked loops : a tutorial survey," IEEE Trans. Commun., vol. COM–30, pp. 2228–2241, Oct. 1982.

[Asc'89] G. Ascheid, M. Oerder, J. Stahl and H. Meyr, "An all digital receiver architecture for bandwidth efficient transmission at high data rates," IEEE Trans. Commun., vol. COM–37, pp. 804–813, Aug. 1989.

[DAn'85] N.A. D'Andrea, U. Mengali and R. Reggiannini, "Synchronization of CPM signals," Proc. 2nd Tirrenia International Workshop on Digital Communications, Tirrenia, Italy, pp. 97–106, Sept. 1985.

[deB'72] R. de Buda, "Coherent demodulation of frequency–shift keying with low deviation ratio," IEEE Trans. Commun., vol. COM–20, pp. 429–435, June 1972.

[Fal'77] D.D. Falconer and J. Salz, "Optimal reception of digital data over the Gaussian channel with unknown delay and phase jitter," IEEE Trans. Inform. Theory, vol. IT–23, pp.117–126, Jan. 1977.

[Fra'74] L.E. Franks and J.B. Bubrouski, "Statistical properties of timing jitter in a PAM timing recovery scheme," IEEE Trans. Commun., vol. COM–22, pp. 913–920, July 1974.

[Fra'80] L.E. Franks, "Carrier and bit synchronization in data communication – a tutorial review," IEEE Trans. Commun., vol. COM–28, pp. 1107–1121, Aug. 1980.

[Gar'86] F.M. Gardner, "A BPSK/QPSK timing–error detector for sampled receivers," IEEE Trans. Commun., vol. COM–34, pp. 423–429, May 1986.

[Gar'88] F.M. Gardner, "Demodulator reference recovery techniques suited for digital implementation," Final report to ESA contract No. 6847/86/NL/DG, Aug. 1988.

[Gar'90a] F.M. Gardner, "Timing adjustment via interpolation in digital demodulators" Final report (part I) to ESA contract No.

8022/88/NL/DG, June 1990.

[Gar'90b] F.M. Gardner, "Frequency detectors for digital demodulators via maximum–likelihood derivation," Final report (part II) to ESA contract No. 8022/88/NL/DG, June 1990.

[Git'71] R.D. Gitlin and J. Salz, "Timing recovery in PAM systems," BSTJ, vol. 50, pp. 1645–1669, May–June 1971.

[Har'91] R.A. Harris and M. Yarwood, "Carrier recovery and inter–burst interference in an MF–TDMA system," to appear in Int. Journal Satellite Commun., 1991.

[Kob'71] H. Kobayashi, "Simultaneous adaptive estimation and decision algorithm for carrier modulated data transmission systems, IEEE Trans. Commun., vol. COM–19, pp. 268–280, June 1971.

[Lin'72] W.C. Lindsey and M.K. Simon, Telecommunication systems engineering. Englewood Cliffs, NJ : Prentice–Hall, 1972.

[Lin'77] W.C. Lindsey and H. Meyr, "Complete statistical description of the phase–error process generated by correlative tracking systems," IEEE Trans. Inform. Theory, vol. IT–23, pp. 194–202, March 1977.

[Max'60] J. Max, "Quantization for minimum distortion," IRE Trans. Inform. Theory, vol. IT–6, pp.7–12, March 1960.

[Men'77] U. Mengali, "Joint phase and timing acquisition in data transmission," IEEE Trans. Commun., vol. COM–25, pp.1174–1185, Oct. 1977.

[Mey'75] H. Meyr, "Nonlinear analysis of correlative tracking systems using renewal process theory," IEEE Trans. Commun., vol. COM–23, pp. 192–203, Febr. 1975.

[Mey'80a] M.H. Meyers and L.E. Franks, "Joint carrier phase and symbol timing recovery for PAM systems," IEEE Trans. Commun., vol. COM–28, pp. 1121–1129, Aug. 1980.

[Mey'80b] H. Meyr and L. Popken, "Phase acquisition statistics for phase–locked loops," IEEE Trans. Commun., vol. COM–28, pp. 1365–1372, Aug. 1980.

[Mey'90] H. Meyr and G. Ascheid, Synchronization in digital communications, volume I : Phase–, frequency–locked loops and amplitude control. New York : John Wiley and Sons, 1990.

[Moe'83] M. Moeneclaey, "A comparison of two types of symbol synchronizers for which self–noise is absent", IEEE Trans. Commun., vol. COM–31, pp. 329–334, March 1983.

[Moe'84a] M. Moeneclaey, "Prefilter optimization for the filter and square synchronizer," Archiv Elektr. Ubertr., vol. AEU–38, pp. 257–261,

July 1984.

[Moe'84b] M. Moeneclaey, "Two maximum–likelihood synchronizers with superior tracking performance," IEEE Trans. Commun., vol. COM–32, pp. 1178–1185, Nov. 1984.

[Moe'85a] M. Moeneclaey, "The asymptotic cycle slip rate for a synchronization loop with arbitrary phase–detector characteristic," Archiv Elektr. Ubertr., vol. AEU–39, pp. 72–74, Jan. 1985.

[Moe'85b] M. Moeneclaey, "The influence of phase–dependent loop noise on the cycle slipping of symbol synchronizers," IEEE Trans. Commun., vol. COM–33, pp. 1234– 1239, Dec. 1985.

[Moe'86] M. Moeneclaey and I. Bruyland, "The joint carrier and symbol synchronizability of continuous phase modulated waveforms," Proc. ICC'86. Toronto, Canada, pp. 987–991, June 1986.

[Moe'88a] M. Moeneclaey, S. Starzak and H. Meyr, "Cycle slips in synchronizers subject to smooth narrowband loop noise," IEEE Trans. Commun., vol. COM–36, pp. 867–874, July 1988.

[Moe'88b] M. Moeneclaey, "Tracking performance comparison of clock synchronization algorithms suited for digital implementation," Proc. First International Workshop on Digital Signal Processing Techniques applied to Space Communications, Noordwijk, The Netherlands, pp. 98–104, 9–10 Nov. 1988.

[Moe'90a] M. Moeneclaey and U. Mengali, "Sufficient conditions on trellis–coded modulation for code–independent synchronizer performance," IEEE Trans. Commun., vol. COM–38, pp. 595–601, May 1990.

[Moe'90b] M. Moeneclaey and T. Batselé, "Carrier–independent NDA symbol synchronization for M–PSK, operating at only one sample per symbol," Globecom'90, paper 407.2, San Diego, Dec. 2–5, 1990.

[Moe'90c] M. Moeneclaey and G. Ascheid, "Extension of the Viterbi and Viterbi carrier synchronization algorithm to OQPSK transmission," Proc. Second international workshop on digital signal processing techniques applied to space communications, Torino, paper 1–7, Sept. 1990.

[Mue'76] K.H. Mueller and M. Müller, "Timing recovery in digital synchronous data receivers," IEEE Trans. Commun., vol. COM–24, pp. 516–530, May 1976.

[Oer'88] Oerder and H. Meyr, "Digital filter and square timing recovery," IEEE Trans. Commun., vol. COM–36, pp. 605–612, May 1988.

[Rya'80] C.R. Ryan, A.R. Hambley and D.E. Voght, "760 Mbit/s serial MSK

microwave modem," IEEE Trans. Commun., vol. COM–28, pp. 771–777, May 1980.

[Ryt'78] D. Ryter and H. Meyr, "Theory of phase tracking systems of arbitrary order : statistics of cycle slips and probability distribution of the state vector," IEEE Trans. Inform. Theory, vol. IT–24, pp. 1–7, Jan. 1978.

[Sim'77] M.K. Simon and W.C. Lindsey, "Optimum performance of suppressed carrier receivers with Costas loop tracking," IEEE Trans. Commun., vol. COM–25, pp. 215–227, Febr. 1977.

[Sim'78a] M.K. Simon, "Tracking performance of Costas loops with hard–limited in–phase channel," IEEE Trans. Commun., vol. COM–26, pp. 420–432, April 1978.

[Sim'78b] M.K. Simon, "Optimum receiver structures for phase– multiplexed receivers," IEEE Trans Commun., vol. COM–26, pp. 865–872, June 1978.

[Sim'79] M.K. Simon, "On the optimality of the MAP estimation loop for carrier phase tracking BPSK and QPSK signals," IEEE Trans. Commun., vol. COM–27, pp. 158–165, Jan. 1979.

[Vit'83] A.J. Viterbi and A.M. Viterbi, "Nonlinear estimation of PSK–modulated carrier phase with application to burst digital transmission," IEEE Trans. Inform. Theory, vol. IT–29, pp. 543–551, July 1983.

EUROPEAN SPACE AGENCY

FINAL REPORT
ESTEC Contract No. 8437/89/NL/RE

DIGITAL DEMODULATOR SYNCHRONIZATION

ANNEX A

COSSAP PRIMITIVE MODEL DOCUMENTATION

R.A. Harris (TST),
ESTEC Technical Manager

June, 1991

SAIT Electronics S.A.
Chaussée de Ruisbroek, 66
B-1190 Brussels
(BELGIUM)

Rijksuniversiteit Gent
Sint-Pietersnieuwstraat, 41
B-9000 Gent
(BELGIUM)

CADIS GmbH
Jülicher Str. 320
D-5100 Aachen
(GERMANY)

Table of Contents

APLC . . . Automatic Power Level Control 1
BSRC2 . . Pseudo Random Binary Source (31 Bit Shift Register) 3
DELAYR . Delays Real Signals (by multiples of the sampling time) 4
DELR . . . Deletes First Signal Elements (Real Signals) 5
DMUXR . Demultiplexer (1 in 2, blockwise alternating, Real Signals) 6
HOLDR . . Repeats Each Signal Element n Times (Real Signals) 7
ILPF . . . Ideal Lowpass Filter (Real Input Signal) 8
IMPMOD . Impulse Modulator 10
INTDMP . Integrate and Dump 12
KILLR . . Signal Sink (Real Signals) 13
LPDIQC . Differentiating Filter (Complex Input Signal) 14
MATAN2 . Computes Arcus Tangens of a Complex Input Signal 15
MAXRP . Searches Maximum Position in a Real Input Signal Block 16
MCVMUL . Computes the Product of Two Complex Input Signals 17
MCVSUB . Subtracts Two Complex Input Signals 19
MPOLAR . Converts Complex Signal from Cartesian to Polar Representaion 20
MPSKSD . M-ary PSK Demodulator 21
MPSKSM . M-ary PSK Modulator 23
MRECT . . Converts Complex Signal from Polar to Cartesian Representation 25
MVADD . Adds Two Real Input Signals 26
MVCOS . Computes Cosine of a Real Input Signal 27
MVMUL . Multiplies Two Real Input Signals 28
MVSIN . . Computes Sine of a Real Input Signal 29
MVSMUL . Multiplies a Real Input Signal by a Real Scalar 30
MVSUB . Subtracts Two Real Input Signals 31
P2SR . . . Parallel to Serial Conversion (N in 1, Sample by Sample Multiplexer, Real Signals) . 32
PSD1 . . . Power Spectral Density Estimation (Bartlett Method, Real Signal) 33
RCRFQC . Root Raised Cosine Filter (Complex Input Signal) 36
S2PR . . . Serial to Parallel Conversion (1 in N, sample by sample Demultiplexer, Real Signals) 38
SRDR . . Sampling Rate Decimation (Real Signals) 39
STAT . . . Signal Statistics - Mean, Power, Variance (Text Chart) 40
STATP . . Signal Statistics - Mean, Power, Variance (Line Chart) 42
SYMGCG Binary to M-ary Conversion, Gray Code 44
TWT5AP . Traveling Wave Tube (Extension of Amplitude-Phase Model by Saleh) 46

Note that the following models have been renamed in COSSAP release 6.4:

Old Name	**New Name**
DELETE	DELR
DEMUX	DMUXR
HOLD	HOLDR
KILL	KILLR
P2S	P2SR
S2P	S2PR
SRD	SRDR

APLC

Automatic Power Level Control

Classes

ESTIMATOR
CONTROL

Description

This model adjusts the power of the output signal to a constant level. The power level is given by the parameter OUTPUT_POWER.

Algorithm:

The input signal power estimate (Input_Power) equals the average power of all received input signal samples. The estimate is updated once per call (i.e. blockwise) using all received input signal samples including the current input buffer contents.

The output signal equals the input signal multiplied by the square root of the ratio of OUTPUT_POWER to the estimated power of the input signal, i.e.

```
OUTPUT = INPUT * sqrt(   OUTPUT_POWER
                       / Power_of_input_signal )
```

The power of the input signal is assumed to be non time varying.

Applications:

This model can be used to adjust the power level of signals of unknown, constant power (e.g. coloured noise or modulated signals after filtering).

Note:
For the first signal elements, the accuracy of the estimate of the power of the input signal will strongly depend on the number of signal elements processed.
In case this model is part of a feed back loop, the

estimate of the power of the input signal will be poor for the first signal elements.

The number of signal elements processed each time this model is executed can be influenced via the input scheduling conditions.

Parameters

OUTPUT_POWER type: **real**

Establishes the signal power of the real output signal

Input Ports

INPUT* type: **real**

Real input signal

Output Ports

OUTPUT* type: **real**

Real output signal of known signal power

BSRC2

Pseudo Random Binary Source (31 Bit Shift Register)

Classes

SOURCE
NOISE
DIGITAL

Description

This model is a pseudo random binary source implemented as a maximum period linear feedback 31 bit shift register.

Parameters

SHIFT_REGISTER_START_VALUE type: **Integer**
Start value of the linear feedback shift register

Output Ports

OUT_BIN type: **Integer**
Integer output signal

Error Codes

WRONG_VALUE_SHIFT_REGISTER_START_VAL error code: **1**
The value assigned to the parameter SHIFT_REGISTER_START_VAL generates a zero sequence.

DELAYR

Delays Real Signals (by multiples of the sampling time)

Classes

FLOW_CONTROL

Description

This model delays a real input signal by a fixed number of signal elements. The number of signal elements is given by the parameter DELAY.

The delay is introduced by insertion of signal elements with value 0 at the beginning of the output signal.

Parameters

DELAY type: **Integer**

Establishes the number of signal elements by which the real input signal is delayed, i.e. the number of 0 inserted at the beginning of the real output signal

Input Ports

INPUT* type: **real**

Real input signal

Output Ports

OUTPUT* type: **real**

Real output signal delayed by DELAY signal elements with respect to the real input signal

DELR

Deletes First Signal Elements (Real Signals)

Classes

FLOW_CONTROL

Description

This model deletes the first NUM_ITEMS signal elements in the input signal and then behaves like the unit transfer channel.

Parameters

NUM_ITEMS type: **integer**
Establishes the number of signal elements to deleted in the input signal.

Input Ports

INPUT* type: **real**
Real input signal

Output Ports

OUTPUT* type: **real**
Real output signal

DMUXR

Demultiplexer (1 in 2, blockwise alternating, Real Signals)

Classes

FLOW_CONTROL

Description

This model demultiplexes one input signal into two output signals by alternatingly writing ITEMS_TO_OUTPUT1 items to output port 1 and ITEMS_TO_OUTPUT2 items to output port 2.

The corresponding multiplexer model is MUXR.

Parameters

ITEMSS_TO_OUTPUT1 type: **integer**
Number of items that are written to OUTPUT1 at each demultiplexing operation

ITEMS_TO_OUTPUT2 type: **integer**
Number of items that are written to OUTPUT2 at each demultiplexing operation

Input Ports

INPUT type: **real**
Real input signal

Output Ports

OUTPUT1 type: **real**
First demultiplexed real output signal

OUTPUT2 type: **real**
Second demultiplexed real output signal

HOLDR

Repeats Each Signal Element n Times (Real Signals)

Classes

DIGITAL

Description

This model writes each signal element in the input signal HOLD_FACTOR times to the output port.

Note:

The value assigned to the parameter HOLD_FACTOR should not exceed the execution site's standard block length, usually 512. In case larger values are required, the user is advised to cascade more than one HOLD model.

Parameters

HOLD_FACTOR type: **integer**
Establishes how many times each signal element is repeated.

Input Ports

INPUT* type: **real**
Real input signal

Output Ports

OUTPUT* type: **real**
Real output signal

Error Codes

WRONG_VALUE_HOLD_FACTOR error code: **1**
The value assigned to the variable HOLD_FACTOR is less than 1.

ILPF

Ideal Lowpass Filter (Real Input Signal)

Classes

FILTER_REAL

Description

ILPF is a filter model, which approximates the transfer function of an ideal low pass filter for real valued input signals. The impulse response is weighted by a Blackman-window of length FFT_LENGTH.

Parameter SLOPE_WIDTH returnes the approximate width of the slope of the transfer function, i.e. the difference between the highest undamped frequency and the first zero of the transfer function. Parameter ATT_1ST_SIDE_LOBE is the attenuation (in dB) of the first sidelobe of the transfer function (between the first two zeros) and is nearly equal to the minimum stopband attenuation.

Reference:

Oppenheim, Schafer, "Digital Signal Processing", Prentice Hall 1975, p. 242

Parameters

SAMPLING_TIME type: **real**
Sampling time

CUTOFF_FREQUENCY type: **real**
Cutoff frequency: FCUT will be rounded to an integer multiple of 1/(N*TSMPL);the rounding error is small compared with the slope width.

GAIN type: **real**
Filter gain (value of transfer function at f=0)

FFT_LENGTH type: **integer**
FFT length; the impulse response consists of N/2+1 samples

SLOPE_WIDTH type: **real**
Slope width (see DESCRIPTION)

ATT_1ST_SIDE_LOBE type: **real**

Attenuation of the first side lobe (see DESCRIPTION)

Input Ports

INPUT type: **real**

Real input signal

Output Ports

OUTPUT type: **real**

Real output signal

IMPMOD

Impulse Modulator

Classes

MODEM

Description

This model represents an impulse modulator, followed by an ideal lowpass filter and a sampling device.

The impulse modulator generates impulses which are weighted by the incoming signal elements and have a sampling rate of

```
1/SAMPLING_TIME = RATE_FACTOR * Symbol_Rate
                = RATE_FACTOR / Symbol_Time
```

The lowpass filter has a cutoff frequency of

```
0.5/SAMPLING_TIME
```

Thus the model simply multiplies each input signal element by 1./SAMPLING_TIME and inserts RATE_FACTOR-1 zeros after each result.

Parameters

RATE_FACTOR type: **integer**

Establishes the factor between the sampling rate of the input signal and the sampling rate of the output signal (i.e. of the impulse)

SAMPLING_TIME type: **real**

Establishes the sampling time of the output signal.

Input Ports

INPUT* type: **real**

Real inpur signal

Output Ports

OUTPUT* type: **real**

Expanded real output signal

INTDMP

Integrate and Dump

Classes

ARITH_REAL
FILTER_REAL
DIGITAL

Description

INTDMP performs an integrate and dump upon the input signals by summing up NUM_SAMPLES signal elements of the input signals and dividing the result by the NUM_SAMPLES. For each NUM_SAMPLES signal elements, i.e. for each integrate and dump operation, only one signal element is written to the output port.

Parameters

NUM_SAMPLES type: **integer**
Establishes the number of signal elements involved in one integrate and dump operation.

Input Ports

INPUT* type: **real**
Real input signal

Output Ports

OUTPUT* type: **real**
Real output signal

KILLR

Signal Sink (Real Signals)

Classes

FLOW_CONTROL

Description

This model reads all signal elements from its input ports. No further processing is done.

Application:

This model can be used to connect otherwise unconnected signals.

Input Ports

INPUT* type: **real**

Real input signal

LPDIQC

Differentiating Filter (Complex Input Signal)

Classes

FILTER_CMPLX

Description

LPDIQC is a filter model which approximates the transfer function of an ideal differentiation filter. The impulse response is weighted by a rect window of length N/2. The transfer function is computed by subroutine LPDTRF.

Parameters

SAMPLING_TIME type: **real**
Establishes the sampling time in seconds.

FFT_LENGTH type: **integer**
Establishes the FFT length. The length of the impulse response generated will be FFT_LENGTH/2. Note that the FFT lenght must be a power of 2.

Input Ports

IN_REAL type: **real**
Real part of the complex input signal

IN_IMAG type: **real**
Imaginary part of the complex input signal

Output Ports

OUT_REAL type: **real**
Real part of the complex output signal

OUT_IMAG type: **real**
Imaginary part of the complex output signal

MATAN2

Computes Arcus Tangens of a Complex Input Signal

Classes

ARITH_CMPLX

Description

This model computes the arcus tangens of a complex input signal.

```
OUTPUT = atan(IN_REAL,IN_IMAG)
```

Note:

This model implements the APMATH64 routine VATAN2.

Applications:

This model can be used to compute the phase of a complex input signal.

Input Ports

IN_REAL type: **real**
Real part of the complex input signal

IN_IMAG type: **real**
Imaginary part of the complex input signal

Output Ports

OUT_PHASE type: **real**
Phase of the complex input signal

MAXRP

Searches Maximum Position in a Real Input Signal Block

Classes

ARITH_REAL

Description

This module searches the maximum position in a real input signal block and writes it to the output port. The maximum position is given relative to the individual block center positions and normalized to the range (-1/2,+1/2), i.e. a mid block maximum corresponds to position 0. The number of real signal elements in each block is given by the parameter BLOCK_LENGTH.
Thus, the real output signals are in the range
[-(1-1/BLOCK_LENGTH)/2 , +(1-1/BLOCK_LENGTH)/2].
Note: For each block processed one real signal element, namely the maximum position, is written to the output port. Therefore the output rate of this module is only 1/BLOCK_LENGTH of the input rate.

Parameters

BLOCK_LENGTH type: **integer**
Establishes the number of signal elements per block.

Input Ports

INPUT* type: **real**
Real input signal

Output Ports

OUT_MAX_POS* type: **real**
Maximum positions relative to block center position

MCVMUL

Computes the Product of Two Complex Input Signals

Classes

ARITH_CMPLX

Description

This model computes the product of two complex input signals. Depending on the value of the parameter CONJUGATE_FLAG, the complex conjugate of the first complex input signal is multiplied by the second complex input signal, i.e.

```
if (CONJUGATE_FLAG = 0), then
  (OUT_REAL,OUT_IMAG) =    (IN1_REAL,IN1_IMAG)
                         * (IN2_REAL,IN2_IMAG)

if (CONJUGATE_FLAG = -1), then

  (OUT_REAL,OUT_IMAG) =    (IN1_REAL,-IN1_IMAG)
                         * (IN2_REAL, IN2_IMAG)
```

Note:

This model implements the APMATH64 routine CVMUL.

Parameters

CONJUGATE_FLAG type: **Integer**

The conjugate flag establishes whether the complex conjugate of the first complex input signal must be computed before the complex multiplication is performed or not.

If CONJUGATE_FLAG = 0, then C = A * B

If CONJUGATE_FLAG = -1, then C = (conjugate complex (A)) * B

Input Ports

IN1_REAL type: **real**

Real part of first complex input signal

IN1_IMAG type: **real**

Imaginary part of first complex input signal

IN2_REAL type: **real**

Real part of second complex input signal

IN2_IMAG type: **real**

Imaginary part of second complex input signal

Output Ports

OUT_REAL type: **real**

Real part of the complex output signal

OUT_IMAG type: **real**

Imaginary part of the complex output signal

MCVSUB

Subtracts Two Complex Input Signals

Classes

ARITH_CMPLX

Description

This model subtracts two complex input signals. The signals at the IN_SUB_* input ports are subtracted from the signals at the IN_MIN_* input ports, i.e.

```
OUT_REAL = IN_MIN_REAL - IN_SUB_REAL
OUT_IMAG = IN_MIN_IMAG - IN_SUB_IMAG
```

Note:

This model implements the APMATH64 routine CVSUB.

Input Ports

IN_SUB_REAL type: **real**
Real part of complex subtrahend

IN_SUB_IMAG type: **real**
Imaginary part of complex subtrahend

IN_MIN_REAL type: **real**
Real part of complex minuend

IN_MIN_IMAG type: **real**
Imaginary part of complex minuend

Output Ports

OUT_REAL type: **real**
Real part of complex output signal

OUT_IMAG type: **real**
Imaginary part of complex output signal

MPOLAR

Converts Complex Signal from Cartesian to Polar Representaion

Classes

ARITH_CMPLX

Description

This model converts a complex signal from cartesian to polar representation, i.e.

```
OUT_AMPLITUDE = sqrt(IN_REAL ** 2 + IN_IMAG **2)
OUT_PHASE     = arctan (IN_IMAG / IN_REAL)
```

The inverse operation, i.e. convert a complex signal from polar to cartesian representation, is performed by the model MRECT.

Note:

This model implements the APMATH64 routine MPOLAR

Input Ports

IN_REAL — type: **real**
Real part of the complex input signal

IN_IMAG — type: **real**
Imaginary part of the complex input signal

Output Ports

OUT_AMPLITUDE — type: **real**
Amplitude of the complex output signal

OUT_PHASE — type: **real**
Phase of the complex output signal

MPSKSD

M-ary PSK Demodulator

Classes

MODEM

Description

This model is a M-ary PSK demodulator. The detected channel symbols are mapped onto the symbol alphabet

k = 0, ... ,M-1

The transmitted symbols are assumed to be

exp(j*2*PI*(k+1/2)/M)

except for BPSK where exp(j*2*PI*k/M) is assumed.

First the squared Euclidian distance of the received complex symbol to each complex channel symbol defined for the modulation scheme is computed. The symbol with the lowest squared Euclidian distance to the received complex symbol is written to the output port.

Note:

The inverse operation, mapping from the symbol alphabet

k = 0, ... ,M-1

onto the complex symbol space, is done by the model MPSKSM.

This model is similar to the model MPSKD. In that model the transmitted symbols are assumed to be

exp(j*2*PI*k/M)

for all orders M of the signal space.

Parameters

M type: **Integer**

Establishes the order of the signal space. M=2 corresponds to BPSK, M=4 to QPSK, etc.

Input Ports

IN_REAL type: **real**

Real part of the complex input signal

IN_IMAG type: **real**

Imaginary part of the complex input signal

Output Ports

OUT_SYMBOL type: **Integer**

Integer output signal with output symbols from the demodulator. The symbol alphabet is 0, ... ,M-1.

Error Codes

WRONG_VALUE_M error code: **1**

The value assigned to the parameter M is less than 2.

MPSKSM

M-ary PSK Modulator

Classes

MODEM

Description

This model is a M-ary PSK modulator. The order of the signal space is given by the parameter M. The input symbol alphabet

```
k = 0, ... ,M-1
```

is mapped onto a two dimensional complex symbol space

```
exp(j*2*pi*(k+1/2)/M)
```

i.e.,

```
(OUT_REAL,OUT_IMAG) = exp(j*2*pi*(IN_SYMBOLS+1/2)/M)
```

except for BPSK where the complex symbol space is

```
exp(j*2*PI*k/M)
```

i.e.,

```
(OUT_REAL,OUT_IMAG) = exp(j*2*pi*IN_SYMBOLS/M)
```

Note:

The inverse operation, mapping from the complex symbol space onto the symbol alphabet

```
k = 0, ... ,M-1
```

is done by the model MPSKSD.

This model is similar to the model MPSKM. In that model the input symbol alphabet is mapped onto a two dimensional complex symbol space

```
exp(j*2*PI*k/M)
```

for all orders M of the signal space.

Parameters

M type: **Integer**

Establishes the order of the signal space. For M=2 BPSK is obtained, for M=4 QPSK, etc.

Input Ports

IN_SYMBOL type: **Integer**

Integer input signal with input symbols to the modulator. The symbol alphabet is 0, ... ,M-1.

Output Ports

OUT_REAL type: **real**

Real part of complex output signal: cannel symbols

OUT_IMAG type: **real**

Imaginary part of complex output signal: channel symbols

Error Codes

WRONG_VALUE_M error code: **1**

The value assigned to the parameter M is less than 2.

MRECT

Converts Complex Signal from Polar to Cartesian Representation

Classes

ARITH_CMPLX

Description

This model converts a complex signal from polar to cartesian representation, i.e.

```
OUT_REAL = IN_AMPLITUDE * cos(IN_PHASE)
OUT_IMAG = IN_AMPLITUDE * sin(IN_PHASE)
```

Note:

This model implements the APMATH64 routine RECT

Input Ports

IN_AMPLITUDE type: **real**
Amplitude of the complex input signal

IN_PHASE type: **real**
Phase of the complex input signal

Output Ports

OUT_REAL type: **real**
Real part of the complex output signal

OUT_IMAG type: **real**
Imaginary part of the complex output signal

MVADD

Adds Two Real Input Signals

Classes

ARITH_REAL

Description

This model adds two real input signals, i.e.

```
OUTPUT = INPUT1 + INPUT2
```

Note:

This model implements the APMATH64 routine VADD.

Input Ports

INPUT1 type: **real**
First real input signal

INPUT2 type: **real**
Second real input signal

Output Ports

OUTPUT type: **real**
Real output signal resulting from adding the two real input signals

MVCOS

Computes Cosine of a Real Input Signal

Classes

ARITH_REAL

Description

This model computes the cosine of a real input signal, i.e.

```
OUTPUT = cos(INPUT)
```

Note:

This model implements the APMATH64 routine VCOS.

Input Ports

INPUT type: **real**
Real input signal

Output Ports

OUTPUT type: **real**
Cosine of the real input signal

MVMUL

Multiplies Two Real Input Signals

Classes

ARITH_REAL

Description

This model multiplies two real input signals, i.e.

```
OUTPUT = INPUT1 * INPUT2
```

Note:

This model implements the APMATH64 routine VMUL.

Input Ports

INPUT1
First real input signal — type: **real**

INPUT2
Second real input signal — type: **real**

Output Ports

OUTPUT
Real output signal — type: **real**

MVSIN

Computes Sine of a Real Input Signal

Classes

ARITH_REAL

Description

This model computes the sine of a real input signal, i.e.

```
OUTPUT = sin(INPUT)
```

Note:

This model implements the APMATH64 routine VSIN.

Input Ports

INPUT type: **real**
Real input signal

Output Ports

OUTPUT type: **real**
Sine of the real input signal

MVSMUL

Multiplies a Real Input Signal by a Real Scalar

Classes

ARITH_REAL

Description

This model multiplies each signal element in a real input signal by a real scalar. The value of the scalar is given by the parameter SCALAR, i.e.

```
OUTPUT = INPUT * SCALAR
```

Note:

This model implements the APMATH64 routine VSMUL.

Parameters

SCALAR type: **real**

Establishes the value of the real scalar by which each signal element in the real input signal is multiplied.

Input Ports

INPUT type: **real**

Real input signal

Output Ports

OUTPUT type: **real**

Real output signal resulting from multiplying each signal element in the real input signal by the real scalar

MVSUB

Subtracts Two Real Input Signals

Classes

ARITH_REAL

Description

This model subtracts two real input signals. The signal at the IN_SUB input port is subtracted from the signal at the IN_MIN input port, i.e.

```
OUTPUT = IN_MIN - IN_SUB
```

Note:

This model implements the APMATH64 routine VSUB.

Input Ports

IN_SUB type: **real**
Real input signal: subtrahend

IN_MIN type: **real**
Real input signal: minuend

Output Ports

OUTPUT type: **real**
Difference of the two real input signals

P2SR

Parallel to Serial Conversion (N in 1, Sample by Sample Multiplexer, Real Signals)

Classes

FLOW_CONTROL

Description

P2S performs parallel to serial conversion of an arbitrary number of real input signals.

Let INPUT1, INPUT2, ... INPUTn be n input signals with samples

INPUT1(1+k), INPUT2(1+k), ..., INPUTn(1+k), k=0,1,2,...

Then the output signal is given by

```
OUTPUT(1 +n*k) = INPUT1(1+k)
OUTPUT(2 +n*k) = INPUT2(1+k)
...
OUTPUT(n +n*k) = INPUTn(1+k) ,          k=0,1,2,...
```

Input Ports

INPUT*
Real input signal — type: **real**

Output Ports

OUTPUT
Serialized real output signal from input signals — type: **real**

PSD1

Power Spectral Density Estimation (Bartlett Method, Real Signal)

Classes

ESTIMATOR
OUTPUT
STATISTICS

Description

This model computes an estimate of the power spectral density for real valued stochastic input signals by the Bartlett periodogram method and writes it into an output dataset. The output dataset will be diplayed as a line chart.

The user can select from two dataset types

<i>nternal
<e>xternal

<i>nternal datasets are store in the simulation output data file, the Y-file.

<e>xternal datasets are store in part in the simulation output data file, and in a plain file. The portion of the output dataset containing definitions for grafical presentation of the signal elements is store in the output data file. The signal elements are stored in the plain file. Since the plain file contains nothing but the signal elements, it is well suited for further processing. You can use it as an <e>xternal input dataset, or read it from within user written programs.

The plain file can be read with the sbio routines. In the plain file the power spectral density values are arranged as follows:

P1,P2,P3, ... ,Pk

The periodogram method works by accumulating NUMBER_OF_SEGMENTS short time spectra of length SEGMENT_LENGTH, which must be a power of 2 because of the FFT involved in the algorithm.

The input signal elements are not weighted (rectangular data window with unity value). Smoothing is done by multiplying the estimated autocorrelation function by a Bartlett lag window.

The evaluated signal segments are consecutive from FIRST_ITEM and nonoverlapping.

As the power spectral density of real valued signals is even, the number of frequency samples effectively obtained is (SEGMENT_LENGTH/2 +1).

Related COSSAP models for spectral and autocorrelation estimation are: ACF1, PSD1QC, PSD2, ACF2, PSD2QC.

Reference:

IEEE PRESS: Programs for Digital Signal Processing.
Ed. by the Digital Signal Processing Committee,
IEEE ASSP Society, N.Y., 1971.

Parameters

SEGMENT_LENGTH type: **integer**
Establishes the number of signal elements per signal segment. Note that it must be a power of 2.

FIRST_ITEM type: **integer**
Establishes the first signal element of the input signal to be involved in the estimation process. All previous signal elements of the input signal are discarded.

NUMBER_OF_SEGMENTS type: **integer**
Establishes the number of consecutive, non overlapping segments of the input signal that will be involved in the estimation process.

SAMPLING_TIME type: **real**
Establishes the sampling time of the real input signal.

Input Ports

INPUT type: **real**
Real input signal

Exit Flag

The exit flag is set as soon as NUMBER_OF_SEGMENTS segments of the input signal have been processed.

Output Datasets

SPECTRAL_DENSITY
Estimated power spectrum of input signal

Chart Type: LINE

Default Chart Format: 1

List of Chart Formats:

PSD OVER FREQUENCY (line)	format:	**1**
PSD OVER FREQUENCY (marker)	format:	**2**
PSD OVER INDEX (line)	format:	**3**
PSD OVER INDEX (marker)	format:	**4**

RCRFQC

Root Raised Cosine Filter (Complex Input Signal)

Classes

FILTER_CMPLX

Description

RCRFQC is a filter model, which approximates the transfer function of a low-pass filter with slopes of the form

```
SQRT(0.5*(1-SIN(Pi*(f-F3DB)/(2*F3DB*ROLLOFF))))
```

i.e. the square of this transfer function is of the cosine-rolloff type. Note that F3DB also is the center frequency of the slope.

Parameter ATTENUATION_ISI (ATTENUATION InterSymbol Interference) returns the attenuation of intersymbol interference (see first Nyquist-condition).

Parameter ATTENUATION_NCD (ATTENUATION Neighbour Channel Distor- tio) returns the attenuation of distortion caused by a signal sequence in a neighbour channel (freq. offset: 2*(F3DB+ROLLOFF)). This sequence is assumed to be of the same form (except mean frequency) as a sequence matched to the impulse response.

Reference:

Proakis, Digital Communications, p.338.

Parameters

SAMPLING_TIME type: **real**
Sampling time

CUTOFF_FREQUENCY_3DB type: **real**
3 dB cutoff-frequency

ROLLOFF type: **real**
ROLLOFF is the ratio beween one half of the slope width and F3DB

GAIN type: **real**
Filter gain (value of transfer function at f=0)

FFT_LENGTH type: **integer**
FFT-length; the impulse response consists of N/2+1 values must be a power of 2

ATTENUATION_ISI type: **real**
(see DESCRIPTION)

ATTENUATION_NCD type: **real**
(see DESCRIPTION)

Input Ports

IN_REAL type: **real**
Real part of the complex input signal

IN_IMAG type: **real**
Imaginary part of the complex input signal

Output Ports

OUT_REAL type: **real**
Real part of the complex output signal

OUT_IMAG type: **real**
Imaginary part of the complex output signal

Error Codes

ROLLOFF_OUT_OF_RANGE error code: **1**
Is set if a rolloff parameter outside the allowed range was specified. Reparameterize the rolloff so that it is in the range 0.LE.ROLLOF.LE.1 .

S2PR

Serial to Parallel Conversion (1 in N, sample by sample Demultiplexer, Real Signals)

Classes

FLOW_CONTROL

Description

S2PR performs serial to parallel conversion to an arbitrary number of parallel real output signals.

Let OUTPUT1, OUTPUT2,... , OUTPUTn be n output signals with samples

```
OUTPUT1(1+k), OUTPUT2(1+k), ..., OUTPUTn(1+k), k=0,1,2,...
```

Then the n output signals are given by

```
OUTPUT1(1+k) = INPUT(1+ n*k)
OUTPUT2(1+k) = INPUT(2+ n*k)
...
OUTPUTn(1+k) = INPUT(n+ n*k) ,            k=0,1,2,...
```

Input Ports

INPUT — type: **real**
Real input signal (serial)

Output Ports

OUTPUT* — type: **real**
Real output signal

SRDR

Sampling Rate Decimation (Real Signals)

Classes

DIGITAL

Description

This model decimates a real input signal. Beginning with the first signal element in the input signal, only each DECIMATOPN_FACTOR-th signal element is written to the output port.

Application:

This model can be used as an ideal sampling device.

Parameters

DECIMATION_FACTOR type: **integer**
Establishes the factor by which the input signal is decimated.

Input Ports

INPUT* type: **real**
Real input signal

Output Ports

OUTPUT* type: **real**
Decimated real output signal

Error Codes

WRONG_VALUE_DECIMATION_FACTOR error code: **1**
The value assigned to the parameter DECIMATION_FACTOR is less than 1.

STAT

Signal Statistics - Mean, Power, Variance (Text Chart)

Classes

OUTPUT

Description

This model computes the statistics of a real input signal. The parameter FIRST_ITEM establishes the first signal element that will be processed. The parameter NUM_ITEMS establishes the total number of signal elements that will be processed. The statistical information listed below is written into an output dataset.

The number of processed items : n

The signal mean value:

```
mean = sum (x(i);i=FIRST_ITEM to FIRST_ITEM+NUM_ITEMS) /n
```

The signal average power:

```
avgpow = sum (x(i)**2;i=FIRST_ITEM to
FIRST_ITEM+NUM_ITEMS) /n
```

The signal variance:

```
var = (avgpow - mean**2) *n/(n-1)
```

Parameters

FIRST_ITEM type: **Integer**
Establishes the first item to be processed.

NUM_ITEMS type: **Integer**
Establishes the number of items to be processed.

Input Ports

INPUT type: **real**
Real input signal for which statistics are computed.

Exit Flag

The exit flag is set as soon as NUM_ITEMS items have been processed.

Output Datasets

STATISTICS

Chart Type: STATISTICS

Default Chart Format: 1

List of Chart Formats:

STATISTICS	format:	1

STATP

Signal Statistics - Mean, Power, Variance (Line Chart)

Classes

OUTPUT

Description

This model computes the statistics of a real input signal. The parameter FIRST_ITEM establishes the first signal element that will be processed. The parameter NUM_ITEMS establishes the total number of signal elements that will be processed. The statistical information listed below is written into an output dataset.

The signal mean value:

```
mean = sum (x(i);
            i=FIRST_ITEM to FIRST_ITEM+NUM_ITEMS) / n
```

The signal average power:

```
avgpow = sum (x(i)**2;
              i=FIRST_ITEM to FIRST_ITEM+NUM_ITEMS) / n
```

The signal variance:

```
var = (avgpow - mean**2) *n/(n-1)
```

where n is the number of processed items.

The output dataset will be plotted as line charts containing a single marker. The x-value of the markers is established by the parameter X_VALUE, the y-value is one of the values listed above, depending on the chart format selected.

Parameters

FIRST_ITEM type: **integer**

Establishes the first signal element to be processed.

NUM_ITEMS type: **integer**

Establishes the number of signal elements to be processed.

X_VALUE type: **real**

Establishes the x-value in dB.

Input Ports

INPUT type: **real**

Real input signal for which the statistics are computed.

Exit Flag

The exit flag is set as soon as NUM_ITEMS items have been evaluated.

Output Datasets

STATISTICS

Chart Type: LINE

Default Chart Format: 1

List of Chart Formats:

MEAN	format:	**1**
POWER	format:	**2**
VARIANCE	format:	**3**

SYMGCG

Binary to M-ary Conversion, Gray Code

Classes

CODEC

Description

This model generates a Gray coded integer output signal made up of M-ary symbols in the range

0, ... ,2**(NUM_BITS) -1

from an integer input signal made up of 0s and 1s. The number of integer input signal elements (bits) to be packed into one symbol is given by the parameter NUM_BITS. The integer input signal is assumed to be ordered LSB first.

For each NUM_BITS integer input signal elements (bits) one integer output signal element (symbol) is generated.

Note:

The inverse operation, mapping from the symbol space to the bits, is accomplished by the model SYMGCD.

Parameters

NUM_BITS type: **Integer**
Establishes the number of bits to be packed into one symbol

Input Ports

IN_BINARY type: **Integer**
Integer input signal: bits, LSB first

Output Ports

OUT_SYMBOL type: **Integer**
Integer output signal: symbols

Error Codes

WRONG_VALUE_NUM_BITS error code: **1**

The value assigned to the parameter NUM_BITS is less than 1 or greater than 31.

TWT5AP

Traveling Wave Tube (Extension of Amplitude-Phase Model by Saleh)

Classes

NONLIN

Description

TWT5AP models a traveling wave tube based on an amplitude-phase representation of AM-AM and AM-PM nonlinearities. This amplitude-phase model is completely described by the ten parameters
A1_A,A2_A,B1_A,B2_A,B3_A
A1_P,A2_P,A3_P,B1_P,B2_P,
Formula:
OUTA(INA) =
INA*(A2_A*INA+A1_A)/(B3_A*INA**3+B2_A*INA**2+B1_A*INA+1)
OUTP(INA) = INP +
(A3_P*INA**3+A2_P*INA**2+A1_P*INA)/(B2_P*INA**2+B1_P*INA+1)
where INA = IN_AMPLITUDE, INP = IN_PHASE
OUTA = OUT_AMPLITUDE, OUTP = OUT_PHASE
For the Ku-band TWTA for OTS satellite the following parameters were obtained by curve fitting of measured data (phase in rad):

A2_A	= 0.012	A1_A	= 2.102	B3_A	= -0.102
B2_A	= 1.215	B1_A	= 0.000	A3_P	= -0.873
A2_P	= -3.491	A1_P	= 0.000	B2_P	= 4.000
B1_P	= -0.100				

The corresponding saturation level is IN_AMPLITUDE = 0 dB.
Note that this module works with amplitude and phase of input and output signal and not on their quadrature components.

Parameters

A2_A type: **real**

2nd order coefficient of nominator of AM-AM characteristic reference value for Ku band TWT for OTS satellite is
A2_A = 0.012

A1_A type: **real**

1st order coefficient of nominator of AM-AM characteristic reference value for Ku band TWT for OTS satellite is
A1_A = 2.102

B3_A type: **real**

3rd order coefficient of denominator of AM-AM characteristic reference value for Ku band TWT for OTS satellite is
B3_A = -0.102

B2_A type: **real**

2nd order coefficient of denominator of AM-AM characteristic reference value for Ku band TWT for OTS satellite is
B2_A = 1.215

B1_A type: **real**

1st order coefficient of denominator of AM-AM characteristic reference value for Ku band TWT for OTS satellite is
B1_A = 0.000

A3_P type: **real**

3rd order coefficient of nominator of AM-PM characteristic reference value for Ku band TWT for OTS satellite is
A3_P = -0.873

A2_P type: **real**

2nd order coefficient of nominator of AM-PM characteristic reference value for Ku band TWT for OTS satellite is
A2_P = -3.491

A1_P type: **real**

1st order coefficient of nominator of AM-PM characteristic reference value for Ku band TWT for OTS satellite is
A1_P = 0.000

B2_P type: **real**

2nd order coefficient of denominator of AM-PM characteristic reference value for Ku band TWT for OTS satellite is
B2_P = 4.000

B1_P type: **real**

1st order coefficient of denominator of AM-PM characteristic reference value for Ku band TWT for OTS satellite is
B1_P = -0.100

Input Ports

IN_AMPLITUDE type: **real**

Amplitude of input signal (V)

IN_PHASE type: **real**

Phase of input signal

Output Ports

OUT_AMPLITUDE type: **real**

Amplitude of output signal (V)

OUT_PHASE type: **real**

Phase of output signal