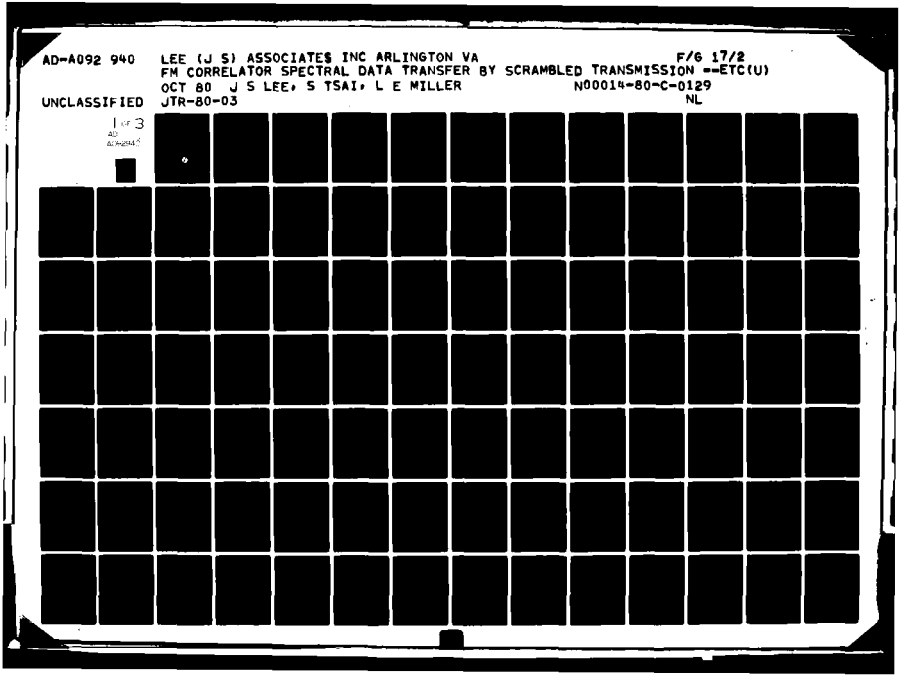
AD-A092 940 LEE (J S) ASSOCIATES INC ARLINGTON VA F/G 17/2
FM CORRELATOR SPECTRAL DATA TRANSFER BY SCRAMBLED TRANSMISSION --ETC(U)
OCT 80 J S LEE, S TSAI, L E MILLER N00014-80-C-0129
UNCLASSIFIED JTR-80-03 NL

1 OF 3
AD
A092940



JTR-80-03

FM CORRELATOR SPECTRAL DATA TRANSFER BY SCRAMBLED TRANSMISSION SYSTEM

PREPARED FOR
THE OFFICE OF NAVAL RESEARCH
STATISTICS AND PROBABILITY PROGRAM
ARLINGTON, VIRGINIA 22217

FINAL REPORT
N00014-80-C-0129
(NR 042-435)

OCTOBER 31, 1980

J.S. LEE ASSOCIATES INC.
2001 Jefferson Davis Highway, Suite 1100
Arlington, Virginia 22202

APPROVED FOR PUBLIC RELEASE: DISTRIBUTION UNLIMITED

FM CORRELATOR SPECTRAL DATA TRANSFER BY SCRAMBLED TRANSMISSION SYSTEM

October 1980

JTR-80-03

Contract Number

N00014-80-C-0129

Prepared for:

The Office of Naval Research
Statistics and Probability Program
Arlington, Virginia 22217

Prepared by:

J. S. Lee Associates, Inc.
2001 Jefferson Davis Highway, Suite 1100
Arlington, Virginia 22202

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO. AD-A092940 (10) / 3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) (6) FM Correlator Spectral Data Transfer By Scrambled Transmission System,	5. TYPE OF REPORT & PERIOD COVERED (9) Final rept. Jan[illegible] – Oct[illegible] 1980,
	6. PERFORMING ORG. REPORT NUMBER (14) JTR-80-031
7. AUTHOR(s) (10) Jhong S. Lee, Stephen Tsai [illegible] Leonard E. Miller	8. CONTRACT OR GRANT NUMBER(s) (15) N00014-80-C-0129
9. PERFORMING ORGANIZATION NAME AND ADDRESS J. S. Lee Associates, Inc. 2001 Jefferson Davis Highway, Suite 1100 Arlington, Virginia 22202	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS NR 042-435
11. CONTROLLING OFFICE NAME AND ADDRESS The Office of Naval Research Statistics and Probability Program Arlington, Virginia 22217 (11)	12. REPORT DATE 31 Oct[illegible] 1980 / 13. NUMBER OF PAGES 183 + x
14. MONITORING AGENCY NAME & ADDRESS(If different from Controlling Office) (12) 194	15. SECURITY CLASS. (of this report) UNCLASSIFIED / 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

FM Correlator, Spectral Line Detector, Spectral Data Transfer System, Data Scrambler, Self-Synchronization, Pseudorandom Sequence, Gold Code.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report presents design consideration for a Spectral Data Transfer System. FM correlators are proposed as cost-effective spectral detectors. The spectral data are scrambled and transmitted via satellite channel to a remote center for processing. A unique self-synchronization scheme is introduced for the scrambler/descrambler system and is explained in detail in the report.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED 393892

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

TABLE OF CONTENTS

TABLE OF CONTENTS
(CONT'D)

TABLE OF CONTENTS
(CONT'D)

List of Figures

List of Figures
(Cont'd)

List of Figures
(Cont'd)

List of Figures
(Cont'd)

PAGE

LIST OF TABLES

0.0 INTRODUCTION AND SUMMARY

J. S. LEE ASSOCIATES, INC. (LAI) has shown in a previous study [1], under contract N00014-77-C-0056, that an FM correlator is a viable spectral line detector that can be utilized for detection and classification of undersea targets. The FM correlator was proposed as a cost-effective alternative to the conventional scheme which computes discrete Fourier transforms (DFT) based on the observation of a time series over a finite time interval. The methods presently employed in spectral estimation for the purpose of detecting the targets' spectral band are based on DFT. In some applications, these methods are not practical from the point of view of complexity and cost. For example, the design of expendable sensor array is usually constrained by a limited cost, and hence it is desirable to employ a scheme which is inherently simple and relatively inexpensive.

The study [1] has demonstrated numerically the performance measures that are quite acceptable when the FM correlator is designed under suitable design parameter choices. It was observed that the probabilities of detection can be made to approach close to unity for arbitrary false alarm rates when the design parameters are choser in a carefully coordinated manner.

Having established the concept validation of utilizing the FM correlator for spectral band detection, LAI has then undertaken the study of designing a total system for effectively transferring the spectral data to a remote processing facility via satellite relay channel.

The object of this report is to present design considerations for the Spectral Data Transfer System as shown in an overall block diagram given in Figure 0.1. Each subsystem has been analyzed in detail, with results and recommendation upon which system designs can be based.

- Each FM correlator, employed as spectral line detector makes a measurement in every 10-15 seconds. Based on the resolution requirement, these measurements are quantized into approximately 10 bits.
- In view of the low data rate from the spectral detector, data from a number of the detectors will be multiplexed and transmitted in a short burst, typically at 1.2 kbps or 2.4 kbps.
- Scrambled transmission is mandated in military application. With respect to the scrambled transmission, a self-synchronizing scheme is described. Self-synchronization means that synchronization can be recovered without depending on a separate channel transmitting the sync information. This scheme, believed to be new, is explained in detail for those without background in the theory of finite fields.
- Error correcting codes are briefly reviewed. It is concluded that either short block codes with simple threshold decoding or convolutional codes with short constraint length using Viterbi decoding is adequate for the Spectral Data Transfer System.

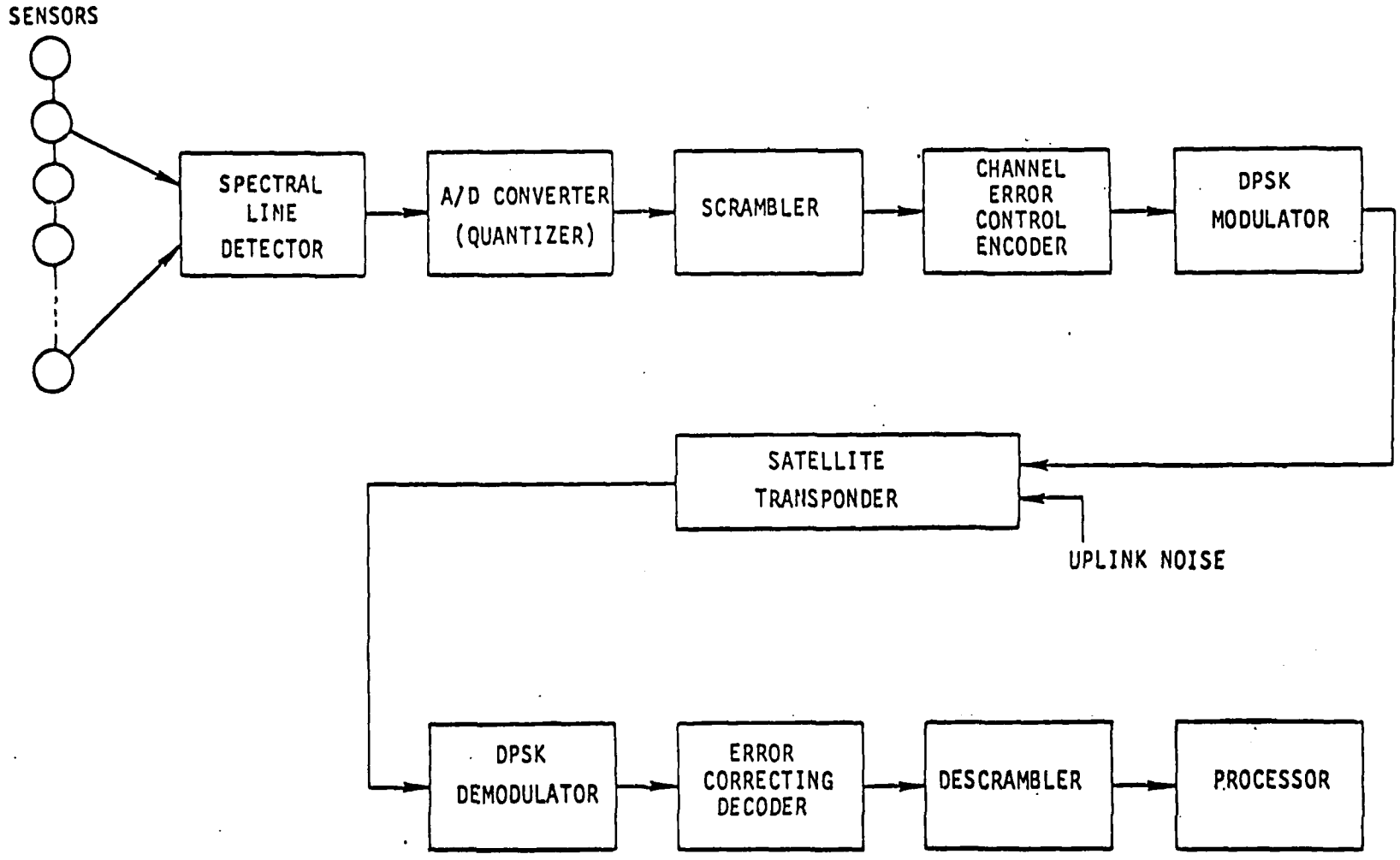


Figure 0.1 SPECTRAL DATA TRANSFER SYSTEM

- Platform instability which severely restricts the antenna gain within 0-3 dB, and the prime power limitation dictate the choice of satellites operating at low carrier frequency, in order to minimize the free-space loss. Link calculations with performance comparison have been analyzed.

In this report it is shown that a Spectral Data Transfer System utilizing FM correlators as line detectors, can be designed which will provide a reliable data transmission with dependable scrambler synchronization scheme. The approach we have taken in connection with data scrambler and self synchronization scheme merits complete understanding of their methodology, for their engineering design approach can be varied in implementation. For these reasons we have provided numerous examples as we discussed the design philosophies.

1.0 SPECTRAL LINE DETECTOR

1.1 The Concept of Using an FM Correlator for Line Detection

Developing the capability to detect and track signal energy contained in selected narrow spectral windows continues to be a significant challenge to the undersea-warfare community. Potential targets can be characterized acoustically by spectra featuring one or more narrowband emissions of uncertain or random bandwidth, whose center frequencies are subject to doppler shifting as the targets move. Therefore, detection of such emissions and estimation of their frequencies as they vary in time (tracking) allow both target identification and localization. In many ways the dynamic behavior of these emissions resembles frequency modulation. In this report the design of systems to exploit this resemblance is presented.

A previous study [1] evaluated the performance of an FM correlator in detecting the presence of spectral lines. The name "FM correlator" refers to the signal processing configuration diagrammed in Figure 1.1, in which the outputs of two sensors are first passed through bandpass filters (BPF), then demodulated as if they were frequency-modulated (FM) signals, then correlated. The correlation operation is performed by multiplying the FM detector (FMD) outputs and then integrating the product by means of a lowpass filter (LPF). In the analysis of this configuration the bandpass filter outputs were modelled by the frequency-modulated waveforms

$$s_i(t) + n_i(t) = A_i \cos[2\pi f_0 t + \phi_{m_i}(t)] + n_i(t), \quad i = 1, 2 \tag{1-1}$$

in which the amplitudes A_i were assumed to be constant and the noise terms $n_i(t)$ assumed to be from independent stationary, zero-mean, narrowband Gaussian random processes. The angle functions $\phi_{m_i}(t)$ were assumed to be

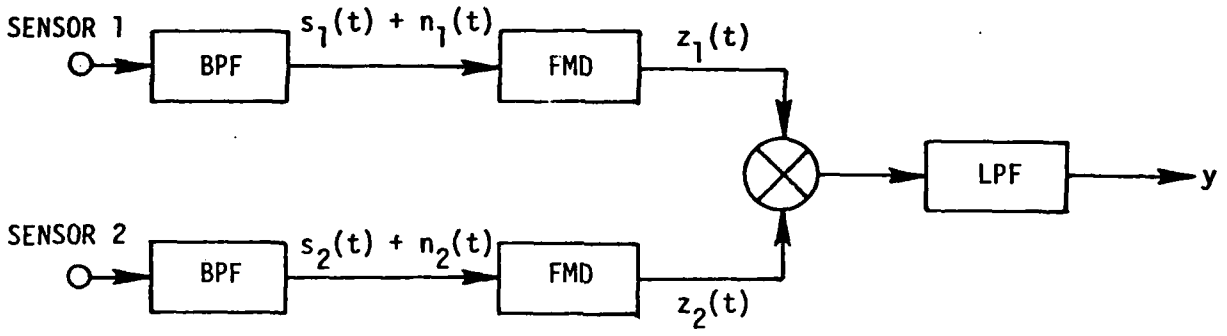


Figure 1.1 FM CORRELATOR CONFIGURATION

given by

$$\phi_{m_i}(t) = \int^{t} d\xi[m_i(\xi) + D(\xi)], \; i = 1, 2 \qquad (1\text{-}2a)$$

where

$$m_2(t) = m_i(t-\Delta t) = m(t-\Delta t) \qquad (1\text{-}2b)$$

and the modulation m(t) is from a zero-mean, stationary, Gaussian random process. D(t) represents a slowly varying doppler frequency shift.

The previous study indicated that successful detection based on the output of the FM correlator can be accomplished for certain values of the system parameters. In the following pages, further details are given concerning the system models and the analytical results which apply to them. Then, starting with Section 1.2, we begin to specify how the FM correlator may be used to measure line frequency remotely for transmission to a central data processing site.

1.1.1 FM Detector Model

FM detectors (FMD), or frequency-to-amplitude conversion circuits, are generally nonlinear but so designed as to provide a characteristic that is nearly linear for an interval of frequencies around some center frequency f_0, as illustrated in Figure 1.2. Many FMD circuits exist; an example [2] of an FMD suitable for integrated circuit implementation is shown in Figure 1.3. This circuit features less than 1.5% departure from linearity for frequencies within one per cent of the center frequency.

Over the linear region of the characteristic we may express the FMD outputs by the general expression

$$z_i(t) - z_0 = k_i[\omega_i(t) - \omega_0]. \qquad (1\text{-}3a)$$

Here we assume the voltage offset $z_0 = 0$ and take the instantaneous angular frequencies to be derivatives of the phases shown in equations

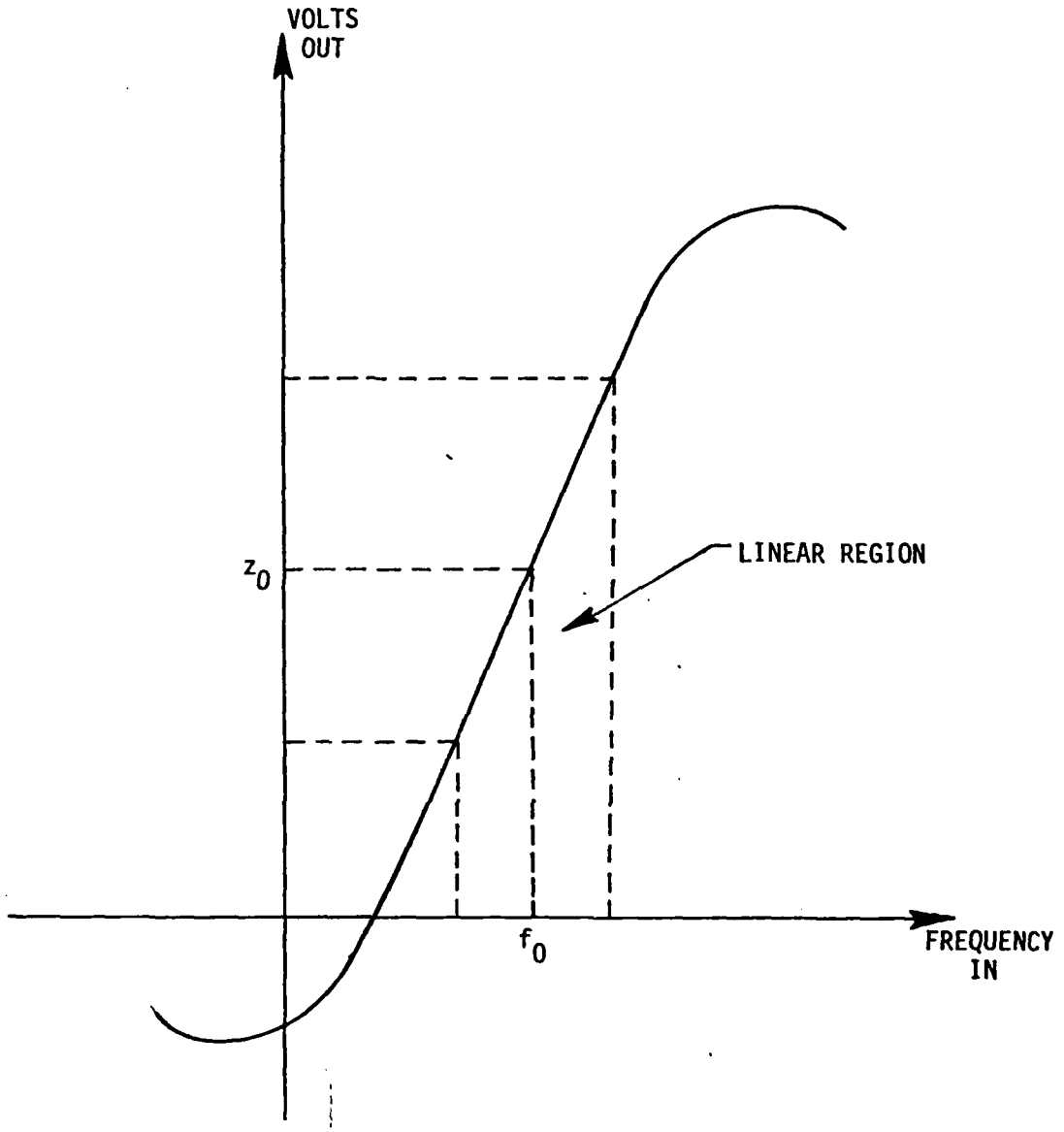


Figure 1-2 FM DETECTOR CHARACTERISTIC

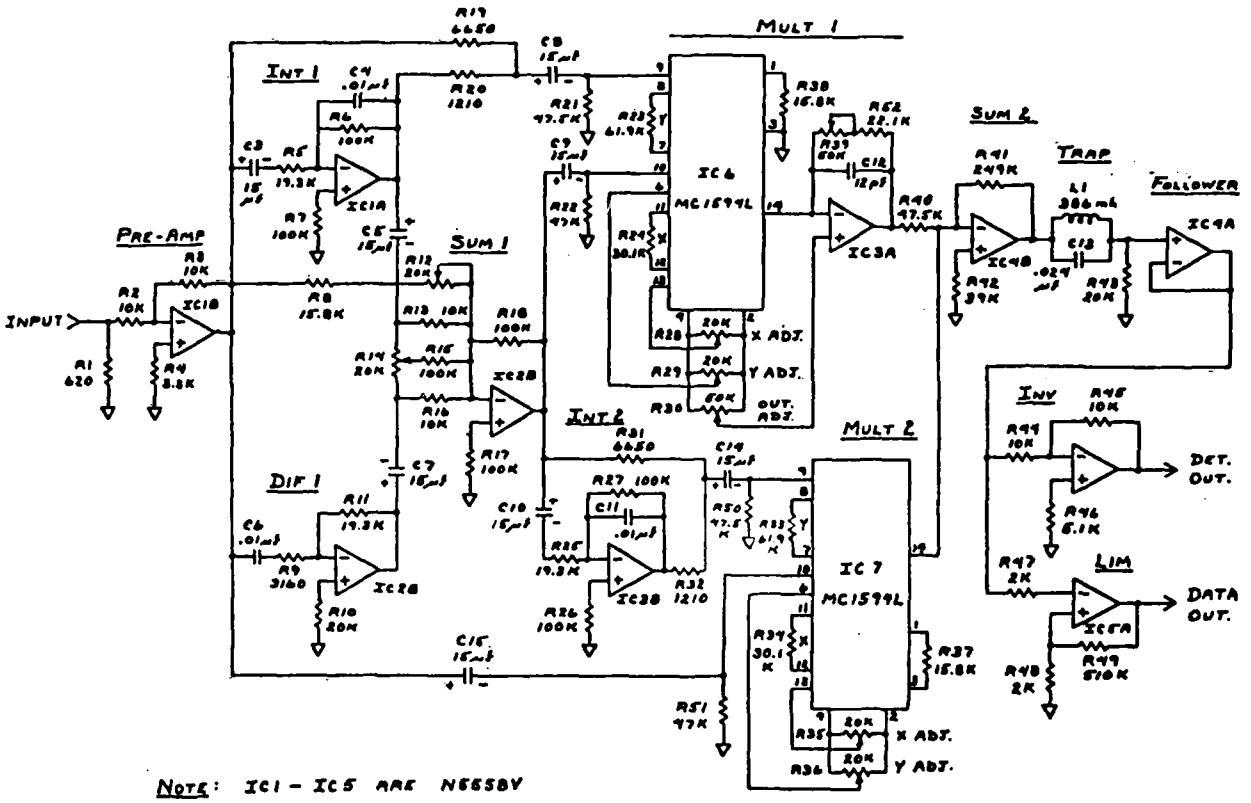


Figure 1.3 SCHEMATIC OF INTEGRATED CIRCUIT FM DETECTOR

(1-1) and (1-2); that is,

$$\omega_i(t) = \omega_0 + \frac{d}{dt}\phi_{m_i}(t). \tag{1-3b}$$

Then

$$z_i(t) = K_i[D + m_i(t)]. \tag{1-4}$$

In this expression we take D = constant (over some interval of time) since it is slowly varying.

1.1.2 Signal and Noise Spectra

The noise components of the received waveforms are assumed to have the autocorrelation functions

$$R_{n_i}(\tau) = N_i \exp\left\{-(\pi W_N \tau)^2\right\} \cos\omega_0\tau \tag{1-5}$$

which correspond to the Gaussian-shaped frequency spectrum

$$S_{n_i}(f) = \frac{N_i}{W_N\sqrt{\pi}} \exp\left\{-(f-f_0)^2/W_N^2\right\}, \quad f > 0. \tag{1-6}$$

These spectra are chosen to represent the class of narrowband spectra, and it is assumed that $W_N << f_0$. The parameter W_N is the one-sided 4.34 dB bandwidth; that is,

$$S_{n_i}(f \pm W_N) = N_i/e = N_i - 4.34 \text{ dB}. \tag{1-7}$$

The same spectral shape is chosen to represent the lowpass random modulation m(t). Its autocorrelation function is assumed to be

$$R_m(\tau) = P_m \exp\left\{-(\pi W_m \tau)^2\right\}. \tag{1-8}$$

1.1.3 The Correlator Output

The previous analysis showed that at the end of a T-second observation interval, the output of the FM correlator shown in Figure 1.1 can be written

$$y(T) = \mu(T) + n(T) \tag{1-9}$$

where $\mu(T)$ is a mean value, given by

$$\mu(T)\Big|_{D=0} = R_m(\Delta t)\,(1-e^{-CNR_1})(1-e^{-CNR_2})\int_0^T dt\; h(t)\cdot K_1K_2 \qquad (1\text{-}10)$$

where h(t) is the impulse response function of the lowpass filter and the CNR_i are the carrier-to-noise power ratios at the FMD inputs. For high CNR_i then,

$$\mu(T)\Big|_{\substack{D=0\\ \Delta t=0}} = \text{const}\cdot P_m\ ; \qquad (1\text{-}11)$$

that is, the mean value of the correlator output is proportional to the frequency modulation power. For nonzero doppler D constant over the T-second interval, we may extend the result (1-10) to obtain

$$\mu(T)\Big|_{\Delta t=0} = \text{const}\cdot(P_m + D^2). \qquad (1\text{-}12)$$

1.2 Using the Correlator Output to Measure Frequency

The form of the FM correlator output,

$$y(T) = \text{const}\,(P_m + D^2) + \text{noise}, \qquad (1\text{-}13)$$

suggests that it may be used to measure the doppler component D when the modulation power P_m is small. The output depends on D^2, however, and at most the absolute value $|D|$ of the doppler can be observed.

A simple modification to the correlator is useful. Consider Figure 1.4. If an appropriately scaled "offset" is added to the FMD outputs, then the correlator output mean value becomes

$$\mu(T) = \text{const.}\,[P_m + (D+\Omega)^2] \approx k(D+\Omega)^2 \qquad (1\text{-}14)$$

for P_m small. This modification allows observation of D including its sign, as illustrated in Figure 1.5.

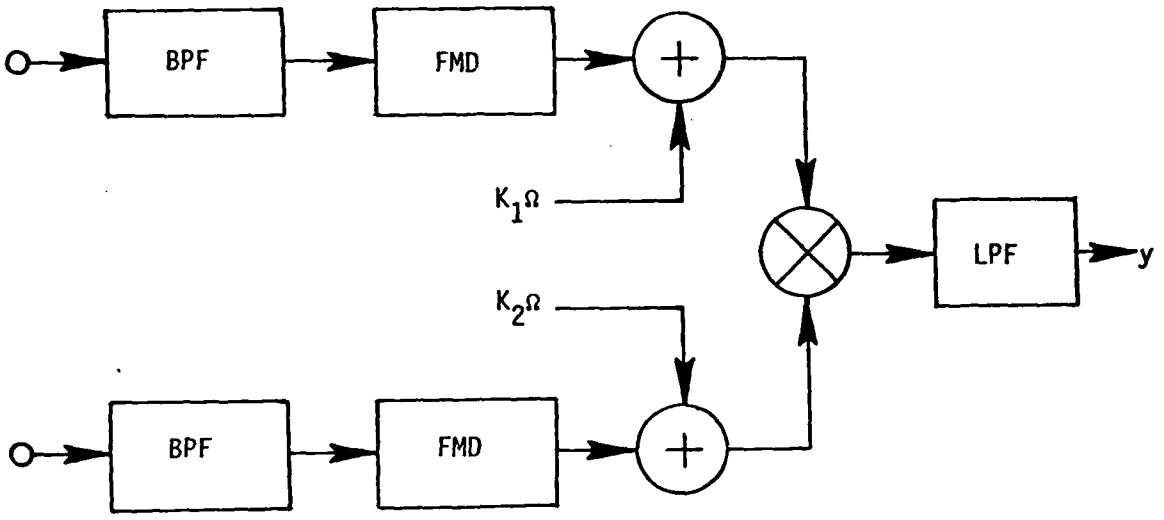


Figure 1.4 FM CORRELATOR WITH OFFSETS INTRODUCED AT FMD OUTPUTS

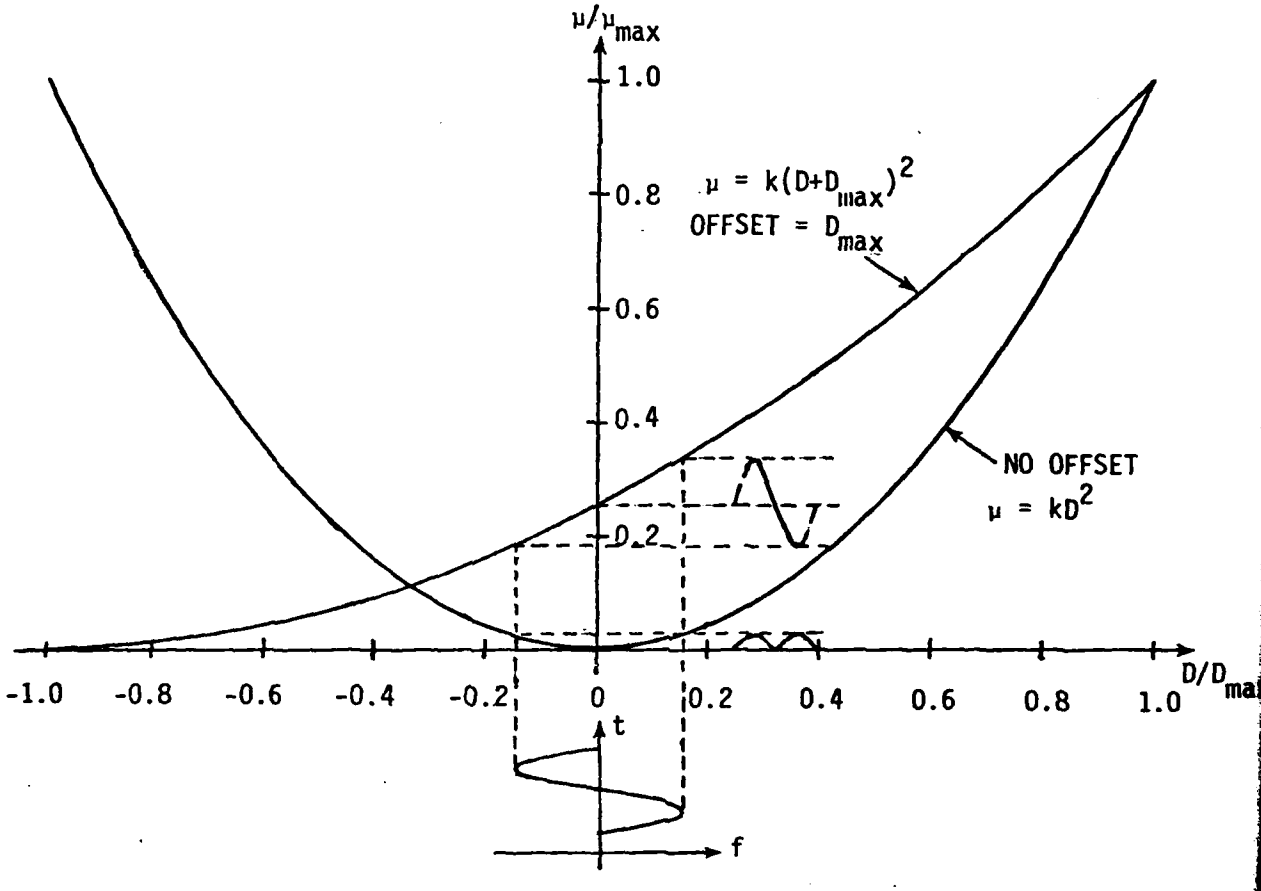


Figure 1.5 TRANSFER CHARACTERISTICS FOR FM CORRELATOR

1.3 Specifying the Frequency Measurement System

The object of this report is to present design considerations for a system to utilize the FM correlator, assumed to be located remotely with the sensors, to acquire measurements or estimates of target line frequency. The overall diagram of such a system is given in Figure 1.6. Treatment of the system components which follow the FM correlator begins in the next chapter. In this section, the basic parameters of the FM correlator are specified.

1.3.1 Required Passband

The bandwidth of a randomly modulated FM signal has been studied by Abramson [3], who defines a mean square signal bandwidth B_s^2 by

$$B_s^2 \equiv \frac{\int df\ (f-f_0)^2 S_s(f)}{\int df\ S_s(f)}\ , \tag{1-15}$$

where $S_s(f)$ is the spectrum of the signal. An alternate form of (1-15) is given by

$$B_s^2 = \frac{1}{(2\pi)^2} \cdot \frac{(-1)}{R_s(0)} \cdot \left.\frac{\partial^2 R_s(\tau)}{\partial \tau^2}\right|_{\tau=0} \tag{1-16}$$

in which $R_s(\tau)$ is the signal envelope autocorrelation function:

$$R_s(\tau) = \frac{A^2}{2} \exp\left\{-R_{\phi_m}(0) + R_{\phi_m}(\tau)\right\}. \tag{1-17}$$

From these expressions we find that the bandwidth of the line is

$$B_s = \frac{1}{2\pi}\sqrt{-\ddot{R}_{\phi_m}(0)} = \frac{1}{2\pi}\sqrt{R_m(0)} = \frac{\sqrt{P_m}}{2\pi}\ . \tag{1-18}$$

Because of doppler, however, there is uncertainty concerning where

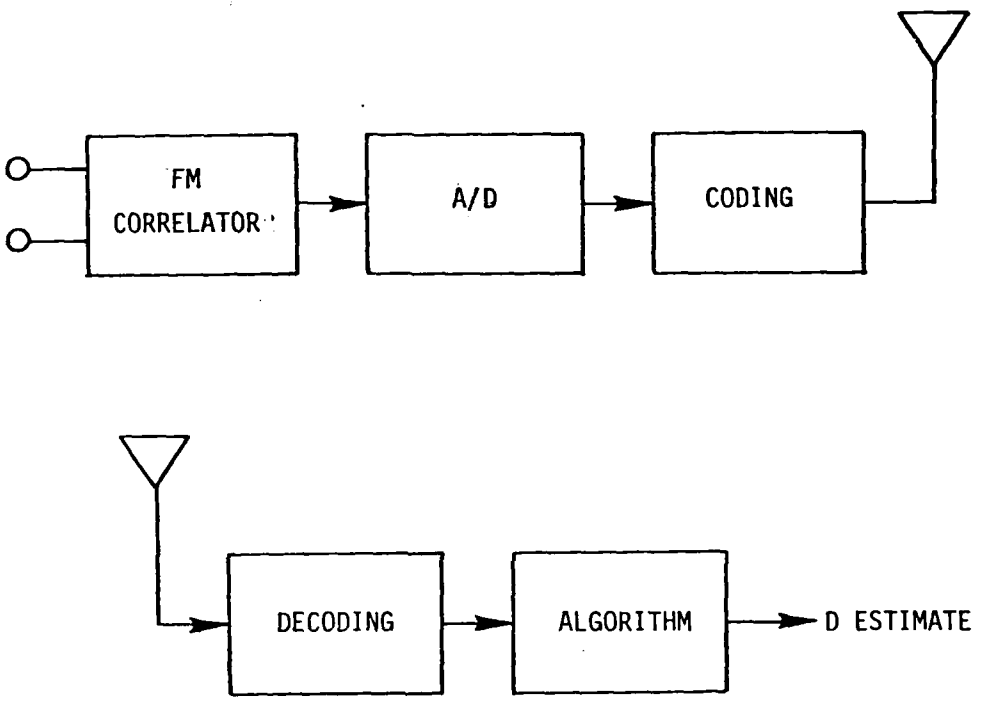


Figure 1.6 REMOTE FREQUENCY-ACQUISITION SYSTEM

in frequency this bandwidth is located. Figure 1.7 illustrates how the signal spectum might appear as a function of time. It is evident that (barring any frequency tracking procedure) the system input passband is determined by D_{max}, the maximum doppler shift, when the random frequency modulation power P_m is small.

If it is assumed that the relative velocity of the acoustic source is limited to within ±30 knots, then it is true that

$$|D| < \frac{2\pi f_0}{100} = D_{max}, \tag{1-19}$$

and the passband required at the inputs is

$$f_0 \pm 1\%, \text{ so that } W_N = f_0/100. \tag{1-20}$$

1.3.2 Smoothing Requirements

As discussed in later sections, the noise remaining in the correlator output is a factor in the design. The noise power is inversely proportional to the time-bandwidth product $W_N T$, where W_N is the (noise) bandwidth at the BPF outputs and T is the integration time.

The lowpass filter is chosen to perform an integrating or smoothing function. Ideally, its impulse response would be

$$h(t) = \begin{cases} \frac{1}{T}, & 0 < t < T \\ 0 \quad \text{elsewhere.} \end{cases} \tag{1-21}$$

A more practical LPF, however, would have a response that approximates this function. For example, a single-pole RC filter has the impulse response

$$h(t) = \begin{cases} e^{-t/RC}, & t > 0 \\ 0 \text{ elsewhere,} \end{cases} \tag{1-22a}$$

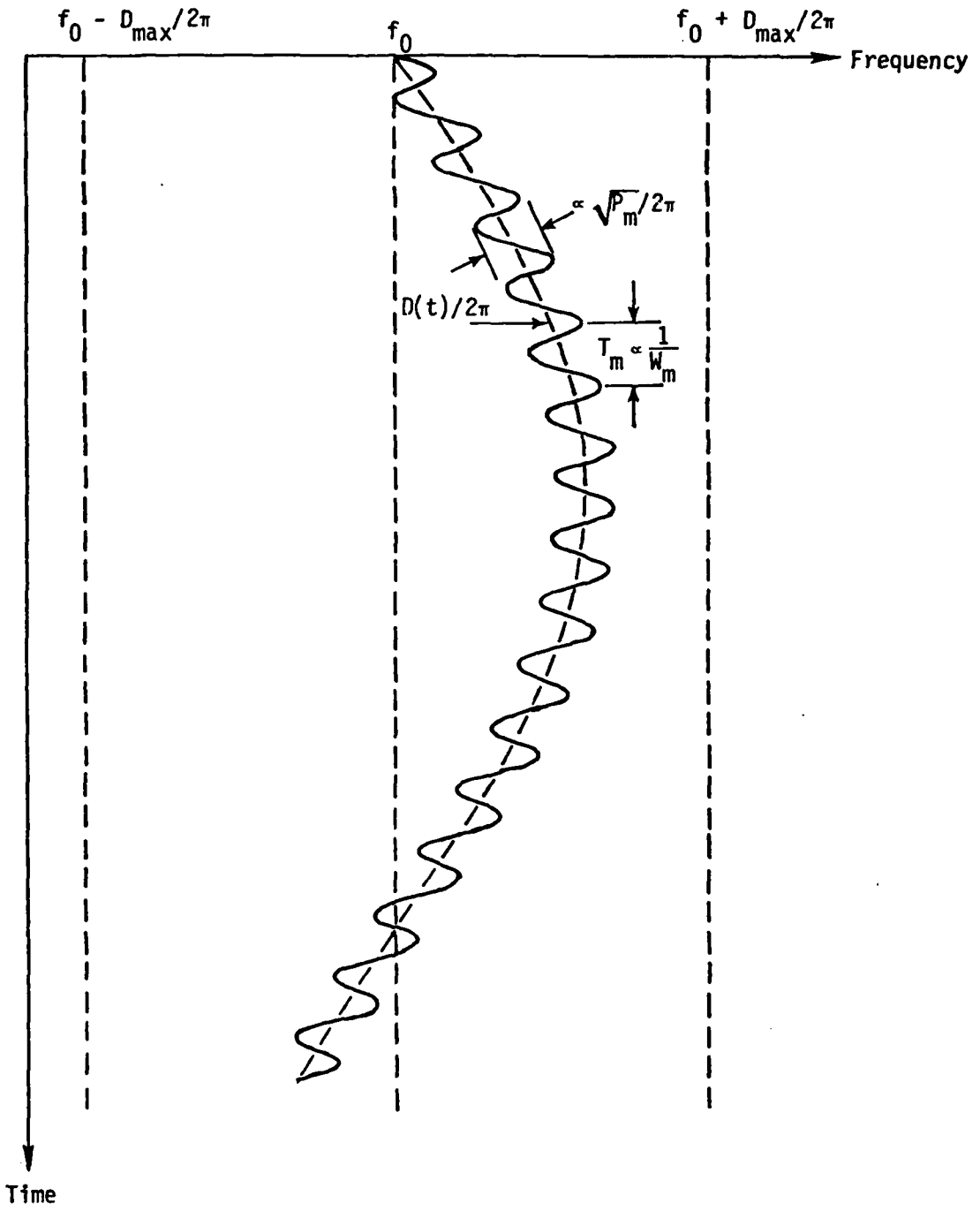


Figure 1.7 TIME-VARYING FREQUENCY

with bandwidth

$$B = \frac{1}{2\pi RC}. \qquad (1\text{-}22b)$$

The filter would be "dumped" or reset at the beginning of the observation interval and its output sampled T seconds later. It was shown in the previous study [1] that the filter bandwidth-time product BT has an optimal value of 0.4.

2.0 QUANTIZATION AND A/D PARAMETERS

2.1 Resolution Requirement

From external data it has been determined that a suitable step size or quantization level for f(t) is

$$q_f = \frac{f_0}{2000} \tag{2-1}$$

or $q_D = 2\pi q_f = D_{max}/20$, since $D_{max} = 2\pi f_0/100$.

What quantization level q_μ then is required on the correlator output to insure this maximum q_D? If we are constrained to use uniform quantization of the output, how many quantization levels are needed--and thus how many bits of A/D? These parameters are determined as follows.

Consider Figure 2.1. For the no-offset case, the maximum q_μ is found from

$$Q_\mu = \frac{q_\mu}{\mu_{max}} = 2\left(\frac{q_D}{2D_{max}}\right)^2 = 2\left(\frac{Q_D}{2}\right)^2 = \frac{2}{(40)^2} = \frac{1}{800} . \tag{2-2}$$

Because the slope of the characteristic is so small near D=0, the correlator output needs to be quantized very finely in order to observe the equivalent input doppler shift. Thus 800 levels are needed to represent 20 levels.

This effect is even more pronounced for the offset case. Near $D = -D_{max}$ we see that the maximum q_μ is found from

$$Q_\mu = \frac{q_\mu}{\mu_{max}} = \frac{1}{2}\left(\frac{q_D}{2D_{max}}\right)^2 = \frac{1}{2}\left(\frac{Q_D}{2}\right)^2 = \frac{1}{3200} . \tag{2-3}$$

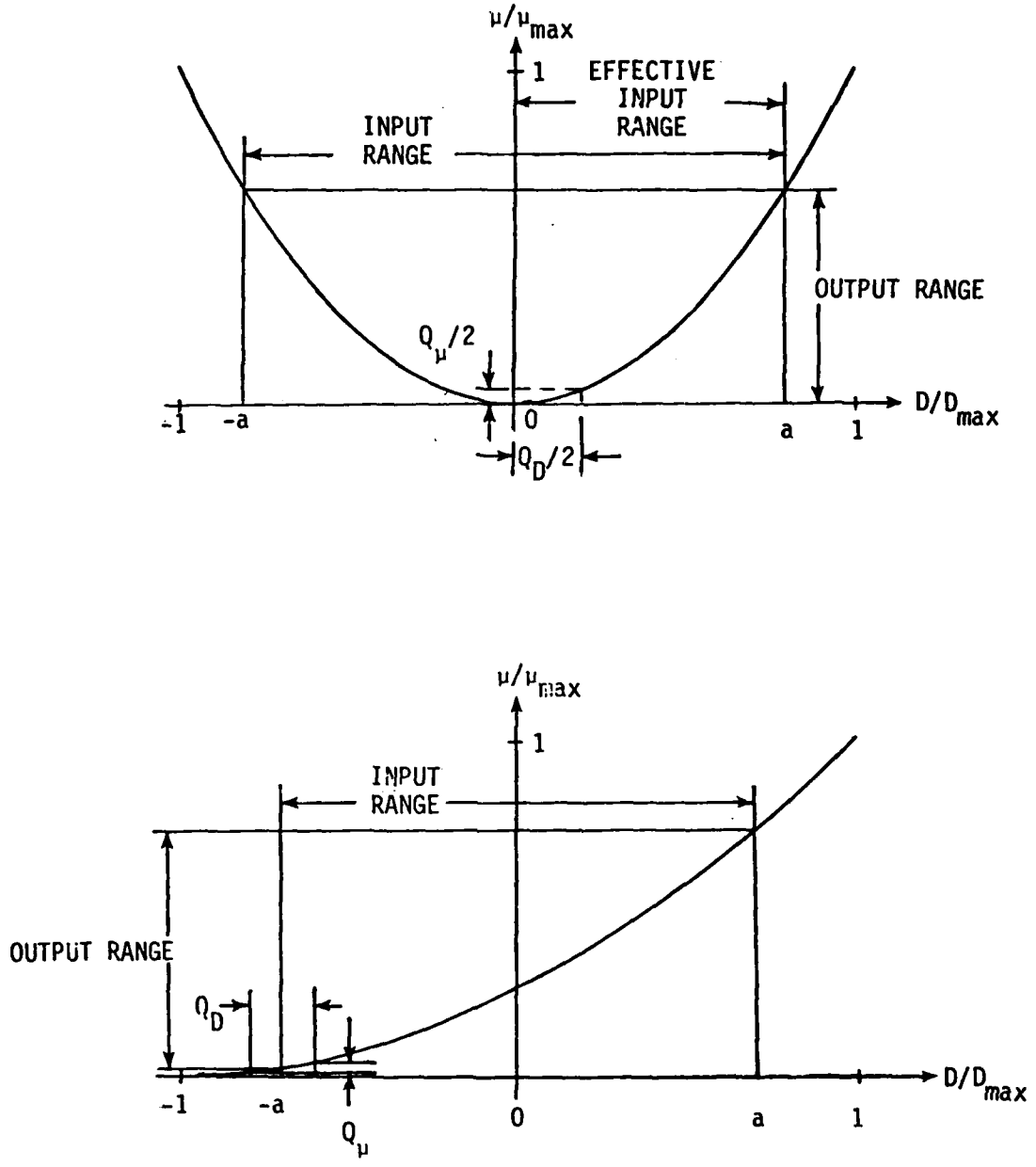


Figure 2.1 DETERMINATION OF REQUIRED QUANTIZATION LEVELS

Here, 3200 levels are needed to represent 20 levels!

The situation can be summarized in the following table:

Table 2.1 Summary of Quantization Parameters

input range	#input levels	#output levels	bit "cost"
$0<\|D/D_{max}\|< 1$ (no offset)	20 (5 bits)	800 (10 bits)	5 bits
$-1<D/D_{max}< 1$ (offset)	40 (6 bits)	3200 (12 bits)	6 bits

Because of the nonlinearity, 5 or 6 bits extra are required to preserve the measurement resolution on the frequency. A nonuniform quantizing scheme would be more efficient, but also more complex. Therefore, since the data rate is quite low (about one sample every 10 or 15 seconds), we are willing to pay the bit "cost" to retain quantization simplicity.

The fact that smaller values of doppler shift are much more likely than larger ones suggests a modified quantization strategy. Suppose that we do the following: restrict the uniform quantization of the correlator output to a smaller interval which corresponds to

$$0 \leq |D/aD_{max}| \leq 1, \quad a \leq 1 \tag{2-4}$$

then, for values outside that interval, simply allow the A/D to indicate "overflow", telling us that a very high doppler has been observed. What effect does this approach have?

Consider Figure 2.1 again. Using the definitions given in that figure, we can construct Table 2.2.

TABLE 2.2 Quantization Parameters for Restricted Input Range

a	input range	#input levels	Q_μ	output range	#output levels	A/D bits	bit cost	
1	1.0	20	1/800	1.0	800	10	5	
.9	.9	18	"	.81	648	10	5	
.8	.8	16	"	.64	512	9	5	
.7	.7	14	"	.49	392	9	5	NO OFFSET
.6	.6	12	"	.36	288	9	5	
.5	.5	10	"	.25	200	8	4	
1	2.0	40	1/3200	1.0	3200	12	6	
.9	1.8	36	1/400	.9	360	9	3	
.8	1.6	32	1/200	.8	160	8	3	OFFSET
.7	1.4	28	3/400	.7	94	7	2	$= D_{max}$
.6	1.2	24	1/100	.6	60	6	1	
.5	1.0	20	1/80	.5	40	6	1	

The meaning of this table is that, when the output interval over which A/D conversion is performed, is reduced,

(a) for no offset, the quantization level does not change, but fewer levels are required because the output range is smaller;

(b) for offset, the quantization level increases and the output range is smaller, giving a two-way reduction in the number of levels required.

Thus, for example, by deciding to declare "overflow" for D greater than 80% of its maximum value, we can reduce the number of A/D bits required from 10 or 12 to 9 or 8. Under this approach, the offset scheme looks very attractive because not only does it require one bit less (8 bits), but also allows us to recover the sign of the doppler shift.

2.2 Correlator Output Noise

Another factor in determining the size of the quantization levels to be used at the correlator output is the amount of noise variation in the output of the correlator. For sufficient integration time, however, the noise

"ripple" in the output can be kept very small compared to the quantization level. The amount of integration time required is included in the discussion in Appendix A.

2.3 Recommended Quantization Scheme

A recommended quantization scheme is given in detail in the Appendix A (Table A-2) and features 7-bit quantization of the reduced input range:

$$|D| \leq .75\ D_{max} \tag{2-5}$$

The 20 input cells then correspond to unequal groupings of output cells.

Implementation of the correlator (including offset) and subsequent quantization operations is diagrammed in Figure 2.2. In the Appendix it is shown that the noise in the correlator output will be sufficiently small to make a 128-level quantization reasonable if T is such that $W_N T$ products on the order of 100 are produced. This is not a stringent requirement.

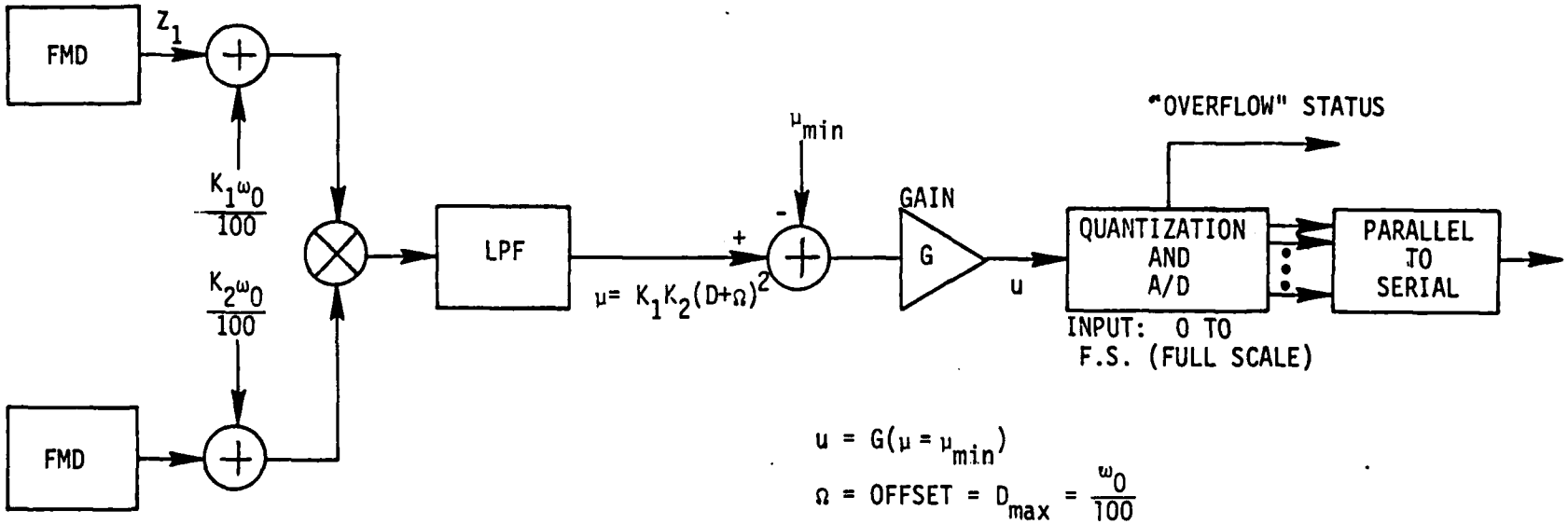
2.4 Detection Considerations

The output of the FM correlator can be utilized for line detection, suitably defined. If the output is tested as to whether it exceeds a threshold, this operation is equivalent to testing the hyptothesis (for no offset)

$$H_0: \text{ no frequency modulation present.} \tag{2-6}$$

Clearly, this test is not equivalent to what is normally meant by "detection", in which the hypothesis to be tested is "no signal present." That is, there is no equivalence unless it can be stated positively that the "signal" by definition contains frequency modulation. If we are willing to make that statement, then the analysis proceeds as follows.

Because of the integrating effect of the lowpass filter, we can treat the correlator output as a Gaussian random variable. Therefore, we have probabilities of false alarm and detection given by



$$u = G(\mu = \mu_{min})$$

$$\Omega = \text{OFFSET} = D_{max} = \frac{\omega_0}{100}$$

$$\mu_{min} = K_1K_2\left[(\Omega - aD_{max})^2 - \frac{q_\mu}{2}\right] = K_1K_2\left(\frac{\omega_0}{100}\right)^2\left[(1-a)^2 - (1-a)Q_D\right]$$

$$\mu_{max} = K_1K_2\left[(\Omega + aD_{max})^2 + \frac{q_\mu}{2}\right] = K_1K_2\left(\frac{\omega_0}{100}\right)^2\left[(1+a)^2 + (1-a)Q_D\right]$$

$$\text{GAIN} = \frac{\text{F.S.}}{\mu_{max}-\mu_{min}} = \frac{\text{F.S.}}{K_1K_2\left(\frac{\omega_0}{100}\right)^2\left[4a + 2(1-a)Q_D\right]}$$

F.S. = full scale specified for input to quantization and A/D device

Figure 2.2 IMPLEMENTATION OF QUANTIZER

$$P_{FA} = Q\left(\frac{\eta - m_0}{\sigma_0}\right) , \quad P_D = Q\left(\frac{\eta - m_1}{\sigma_1}\right) , \tag{2-7}$$

where η is the detection theshold, and (m_i, σ_i^2) are the means and variances under the hypotheses

$$H_0: \text{CNR} = 0 , \quad H_1: \text{CNR} > 0. \tag{2-8}$$

The variance σ_1^2 is given above as σ_y^2; $m_1 \equiv \mu$; $m_0 = 0$; and σ_0^2 is given by

$$\sigma_0^2 = K_1^2 K_2^2 D_{max}^2 \left\{\frac{1.35}{2\pi WT} - \frac{.299}{(2\pi WT)^2}\right\} . \tag{2-9}$$

By selecting η to guarantee a specified false alarm probability we may relate P_D and P_{FA} by the equation

$$x_D \sigma_1 + m_1 = x_{FA} \sigma_0 \tag{2-10}$$

$$\text{or} \quad x_D \left(\frac{K_{11}}{\alpha} + \frac{K_{12}}{\alpha^2}\right) + m = x_{FA}\left(\frac{K_{01}}{\alpha} + \frac{K_{02}}{\alpha^2}\right) , \quad \alpha \equiv 2\pi WT. \tag{2-11}$$

The notation x_p corresponds to inverse probability and for the Gaussian distribution we have the inverse distribution given in Table 2.3.

TABLE 2.3 Inverse Gaussian Distribution

P	x_p
.0001	3.71902
.001	3.09023
.01	2.32635
.5	0
.9	-1.28155
.99	-2.32635

This equation can be solved for α or WT, resulting in the values given in Table 2.4, in which $S \equiv D/D_{max}$ and $\gamma \equiv P_m/(2\pi W_N)^2$.

TABLE 2.4 Required WT Products for Detection Based on FMC Output

P_{FA} \ P_D	.5	.9	.99	.5	.9	.99	
.01	WT=4.40	4.42	4.44	4.90	4.67	4.89	γ = .0025
.001	5.85	5.87	5.88	5.85	6.12	6.34	δ = 1/3
.0001	7.04	7.06	7.07	7.04	7.31	7.53	
.01	39.99	40.01	40.04	39.99	40.29	40.54	γ = .0025
.001	53.12	53.14	53.17	53.12	53.42	53.67	δ = 1/10
.0001	63.93	63.95	63.97	63.93	64.23	64.48	
	CNR = 10 dB			CNR = 3 dB			

With $W=f_0/100$, we can see that an integration time T of 20 sec for low speed targets (δ=1/10) and of 2 sec for high speed targets (δ=1/3) is required for P_{FA}=.01.

Power Detection. If detection of the line is to take place in the usual way by square-law/lowpass filtering as shown in Figure 2.3, detection requires a certain value of the product $WT_D(CNR)$ [4] as given in Table 2.5.

TABLE 2.5 Required WT_D and CNR For Power Detection

P_{FA}	P_D	WT_D	CNR (dB)
.01	.5	1	2.1
		2.5	-.4
		5	-2.2
		10	-3.8
		25	-6.2
.01	.9	1	5.0
		2.5	2.4
		5	0.4
		10	-1.4
		25	-3.7
.001	.5	1	4.9
		2.5	2.2
		5	0.3
		10	-1.5
		25	-3.7
.001	.9	1	7.0
		2.5	4.2
		5	2.1
		10	0.3
		25	-2.3

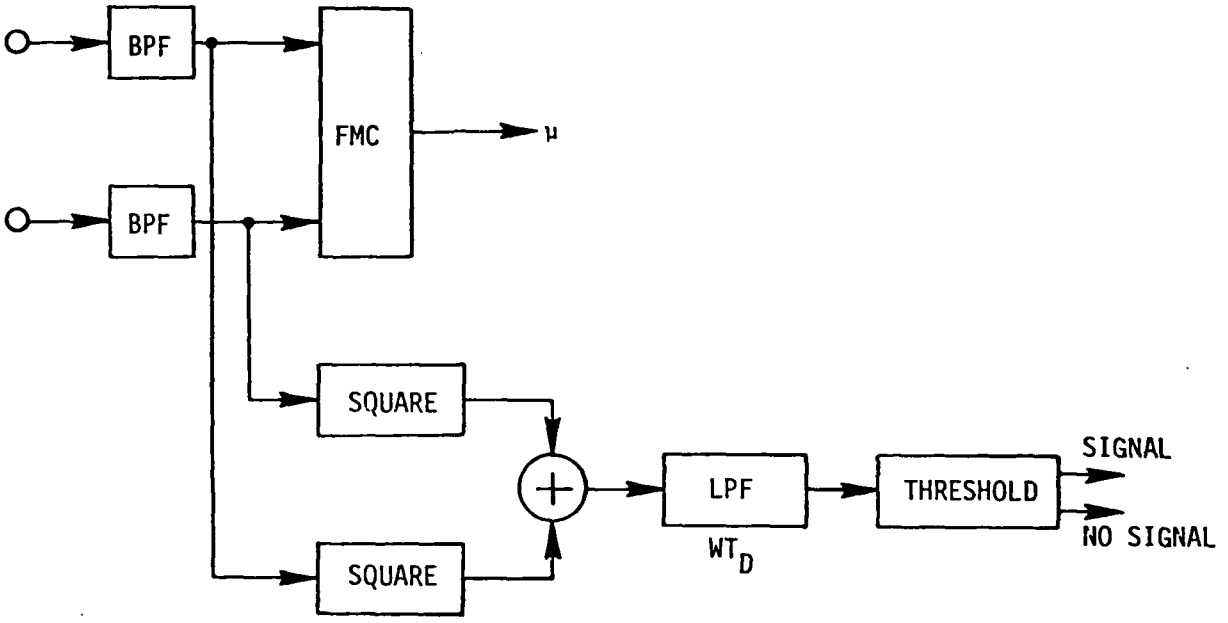


Figure 2.3 SQUARE-LAW DETECTION OF SPECTRAL LINE

Obviously, we can make the required CNR for detection as small as we please by manipulating the integration time T_D. This integration time is independent of that used in the correlator and is inversely proportional to the bandwidth of the lowpass filter.

3.0 SCRAMBLER AND SYNCHRONIZATION SCHEME

In this section we first present the proposed data scrambler system with a brief description of the main parts of the system. We then proceed to discuss each part in detail. In developing a digital data scrambler and descrambler system, careful consideration must be given to the problem of synchronization. A self-synchronization scheme, believed to be new, is thoroughly analyzed both in its principle of operation and its implementation.

3.1 Description of the Proposed System

3.1.1 Scrambler

A block diagram of the scrambler is shown in Figure 3.1. The scrambler will perform the following functions:

(1) Receive and store data from the line detectors

(2) Scramble the data

(3) Arrange the data in proper frame format

(4) Provide signals for sync recovery

K-2 bits of data are received from each of the N spectral detectors and are stored in K-stage buffers with two leading stages containing zeros. At the time of transmission the multiplexer switches out the stored data. The contents of each buffer are switched out continuously until the buffer is empty. The multiplexed data are scrambled by combining with a Gold sequence which is the bit-by-bit modulo-2 sum of two pseudorandom sequences, the X sequence and the Y sequence. The scrambled sequence is blocked into frames. When A=1 and B=0, the output is a frame sync bit; when A=0 and B=1, the output is an original bit from the X sequence; and at the remaining instance both A=0 and B=0, the output is the scrambled data bit. The original Y sequence forms the other part of the output.

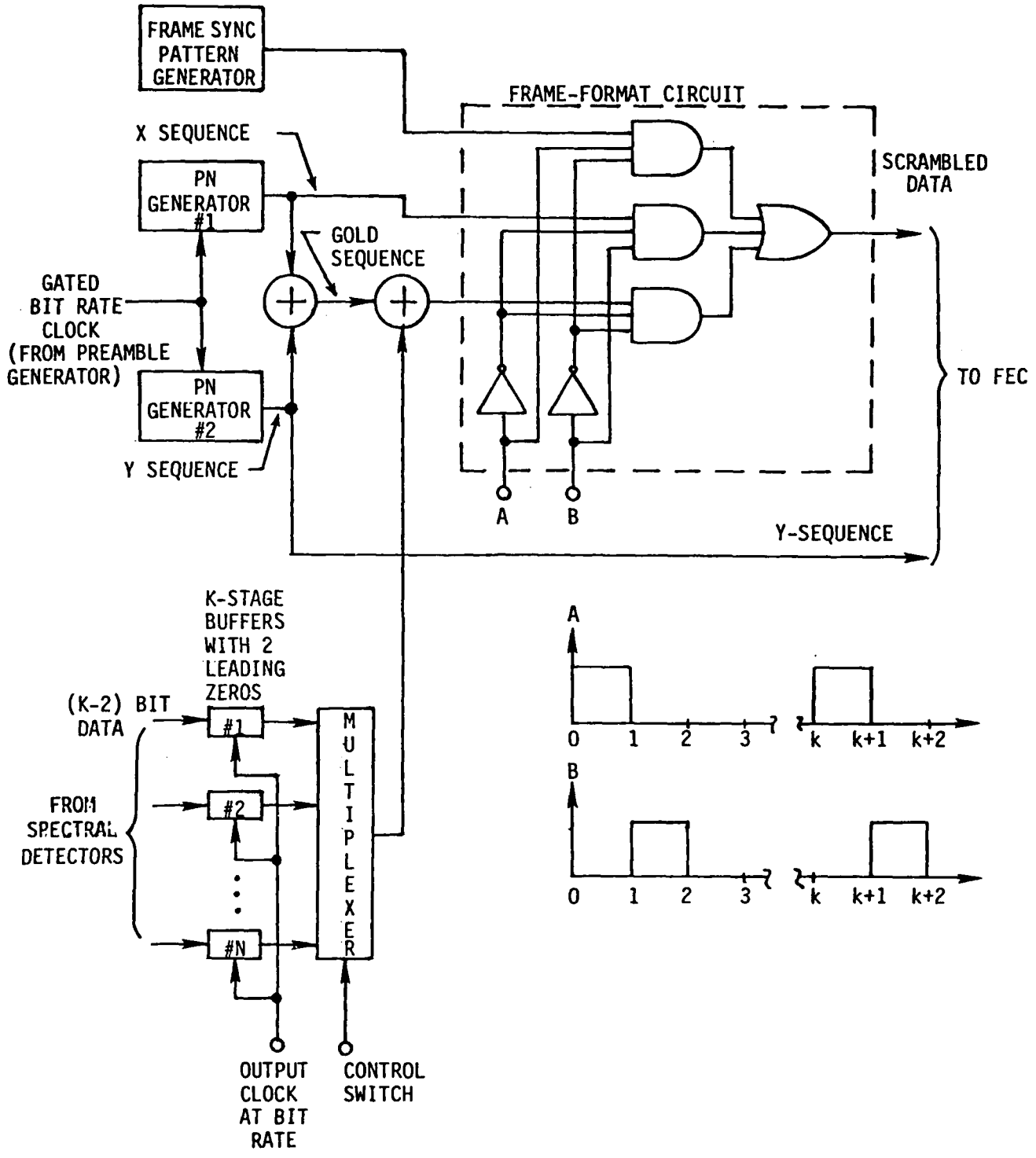


Figure 3.1 DATA SCRAMBLER

Both scrambled sequence and the Y sequence are sent to the FEC encoder.

The system is initialized by a preamble which consists of a burst of the carrier for carrier sync, a zero-one or all-one pattern for bit sync, and a Barker sequence or a Neuman-Hofman sequence for initialization of frame sync. The circuit will be described in Section 3.4.

3.1.2 Receiver and Data Descrambler

A block diagram of the receiver and the data descrambler is shown in Figure 3.2. At the end of the preamble, the receiver establishes the initial clock and the initial sync for the de-interleaver and the pseudo-random sequence generators. The de-interleaver separates the Y sequence and the scrambled data sequence and the frame sync circuits separate the frame sync bits and the sampled bits of the X sequence from the scrambled data sequence. Two pseudorandom sequence generators identical with the corresponding ones in the transmitter and with pre-determined initial states are activated to form the Gold sequence which combines with the scrambled data to recover the original spectral line data. The received Y sequence and the samples of the X sequence are used for the purpose of recovering the synchronization when sync loss is detected.

3.1.3 Gold Sequence and Sync Recovery

Figure 3.3 shows two local PN Generators which generate two pseudo-random sequences. Gold sequences are generated by bit-by-bit modulo-2 addition of the two pseudorandom sequences. Sync recovery for PN Generator #2 is accomplished by direct bit-by-bit comparison between the received Y sequence and the contents of the generator. Sync recovery for PN Generator #1 is accomplished by a self-synchronization circuit with the aid of the received samples of the X sequence. The self-synchronization circuit will be described and analyzed in detail in Section 3.3.

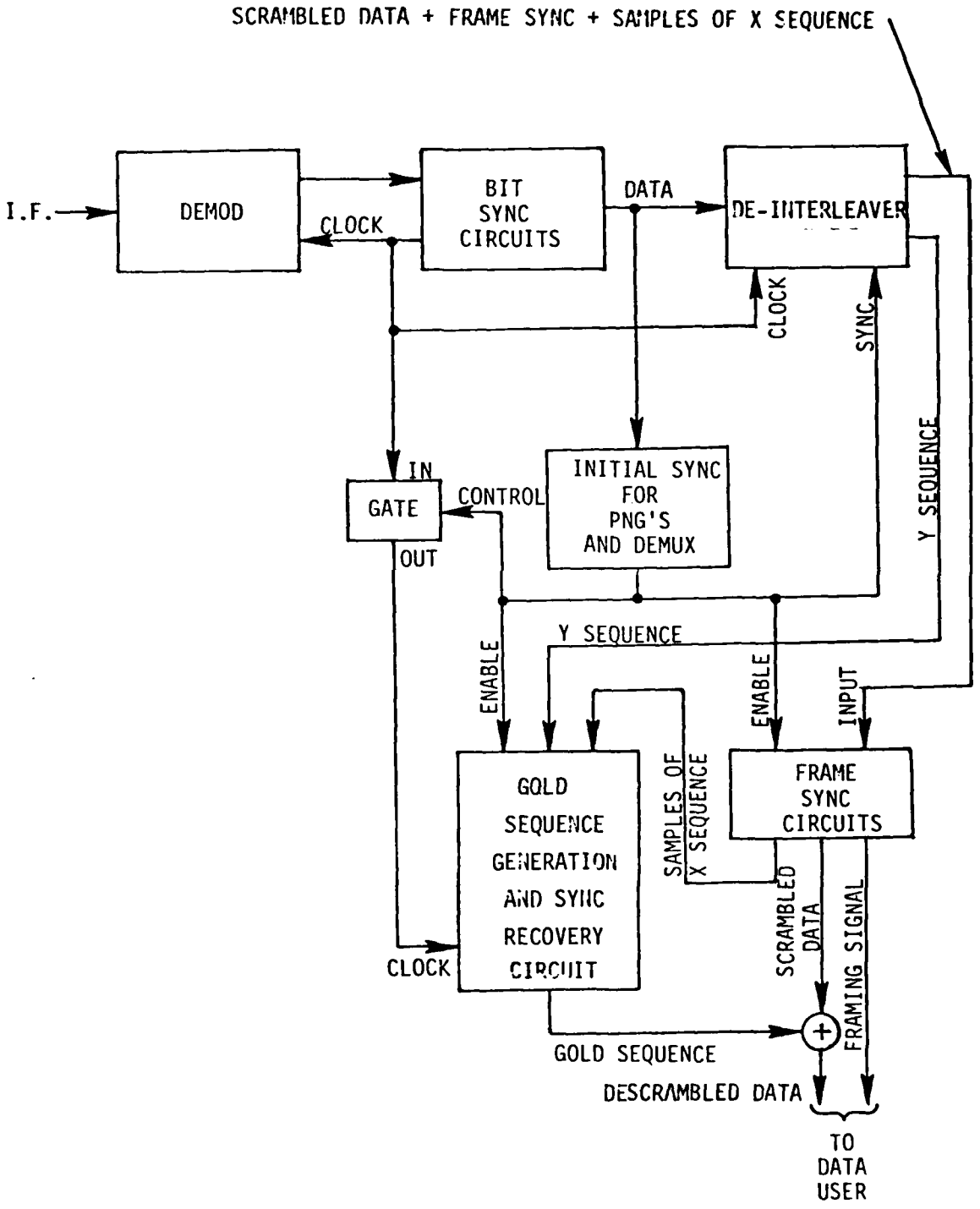


Figure 3.2 RECEIVER AND DATA DESCRAMBLER

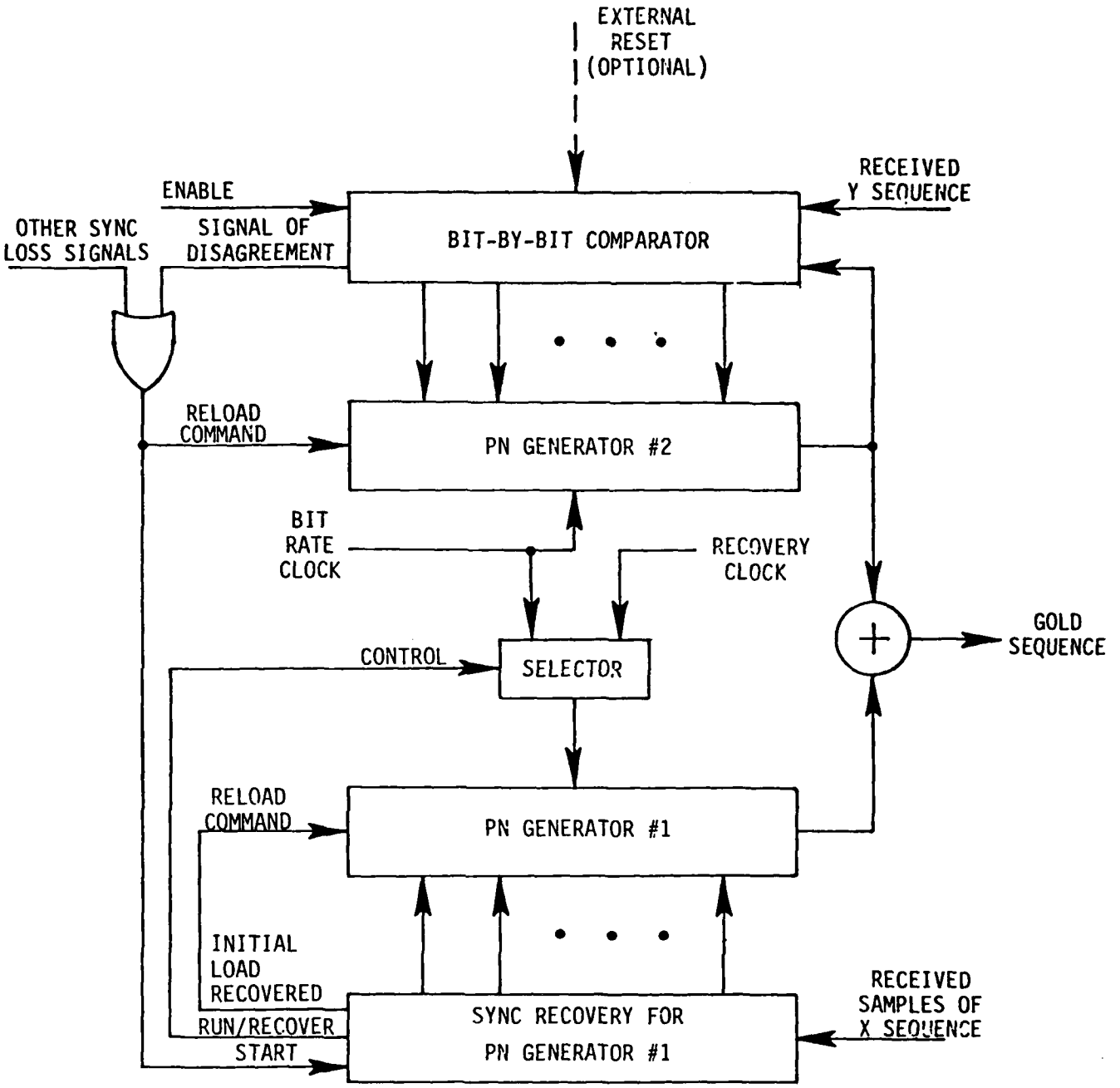


Figure 3.3 GOLD SEQUENCE GENERATION AND SYNC RECOVERY

3.2 Data Scrambler

3.2.1 Pseudorandom Sequences (A Tutorial Discussion)

In principle, data scrambling is achieved by changing the data sequence "randomly" before transmission. At the receiver the scrambled sequence is "changed back to the original data sequence." The problem lies in the two phrases: "randomness" and "changing back". If it is completely random, the receiver will have no way to change back. On the other hand if the receiver knows how to change back it cannot be completely random. Consider the following sequences.

```
      Data Sequence . . . 1 1 0 0 1 0 1 0 0 1 0 1 0 1 . . . .
  "Random" Sequence . . . 1 0 1 0 0 0 0 1 0 1 1 0 1 0 . . . .
Transmitted Sequence . . . 0 1 1 0 1 0 1 1 0 0 1 1 1 1 . . . .
```

The transmitted sequence, or the scrambled sequence, is the bit-by-bit modulo-2 sum of the data sequence and the "random" sequence. At the receiver, an identical "random" sequence is added to the transmitted sequence, yielding the original data sequence.

```
Transmitted Sequence . . . 0 1 1 0 1 0 1 1 0 0 1 1 1 1 . . . .
  "Random" Sequence . . . 1 0 1 0 0 0 0 1 0 1 1 0 1 0 . . . .
      Data Sequence . . . 1 1 0 0 1 0 1 0 0 1 0 1 0 1 . . . .
```

This simple illustration reveals two fundamental requirements on the "random" sequence: it must be reproducible at the receiver and it must be reproduced in synchronism with the sequence which scrambled the data sequence. These two fundamental requirements make it virtually impossible to use a completely "random" sequence. What is required, in practice, is a sequence that has sufficient "randomness" to be unrecognizable

to the unintended observers and yet is deterministic making it relatively easy to generate and synchronize in the receiver.

The most important method of generating binary sequences is by means of a shift register with feedback connections. An n-stage shift register is a device consisting of n consecutive binary storage elements. The contents, either "0" or "1", of each element can be shifted to the next position down on receipt of a regular clock pulse. The feedback will be a "1" or a "0" and may be computed as a logical function of the contents of the shift register stages. In a linear feedback shift register, which is the most useful and practical realization of a shift-register sequence generator, the feedback function is a modulo-2 sum of two or more of the shift register stages. An example of a 5-stage linear feedback shift-register sequence generator is shown in Figure 3.4. Five arbitrary binary digits, not all zeros, are loaded into the shift register as the initial conditions. At each clock time, the content of each shift register stage shifts to the next stage; the content of the last stage and the content of the second stage are modulo-2 added; and the sum is then fed into the first stage. At the same time the content of the last stage is taken out as the output sequence.

For a shift-register sequence generator of n stages, the output sequence will be always periodic because, whatever the initial condition, after a number of clock pulses the inital condition must eventually be reproduced. Since the maximum number of different combinations of n binary digits is 2^n, the period of the output sequence cannot exceed 2^n. In the linear feedback shift register, the state of all zeros in the shift register will produce a "0" as feedback; hence the state of the shift register will be all zeros forever. The "all-zeros" state is a degenerate

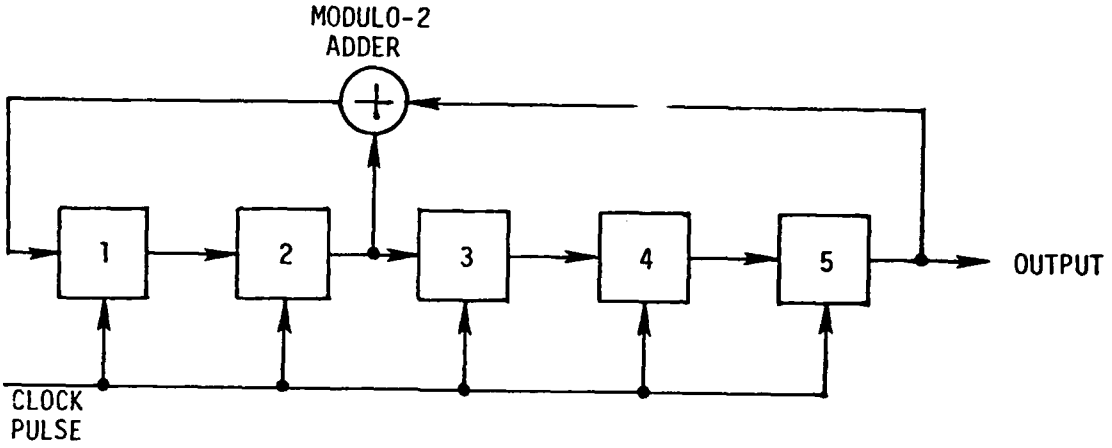


Figure 3.4 LINEAR FEEDBACK SHIFT REGISTER

state; it cannot appear in the shift register if the initial state is not "all zeros"; therefore the maximum number of possible states is 2^n-1. An output sequence whose period is 2^n-1 is called a "maximum length sequence" or "m-sequence" for short. The quantity $L=2^n-1$ is called the length of the sequence.

Maximum-length sequences are also referred as "pseudorandom sequences" or "pseudonoise sequences" or PN sequences for short. They possess the following "randomness" properties.

(1) The Balance Property: In a complete period of the sequence the number of "1" differs from the number of "0" by at most 1.

(2) The Run Property: There are $\frac{L+1}{2}$ runs of consecutive "1" and "0", a half of the runs are of length 1, $\frac{1}{4}$ of the runs are of length 2, $\frac{1}{8}$ of the runs are of length 3, etc.

(3) The Correlation Property: If a complete sequence is compared, bit by bit, with any shift of itself the number of agreements differs from the number of disagreements by at most 1.

For example, the sequence generated by the shift-register sequence generator in Figure 3.4 with 1 0 0 0 0 as the initial state will be:

0 0 0 0 1 0 1 0 1 1 1 0 1 1 0 0 0 1 1 1 1 1 0 0 1 1 0 1 0 0 1.

The length of the sequence is $2^5-1 = 31$. The balance property is satisfied as there are 16 "1" and 15 "0". The run property is satisfied as there are $\frac{L+1}{2} = 16$ runs (8 runs of "1" and 8 runs of "0"), there are 8 runs of length 1 (4 runs of "1" and 4 runs of "0"), there are 4 runs of length 2 (2 runs of "1" and 2 runs of "0"), there are 2 runs of length 3 (1 run of "1" and 1 run of "0"); and finally there is 1 run of "0" of length 4 and 1 run of "1" of length 5. Finally the correlation property is satisfied as follows:

```
0 0 0 0 1 0 1 0 1 1 1 0 1 1 0 0 0 1 1 1 1 1 0 0 1 1 0 1 0 0 1   complete sequence
1 0 0 0 0 1 0 1 0 1 1 1 0 1 1 0 0 0 1 1 1 1 1 0 0 1 1 0 1 0 0   shifted sequence

  a a a         a a   a   a a   a a a a   a   a     a          15 agreements
d     d d d d d   d d   d   d       d   d   d d d   d          16 disagreements.
```

The comparison between a complete sequence with a shifted sequence shows 15 agreements and 16 disagreements.

The feedback connections of a LFSR (abbreviation of Linear Feedback Shift Register) of n stages can be made into a one-to-one correspondence with a polynomial of degree n with either "0" or "1" as coefficients, where a coefficient of 1 represents a connection (including two ends) and 0 represents no connection. Consider the 5-stage sequence generator in Figure 3.4 and redraw it in Figure 3.5. The corresponding polynomial is

$$P(x) = 1x^5 + 0x^4 + 0x^3 + 1x^2 + 0x^1 + 1x^0$$

$$= x^5 + x^2 + 1$$

arranged in an ascending order from left to right. The polynomial which corresponds to the feedback connections of a maximum length sequence generator or PN generator is called the "primitive polynomial." Each primitive polynomial of degree n will generate a unique pseudorandom sequence of length $L = 2^n-1$.

The next questions are: how many primitive polynomials are there of degree n and how can we find the primitive polynomials?

The answer to the first question is: there are exactly $\phi(n)/n$ primitive polynomials of degree n, where $\phi(n)$ is the number of integers larger than 0, less than $L = 2^n-1$, and relatively prime to L (including 1).

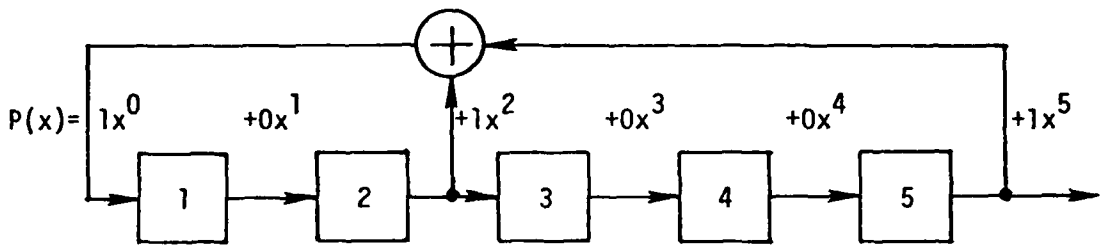


Figure 3.5 FEEDBACK CONNECTIONS AND POLYNOMIAL P(x)

For example, for n=4, $L=2^n-1 = 15$, the integers

$$1, 2, 4, 7, 8, 11, 13, 14$$

are relatively prime to 15; hence

$$\phi(n) = 8$$

and the number of different primitive polynomials is

$$\frac{\phi(n)}{n} = \frac{8}{4} = 2.$$

For n=5, since $L=2^5-1=31$ is a prime number itself, all integers less than 31 are prime to 31; therefore

$$\phi(n) = 30$$

and the number of different primitive polynomials is

$$\frac{\phi(n)}{n} = \frac{30}{5} = 6.$$

A list of the number of possible pseudorandom sequences for shift registers having up to 21 stages is given in Table 3.1.

To find the primitive polynomials of degree n turns out to be a very difficult problem. Fortunately, tables of irreducible polynomials has been published up to degree 34, among which the primitive polynomials are labeled. A well known source is in Peterson's book [5, Appendix C, pp. 472-492]. We shall point out the notations used in that reference in order to be able to make use of the table. Take an entry:

Degree 5 1 45E 3 75G 5 67H.

The numbers 45, 75, and 67 represent the polynomials in octal representation:

0 000	2 010	4 100	6 110
1 001	3 011	5 101	7 111.

Thus, 45 represents 1 0 0 1 0 1

TABLE 3.1 LIST OF THE NUMBER OF POSSIBLE PSEUDORANDOM SEQUENCES

NUMBER OF SHIFT REGISTER STAGES n	SEQUENCE LENGTH $L = 2^n-1$	TOTAL NUMBER OF SEQUENCES $\phi(n)/n$
2	3	1
3	7	2
4	15	2
5	31	6
6	63	6
7	127	18
8	255	16
9	511	48
10	1023	60
11	2047	176
12	4095	144
13	8191	630
14	16383	756
15	32767	1800
16	65535	2048
17	131071	7710
18	262143	7716
19	534287	27594
20	1048575	24000
21	2097151	84672

which in turns represents the polynomial

$$1x^5 + 0x^4 + 0x^3 + 1x^2 + 0x + 1$$

or

$$x^5 + x^2 + 1.$$

The number 75 represents 1 1 1 1 0 1

or the polynomial $x^5 + x^4 + x^3 + x^2 + 1$

and 67 represents 1 1 0 1 1 1

or the polynomial $x^5 + x^4 + x^2 + x + 1$.

The letter after the numbers gives the following information about the polynomial: E, F, G, H indicate that the polynomial is primitive, while A, B, C, D indicate not primitive.

The number before 45, 75, 67, that is 1, 3, 5, has the following meaning. If $\alpha=\alpha^1$ is a root of the polynomial represented by 45, then α^3 is a root of the polynomial 75 and α^5 is a root of polynomial 67; or

α is a root of $x^5 + x^2 + 1$.

α^3 is a root of $x^5 + x^4 + x^3 + x^2 + 1$,

and α^5 is a root of $x^5 + x^4 + x^2 + x + 1$.

For degree 5, three primitive polynomials are listed:

$$45 \leftrightarrow 1\ 0\ 0\ 1\ 0\ 1 \quad \leftrightarrow \quad x^5 + x^2 + 1$$

$$75 \leftrightarrow 1\ 1\ 1\ 1\ 0\ 1 \quad \leftrightarrow \quad x^5 + x^4 + x^3 + x^2 + 1$$

$$67 \leftrightarrow 1\ 1\ 0\ 1\ 1\ 1 \quad \leftrightarrow \quad x^5 + x^4 + x^2 + x + 1.$$

They will generate three different pseudorandom sequences of length L=31. Each polynomial corresponds to a PN generator. We observe that in all three generators the content of the last stage always feeds back to the first stage reflecting the fact that the coefficients of the highest degree term (the x^5 term) and the lowest degree term (the x^0 term or the constant term) always equal 1. In the first generator the content of the second stage also feeds back; in the second generator the contents of the

second, third, and fourth stages feedback; and the contents of first, second, and fourth stages feedback in the third generator. The first generator requires one tap while the other two generators require three taps. The generator corresponding to the polynomial $x^5 + x^2 + 1$ is called "the minimum tap pseudorandom-sequence generator." In the Table of the Irreducible Polynomials the primitive polynomial corresponding to the minimum-tap generator is always listed first.

According to Table 3.1 there are six distinct pseudorandom sequences, yet there are only three primitive polynomials in the Table of Irreducible Polynomials. The three missing polynomials can be obtained as follows. Let

$$P_1(x) = x^5 + x^2 + 1.$$

Replace x by x^{-1}

$$P_1(x^{-1}) = \frac{1}{x^5} + \frac{1}{x^2} + 1.$$

Multiplying $P_1(x^{-1})$ by x^5, we obtain

$$x^5 P_1(x^{-1}) = 1 + x^3 + x^5 = x^5 + x^3 + 1.$$

The new polynomial $x^5 + x^3 + 1$ is called "the reciprocal polynomial" of $P_1(x)$, and is represented by the notation $P_1^*(x)$. Thus, the reciprocal polynomial of a polynomial $P_1(x)$ is defined as

$$P_1^*(x) = x^n P_1(x^{-1}).$$

It can be shown that the reciprocal polynomial of a primitive polynomial is also primitive.

To find the reciprocal polynomials is, in practice, very easy by making the following observation:

$P_1(x) = x^5 + x^2 + 1 \quad \leftrightarrow \quad 100101 \quad - \quad 101001 \quad \leftrightarrow \quad P_1^*(x) = x^5 + x^3 + 1$

$P_2(x) = x^5 + x^4 + x^3 + x^2 + 1 \quad \leftrightarrow \quad 111101 \quad - \quad 101111 \quad \leftrightarrow \quad P_2^*(x) = x^5 + x^3 + x^2 + x + 1$

$P_3(x) = x^5 + x^4 + x^2 + x + 1 \quad \leftrightarrow \quad 110111 \quad - \quad 111011 \quad \leftrightarrow \quad P_3^*(x) = x^5 + x^4 + x^3 + x + 1.$

The procedure is: (1) write down the binary sequence representation of the given polynomial, (2) read the binary sequence backwards, and (3) change the sequence to the reciprocal polynomial.

We now obtain all 6 pseudorandom sequence generators of sequence length L=31:

$$P_1(x) = x^5 + x^2 + 1$$

$$P_2(x) = x^5 + x^4 + x^3 + x^2 + 1$$

$$P_3(x) = x^5 + x^4 + x^2 + x + 1$$

$$P_4(x) = x^5 + x^3 + 1$$

$$P_5(x) = x^5 + x^3 + x^2 + x + 1$$

$$P_6(x) = x^5 + x^4 + x^3 + x + 1.$$

The polynomials $P_1(x)$ and $P_4(x)$ are reciprocal to each other; so are $P_2(x)$ and $P_5(x)$, and $P_3(x)$ and $P_6(x)$.

Take an example. For n=11, we obtain from the Table of Irreducible Polynomials

(1) 1 4005E

4005 $\leftrightarrow$ 1 0 0 0 0 0 0 0 0 1 0 1

$P_1(x) = x^{11} + x^2 + 1.$

Read the sequence backwards

1 0 1 0 0 0 0 0 0 0 0 1 $\leftrightarrow$ 5001

and $P_1^*(x) = x^{11} + x^9 + 1.$

(2) 3 4445E

4445 $\leftrightarrow$ 1 0 0 1 0 0 1 0 0 1 0 1

$P_3(x) = x^{11} + x^8 + x^5 + x^2 + 1$

Read the sequence backwards

$$1\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 1 \leftrightarrow 5111$$

and $P_3^*(x) = x^{11} + x^9 + x^6 + x^3 + 1$.

(3) 13 4143F

$$4143 \leftrightarrow 1\ 0\ 0\ 0\ 0\ 1\ 1\ 0\ 0\ 0\ 1\ 1$$

$$P_{13}(x) = x^{11} + x^6 + x^5 + x + 1$$

Read the sequence backwards

$$1\ 1\ 0\ 0\ 0\ 1\ 1\ 0\ 0\ 0\ 0\ 1 \leftrightarrow 6141$$

and $P_{13}^*(x) = x^{11} + x^{10} + x^6 + x^5 + 1$.

The pseudorandom sequence generator can be implemented with flip-flops and logic gates. The implementation of the generator in Figure 3.4 is shown in Figure 3.6. The flip-flops used in this implementation are JK flip-flops with additional inputs of PRESET or direct SET (S_D) and CLEAR or direct RESET (R_D). These inputs, which function exactly as the S and R inputs of the SR flip-flops do, are used to establish the initial state of the shift register. After RESET clears the register, the LOAD line is activated and the initial states D_1, D_2, D_3, D_4, and D_5 can be entered into the register in parallel. Following the loading operation the data shift one position to the right with each clock pulse and at the same time the contents of the second and fifth stages feed back to the first stage through an EXCLUSIVE OR. The pseudorandom sequence is read out serially at the last stage.

We are now ready to observe how the maximum-length sequence is generated. We do this by considering a series of examples.

Example 1. $P(x) = x^3 + x + 1$

The shift register pseudorandom sequence generator is shown in Figure 3.7. Let F_{1t}, F_{2t}, and F_{3t} be the contents of the corresponding stages

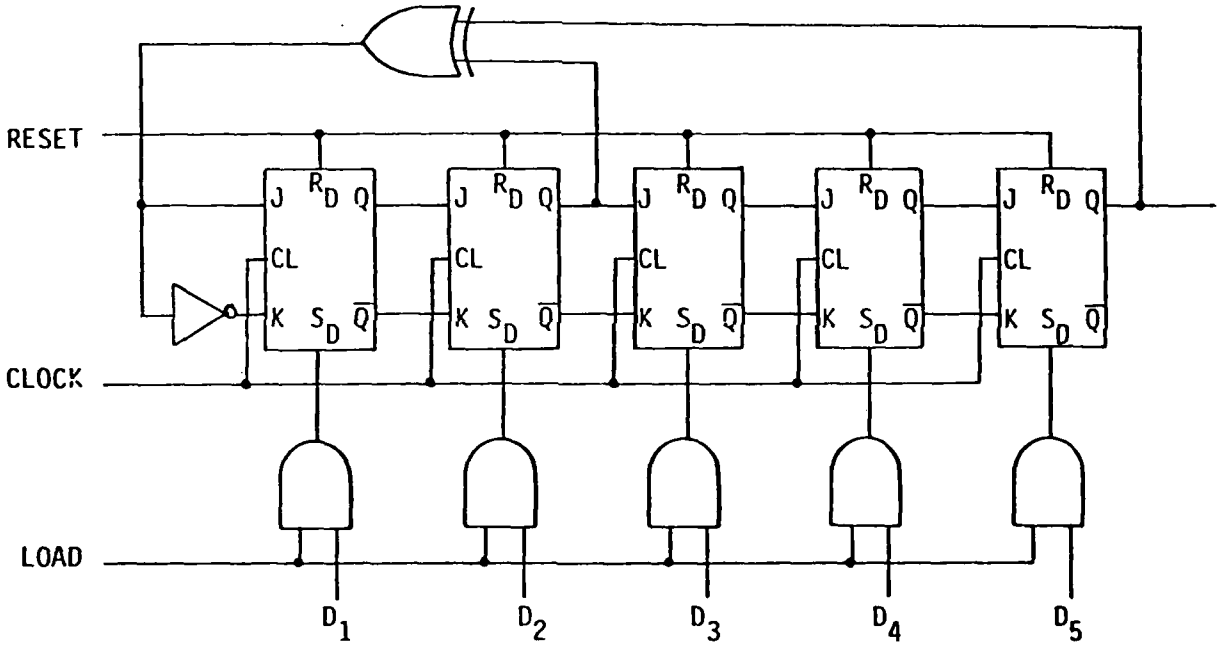
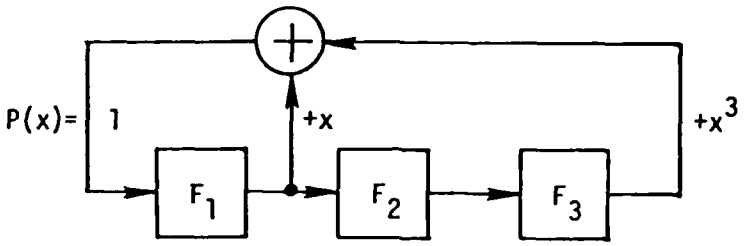


Figure 3.6 PSEUDORANDOM SEQUENCE GENERATOR

$P(x) = x^3 + x + 1$



			t
1	0	1	0
0	1	0	1
0	0	1	2
1	0	0	3
1	1	0	4
1	1	1	5
0	1	1	6
1	0	1	

Figure 3.7 3-STAGE PSEUDORANDOM SEQUENCE GENERATOR

at a particular instant, t; and $F_{1,t+1}$, $F_{2,t+1}$, and $F_{3,t+1}$ be the values at the next instant, t+1. The values $F_{1,t+1}$, $F_{2,t+1}$, and $F_{3,t+1}$, can be calculated from $F_{1,t}$, $F_{2,t}$, and $F_{3,t}$ by observation as follows:

$$F_{1,t+1} = F_{1,t} + F_{3,t}$$

$$F_{2,t+1} = F_{1,t}$$

$$F_{3,t+1} = F_{2,t}.$$

We can formulate this result as a matrix multiplication

$$\begin{bmatrix} F_1 \\ F_2 \\ F_3 \end{bmatrix}_{t+1} = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} F_1 \\ F_2 \\ F_3 \end{bmatrix}_t = T \begin{bmatrix} F_1 \\ F_2 \\ F_3 \end{bmatrix}_t .$$

Let

$$\begin{bmatrix} F_1 \\ F_2 \\ F_3 \end{bmatrix}_{t=0} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} F_1 \\ F_2 \\ F_3 \end{bmatrix}_{t=1} = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} F_1 \\ F_2 \\ F_3 \end{bmatrix}_{t=2} = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

$$\vdots$$

or,

t =	0	1	2	3	4	5	6	7 = 0
F_1	1	0	0	1	1	1	0	1
F_2	0	1	0	0	1	1	1	0
F_3	1	0	1	0	0	1	1	1.

The pseudorandom sequence generated is

1 0 1 0 0 1 1.

Example 2

$$P(x) = x^4 + x + 1$$

The PN generator is shown in Figure 3.8.

$$T = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

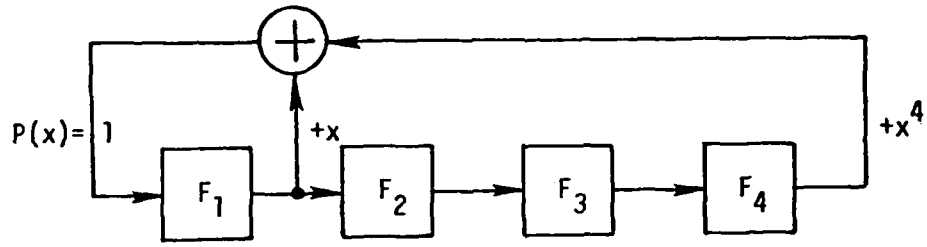
If

$$\begin{bmatrix} F_1 \\ F_2 \\ F_3 \\ F_4 \end{bmatrix}_{t=0} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \end{bmatrix}$$

then the values of F_1, F_2, F_3, and F_4 at t = 1, 2,...,14 can be calculated by matrix multiplication. The results are as follows:

Example 2

$P(x) = x^4 + x + 1$



				t
1	1	0	1	0
0	1	1	0	1
0	0	1	1	2
1	0	0	1	3
0	1	0	0	4
0	0	1	0	5
0	0	0	1	6
1	0	0	0	7
1	1	0	0	8
1	1	1	0	9
1	1	1	1	10
0	1	1	1	11
1	0	1	1	12
0	1	0	1	13
1	0	1	0	14
1	1	0	1	

Figure 3.8 4-STAGE PSEUDORANDOM SEQUENCE GENERATOR

t =	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
F_1	1	0	0	1	0	0	0	1	1	1	1	0	1	0	1
F_2	1	1	0	0	1	0	0	0	1	1	1	1	0	1	0
F_3	0	1	1	0	0	1	0	0	0	1	1	1	1	0	1
F_4	1	0	1	1	0	0	1	0	0	0	1	1	1	1	0.

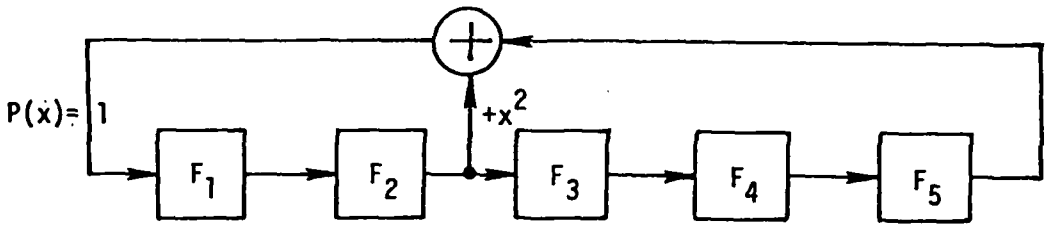
Example 3

$P(x) = x^5 + x^2 + 1$

The 5-stage PN generator is shown in Figure 3.9 with the shift register contents and the output pseudorandom sequence written directly under the generator. The matrix multiplication method which provides a clear understanding of the operation of a PN sequence generator as in Examples 1 and 2, becomes cumbersome as the number of stages increases.

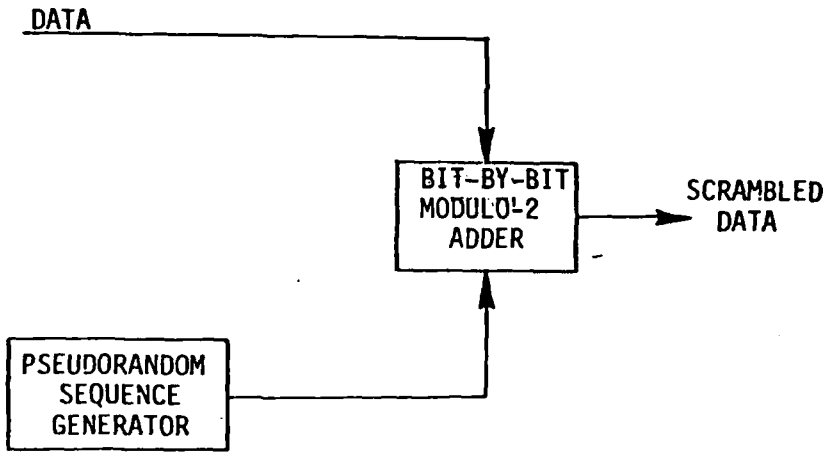
We are now ready to describe the principle of a data scrambling system. When a data sequence is added onto a PN sequence, the resultant sequence becomes unintelligible to an intercepter. But the original data sequence can be recovered by adding an identical PN Sequence. A conceptual block diagram of a data scrambling and descrambling system using a pseudorandom sequence is shown in Figure 3.10. The data sequence is scrambled by a pseudorandom sequence with a bit-by-bit modulo-2 adder. At the receiving end a synchronized identical pseudorandom sequence is bit-by-bit modulo-2 added onto the scrambled sequence to recover the original data sequence. The data scrambling system described thus far and a spread-spectrum system are similar. The only difference is that in the scrambling system the data sequence and pseudorandom sequence are of the same rate while in the spread-spectrum system the pseudorandom sequence is much faster than the data sequence.

Example 3

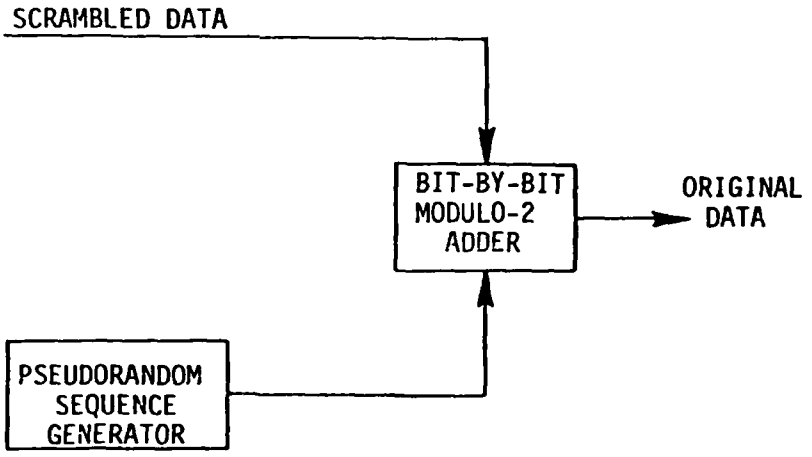


					t
1	0	0	0	0	0
0	1	0	0	0	1
1	0	1	0	0	2
0	1	0	1	0	3
1	0	1	0	1	4
1	1	0	1	0	5
1	1	1	0	1	6
0	1	1	1	0	7
1	0	1	1	1	8
1	1	0	1	1	9
0	1	1	0	1	10
0	0	1	1	0	11
0	0	0	1	1	12
1	0	0	0	1	13
1	1	0	0	0	14
1	1	1	0	0	15
1	1	1	1	0	16
1	1	1	1	1	17
0	1	1	1	1	18
0	0	1	1	1	19
1	0	0	1	1	20
1	1	0	0	1	21
0	1	1	0	0	22
1	0	1	1	0	23
0	1	0	1	1	24
0	0	1	0	1	25
1	0	0	1	0	26
0	1	0	0	1	27
0	0	1	0	0	28
0	0	0	1	0	29
0	0	0	0	1	30

Figure 3.9 5-STAGE PSEUDORANDOM SEQUENCE GENERATOR



SCRAMBLER



DESCRAMBLER

Figure 3.10 SCRAMBLING SYSTEM

3.2.2 Gold Codes

Pseudorandom sequences have extensive applications in spread-spectrum communication systems. In the case of a spread-spectrum system the pseudorandom sequence serves as an encoding mechanism which, when added to the data sequence, results in a wideband signal as shown in Figure 3.11. If the communication system is a multiple access system each pseudorandom sequence is used as a code for a particular user.

The usefulness of the pseudorandom sequences in a spread-spectrum system depends in large part on their ideal autocorrelation properties. One of the randomness properties of the pseudorandom sequence is the correlation property, i.e. if a complete sequence is compared, bit by bit, with any shift of itself the number of agreements differs from the number of disagreements by at most 1. From this property it is easy to see that the autocorrelation function of a pseudorandom sequence is of the shape described in Figure 3.12.

The cross-correlation function between two different pseudorandom sequences of the same length is, however, an entirely different matter. It can have high peaks; and to make the matter worse, there is no method available to calculate the cross-correlation function between two pseudo-random sequences except by simulation. For long sequences this is not possible even with the fastest computers. To visualize the cross-correlation problem and how much is involved in the simulation, consider an example as follows.

The sequence generator $x^3 + x + 1$ with initial state 1 0 0 generates

$$S_1 = 0\ 0\ 1\ 1\ 1\ 0\ 1$$

and the sequence generator $x^3 + x^2 + 1$ with initial state 1 1 1 generates

$$S_2 = 1\ 1\ 1\ 0\ 0\ 1\ 0.$$

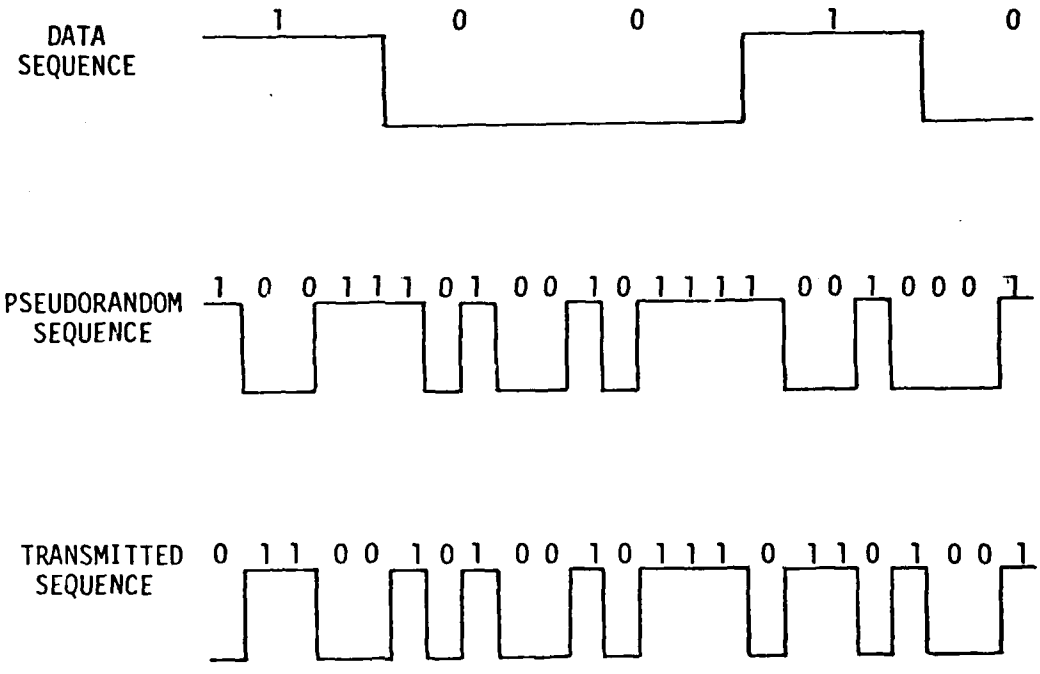


Figure 3.11 SPREAD-SPECTRUM WAVEFORM

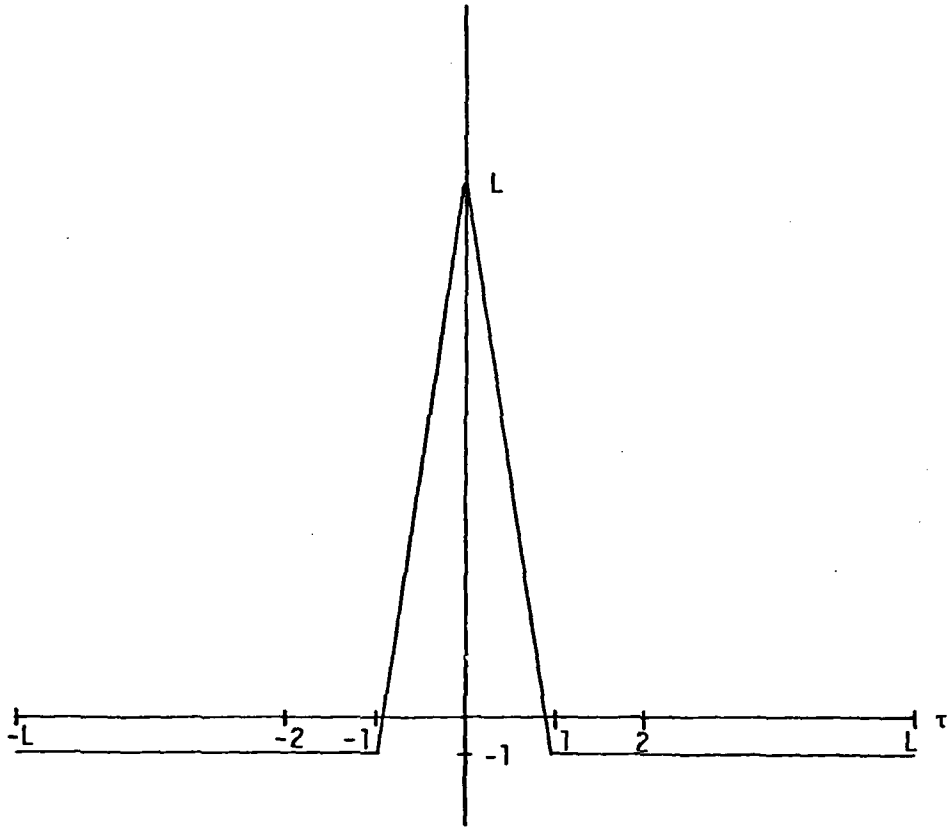


Figure 3.12 AUTOCORRELATION FUNCTION OF A PSEUDORANDOM SEQUENCE

The cross-correlation is computed by holding S_1 fixed, shifting S_2 one bit at a time, and comparing the two sequences to obtain the difference between the number of agreements and the number of disagreements.

1. 0 0 1 1 1 0 1
   +1 1 1 0 0 1 0
   1 1 0 1 1 1 1 → -5

2. 0 0 1 1 1 0 1
   +0 1 1 1 0 0 1
   0 1 0 0 1 0 0 → +3

3. 0 0 1 1 1 0 1
   +1 0 1 1 1 0 0
   1 0 0 0 0 0 1 → +3

4. 0 0 1 1 1 0 1
   +0 1 0 1 1 1 0
   0 1 1 0 0 1 1 → -1

5. 0 0 1 1 1 0 1
   +0 0 1 0 1 1 1
   0 0 0 1 0 1 0 → +3

5. 0 0 1 1 1 0 1
   +1 0 0 1 0 1 1
   1 0 1 0 1 1 0 → -1

7. 0 0 1 1 1 0 1
   +1 1 0 0 1 0 1
   1 1 1 1 0 0 0 → -1

The cross-correlation function is plotted in Figure 3.13.

In Section 3.2.1, we have found all six PN generators of 5 stages. The cross-correlations between sequences generated by $P_1(x)$ and $P_4(x)$, and by $P_4(x)$ and $P_6(x)$ are shown in Figure 3.14.

The cross-correlation function between two distinct pseudorandom sequences is a very important requirement in a multiple access communications system as a user's receiver might lock onto a wrong signal if the cross-correlation peak between the user's pseudorandom sequence and the sequence for the wrong signal is high enough to exceed the threshold for synchronization.

To overcome the cross-correlation problem, Gold considered the bit-by-bit modulo-2 sum of two pseudorandom sequences of same length but generated by two distinct primitive polynomials, $p_1(x)$ and $p_2(x)$, as shown in Figure 3.15. If the length of the two pseudorandom sequences is 2^n-1, then the resultant sequence also repeats itself after 2^n-1 bits. Furthermore if one sequence is kept fixed and the second sequence is shifted in time, a different resultant sequence is generated. 2^n-1 different sequences can be generated this way, one for each different time shift of the second sequence. Joining the two original pseudorandom sequences, altogether 2^n+1 different sequences can be generated with one pair of primitive polynomials. These sequences are sometimes referred to as Gold sequences or Gold codes; they are not maximal except the two original pseudorandom sequences. It should be noted that for n-stage shift register there are only $\phi(2^n-1)/n$ pseudorandom sequences and yet a pair of n-stage shift registers can generate 2^n+1 different Gold sequences. The increase in the number of available sequences is drastic. For instance, from Table 3.1, for n=10, $\phi(2^n-1)/n = 60$ and $2^n+1 = 1025$.

As an example of Gold sequences consider the case of n=3. There are two primitive polynomials

$$P_1(x) = x^3 + x + 1 \quad \text{generates} \quad S_1 = 0\ 0\ 1\ 1\ 1\ 0\ 1$$

and

$$P_1(x) = x^3 + x^2 + 1 \quad \text{generates} \quad S_2 = 1\ 1\ 1\ 0\ 0\ 1\ 0$$

This pair generates nine Gold sequences

$$S_1 = 0\ 0\ 1\ 1\ 1\ 0\ 1$$

$$S_2 = 1\ 1\ 1\ 0\ 0\ 1\ 0$$

$$S_3 = 1\ 1\ 0\ 1\ 1\ 1\ 1$$

$$S_4 = 0\ 1\ 0\ 0\ 1\ 0\ 0$$

$$S_5 = 1\ 0\ 0\ 0\ 0\ 0\ 1$$

$$S_6 = 0\ 1\ 1\ 0\ 0\ 1\ 1$$

$$S_7 = 0\ 0\ 0\ 1\ 0\ 1\ 0$$

$$S_8 = 1\ 0\ 1\ 0\ 1\ 1\ 0$$

$$S_9 = 1\ 1\ 1\ 1\ 0\ 0\ 0$$

We have illustrated that the number of available codes increases from $\frac{\phi(2^n-1)}{n}$ to 2^n+1. However, we have yet to address to the cross-correlation problem. To solve this problem Gold proved the following [6].

Given a pseudorandom sequence generated by a polynomial $P_1(x)$ of degree n having α as a root, let $P_t(x)$ be a polynomial with α^t as a root where

$$t = \begin{cases} 2^{\frac{n+1}{2}} + 1 & \text{if n is odd} \\ 2^{\frac{n+2}{2}} + 1 & \text{if n is even.} \end{cases}$$

Then the cross-correlation between the Gold sequences generated by $P_1(x)$ and $P_t(x)$ satisfies the following:

$$|R_\tau(1,t)| \leq \begin{cases} 2^{\frac{n+1}{2}} + 1 & \text{if n is odd} \\ 2^{\frac{n+2}{2}} + 1 & \text{if n is even.} \end{cases}$$

$P_1(x)$ and $P_t(x)$ form a preferred pair.

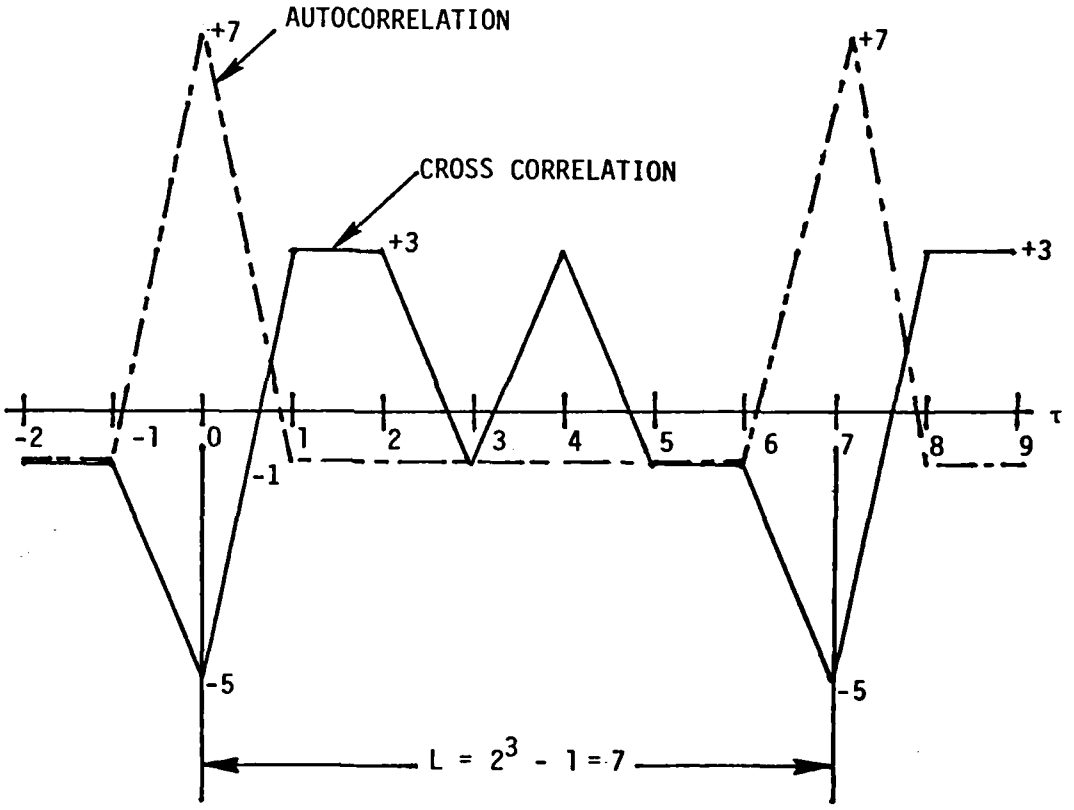
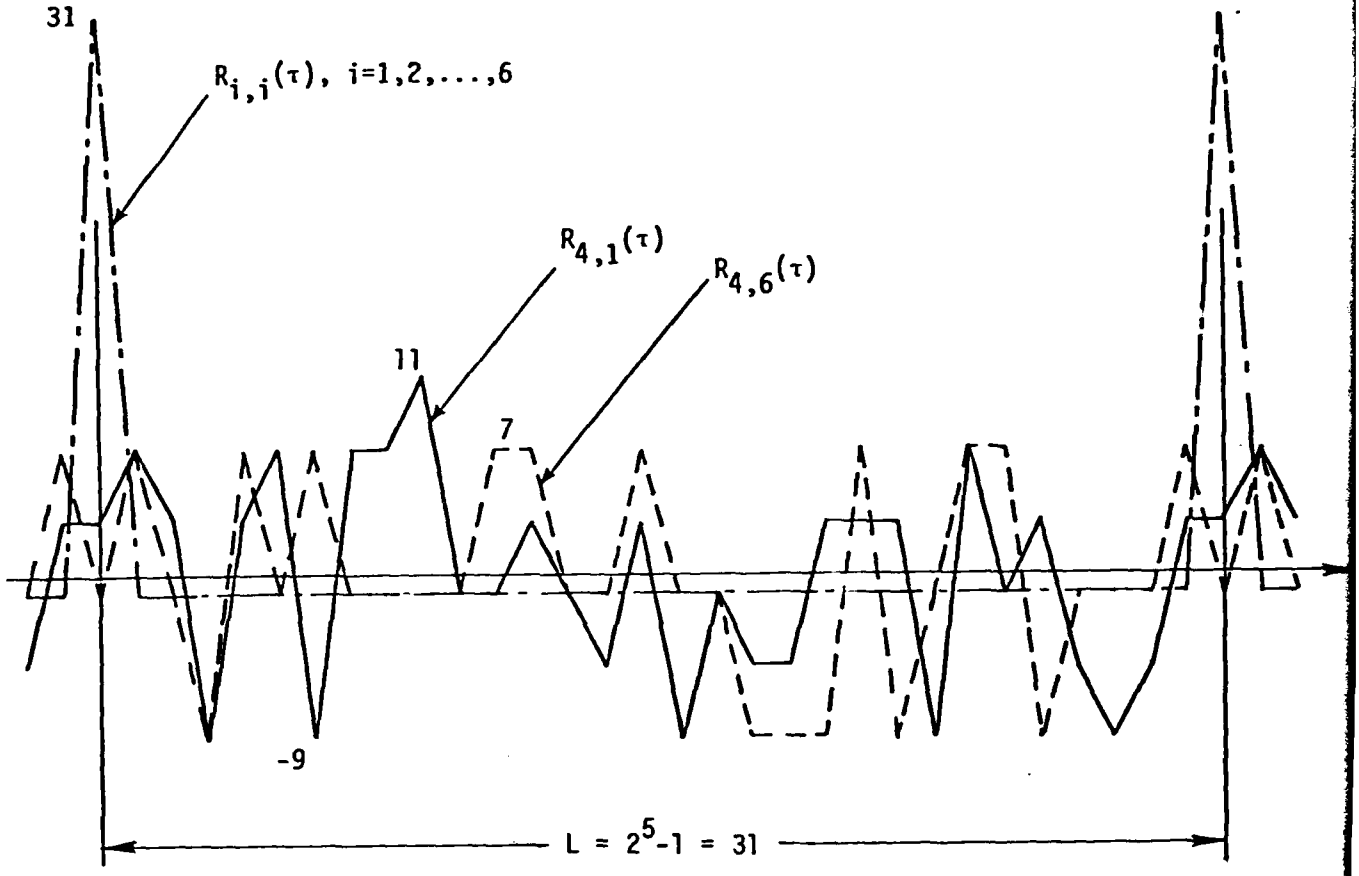


Figure 3.13 AUTOCORRELATION FUNCTION AND CROSS-CORRELATION FUNCTION BETWEEN TWO PSEUDORANDOM SEQUENCES GENERATED BY $x^3 + x + 1$ and $x^3 + x^2 + 1$



$$p_1(X) = X^5 + X^2 + 1$$
$$p_4(X) = X^5 + X^3 + 1$$
$$p_6(X) = X^5 + X^4 + X^3 + X + 1$$

Figure 3.14 CORRELATION OF PSEUDORANDOM SEQUENCES

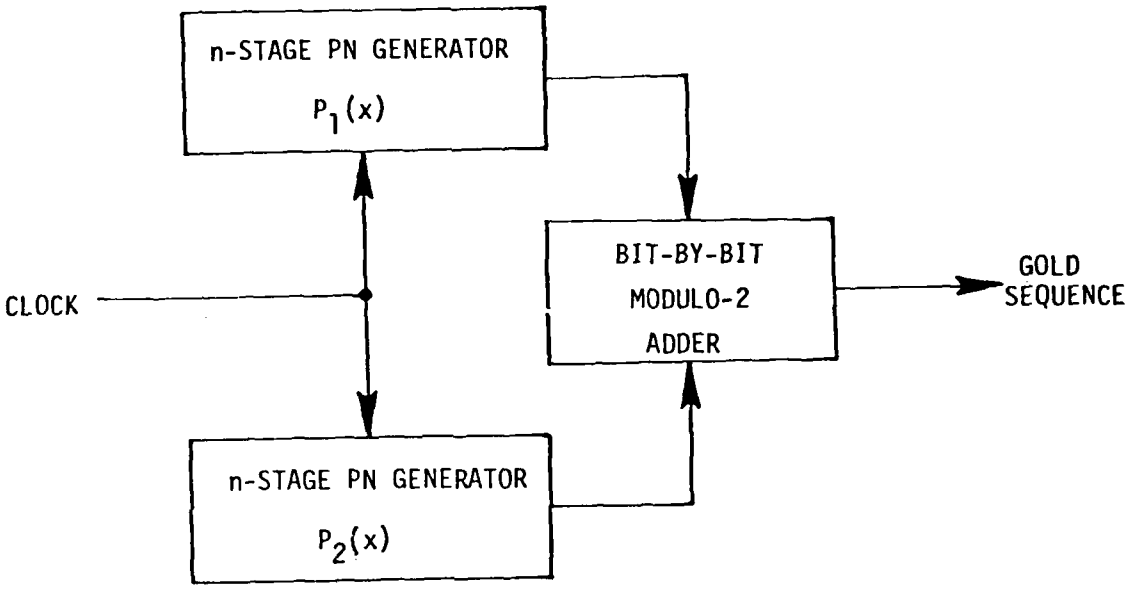


Figure 3.15 GOLD SEQUENCE GENERATOR

Suppose that n=3,

$$P_1(x) = x^3 + x + 1$$

$$2^{\frac{n+1}{2}} + 1 = 2^2 + 1 = 5$$

$$P_5(x) = x^3 + x^2 + 1.$$

The absolute value of the cross-correlation between the Gold sequences generated by $x^3 + x + 1$ and $x^3 + x^2 + 1$ does not exceed 5.

We now consider an example to show how to design a Gold sequence generator with a preferred pair of primitive polynomials of degree n. Suppose that n is given:

$$n = 11.$$

From Peterson's Table of Irreducible Polynomials, we can choose 1 4005E as $P_1(x)$, i.e.

$$P_1(x) = x^{11} + x^2 + 1$$

$$t = 2^{\frac{n+2}{2}} + 1 = 65.$$

We want to find $P_{65}(x)$ which has α^{65} as a root. But the polynomial with 65 as leading number is not listed in the table. In this case we can do the following.

(1) Calculate 2^m x 65, for m=1,2,...,11. If 2^m x 65 is larger than $2^{11}-1=2047$, divide 2^m x 65 by 2047 and replace 2^m x 65 by the remainder, and then continue the doubling process.

65, 130, 260, 520, 1040, 2080 = 33, 66, 132, 264, 528, 1056, 2112 = 65

(2) The doubling process will end at the 11-th time; at the 12-th time the number will go back to the original number, 65.

(3) $\alpha^{65}, \alpha^{130}, \ldots \alpha^{1056}$ are roots of the same polynomial.

(4) One of the eleven numbers, 33, is listed in the table:

33 7335G.

(5) Therefore,

$$P_{65}(x) = P_{33}(x)$$

$$P_{33}(x) = x^{11} + x^{10} + x^9 + x^7 + x^6 + x^4 + x^3 + x^2 + 1.$$

There are $2^{11} + 1 = 2049$ Gold sequences of length L=2047 with cross-correlation

$$|R_{i,j}(\tau)| \leq 65.$$

Gold sequences can be used for scrambling just like pseudorandom sequences. They are especially attractive in systems which do not require long sequences but need to accommodate a large number of users. A conceptual block diagram of a scrambler using a Gold sequence is shown in Figure 3.16. The two pseudorandom-sequence generators are of the same number of stages (n stages, say). The bit-by-bit modulo-2 sum of the two pseudorandom sequences is a Gold sequence which is a (2^n-1)-digit non-maximal sequence (which means the Gold sequence is not a maximum-length sequence although both original pseudorandom sequences are).

3.2.3 Frame Synchronization

In Section 1 of this report we have established the feasibility of using FM correlator to measure the frequency of the spectral line. The spectral line detector makes one measurement in every 10-15 seconds and the measurement value is quantized and encoded into a 7-11 bit word for transmission to a central location for processing. The information rate from a single line detector is very low. Even though a line detection system could combine a large number of detectors the information rate would still be so low that the only viable transmission method would be in

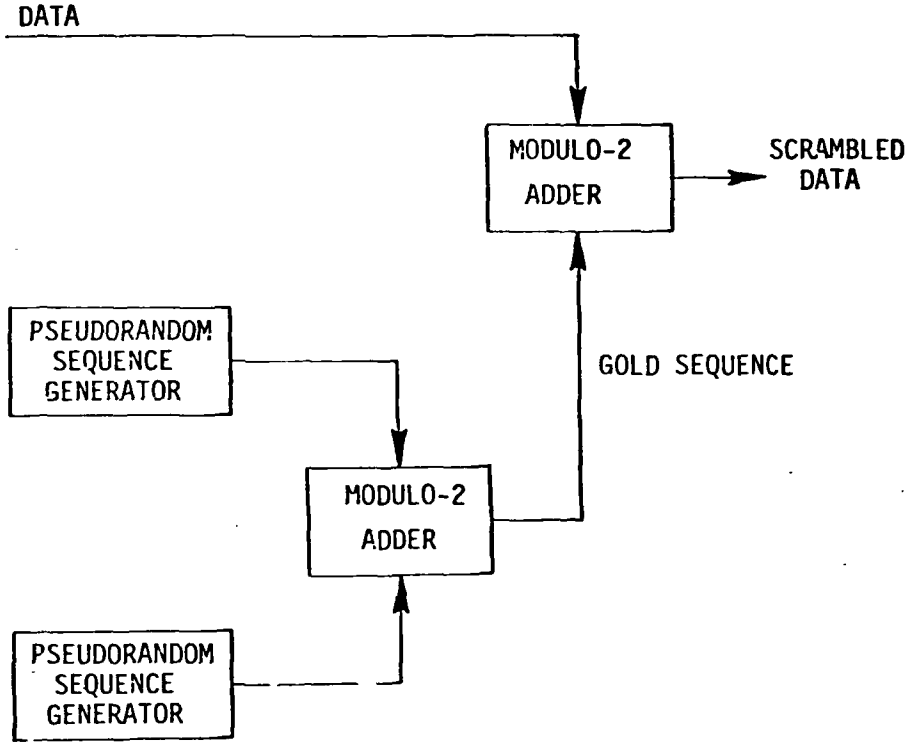


Figure 3.16 SCRAMBLER USING GOLD SEQUENCE

burst mode; i.e. all the spectral data are stored and are being transmitted in a short burst of time. The typical transmission rate of a communication link via satellite could be either 1200 bps or 2400 bps. The function of the frame-synchronization scheme is to separate the data from different detectors into blocks called frames. The start of a frame is indicated by one or more bits periodically inserted at the beginning of each frame. For the system under discussion, because of the low data rate a single bit for an alternating zero-one pattern is sufficient for the receiver to obtain frame synchronization. The frame structure is shown in Figure 3.17. It should be noted that the bit number is arbitrarily chosen just for illustration purpose. Each frame contains the scrambled spectral data from one FM line detector. At the beginning of each frame one bit from an alternating zero-one pattern is used for frame sync; the second bit is from the X-sequence which will be discussed in detail later; and the remaining 11 bits are the scrambled data from one spectral detector.

3.2.4 Data Scrambler System and Block Diagram

As mentioned in Section 3.1.1, the function of the data scrambler system is:

(1) to receive and store data from the spectral detectors

(2) to scramble the data

(3) to arrange the data in proper frame format

(4) to provide signals for sync recovery.

As shown in the block diagram in Figure 3.1, K-2 bits of data are received from one of the N spectral detectors and are stored in a K-stage buffer with two leading stages containing zeros. At the time of transmission, the multiplexer switches out the data emptying the buffers one by one with the output clock running at the same rate as the bit rate clock. The

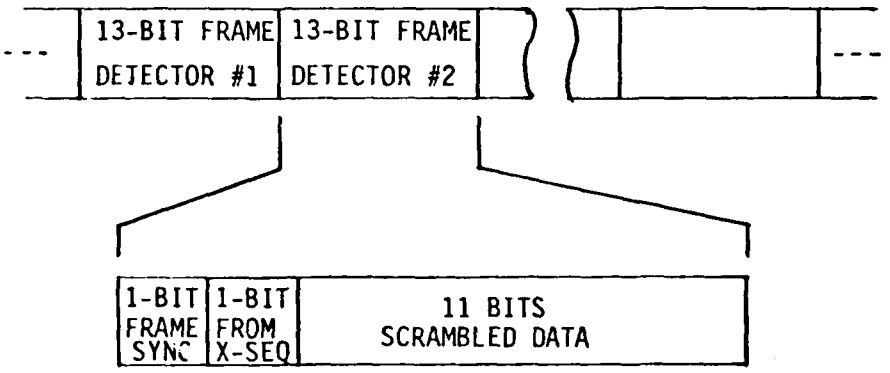


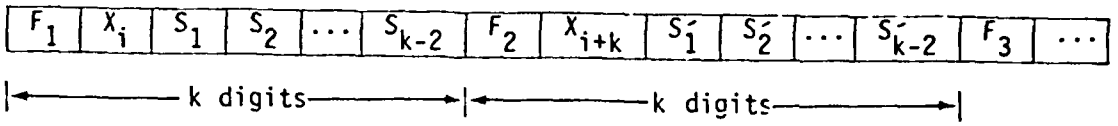
Figure 3.17 FRAME STRUCTURE

output of the multiplexer is a continuous bit stream separated into blocks of K bits; each block begins with two zeros followed by K-2 data bits from one spectral detector.

For data scrambling, a Gold sequence is chosen over a single pseudorandom sequence. A Gold sequence possesses two advantages over a single pseudorandom sequence: namely, the bounded cross-correlations and the large number of available sequences. In the Spectral Data Transfer System, although the bounded cross-correlation characteristic is not being utilized, a Gold sequence still offers more security than a single pseudorandom sequence because there are more codes available for altering.

Two pseudorandom-sequence generators, #1 and #2, of the same number of stages, and with timing pulses provided by a Bit-Rate Clock, generate two sequences, the X sequence and the Y sequence respectively. The X sequence and the Y sequence are bit-by-bit modulo-2 added to form a Gold sequence. The multiplexed data sequence is then scrambled by the Gold sequence through a bit-by-bit modulo-2 adder.

The frame-format circuit has three inputs: the scrambled data sequence and the X sequence both at the bit rate and the frame sync pattern sequence which is an alternating zero-one sequence at $\frac{1}{K}$ of the bit rate. A and B form the select signal: when A=1 and B=C, the output of the frame-format circuit is the frame sync bit; when A=0 and B=1, the output is a digit from the X sequence; and at the remaining K-2 instances both A=0 and B=0, the outputs are the scrambled data bits. Thus, the output sequence of the frame-format circuit has the following frame structure:



where F's are the frame sync bits; X_i, X_{i+k},...are digits from the X sequence; and s_i and s_i^{-1}, $i = 1, 2,\ldots,K-2$ are the scrambled data bits.

There are two sequences at the output of the data scrambler: the scrambled data sequence described above and the Y sequence. These two sequences are to be interleaved at the FEC encoder. Sync recovery will be discussed in detail in later sections; it suffices to indicate here that the digits, x_i, x_{i+k}, x_{i+2k},..., which form a sampled X sequence, sampled every K digits, are used to recover the sync of the X sequence at the receiver while the Y sequence transmitted with the scrambled data is for recovery of the sync of the corresponding sequence.

3.3 Self-Synchronization

At the receiving end the descrambler performs the reverse process of the scrambler as shown in Figure 3.18. Two pseudorandom-sequence generators identical to the ones in the scrambler with predetermined initial states are required. When they are activated at the right moment, the correct Gold sequence is generated and the data is recovered. However, in case synchronization is lost means must be provided for its recovery.

The methods of sync recovery for X sequence and Y sequence are different. As mentioned in Section 3.2.4, the Y sequence is being transmitted along with the scrambled data. The received Y sequence is stored in an n-stage buffer register and is compared with the register contents of the local PN generator #2 through a bit-by-bit comparator as shown in Figure 3.3. When a disagreement is detected a signal is sent to replace the register contents of the local PN generator #2 with that of the received Y sequence. The disagreement signal also activates the sync-recovery process for the X sequence. Thus the Y sequence performs two functions: sync recovery for the Y sequence and sync loss detection.

3.3.1 Preliminary Description

The scrambled data sequence is the bit-by-bit modulo-2 sum of the data sequence and the Gold sequence which, in turn, is the bit-by-bit modulo-2 sum of the X sequence and the Y sequence. The synchronization of both the X sequence and the Y sequence is required for descrambling the data. To recover the sync for the Y sequence, external information, besides the scrambled data sequence, is needed; in our case the external information is the Y sequence itself transmitted along with the scrambled data sequence. On the other hand no external information is needed to recover the sync of the X sequence; the information needed is embedded in the scrambled data

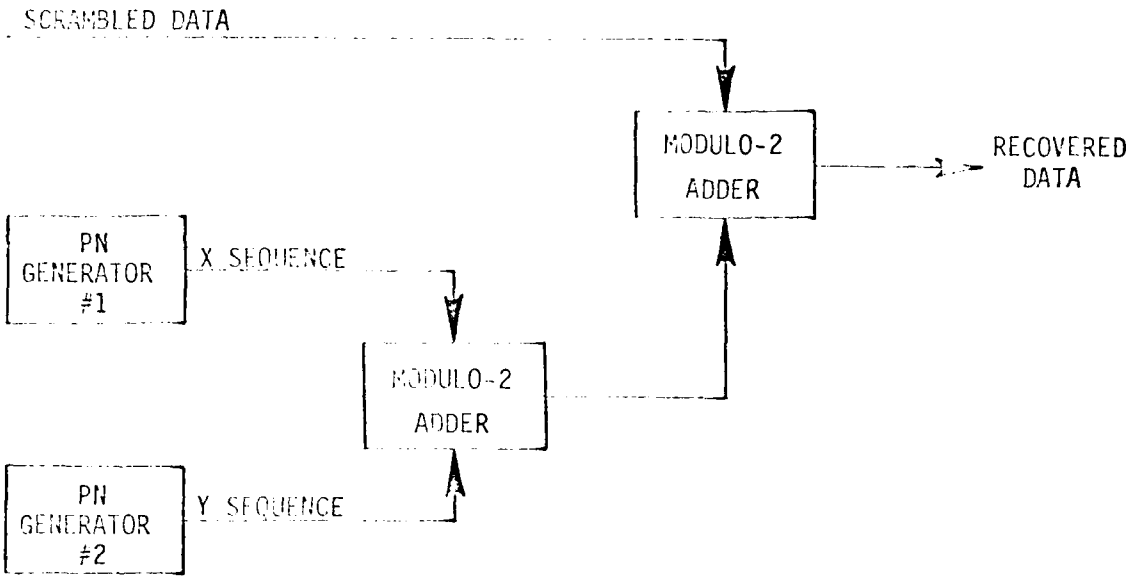
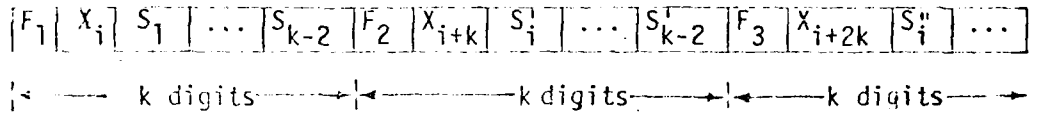


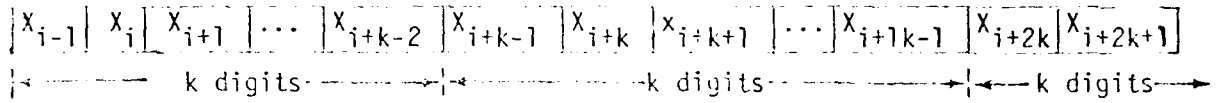
Figure 3.18 DESCRAMBLER

sequence. This is called "self-synchronization".

Recall the frame structure of the scrambled data sequence.



The X's in the scrambled data sequence are from the original X sequence.



At the receiver these digits

$$x_i \cdots x_{i+k} \cdots x_{i+2k} \cdots$$

are separated from the scrambled data sequence. When sync loss is detected n of these digits

$$x_i, x_{i+k}, x_{i+2k}, \ldots x_{i+(n-1)k}$$

are used to calculate the proper initial state of the pseudorandom generator for sync recovery. The logic steps of the sync recovery process is shown in Figure 3.19.

Before we attempt to explain how this self synchronization scheme works we shall review some basics of Finite Fields.

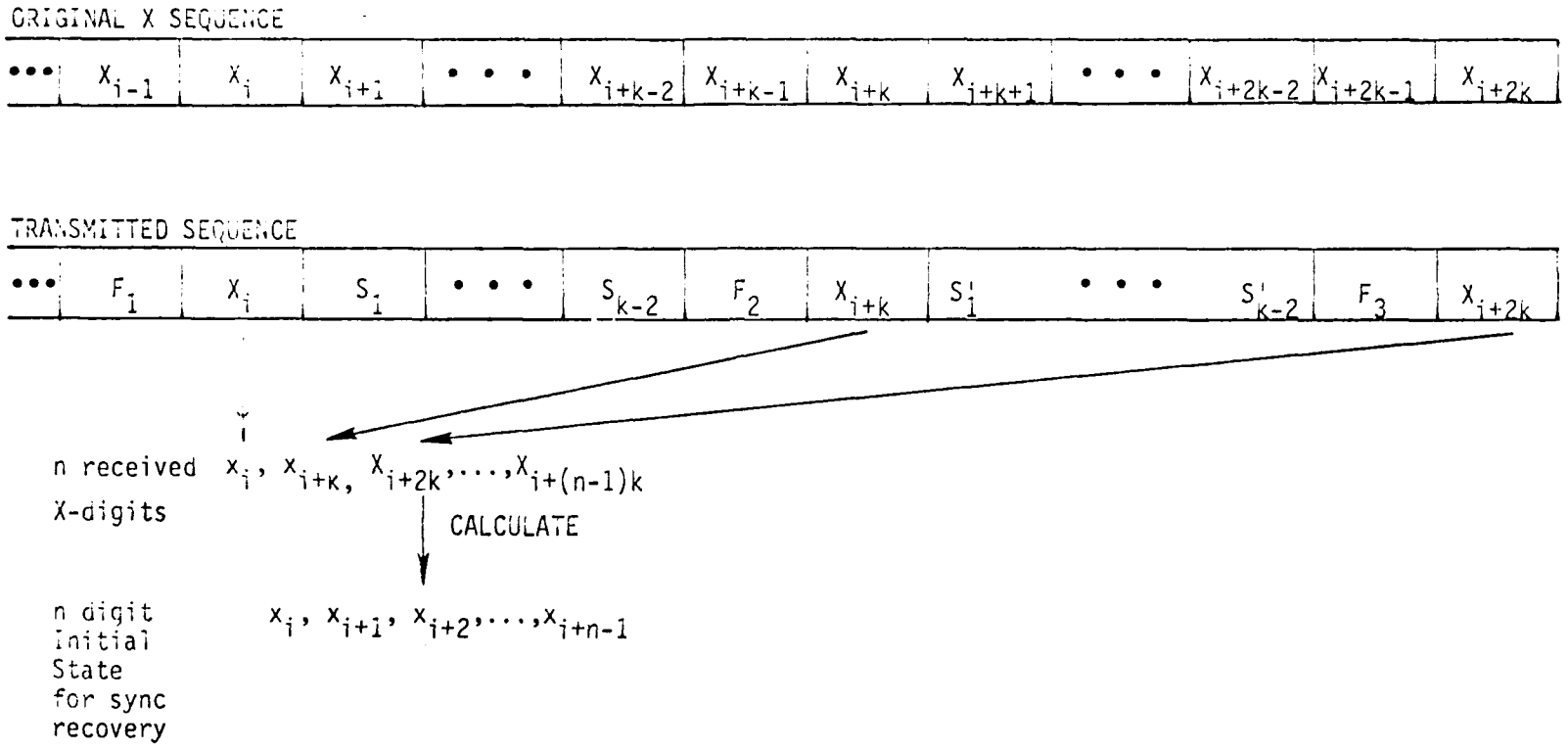


Figure 3.19 INITIAL STATE RECOVERY

3.3.2 Pseudorandom Sequence and Elements of a Finite Field

It has been described in Section 3.2.1 how a primitive polynomial of degree n, say, P(x) can generate a pseudorandom sequence of length $L=2^n-1$. The same primitive polynomial also defines a finite field of 2^n elements and we want to describe how these field elements are defined.

Consider the totality of polynomials with one indeterminant x and with coefficients either 0 or 1. Let the polynomials be represented by F(x), and let P(x) be a primitive polynomial of degree n. Divide F(x) by P(x) resulting in

$$F(x) = Q(x)P(x) + R(x)$$

where Q(x) is the quotient polynomial and R(x) is the remainder polynomial. The remainder R(x) is of degree less than or equal to n-1;

$$R(x) = C_{n-1}\, x^{n-1} + C_{n-2}\, x^{n-2} + \ldots + C_1 x + C_0$$

where C_i is either 0 or 1. There are altogether 2^n possible remainder polynomials and they form a finite field of 2^n elements, sometimes referred to as Galois field of 2^n elements or $GF(2^n)$ for short.

Addition and multiplication in the field are the same as that between polynomials except that in the case of multiplication when the product is of degree higher than n-1, it should be divided by P(x) and replaced by the remainder polynomial. For example, let

$$F_1(x) = x^4 + x^2 + 1,$$

$$F_2(x) = x^3 + x + 1,$$

and

$$P(x) = x^5 + x^2 + 1.$$

The product between $F_1(x)$ and $F_2(x)$ is

$$F_1(x) \cdot F_2(x) = (x^4 + x^2 + 1)(x^3 + x + 1)$$

$$= x^7 + x^4 + x^2 + x + 1.$$

Divide $F_1(x) \cdot F_2(x)$ by $P(x)$ resulting in

$$x^7 + x^4 + x^2 + x + 1 = x^2 (x^5 + x^2 + 1) + (x + 1)$$

where the quotient polynomial is x^2 and the remainder polynomial is $x + 1$. Therefore in $GF(2^5)$ the product between the element $F_1(x)$ and the element $F_2(x)$ is the element $x + 1$:

$$F_1(x) \cdot F_2(x) = x + 1.$$

Since a polynomial can be put into a one-to-one correspondence with its coefficient sequence, i.e.

$F(x) = C_{n-1} x^{n-1} + C_{n-2} x^{n-2} + \ldots + C_1 x + C_o \leftrightarrow C_{n-1}, C_{n-2}, \ldots C_1, C_0$,

an n-tuple of zeros and ones can be regarded as an element in $GF(2^n)$. Consider the example in Figure 3.9. The polynomial $P(x) = x^5 + x^2 + 1$ is a primitive polynomial; it generates a maximum length sequence of length L=31 and at each clock time it generates one digit of the sequence. In the mean time, at each clock time the contents of the generator are a 5-tuple which can represent a polynomial. For instance, at t=5, the output sequence digit is "0" and the field element is

1 1 0 1 0

which represents the polynomial

$$1 + x + x^3$$

and at t=24, the output-sequence digit is "1" and the field element is

0 1 0 1 1

which represents the polynomial

$$x + x^3 + x^4.$$

There are 31 distinct 5-tuples; joining all the zero 5-tuple, 0 0 0 0 0, they are the entire 32 elements of $GF(2^5)$.

We have established a one-to-one correspondence between the digits of a maximum-length sequence and the elements of the field. In the example shown in Figure 3.9 if the digit of the output sequence at t=0 is identified as the 0-th digit, "0", then it corresponds to the field element represented by

1 0 0 0 0

and the 5-th digit of the output sequence, "0", corresponds to the field element

1 1 0 1 0

and the 24-th digit of the output sequence, "1", corresponds to the field element

0 1 0 1 1.

Consider again the example shown in Figure 3.9. The polynomial generating the pseudorandom sequence is

$$P(x) = x^5 + x^2 + 1.$$

The reciprocal polynomial of P(x) is

$$P^*(x) = x^5 + x^3 + 1.$$

Connect a linear feedback shift register according to $P^*(x)$, but this time the modulo-2 adder is placed between the shift register stages as shown in Figure 3.20. An arbitrary initial state of a 5-tuple, say 1 0 0 1 1, is loaded into the shift register; after 30 shifts the register contents returns to their initial state, 1 0 0 1 1. The 31 5-tuples at each clock time are all the possible 5-tuples except the all-zero 5-tuple, 0 0 0 0 0. They are the elements of $GF(2^5)$ just like the ones in

1	0	0	1	1	β^{18}
1	1	0	1	1	β^{19}
1	1	1	1	1	β^{20}
1	1	1	0	1	β^{21}
1	1	1	0	0	β^{22}
0	1	1	1	0	β^{23}
0	0	1	1	1	β^{24}
1	0	0	0	1	β^{25}
1	1	0	1	0	β^{26}
0	1	1	0	1	β^{27}
1	0	1	0	0	β^{28}
0	1	0	1	0	β^{29}
0	0	1	0	1	β^{30}
1	0	0	0	0	$\leftarrow \beta^{0} = 1$
0	1	0	0	0	$\beta^{1} = \beta$
0	0	1	0	0	β^{2}
0	0	0	1	0	β^{3}
0	0	0	0	1	β^{4}
1	0	0	1	0	β^{5}
0	1	0	0	1	β^{6}
1	0	1	1	0	β^{7}
0	1	0	1	1	β^{8}
1	0	1	1	1	β^{9}
1	1	0	0	1	β^{10}
1	1	1	1	0	β^{11}
0	1	1	1	1	β^{12}
1	0	1	0	1	β^{13}
1	1	0	0	0	β^{14}
0	1	1	0	0	β^{15}
0	0	1	1	0	β^{16}
0	0	0	1	1	β^{17}
1	0	0	1	1	β^{18}

Figure 3.20 MODULAR SHIFT REGISTER GENERATOR

Figure 3.9. The only difference between these two sets is the order in which the elements are arranged.

Let us make two important observations.

(1) Compare the sequences taken from the 5-th stage of the two shift registers. They are identical except for the point at which each sequence begins. We observe that these two sequences are identical up to a phase shift. The amount of phase difference depends on the initial states of the two shift registers.

The shift register in Figure 3.9 is sometimes referred to as simple shift-register generator, or SSRG, and the one in Figure 3.20 is referred to as a modular shift-register generator, or MSRG. We have illustrated that a pseudorandom sequence generated by a primitive polynomial, $P(x)$, using an SSRG is identical up to a phase shift with that generated by its reciprocal polynomial, $P^*(x)$, but using an MSRG.

(2) Starting from the 5-tuple 1 0 0 0 0 in the list of the contents of the shift register generated by $P^*(x) = x^5 + x^3 + 1$ using the MSRG in Figure 3.20, write down the corresponding polynomial in ascending powers.

1 0 0 0 0	$1 + 0x + 0x^2 + 0x^3 + 0x^4$
0 1 0 0 0	x
0 0 1 0 0	x^2
0 0 0 1 0	x^3
0 0 0 0 1	x^4
1 0 0 1 0	$1 + x^3$
0 1 0 0 1	$x + x^4$
1 0 1 1 0	$1 + x^2 + x^3$
0 1 0 1 1	$x + x^3 + x^4$
⋮	⋮

From the simple calculations,

$$x^5 = (1)(1 + x^3 + x^5) + (1 + x^3) \quad \leftrightarrow \quad x^5 = 1 + x^3$$

$$x^6 = (x)(1 + x^3 + x^5) + (x + x^4) \quad \leftrightarrow \quad x^6 = x + x^4$$

$$x^7 = (1 + x^2)(1 + x^3 + x^5) + (1 + x^2 + x^3) \quad \leftrightarrow \quad x^7 = 1 + x^2 + x^3$$

$$x^8 = (x + x^3)(1 + x^3 + x^5) + (x + x^3 + x^4) \quad \leftrightarrow \quad x^8 = x + x^3 + x^4$$

$$\vdots$$

it can be verified that the field elements are arranged in the order of the powers of α, i.e.

$$x^i \pmod{P^*(x)}, \quad \text{for } i = 0, 1, 2, \ldots, 30.$$

Now if we let β be the field element represented by the polynomial x or the n-tuple 0 1 0 0 0, then

$$\beta^0 = 1$$

$$\beta = x$$

$$\beta^2 = x^2$$

$$\beta^3 = x^3$$

$$\beta^4 = x^4$$

$$\beta^5 = 1 + x^3 = 1 + \beta^3$$

$$\beta^6 = x + x^4 = \beta + \beta^4$$

$$\beta^7 = 1 + x^2 + x^3 = 1 + \beta^2 + \beta^3$$

$$\beta^8 = x + x^3 + x^4 = \beta + \beta^3 + \beta^4$$

$$\vdots$$

Since $\beta^5 = 1 + \beta^3$,

$$\beta^5 + \beta^3 + 1 = 0$$

i.e. $P^*(\beta) = \beta^5 + \beta^3 + 1 = 0$,

which means β is a root of $P^*(x)$. Thus we have reached an important result. The digits of a pseudorandom sequence generated by a primitive

polynomial, $P(x)$, of degree n can be put into a one-to-one correspondence with the field elements of $GF(2^n)$ ordered as the consecutive powers of a root β of $P^*(x)$, which is the reciprocal polynomial of $P(x)$. Using the same example the pseudorandom sequence generated by $P(x) = x^5 + x^2 + 1$ in Figure 3.9 can be put into a one-to-one correspondence with powers of β where β is a root of $P^*(x) = x^5 + x^3 + 1$. This correspondence is shown in Figure 3.21.

PSEUDORANDOM SEQUENCE GENERATED BY $P(x) = x^5 + x^2 + 1$	POWERS OF β $P^*(\beta) = \beta^5 + \beta^3 + 1 = 0$
0	$\beta^0 = 1$
0	$\beta^1 = \beta$
0	β^2
0	β^3
1	β^4
0	β^5
1	β^6
0	β^7
1	β^8
1	β^9
1	β^{10}
0	β^{11}
1	β^{12}
1	β^{13}
0	β^{14}
0	β^{15}
0	β^{16}
1	β^{17}
1	β^{18}
1	β^{19}
1	β^{20}
1	β^{21}
0	β^{22}
0	β^{23}
1	β^{24}
1	β^{25}
0	β^{26}
1	β^{27}
0	β^{28}
0	β^{29}
1	β^{30}

Figure 3.21 CORRESPONDENCE BETWEEN A PSEUDORANDOM SEQUENCE AND POWERS OF AN ELEMENT IN A FINITE FIELD

3.3.3 Sampled Version of a Pseudorandom Sequence

In the data scrambler system described in Section 3.2, two pseudorandom sequences, the X sequence and the Y sequence, are generated to construct the Gold sequence. The Y sequence is transmitted along with the scrambled data sequence. In every k digits of the transmitted data sequence there is one digit which comes from the original X sequence. If we pick out this digit from the transmitted sequence, the resultant sequence is the sampled version of the original X sequence, sampled every k digits. We now want to analyze the relationship between this new sequence and the original X sequence and to find out how this new sequence can be utilized to recover the synchronization between the transmitted X sequence and the locally generated X sequence at the receiving end. For notational convenience in the following discussion, the X sequence will be called the "A sequence" and the new sequence, which is the sampled version of the A sequence, will be called the "B sequence."

Consider the pseudorandom sequence (A sequence) of length $L=2^n-1$, which is generated by a primitive polynomial of degree n, $P_A(x)$. Sample the A sequence every k digits, where k is relatively prime to L, and form the B sequence:

$$\begin{array}{ccccccccc} A_0, & A_1, \ldots & A_k, & A_{k+1}, \ldots, & A_{2k}, \ldots \\ \downarrow & & \downarrow & & \downarrow \\ B_0 & & B_1 & & B_2 & \cdots . \end{array}$$

Since k and L are relatively prime to each other, the B sequence will not repeat itself until every digit of the A sequence has been sampled;

hence the B sequence is also a maximum-length sequence and the length of the B sequence is also L.

Since the B sequence is a maximum-length sequence, there must be a primitive polynomial, $P_B(x)$, which will generate the B sequence. By the important result obtained in Section 3.3.2, that the digits of a pseudorandom sequence generated by a primitive polynomial, P(x), of degree n can be put into a one-to-one correspondence with the field elements of $GF(2^n)$ ordered as the consecutive powers of a root of the reciprocal polynomial, P*(x), of P(x) the A sequence corresponds to powers of α^{-1} and the B sequence corresponds to powers of β^{-1}:

$$
\begin{array}{ccccccccccc}
A_0, & & A_1, & \cdots, & A_k & A_{k+1}, & \cdots, & A_{2k}, & A_{2k+1}, & \cdots \\
\updownarrow & & \updownarrow & & \updownarrow & \updownarrow & & \updownarrow & \updownarrow & \\
(\alpha^{-1})^0=1, & & \alpha^{-1}, & \cdots, & \alpha^{-k}, & \alpha^{-(k+1)}, & \cdots, & \alpha^{-2k}, & \alpha^{-(2k+1)} & \\
\downarrow & & & & \downarrow & & & \downarrow & & \\
B_0, & & & & B_1, & & & B_2, & \cdots & \\
\updownarrow & & & & \updownarrow & & & \updownarrow & & \\
(\beta^{-1})^0=1, & & & & \beta^{-1}, & & & \beta^{-2}, & \cdots &
\end{array}
$$

where α^{-1} is a root of $P_A^*(x)$ and β^{-1} is a root of $P_B^*(x)$ and $\beta^{-1} = (\alpha^{-1})^k$.

In Section 3.2.1 we have defined the reciprocal polynomial without referring to field elements as follows: if P(x) is a polynomial of degree n, then the reciprocal polynomial

$$P^*(x) = x^n P(\tfrac{1}{x}).$$

In terms of field elements, reciprocal polynomial has the following meaning: if a field element, γ, is a root of a polynomial P(x), i.e.

$$P(\gamma) = 0$$

then the inverse element of γ, γ^{-1}, is a root of the reciprocal polynomial $P^*(x)$ or

$$P^*(\gamma^{-1}) = 0.$$

As an example, consider the finite field $GF(2^5)$ generated by the primitive polynomial

$$P(x) = x^5 + x^3 + 1$$

the field element

$$\gamma = 0\ 1\ 0\ 0\ 0$$

is a root of $P(x)$, i.e. $P(\gamma) = 0$.

This can be verified as follows: from Figure 3.20, we observe that

$$\gamma^5 = (1\ 0\ 0\ 1\ 0)$$
$$\gamma^3 = (0\ 0\ 0\ 1\ 0)$$
$$1 = (1\ 0\ 0\ 0\ 0)$$
$$\gamma^5 + \gamma^3 + 1 = (0\ 0\ 0\ 0\ 0)$$

Since the 5-tuple, 0 0 0 0 0, is the field element, 0, therefore

$$P(\gamma) = \gamma^5 + \gamma^3 + 1 = 0.$$

The reciprocal polynomial $P^*(x)$ is

$$P^*(x) = x^5\, P\left(\frac{1}{x}\right)$$
$$= x^5 \left(\frac{1}{x^5} + \frac{1}{x^3} + 1\right)$$
$$= x^5 + x^2 + 1$$

and

$$(\gamma^{-1})^5 = \gamma^{-5} = \gamma^{26} = (1\ 1\ 0\ 1\ 0)$$

$$(\gamma^{-1})^2 = \gamma^{-2} = \gamma^{29} = (0\ 1\ 0\ 1\ 0)$$

$$\underline{\qquad 1 \quad = \qquad\qquad (1\ 0\ 0\ 0\ 0)}$$

$$(\gamma^{-1})^5 + (\gamma^{-1})^2 + 1 = (0\ 0\ 0\ 0\ 0)$$

hence

$$P^*(\gamma^{-1}) = (\gamma^{-1})^5 + (\gamma^{-1})^2 + 1 = 0$$

Now from the results obtained above,

$$\beta^{-1} = (\alpha^{-1})^k,$$

since each field element has a unique inverse, it follows that

$$\beta = \alpha^k.$$

Furthermore α^{-1} is a root of $P_A^*(x)$ and β^{-1} is a root of $P_B^*(x)$ imply that α is a root of $P_A(x)$ and β is a root of $P_B(x)$ respectively. It can, therefore, be concluded that if α is a root of a primitive polynomial, $P_A(x)$, generating the A sequence and β is a root of $P_B(x)$ generating the B sequence; and if the B sequence is a sampled version of the A sequence, sampled every k digits then β is the k-th power of α, i.e.

$$\beta = \alpha^k.$$

AD-A092 940 LEE (J S) ASSOCIATES INC ARLINGTON VA F/G 17/2
FM CORRELATOR SPECTRAL DATA TRANSFER BY SCRAMBLED TRANSMISSION --ETC(U)
OCT 80 J S LEE, S TSAI, L E MILLER N00014-80-C-0129
UNCLASSIFIED JTR-80-03 NL

2 of 3
AD-A092

When $k = 2^m$, for $m \leq n-1$, $P_B(x) = P_A(x)$. This follows from the fact that in $GF(2^n)$ if α is a root of a polynomial $P(x)$, then α^2, α^4,..., α^{2^m} are also roots of $P(x)$. The notation $P_B(x) = P_A(x)$ means that the B sequence is identical to the A sequence up to a phase shift [7].

Consider the following examples.

Example 1.

$n = 3$, $L = 2^3-1 = 7$

Choose $P_A(x) = x^3 + x + 1$ with 1 0 1 as the initial state as in Figure 3.7. The generated A sequence is:

A_0	A_1	A_2	A_3	A_4	A_5	A_6
1	0	1	0	0	1	1.

If $k = 2^2 = 4$, then

$B_0 = A_0 = 1$ $\qquad$ $B_4 = A_{16} = A_2 = 1$

$B_1 = A_4 = 0$ $\qquad$ $B_5 = A_{20} = A_6 = 1$

$B_2 = A_8 = A_1 = 0$ $\qquad$ $B_6 = A_{24} = A_3 = 0$

$B_3 = A_{12} = A_5 = 1.$

The B sequence is

B_0	B_1	B_2	B_3	B_4	B_5	B_6
1	0	0	1	1	1	0.

Comparison between these two sequence shows that the sampled sequence is same as the original sequence except for the starting digits.

If $k = 3$, then

$B_0 = A_0 = 1$ $\qquad$ $B_4 = A_{12} = A_5 = 1$

$B_1 = A_3 = 0$ $\qquad$ $B_5 = A_{15} = A_1 = 0$

$B_2 = A_6 = 1$ $\qquad$ $B_6 = A_{18} = A_4 = 0$

$B_3 = A_9 = A_2 = 1.$

The B sequence is:

B_0	B_1	B_2	B_3	B_4	B_5	B_6
1	0	1	1	1	0	0.

This sequence is different from the original A sequence. Since there are only two different pseudorandom sequences for n=3, the polynomial $P_B(x)$ which generates the B sequence must be the reciprocal polynomial of $P_A(x)$; hence

$$P_B(x) = P_A^*(x) = x^3 + x^2 + 1.$$

Example 2.

$$n = 4,\ L = 2^4 - 1 = 15$$

Choose $P_A(x) = x^4 + x + 1$ with 1 1 0 1 as the initial state as in Figure 3.8. The A sequence is

A_0	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8	A_9	A_{10}	A_{11}	A_{12}	A_{13}	A_{14}
1	0	1	1	0	0	1	0	0	0	1	1	1	1	0.

For $k = 2^2 = 4$, the B sequence is

B_0	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8	B_9	B_{10}	B_{11}	B_{12}	B_{13}	B_{14}
1	0	0	1	0	0	0	1	1	1	1	0	1	0	1

which is a shifted version of the A sequence.

For k = 7, the B sequence is

B_0	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8	B_9	B_{10}	B_{11}	B_{12}	B_{13}	B_{14}
1	0	0	1	1	0	1	0	1	1	1	1	0	0	0.

This sequence is different from the original A sequence. Since there are only two distinct pseudorandom sequences for n = 4, the polynomial

which generates the B sequence must be

$$P_B(x) = P_A^*(x) = x^4 + x^3 + 1.$$

Example 3.

$$n = 5,\ L = 2^5 - 1 = 31$$

Choose $P_A(x) = x^5 + x^2 + 1$ with 1 0 0 0 0 as the initial state as in Figure 3.9. The A sequence, the B sequence by sampling every 4-th digit, and the B sequence by sampling every 5-th digit, are shown in Figure 3.22. Again, the B sequence for $k = 2^2 = 4$ is identical with the A sequence up to a phase shift and the B sequence for k = 5 is different from the A sequence. From the Table of Irreducible Polynomials we found the polynomial

5 67H.

Thus, for k = 5

$$P_B(x) = x^5 + x^4 + x^2 + x + 1.$$

We now summarize the results as follows. Let $P_A(x)$ be a primitive polynomial of degree n which generates a pseudorandom A sequence of length $L = 2^n - 1$. Sample the A sequence at every k-th digit; when (K,L)=1, the sampled sequence (the B sequence) is also pseudorandom and is generated by a primitive polynomial $P_B(x)$. If $k=2^m$, for $m \leq n-1$, then

$$P_B(x) = P_A(x)$$

which means that the B sequence and the A sequence are identical up to a phase shift. If $k \neq 2^m$, then

$$P_B(x) \neq P_A(x).$$

A SEQUENCE		B SEQUENCE k = 4		B SEQUENCE k = 5	
A_0	0	B_0	0	B_0	0
A_1	0	B_1	1	B_1	0
A_2	0	B_2	1	B_2	1
A_3	0	B_3	1	B_3	0
A_4	1	B_4	0	B_4	1
A_5	0	B_5	1	B_5	1
A_6	1	B_6	1	B_6	1
A_7	0	B_7	0	B_7	1
A_8	1	B_8	0	B_8	1
A_9	1	B_9	0	B_9	0
A_{10}	1	B_{10}	1	B_{10}	1
A_{11}	0	B_{11}	1	B_{11}	1
A_{12}	1	B_{12}	1	B_{12}	0
A_{13}	1	B_{13}	1	B_{13}	0
A_{14}	0	B_{14}	1	B_{14}	1
A_{15}	0	B_{15}	0	B_{15}	1
A_{16}	0	B_{16}	0	B_{16}	1
A_{17}	1	B_{17}	1	B_{17}	0
A_{18}	1	B_{18}	1	B_{18}	0
A_{19}	1	B_{19}	0	B_{19}	0
A_{20}	1	B_{20}	1	B_{20}	0
A_{21}	1	B_{21}	0	B_{21}	1
A_{22}	0	B_{22}	0	B_{22}	1
A_{23}	0	B_{23}	1	B_{23}	0
A_{24}	1	B_{24}	0	B_{24}	1
A_{25}	1	B_{25}	0	B_{25}	0
A_{26}	0	B_{26}	0	B_{26}	1
A_{27}	1	B_{27}	0	B_{27}	0
A_{28}	0	B_{28}	1	B_{28}	0
A_{29}	0	B_{29}	0	B_{29}	1
A_{30}	1	B_{30}	1	B_{30}	0

Figure 3.22 PSEUDORANDOM SEQUENCE AND ITS SAMPLED SEQUENCES

However, if α is a root of $P_A(x)$, then

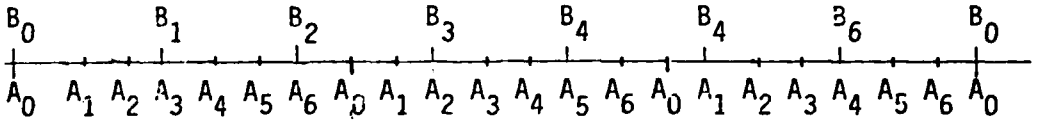
$$\beta = \alpha^k$$

is a root of $P_B(x)$. The polynomial $P_B(x)$ can be determined with aid of the Table of Irreducible Polynomials.

3.3.4 Central Idea of the Self-Synchronization Scheme

The A sequence, which is the original X sequence, is a component of the Gold sequence. At the receiving end an identical pseudorandom-sequence generator generates the A sequence in order to perform the descrambling operation. When a sync loss is detected, the contents of that generator must be cleared and the "correct initial state" must be loaded into the shift register to recover the synchronization. The problem is how to determine the correct initial state.

Every k-th digit of the A sequence is transmitted along with the scrambled data sequence and is received at the receiving end. The sequence of these received digits is called the B sequence which is the sampled version of the A sequence. Suppose that at the time sync loss is detected one such B digit is just received. We label this digit as B_0, corresponding to A_0 in the A sequence. The (n-1) succeeding digits, $A_1, A_2, \ldots, A_{n-1}$, together with A_0 are the correct initial state beginning from A_0. These digits, $A_1, A_2, \ldots, A_{n-1}$, can be obtained from the B sequence. Specifically, since the sample interval, k, and the length of the A sequence, L, are relatively prime to each other, the B sequence will not repeat itself until every digit in the A sequence is sampled and this occurs after exactly k cycles of the A sequence. For instance, if L = 7 and k = 3 the B sequence will repeat after 3 cycles of the A sequence.



Now, there must be a B-digit which equals A_1. Let this B-digit be B_p; then

$$kp = 1 \pmod{L}.$$

This is equivalent to measuring the same distance with two different yardsticks: p multiples of k must be equal to m multiples of L plus one. Therefore,

$$A_0 = B_0$$
$$A_1 = B_p \text{ , where } kp = 1 \pmod{L}.$$

It is not difficult to show that

$$A_2 = B_{2p}$$
$$\vdots$$
$$A_{n-1} = B_{(n-1)p}.$$

However, this is not acceptable because it takes too long to wait for all these digits. To have some idea as to how long it will take to accumulate these digits, let us consider an example. Suppose the PN generator has 11 stages, i.e

$$n = 11.$$

Then $L = 2^{11} - 1 = 2047.$

Let $k = 13.$

Solve $kp = 1 \pmod{2047}$

giving $p = 315,$

for $13 \times 315 = 4095 = 2 \times 2047 + 1.$

Therefore

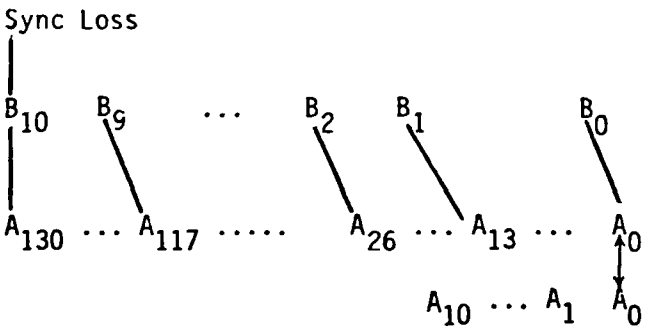
$$A_0 = B_0$$
$$A_1 = B_{315}$$
$$A_2 = B_{630}$$
$$A_3 = B_{945}$$
$$A_4 = B_{1260}$$
$$A_5 = B_{1575}$$
$$A_6 = B_{1890}$$

$$A_7 = B_{2205} = B_{158}$$
$$A_8 = B_{2520} = B_{473}$$
$$A_9 = B_{2835} = B_{788}$$
$$A_{10} = B_{3150} = B_{1103}.$$

We need to wait for 1890 B-digits which is equivalent to 1890 x 13 = 24,570 A-digits. This means that when we lose sync, 24,570 digits later we will find the current initial state beginning at A_0. If we want to recover the sync, we have to speed up the shift register to catch up these 24,570 digits.

The central idea of the self synchronization scheme is to reduce this waiting time to less than a bit duration. When the Spectral Data Transfer System is running in a normal mode, the scrambled data sequence is being received and being descrambled; in every frame of k bits one B-digit is also received. The B-digits are loaded serially into an n-stage shift register; thus n B-digits are being stored and up-dated at all times. When sync loss is detected the most recent digit entering the shift register is designated as B_{n-1}; and the one ready to be discarded, as B_0. The digit $B_0 = A_0$ occurred (n-1)k digits ago in the A sequence. The self-synchronization scheme is to determine the initial state, A_0, A_1, $A_2,\ldots,A_{n-1}$, beginning at A_0 from the presently available B-digits, B_0, B_1, $B_2,\ldots,B_{n-1}$.

Take the same example of n = 11.



It can be seen from this example that if we can determine the initial state $A_0, A_1, \ldots, A_{10}$ beginning at A_0 immediately at the instant sync loss is detected we have 130 digits to catch up instead of 24,570 digits.

Since it has been recognized that

$$\begin{aligned} A_0 &= B_0 \\ A_1 &= B_p \\ A_2 &= B_{2p} \\ &\vdots \\ A_{n-1} &= B_{(n-1)p} \end{aligned}$$

the problem is now to calculate $B_p, B_{2p}, \ldots, B_{(n-1)p}$ in terms of $B_0, B_1, B_2, \ldots, B_{n-1}$.

3.3.5 Determination of $B_m = \sum_{i=0}^{n-1} b_i B_i$ for all $m \leq L$

It has been established in Section 3.3.2 that the digits of a pseudorandom sequence generated by a primitive polynomial of degree n, say $P_B(x)$, can be put into a one-to-one correspondence with the powers of an element of $GF(2^n)$ and that this field element is a root of the reciprocal polynomial of $P_B(x)$ denoted by $P_B^*(x)$. For example, if $P_B(x) = x^5 + x^2 + 1$ the pseudorandom sequence generated by $x^5 + x^2 + 1$ is shown in Figure 3.9. The reciprocal polynomial $P_B^*(x)$ is $x^5 + x^3 + 1$. Powers of β, $P_B^*(\beta) = 0$, are generated in Figure 3.20. Since

$$\begin{aligned} \beta^0 &= 1\ 0\ 0\ 0\ 0 \\ \beta &= 0\ 1\ 0\ 0\ 0 \\ \beta^2 &= 0\ 0\ 1\ 0\ 0 \\ \beta^3 &= 0\ 0\ 0\ 1\ 0 \\ \beta^4 &= 0\ 0\ 0\ 0\ 1 \end{aligned}$$

and any power of β is an n-tuple, β^m can be expressed as a linear combination of β^0, β, β^2, β^3, and β^4, i.e.

$$\beta^m = \sum_{i=0}^{n-1} b_i \beta^i.$$

On the other hand, B_m is just the n-th component of the n-tuple representing β^m; therefore

$$\beta^m = \sum_{i=0}^{n-1} b_i \beta^i$$

implies [7]

$$B_m = \sum_{i=0}^{n-1} b_i B_i.$$

For example in Figure 3.20

$$\beta^{19} = 1\ 1\ 0\ 1\ 1$$
$$= 1\ 0\ 0\ 0\ 0 + 0\ 1\ 0\ 0\ 0 + 0\ 0\ 0\ 1\ 0 + 0\ 0\ 0\ 0\ 1$$
$$= 1 + \beta + \beta^3 + \beta^4$$

and in Figure 3.9

$$B_0 = 0$$
$$B_1 = 0$$
$$B_3 = 0$$
$$B_4 = 1$$
$$B_{19} = B_0 + B_1 + B_3 + B_4 = 0 + 0 + 0 + 1 = 1.$$

Moreover, in the pseudorandom-sequence generator using the simple shift-register generator the consecutive digits of the sequence shift one stage down from left to right at each clock pulse; hence B_m can be

obtained with a modulo-2 adder and proper connections as shown in Figure 3.23. At the next clock pulse B_5 replaces B_4, B_4 replaces B_3,..., B_1 replaces B_0, and the output of the modulo-2 adder is

$$B_5 + B_4 + B_2 + B_1 = B_{20}.$$

The modulo 2 adder and the connections are sometime called the phase shift network.

It should be noted that this property is not applicable to the modular shift-register generator with the reciprocal polynomial although the output at the last stage produces the identical pseudorandom sequence.

We are now able to determine A_1 which is equal to B_p,

$$B_p = \sum_{i=0}^{n-1} b_i B_i.$$

Let us consider the following examples.

Example 1:.

$n = 3$

$P_A(x) = x^3 + x + 1$

(i) $k = 2^2 = 4$

$P_B(x) = P_A(x) = x^3 + x + 1$

and $p = 2^{n-m} = 2$

$$B_p = \sum_{i=0}^{2} b_i B_i = B_2$$

The circuit to calculate B_p is shown in Figure 3.24.

(ii) $k = 3$

$P_B(x) = x^3 + x^2 + 1$

Solve for p in

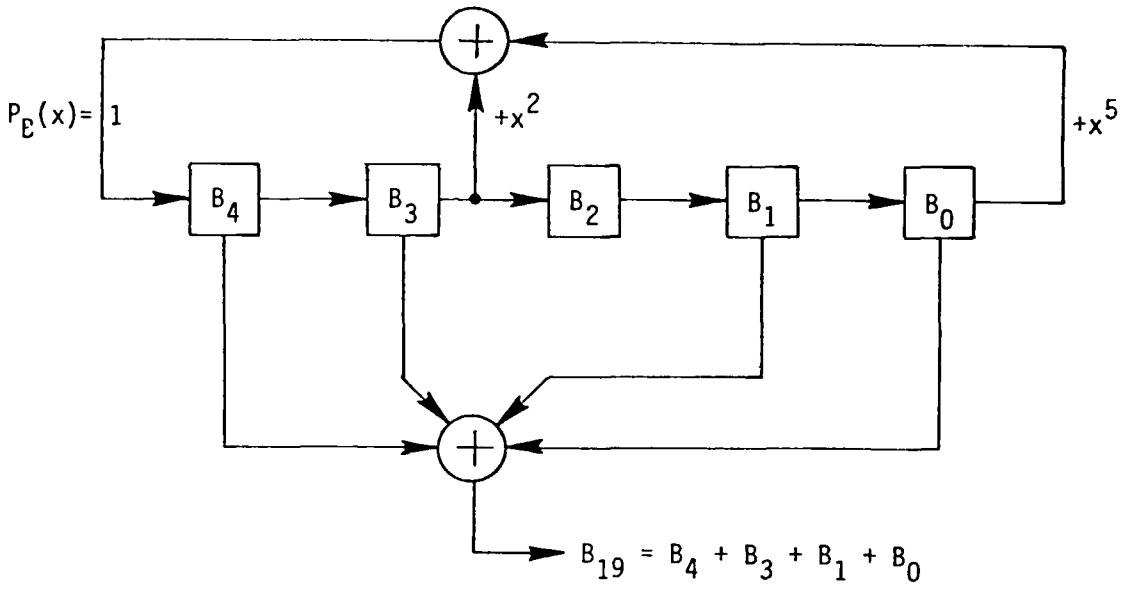


Figure 3.23 PHASE SHIFT NETWORK

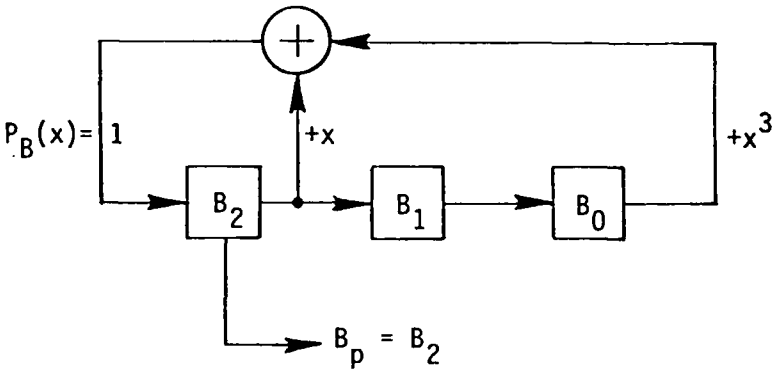


Figure 3.24 PHASE SHIFT NETWORK
(n = 3, k = 4)

$$kp = 1 \pmod 7$$

$$p = 5$$

$$P_B^*(x) = x^3 + x + 1$$

$$P_B^*(\beta) = \beta^3 + \beta + 1 = 0$$

$$\beta^3 = \beta + 1$$

$$\beta^4 = \beta\,\beta^3 = \beta(\beta + 1) = \beta^2 + \beta$$

$$\beta^5 = \beta\,\beta^4 = \beta(\beta^2 + \beta) = \beta^3 + \beta^2$$

$$= \beta + 1 + \beta^2$$

$$= \beta^2 + \beta + 1$$

$$B_5 = B_2 + B_1 + B_0$$

The circuit for calculating B_p is shown in Figure 3.25.

Example 2.

$$n = 4$$

$$P_A(x) = x^4 + x + 1$$

(i) $k = 2^2 = 4$

$$P_B(x) = P_A(x) = x^4 + x + 1$$

$$P = 2^{n-m} = 4$$

$$P_B^*(x) = x^4 + x^3 + 1$$

$$P_B^*(\beta) = \beta^4 + \beta^3 + 1 = 0$$

$$\beta^4 = \beta^3 + 1$$

$$B_p = B_4 = B_3 + B_0$$

The circuit for calculating B_p is shown in Figure 3.26.

(iii) $k = 7$

$$P_B(x) = x^4 + x^3 + 1$$

$$7p = 1 \pmod{15}$$

$$p = 13 \quad [13 \times 7 = 91 = 6 \times 15 + 1]$$

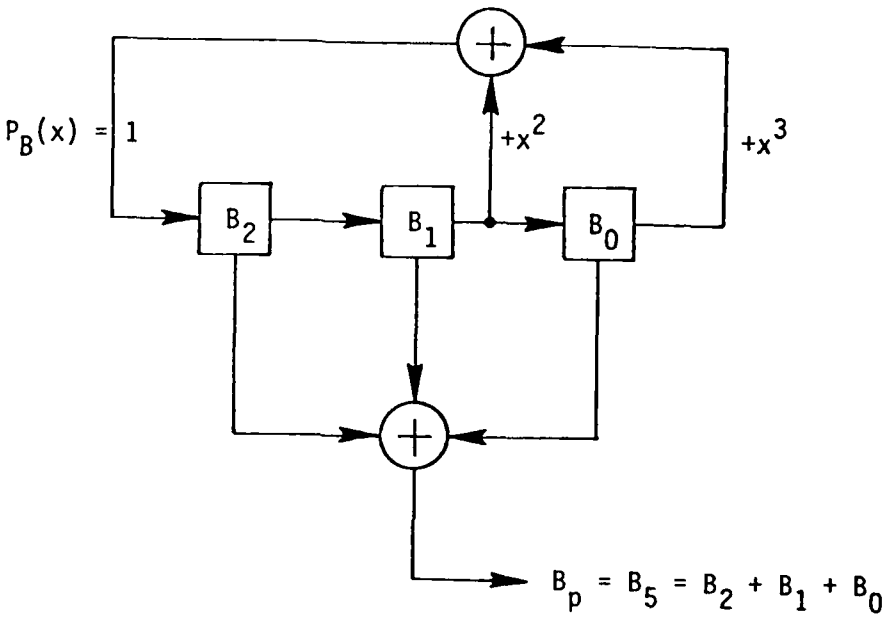


Figure 3.25 PHASE SHIFT NETWORK
(n = 3, k = 5)

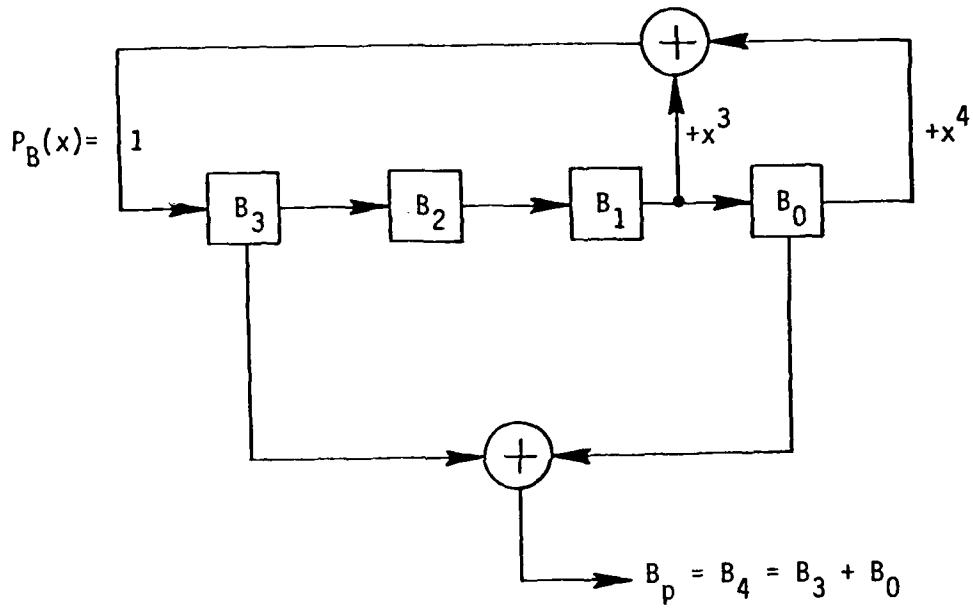


Figure 3.26 PHASE SHIFT NETWORK
(n = 4, k = 4)

$$P_B^*(x) = x^4 + x + 1$$

$$P_B^*(\beta) = \beta^4 + \beta + 1 = 0$$

$$\beta^4 = \beta + 1$$

$$\beta^5 = \beta^2 + \beta$$

$$\beta^6 \quad \beta^3 + \beta^2$$

$$\beta^7 = \beta^4 + \beta^3$$

$$= \beta^3 + \beta + 1$$

$$\beta^8 = \beta^4 + \beta^2 + \beta$$

$$= \beta^2 + 1$$

$$\beta^9 = \beta^3 + \beta$$

$$\beta^{10} = \beta^4 + \beta^2$$

$$= \beta^2 + \beta + 1$$

$$\beta^{11} = \beta^3 + \beta^2 + \beta$$

$$\beta^{12} = \beta^4 + \beta^3 + \beta^2$$

$$= \beta^3 + \beta^2 + \beta + 1$$

$$\beta^{13} = \beta^4 + \beta^3 + \beta^2 + \beta$$

$$= \beta^3 + \beta^2 + 1$$

$$B_p = B_{13} = B_3 + B_2 + B_0$$

The circuit for calculating B_p is shown in Figure 3.27.

Example 3.

$$n = 5$$

$$P_A(x) = x^5 + x^2 + 1$$

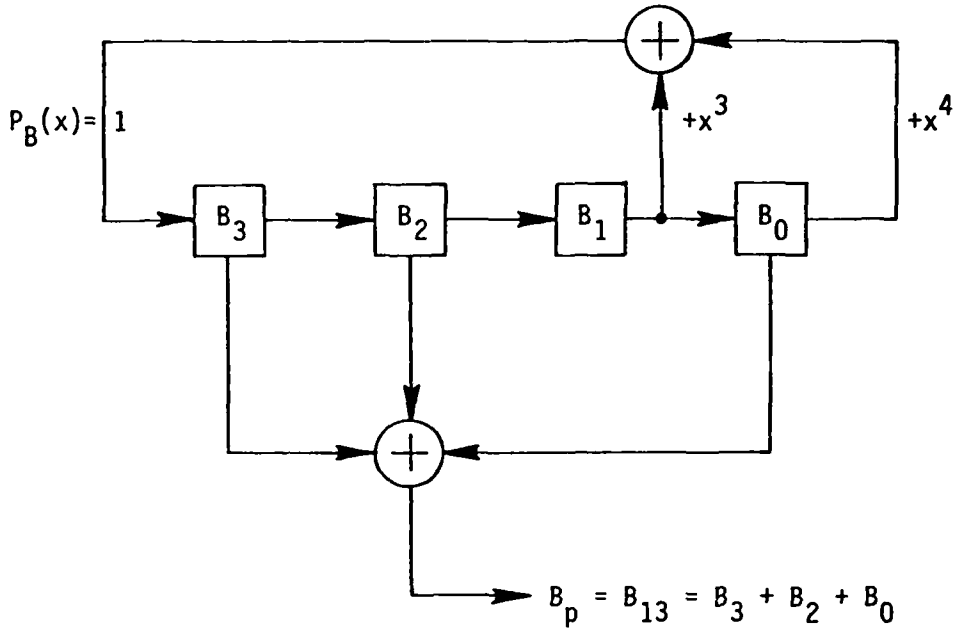


Figure 3.27 PHASE SHIFT NETWORK
(n = 4, k = 7)

(i) $k = 2 = 4$

$$P_B(x) = P_A(x) = x^5 + x^2 + 1$$

$$p = 2^{n-m} = 2^{5-2} = 8$$

$$P_B^*(x) = x^5 + x^3 + 1$$

$$P_B^*(\beta) = \beta^5 + \beta^3 + 1 = 0$$

$$\beta^5 = \beta^3 + 1$$

$$\beta^6 = \beta^4 + \beta$$

$$\beta^7 = \beta^5 + \beta^2$$

$$= \beta^3 + \beta^2 + 1$$

$$\beta^8 = \beta^4 + \beta^3 + \beta$$

$$B_p = B_8 = B_4 + B_3 + B_1$$

The circuit for calculating B_p is shown in Figure 3.28.

(ii) $k = 5$

$$P_B(x) = x^5 + x^4 + x^2 + x + 1$$

$$5p = 1 \pmod{31}$$

$$p = 25 \quad [5 \times 25 = 125 = 4 \times 31 + 1]$$

$$P_B^*(x) = x^5 + x^4 + x^3 + x + 1$$

$$P_B^*(\beta) = \beta^5 + \beta^4 + \beta^3 + \beta + 1 = 0$$

$$\beta^5 = \beta^4 + \beta^3 + \beta + 1$$

$$\beta^6 = \beta^5 + \beta^4 + \beta^2 + \beta$$

$$= \beta^3 + \beta^2 + 1$$

$$\beta^{12} = \beta^6 + \beta^4 + 1$$

$$= \beta^4 + \beta^3 + \beta^2$$

$$\beta^{24} = \beta^8 + \beta^6 + \beta^4$$

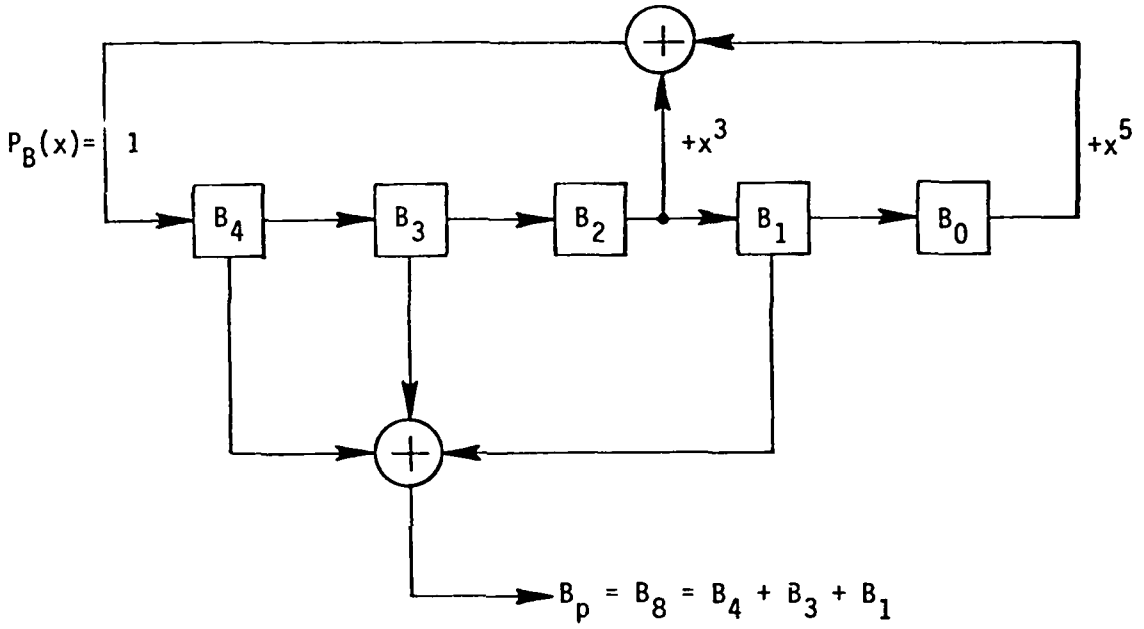


Figure 3.28 PHASE SHIFT NETWORK
($n = 5$, $k = 4$)

$$= \beta^2(\beta^3 + \beta^2 + 1) + \beta^3 + \beta^2 + 1 + \beta^4$$

$$= \beta^4 + \beta^3 + \beta + 1 + \beta^4 + \beta^2 + \beta^3 + \beta^2 + 1 + \beta^4$$

$$= \beta^4 + \beta$$

$$\beta^{25} = \beta^4 + \beta^3 + \beta + 1 + \beta^2$$

$$= \beta^4 + \beta^3 + \beta^2 + \beta + 1$$

$$B_p = B_{25} = B_4 + B_3 + B_2 + B_1 + B_0$$

The circuit for calculating B_p is shown in Figure 3.29.

3.3.6 Calculation of $B_{2p}, B_{3p}, \ldots, B_{(n-1)p}$

In Section 3.3.5 we have determined the coefficients b_i for calculating

$$B_m = \sum_{i=0}^{n-1} b_i B_i \quad \text{for all } m \leq L$$

and we have, in particular, found the coefficients for B_p. These coefficients enable us to construct a connection network, which is called the phase-shift network, to obtain

$$A_1 = B_p$$

directly from the sampled digits of the original pseudorandom sequence.

In principle we can use the same procedure to construct a phase shift network for each component of the initial state. However, a close observation reveals that the connection network already obtained can calculate the remaining components, $A_2 = B_{2p}, \ldots, A_{n-1} = B_{(n-1)p}$, as well.

Given

$$B_p = \sum_{i=0}^{n-1} b_i B_i$$

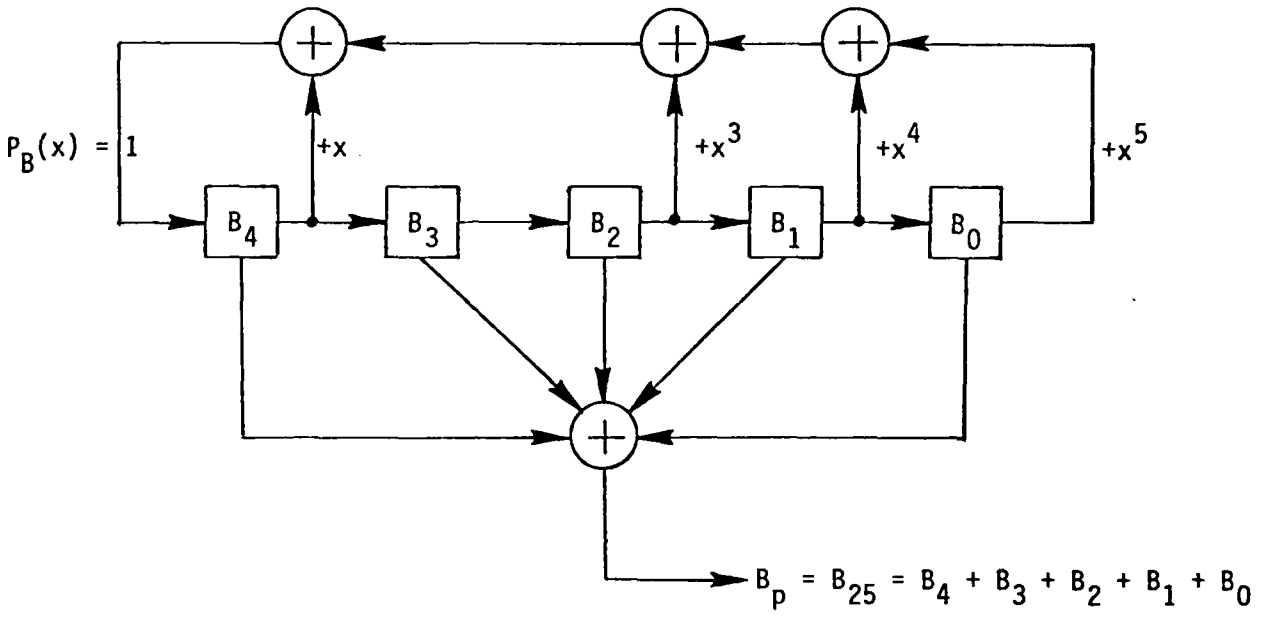


Figure 3.29 PHASE SHIFT NETWORK
(n = 5, k = 5)

which corresponds to

$$\beta^p = \sum_{i=0}^{n-1} b_i \beta^i$$

where B_p is the p-th digit of the pseudorandom-sequence generated by the polynomial $P_B(x)$ and β^p is the p-th power of a field element, β, which is a root of $P_B^*(x)$, the reciprocal polynomial of $P_B(x)$, B_{2p} can be calculated as follows:

$$\beta^{2p} = \beta^p \cdot \beta^p = \beta^p \sum_{i=0}^{n-1} b_i \beta^i$$

$$= \sum_{i=0}^{n-1} b_i \beta^{p+i}$$

which corresponds to

$$B_{2p} = \sum_{i=0}^{n-1} b_i B_{p+i} .$$

Using a similar derivation, we obtain

$$B_{jp} = \sum_{i=1}^{n-1} b_i B_{(j-1)p+i}, \quad j = 2,3,\ldots,n-1.$$

We now want to show how B_{jp} can be calculated through a circuit implementation. Let us consider the following examples.

Example 1.

$n = 3$, $k = 3$, $p = 5$

$P_B(x) = x^3 + x^2 + 1$

and $B_p = B_5 = B_2 + B_1 + B_0$

Construct the pseudorandom-sequence generator and the phase shift network as in Figure 3.30. The digits B_0, B_1, and B_2 are in the shift register at

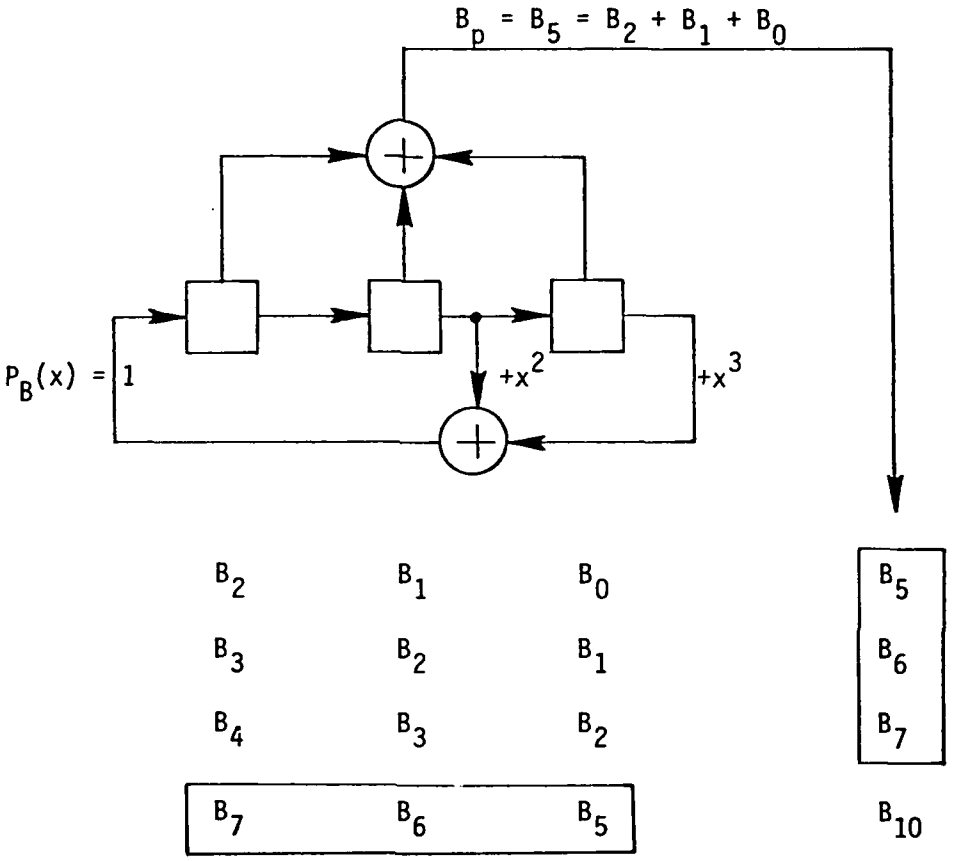


Figure 3.30 INITIAL-STATE CALCULATION
(n = 3, k = 5)

the instant when sync loss is detected. Then

$$A_0 = B_0$$

and $A_1 = B_5$ throught the connection network.

$$A_2 = B_{2p} = B_{10} = B_{5+2} + B_{5+1} + B_5$$

$$= B_7 + B_6 + B_5$$

These digits B_7, B_6, and B_5 are obtained at the output of the phase-shift network after each shift of the pseudorandom sequence generator.

Example 2

For $n = 4$ and $k = 7$.

$$P_B(x) = x^4 + x^3 + 1$$

$$kp = 1 \pmod{2^n-1} \Rightarrow p=13$$

and $$B_p = B_{13} = B_3 + B_2 + B_0$$

$$B_{2p} = B_{26} = B_{p+3} + B_{p+2} + B_p = B_{16} + B_{15} + B_{13}$$

or $$B_{2p} = B_{26} = B_{11} = B_1 + B_0 + B_{13}$$

Similarly,

$$B_{3p} = B_{39} = B_9 = B_{14} + B_{13} + B_{11}$$

The principle of operation is illustrated in Figure 3.31.

Example 3

For $n = 5$ and $k = 6$

$$P_B(x) = x^5 + x^4 + x^3 + x + 1$$

$$p = 25$$

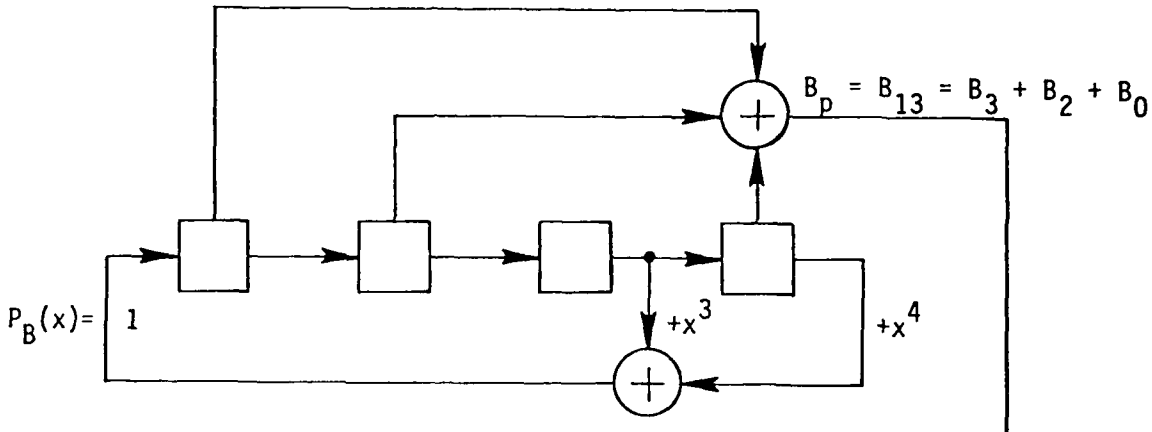
and $$B_p = B_{25} = B_4 + B_3 + B_2 + B_1 + B_0$$

$$B_{2p} = B_{50} = B_{19} = B_{29} + B_{28} + B_{27} + B_{26} + B_{25}$$

$$B_{3p} = B_{75} = B_{13} = B_{23} + B_{22} + B_{21} + B_{20} + B_{19}$$

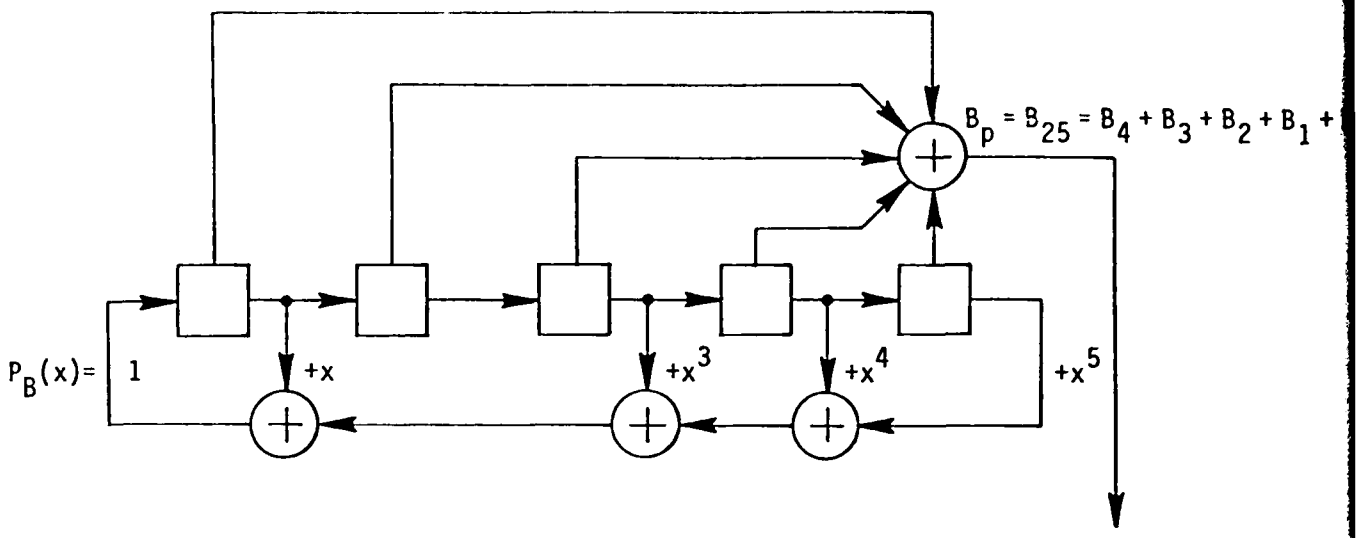
$$B_{4p} = B_{100} = B_7 = B_{17} + B_{16} + B_{15} + B_{14} + B_{13}$$

The principle of operation is illustrated in Figure 3.32.



B_3	B_2	B_1	B_0	B_{13}
B_4	B_3	B_2	B_1	B_{14}
B_5	B_4	B_3	B_2	$B_{15} = B_0$
B_6	B_5	B_4	B_3	$B_{16} = B_1$
B_1	B_0	B_{14}	B_{13}	$B_{26} = B_{11}$
B_2	B_1	B_0	B_{14}	$B_{27} = B_{12}$
B_3	B_2	B_1	B_0	$B_{28} = B_{13}$
B_4	B_3	B_2	B_1	$B_{29} = B_{14}$
B_{14}	B_{13}	B_{12}	B_{11}	$B_{39} = B_9$

Figure 3.31 INITIAL-STATE CALCUALTION
(n = 4, k = 13)



B_4	B_3	B_2	B_1	B_0	B_{25}
B_5	B_4	B_3	B_2	B_1	B_{26}
B_6	B_5	B_4	B_3	B_2	B_{27}
B_7	B_6	B_5	B_4	B_3	B_{28}
B_8	B_7	B_6	B_5	B_4	B_{29}
B_{29}	B_{28}	B_{27}	B_{26}	B_{25}	$B_{50} = B_{19}$
B_{30}	B_{29}	B_{28}	B_{27}	B_{26}	B_{20}
B_0	B_{30}	B_{29}	B_{28}	B_{27}	B_{21}
B_1	B_0	B_{30}	B_{29}	B_{28}	B_{22}
B_2	B_1	B_0	B_{30}	B_{29}	B_{23}
B_{23}	B_{22}	B_{21}	B_{20}	B_{19}	$B_{75} = B_{13}$
B_{24}	B_{23}	B_{22}	B_{21}	B_{20}	B_{14}
B_{25}	B_{24}	B_{23}	B_{22}	B_{21}	B_{15}
B_{26}	B_{25}	B_{24}	B_{23}	B_{22}	B_{16}
B_{27}	B_{26}	B_{25}	B_{24}	B_{23}	B_{17}
B_{17}	B_{16}	B_{15}	B_{14}	B_{13}	$B_{100} = B_7$

Figure 3.32 INITIAL-STATE CALCULATION
(n = 5, k = 5)

Following the above examples, we can describe the principle of the operation of the essential part of the sync recovery circuit as follows.

The receiver receives and stores n sampled digits of the original pseudorandom sequence in an n-stage shift register; at the instant when sync loss is detected, the feedback connections corresponding to $P_B(x)$ are activated and the shift register becomes a pseduorandom-sequence generator generating the B sequence which is the sampled sequence corresponding to original sequence. At the same time the phase shift network is also activated. The values of $A_0=B_0$ and $A_1=B_p$ are thus obtained immediatelv.

At each clock time another digit, B_{p+i}, appears at the output of the phase-shift network. These digits, B_p, B_{p+1}....B_{p+n-1} are to be stored in an n-stage shift register buffer. At the (n-1)-st clock time the buffer is filled up, the contents of the pseudorandom sequence generator $P_B(x)$ are cleared, and replaced by the contents of the buffer. The value of $A_2=B_{2p}$ now appears at the output or the phase shift network. Repeat this procedure n-2 times in order to obtain the entire initial state:

$$A_0, A_1, A_2,\ldots,A_{n-1}.$$

Load this initial state into the pseudorandom sequence generator $P_A(x)$ of the original pseudorandom sequence, and speed up the clock to recover the sync.

3.3.7 Determination of the Phase Shift Network

The connections of the phase shift network are constructed according to the coefficients b_i in

$$B_p = \sum_{i=0}^{n-1} b_i B_i$$

or

$$\beta^p = \sum_{i=0}^{n-1} b_i \, \beta^i.$$

Since β is a root of the reciprocal polynomial $P_B^*(x)$,

$$P_B^*(\beta) = 0$$

and the degree of the polynomial $P_B^*(x)$ is n and β^n is a linear combination of the terms of degree less than or equal to (n-1). Therfore β^p can be calculated by successive application of the recursion formula

$$\beta^{j+1} = \beta \cdot \beta^j$$

and successive reduction of the degree of the term β^n.

This procedure is simple in concept, but very lengthy and tedious in practice, and almost impossible if n is very large. We shall present a much shorter method by considering the following two examples.

Example 1.

$n = 11$, $k = 13$

Choose $P_A(x) = x^{11} + x^2 + 1$.

Since $k = 13 \neq 2^m$, for $m < n = 11$

$$P_B(x) \neq P_A(x)$$

and $P_B(x)$ has a root which is the 13-th power of a root of $P_A(x)$. From the Table of Irreducible Polynomials, it is found that

$$P_B(x) = x^{11} + x^6 + x^5 + x + 1 .$$

The reciprocal polynomial of $P_B(x)$ is

$$P_B^*(x) = x^{11} + x^{10} + x^6 + x^5 + 1.$$

Next, solve for p from

$$kp = 1 \pmod{2047}, \quad L = 2^{11} - 1 = 2047$$

$$13 \times 315 = 4095 - 2 \times 2047 + 1$$

yielding $p = 315$.

Hence $A_0 = B_0$

and $A_1 = B_{315}$.

We need to find an expression for B_{315} in terms of $B_0, B_1, B_2, \ldots, B_{10}$.

The next step is to write a computer program simulating the linear feedback shift register in an MSRG configuration according to $P_B^*(x) = x^{11} + x^{10} + x^6 + x^5 + 1$ as shown in Figure 3.33 with

$$\beta^0 = 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0$$
$$\beta = 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0$$
$$\beta^2 = 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0$$
$$\vdots$$
$$\beta^{10} \quad 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1$$

and calculate β^{315} according to

$$\beta^{315} = \left(\left(\left(\beta^{39}\right)^2\right)^2\right)^2 \cdot \beta^3$$
$$= \left(\left(\beta^{78}\right)^2\right)^2 \cdot \beta^3$$
$$= \left(\beta^{156}\right)^2 \cdot \beta^3$$
$$= \beta^{312} \cdot \beta^3.$$

(a) Calculate $\beta^{11}, \beta^{12}, \ldots, \beta^{39}$ resulting in

$$\beta^{39} = b_0 + b_1\beta + b_2\beta^2 + \ldots + b_{10}\beta^{10}.$$

(b) Calculate $\beta^{39}, \beta^{39+1}, \beta^{39+2}, \ldots, \beta^{39+10}$ and

$$\beta^{78} = b_0\beta^{39} + b_1\beta^{39+1} + b_2\beta^{39+2}, \ldots, b_{10}\beta^{39+10}.$$

(c) Repeat (b) until

$$\beta^{312} = b_0\beta^{4x39} + b_1\beta^{4x39+1} + b_2\beta^{4x39+2} + \ldots + b_{10}\beta^{4x39+10}.$$

(d) Calculate $\beta^{313}, \beta^{314}, \beta^{315}$ obtaining

$$\beta^{315} = \beta^{10} + \beta^7 + \beta^2 + \beta + 1.$$

The sync recovery circuit is constructed as in Figure 3.34.

Example 2.

$n = 39, k = 8$

Choose $P_A(x) = x^{39} + x^4 + 1$

Since $k = 8 = 2^3$

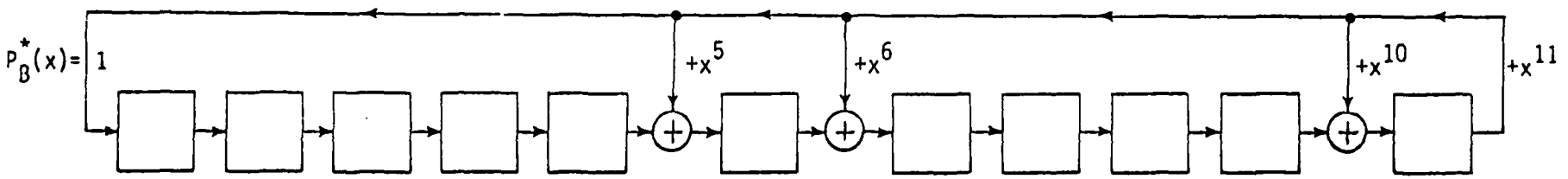


Figure 3.33 MODULAR SHIFT REGISTER GENERATOR,
$P_B(x) = x^{11} + x^{10} + x^6 + x^5 + 1$

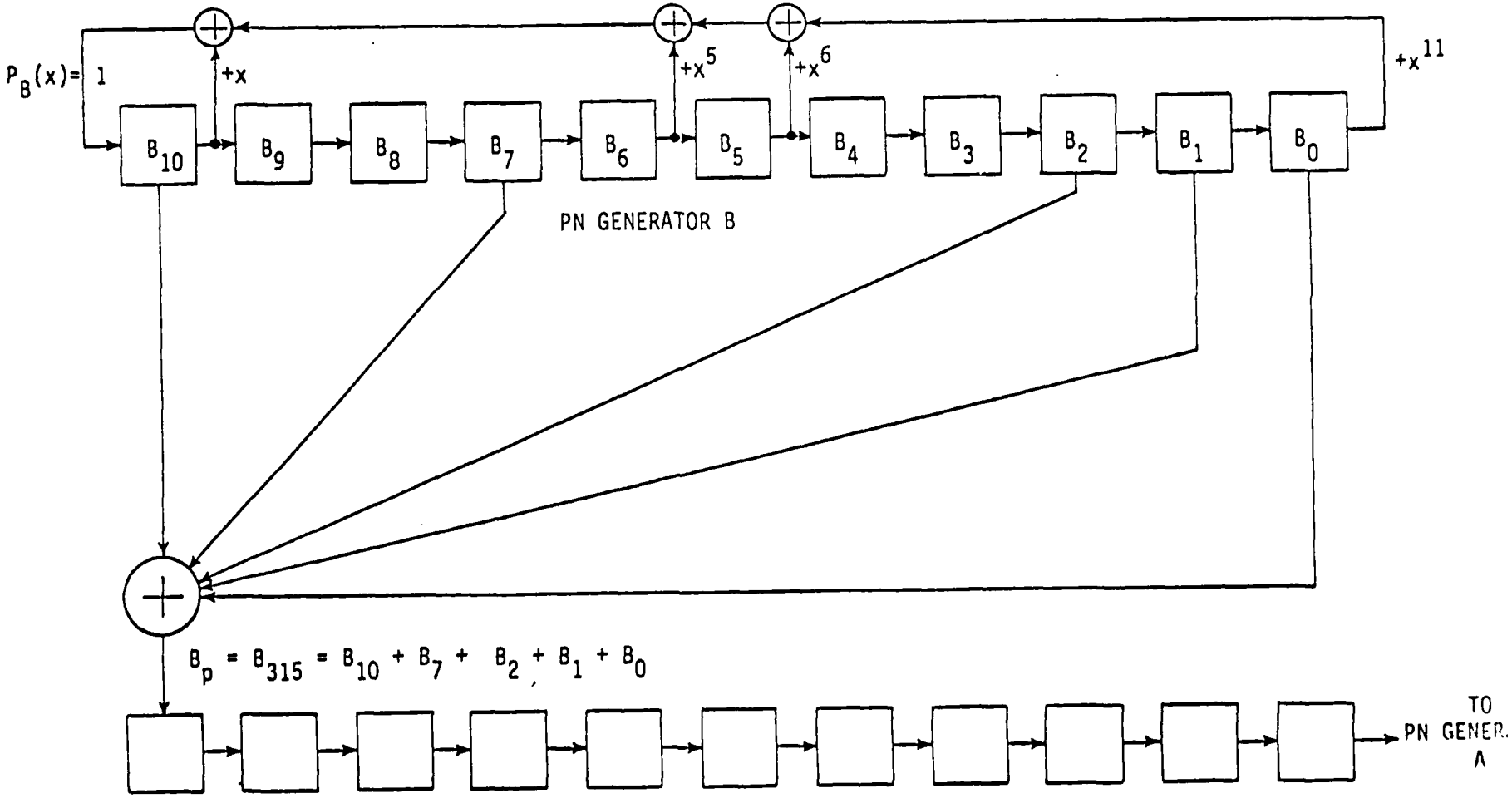


Figure 3.34 SYNC RECOVERY CIRCUIT

$$P_B(x) = P_A(x) = x^{39} + x^4 + 1.$$

The reciprocal polynomial of $P_B(x)$ is

$$P_B^*(x) = x^{39} + x^{35} + 1.$$

Solve for p from

$$kp = 1 \pmod{2^{39} - 1}$$

giving $p = 2^{39-3} = 2^{36}$.

We need to determine $\beta^{2^{36}}$ in terms of β^0, β^1, β^2,...,β^{38}. First we write a computer program simulating the modular shift register for $P_B^*(x)$ as shown in Figure 3.35 with

$$\beta^0 = 1\ 0\ 0\ 0 \ldots\ldots\ldots\ldots 0$$

$$\beta = 0\ 1\ 0\ 0 \ldots\ldots\ldots\ldots 0$$

$$\beta^2 = 0\ 0\ 1\ 0 \ldots\ldots\ldots\ldots 0$$

$$\vdots$$

$$\beta^{38} = \underbrace{0\ 0\ 0\ 0 \ldots\ldots\ldots\ldots 01.}_{39 \text{ digits}}$$

The calculation procedure is as follows.

(a) Calculate β^{39}, β^{40},...,β^{64} and

$$\beta^{64} = b_0 + b_1\beta + b_2\beta^2 + \ldots + b_{38}\beta^{38}.$$

(b) Calculate β^{64}, β^{64+1}, β^{64+2},...,β^{64+38} and

$$\beta^{128} = \beta^2 = b_0\beta^{64} + b_1\beta^{64+1} + b_2\beta^{64+2} + \ldots + b_{38}\beta^{64+38}.$$

(c) Repeat (b).

$$\beta^{2^8} = b_0\beta^{2^7} + b_1\beta^{2^7+1} + b_2\beta^{2^7+2} + \ldots + b_{38}\beta^{2^7+38}$$

$$\vdots$$

$$\beta^{2^{36}} = b_0\beta^{2^{35}} + b_1\beta^{2^{35}+1} + b_2\beta^{2^{35}+2} + \ldots + b_{38}\beta^{2^{35}+38}$$

(d) The solution is

$$\beta^{2^{36}} = \beta^{24} + \beta^{22} + \beta^5.$$

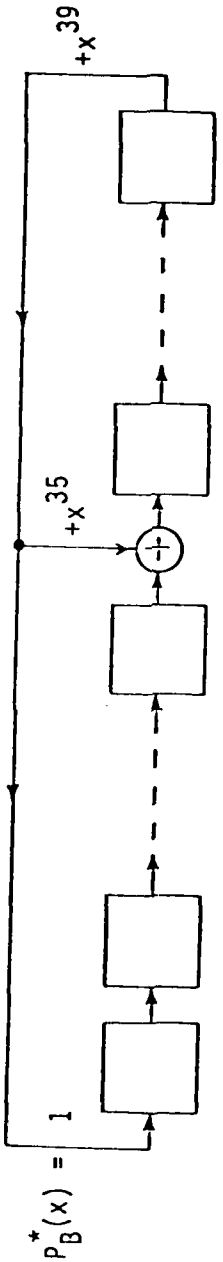


Figure 3.35 MODULAR SHIFT REGISTER FOR $P_B^*(x)$

Therefore the sync recovery circuit is constructed according to Figure 3.36. From the two examples we can appreciate the powerfulness of the method, especially in Example 2. We can roughly estimate the number of calculations:

	initial calculation (step a) = 64 - 38 =	26
+	(36-6) x 38 =	1,140
	Total	1,166.

Otherwise the number will be approximately $2^{26} = 6.8 \times 10^{10}$ calculations.

3.3.8 Functional Block Diagram of a Sync-Recovery Circuit

The block diagram of a sync recovery circuit for the specific case of Example 1 in Section 3.3.7 is shown in Figure 3.37. In this case

$$n = 11,\ k = 13,\ \text{and}\ L = 2047$$

$$P_A(x) = x^{11} + x^2 + 1$$

$$P_B(x) = x^{11} + x^6 + x^5 + x + 1$$

and $$B_p = B_{315} = B_{10} + B_7 + B_2 + B_1 + B_0.$$

The feedback connections for PN generator A are omitted in the diagram; but the feedback connections and the phase-shift network connections for PN generator B and the lower buffer register are redrawn and shown in Figure 3.38.

In the normal operating mode, all the connections on PN generator B are disconnected and the generator acts as a buffer register receiving a B-digit every 13 bit-rate clock times. At the instant when sync loss is detected, the circuit enters into the sync recovery mode and the clock is changed to the "recovery rate" which is much faster than the normal bit rate clock. The recovery clock rate will be discussed in a later section. For convenience in the discussion we set the time as t=0. The recovery circuit functions as follows.

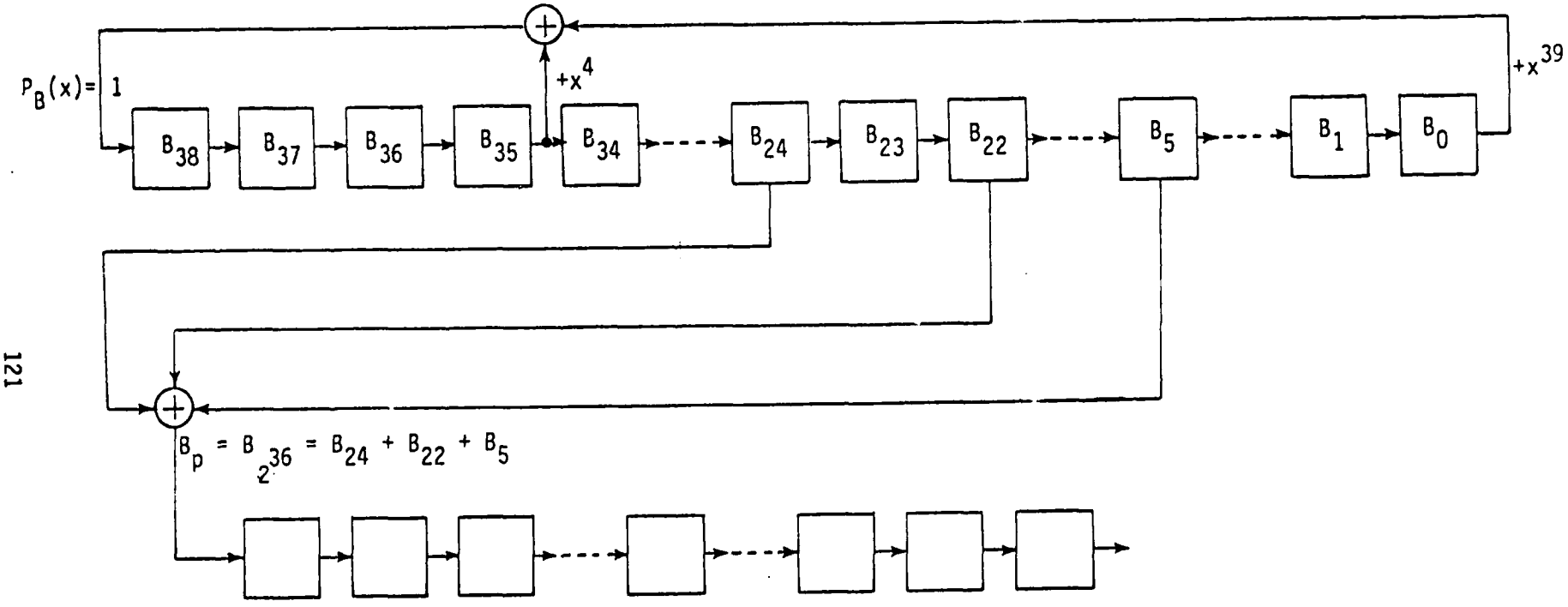


Figure 3.36 SYNC RECOVERY CIRCUIT

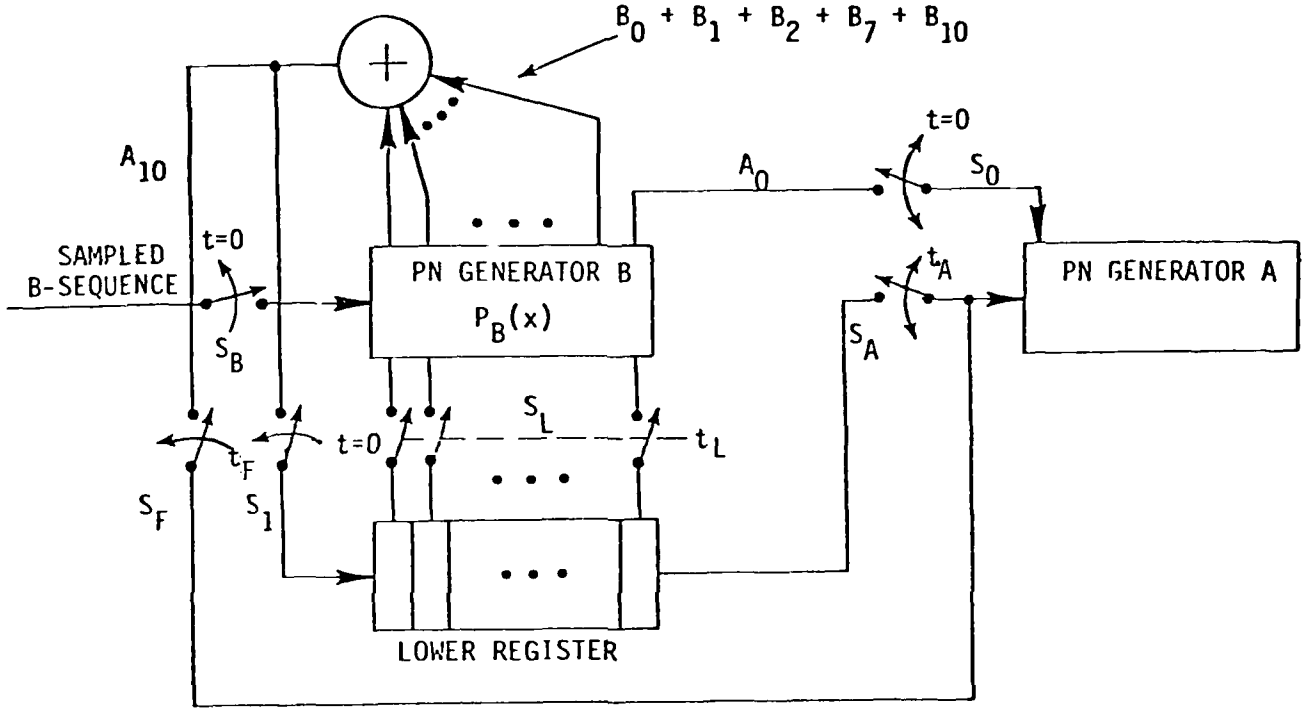


Figure 3.37 BLOCK DIAGRAM OF SYNC RECOVERY CIRCUIT

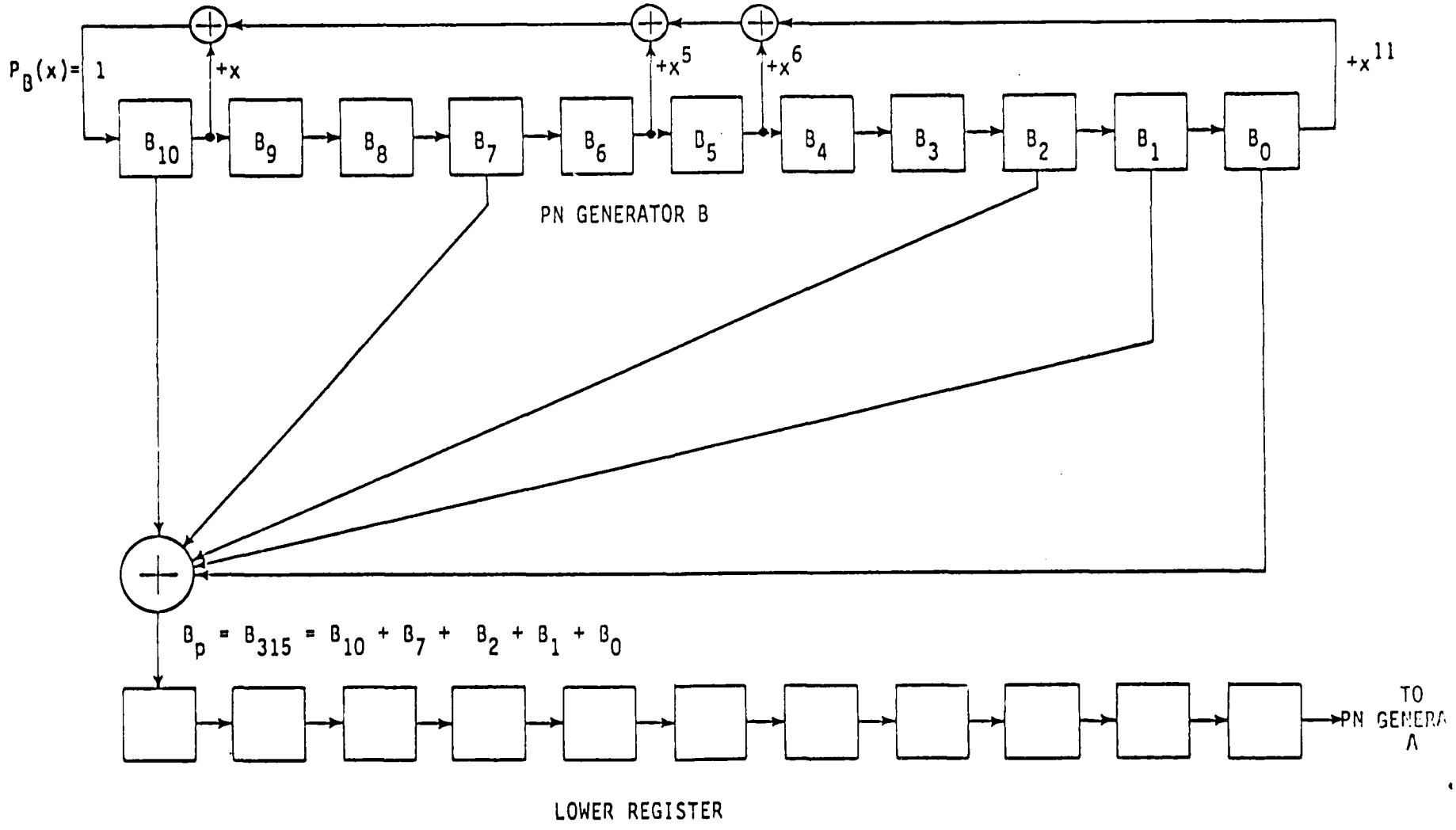


Figure 3.38 PN GENERATOR B AND LOWER REGISTER

1. At $t = 0$

(a) Switch S_B opens, the content of the last stage of the generator is designated as B_0 and that of the rest of the stages as B_1, B_2,... respectively and that of the first stage as B_{10}. The digit $A_0 = B_0$ is ready at the last stage and $A_1 = B_p$ appears at the output of the phase-shift network.

(b) Switch S_0 closes to let A_0 enter PN generator A or a buffer register to be loaded into generator A later. Switch S_0 opens then and remains open throughout the recovery mode.

(c) Switch S_1 closes to let $A_1 = B_p$ enter the first stage of the lower buffer register. Switch S_1 remains closed throughout the recovery mode. At each clock time one digit, B_{jp+i}, $i = 0, 1, 2,...,10$ and $j = 1, 2,...,9$, enters the lower register.

2. At $t = t_L$, $t_L = 10j$, $j = 1, 2,...,9$, the lower buffer register is filled up with the following contents:

$$B_{jp+10},\ B_{jp+9},\cdots B_{jp+1},\ B_{jp} = A_j.$$

Generator B is cleared and switch S_L closes allowing the contents of the lower buffer register to be loaded into generator B.

3. At $t = t_A$ where $t_A = t_L = 10j$, switch S_A closes to let $A_j = B_{jp}$ shift into generator A. Switch S_A opens until j increments.

4. At $t = t_F$, $t_F = 90$, A_{10} appears at the output of the phase-shift network; and switch S_F closes to let A_{10} enter generator A. The initial state to generator A is filled up and generator A begins to shift until sync is caught up.

3.3.9 Estimation of Sync-Recovery Time

When a declaration of sync loss is made, let i_L be the number of bit times elapsed after the most recent B-digit, namely B_{n-1}, entered generator B and let i_R be the corresponding number when sync is recovered, as illustrated in the recovery timing diagram in Figure 3.39.

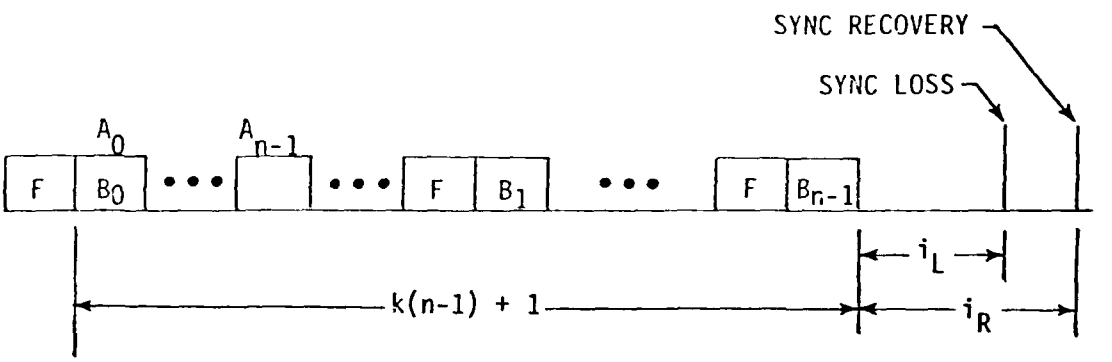


Figure 3.39 RECOVERY TIMING DIAGRAM

It will take generator A

$$k(n-1) + 1 + i_R = 131 + i_R$$

shifts to catch up after the initial state has been determined and it will take

$$(n-1)(n-2)$$

shifts for generator B to determine the initial state. Thus the combined number of shifts required to recover the sync is

$$k(n-1) + 1 + (n-1)(n-2) + i_R$$

If recovery is required within no more than one bit period of the original bit-rate clock, then

$$i_R = i_L + 1$$

where i_L is the number of bit times elapsed after the most recent B-digit entered generator B.

$$0 \leq i_L \leq k-1,$$

and

$$1 \leq i_R \leq k.$$

Therefore, the total number of shifts required by both generator A and generator B to recovery the sync is

$$k(n-1) + 1 + (n-1)(n-2) + 1 \leq \text{Number of shifts} \leq k(n-1) + 1 + (n-1)(n-2) + k.$$

The recovery clock should be approximately

$$k(n-1) + 1 + (n-1)(n-2) + k = (k + n-3)\ n+3$$

times faster than the bit-rate clock. Thus, if R is the bit rate, then the recovery clock rate is

$$[n(k + n-3) + 3]R.$$

For large values of n and k, the recovery clock rate is approximately

$$n(k + n-3)R.$$

For the example we were discussing,

$$n = 11 \quad \text{and} \quad k = 13$$

the recovery clock is 234 times faster than the bit-rate clock. The bit rate of the Spectral Data Transfer System is on the order of kilobits per second and the recovery clock rate will be below megabits per second which is well within the state of art.

3.4 Preamble

3.4.1 Purpose of the Preamble

In the Spectral Data Transfer System, the receiver must establish a coherent carrier reference in order to demodulate the received signal. It must also establish a bit timing reference in order to provide sampling pulses to the data demodulator and to clock any baseband digital processing. Furthermore, frame synchronization must also be established in order to identify correctly the self-sychronization bits for the scrambler and the scrambled data.

In order to permit the receiver to perform these essential functions efficiently and effectively, each transmission from the transmitter begins with a special preamble. This preamble is designed to facilitate the receiver's synchronization tasks: carrier synchronization, bit synhronization, and initial frame and PN Generator synchronization.

3.4.1.1 Carrier Synchronization

The receiver employs a loop structure to track the noisy received carrier and generate a coherent local reference for the demodulation of the data. The exact details of the carrier-tracking structure will depend upon the type of modulation impressed upon the carrier by the transmitter. If the modulation suppresses the carrier, i.e. there is no spectral line at the carrier frequency, then a structure capable of generating a line related to the carrier frequency must be employed, rather than a simple tracking loop.

The two structures most commonly employed for carrier tracking or suppressed carrier tracking are, respectively, the phase-locked loop (PLL) and the Costas loop. These structures and their tracking performances

are treated thoroughly in the literature [8]-[18]. We will quote the results necessary for the design of the carrier recovery portion of the preamble.

The frequency of the received carrier may differ from the quiescent frequency of the VCO in the receiver's carrier tracker due to effects such as oscillator drift and Doppler shift. Therefore, the tracker must be able to lock in the presence of the maximum expected frequency offset. For a second order loop, which is often employed for carrier tracking, with a loop filter specified by

$$F(s) = K \frac{1 + as}{1 + bs} \tag{3-1}$$

the pull-in range $\Delta\omega$ is given approximately by [8, p. 364]:

$$\Delta\omega \simeq 2\omega_n \sqrt{\zeta\omega_n b - \frac{1}{2}} \tag{3-2}$$

where ω_n is the natural frequency of the loop.

The acquisition time T_{acq} is approximately [8, p. 364]:

$$T_{acq} \simeq \frac{(\Delta\omega)^2}{4\zeta\omega_n^3 - [(\Delta\omega)^2\omega_n^2/K] - (2\omega_n^4/K)} \simeq \frac{(\Delta\omega)^2}{2\zeta\omega_n^3} \text{ if } \omega_n/K \to 0 \tag{3-3}$$

where ζ is the loop damping factor and K is the amplitude scale factor of F(s) in (3-1). The time given by (3-3) is for pull-in to a phase error of $\pi/2$ radians. The additional time T_s to settle to a phase error $\varepsilon_{T\delta}$ in the absence of a frequency offset is approximately [8, p. 365]:

$$T_s = \frac{1}{2B_n} \ln \frac{2}{\varepsilon_{T\delta}} \tag{3-4}$$

where B_n is the one-sided noise bandwidth of the loop.

The preceding discussion has not considered the effects of noise on the acquisition process. The lack of analytical results on the

acquisition time of phase-locked loops is due largely to the inability to solve the nonlinear differential equations describing the loop. However, some results obtained by computer simulation have appeared in the literature [15], [17]. From these results the acquisition time (for 90% probability of acquisition) in the presence of noise with no frequency offset, can be approximated, for loop SNR $\geq$ 20 dB, by

$$T_{acq} \simeq \frac{10}{f_n} = \frac{2.99}{B_n} \tag{3-5}$$

where $f_n = \omega_n/2\pi$. If the loop SNR decreases to 10 dB, then T_{acq} approximately doubles.

The design of the preamble proceeds in the following manner. In order to maximize the carrier power (and hence loop SNR) available during the acquisition process, the first segment of the preamble will be an unmodulated carrier. The receiver designer will employ (3-2) as part of his design of the loop, taking into account the maximum anticipated frequency offset due to oscillator instabilities in both the transmitter and the receiver, Doppler shifts, etc. Commonly the factor ζ appearing in (3-2) is set to $\zeta = 1/\sqrt{2}$ as a good compromise between speed and stability of the loop [9, p. 54]. Once (3-2) and other design criteria are considered, a suitable ω_n will be chosen by the receiver designer. Then (3-5) and the results reported in [15] and [17] can be used to determine the duration of the unmodulated carrier which forms the first segment of the preamble.

3.4.1.2 Bit Synchronization

Once the receiver has acquired carrier synchronization, it is able to recover a baseband signal from the modulated incoming signal. The

next step is to synchronize the local bit-rate clock in frequency and time (epoch) with the bit transitions of the received signal. This process is commonly accomplished by a tracking loop structure, and, thus, is somewhat similar to the carrier synchronization process. However, the bit synchronizer locks to a line, related to the bit rate, in the baseband spectrum, whereas the carrier tracker operates on a line in the RF spectrum. (Either case may require a nonlinear operation to generate a suitable line to track.) Again, the subject of bit synchronization has been covered extensively in the literature [8], [10], [11], [13], [15], [19] and only the pertinent results will be mentioned here.

Two bit synchronizer structures which may be employed are the in-phase/mid-phase tracker shown in Figure 3.40 and the absolute-value early/late-gate tracker shown in Figure 3.41. The in-phase/mid-phase synchronizer has better noise tracking performance if the window (integration interval) is small (less than 1/4 of a bit period) and has a shorter pull-in time, but the early/late-gate synchronizer is less sensitive to D.C. offsets and is simpler to implement [8, p. 445].

Transition-tracking loops used for bit synchronization can be analyzed exactly as with phase-locked loops [13, p. 458]. Thus the results cited above for the carrier tracking loop may be applied to the design of the preamble for bit synchronization. There are, however, two differences to be considered. First, of course, the analysis must use the parameters of the bit synchronizer's loop. Second, the preamble segment for bit-synchronization acquisition will consist of a pattern yielding a maximum bit-transition density. For an NRZ-L baseband format

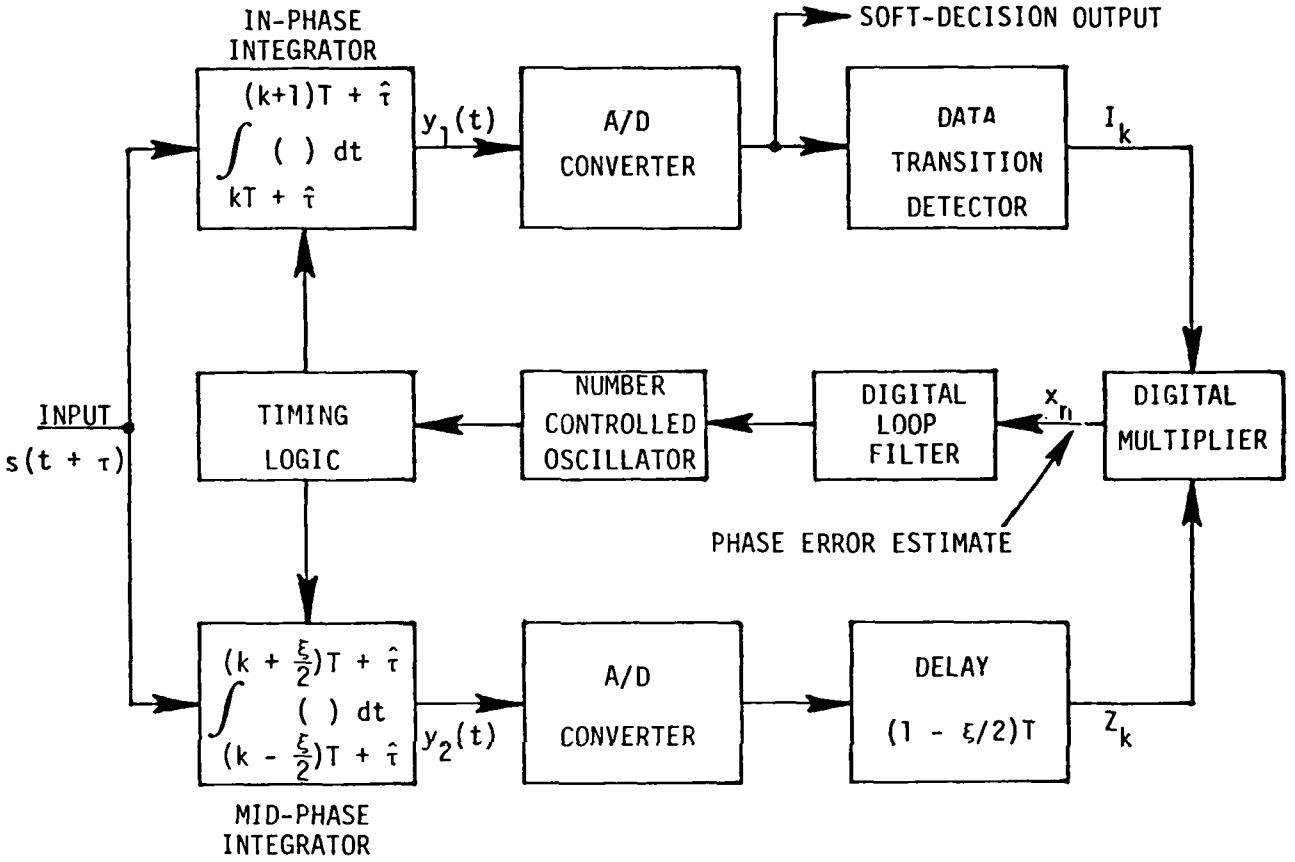


Figure 3.40 IN-PHASE/MID-PHASE BIT SYNCHRONIZER

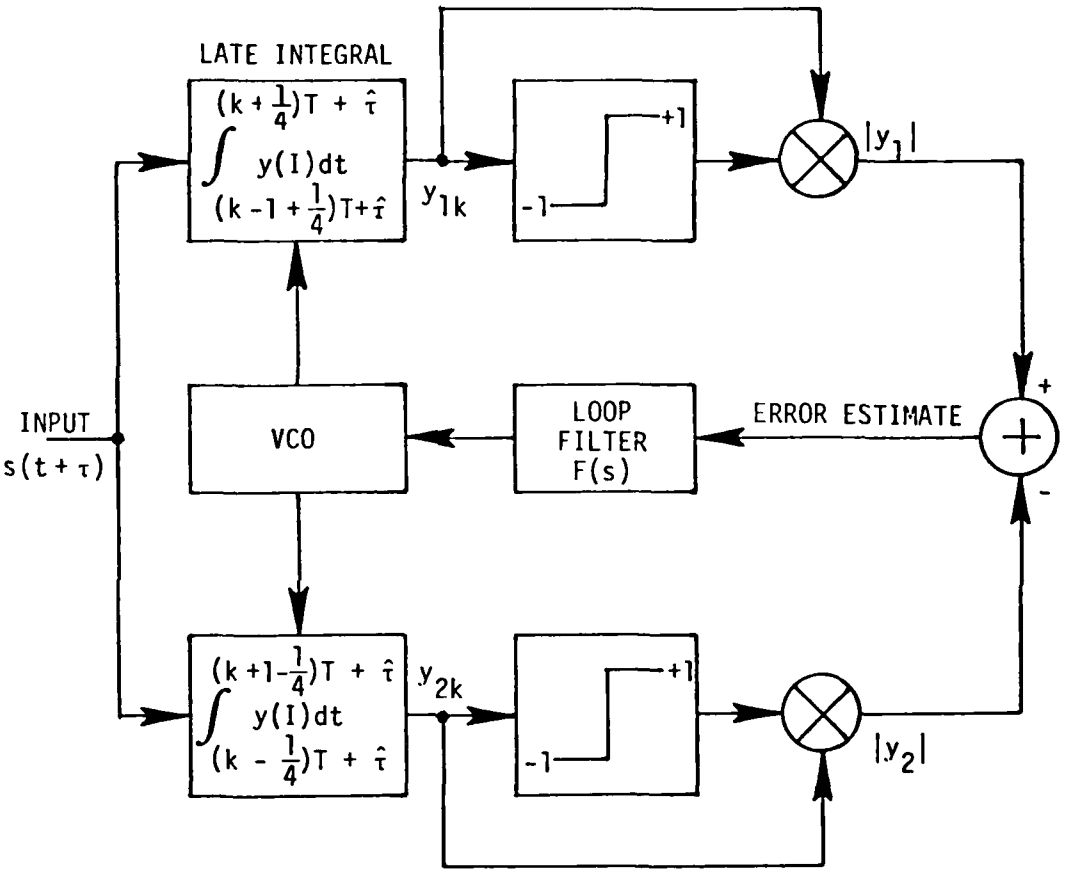


Figure 3.41 ABSOLUTE-VALUE EARLY/LATE-GATE BIT SYNCHRONIZER

the pattern would be alternating 0's and 1's, whereas for a differentially encoded (or NRZ-M) format the pattern would be all-1's. The pattern length will depend upon the loop SNR and the loop bandwidth of the bit synchronizer, but a few tens of bits typically would be expected to suffice.

3.4.1.3 Unique Word

The receiver must have some means of determining when the acquisition-aiding preamble concludes and data transmission begins, in order to initialize the frame synchronization and the PN Generator synchronization. Therefore, the final segment of the preamble is a unique word which can be detected by the receiver to mark the epoch at which the preamble ends and data transmission begins. In order to insure that the epoch determined by recognition of this word is the correct epoch, the word should have a high autocorrelation peak when properly aligned and low correlation when misaligned.

One class of words having such a property is the class known as Barker codes. These codes consist of a finite-length word of ±1 symbols and have a unique property that the autocorrelation function C_k, defined by

$$C_k = \sum_{i=1}^{N-k} x_i \, x_{i+k} \qquad (3\text{-}6)$$

where N is the word length, k is the shift, and x_i is a code symbol, satisfies $|C_k| \leq 1$ for $k \neq 0$.

The definition of the autocorrelation function C_k given in (3-6) yields an ideal shape (high peak for $k = 0$, uniform low value for $k \neq 0$) only if adjacent code symbols are assumed to be zero. If the

adjacent data bits are not zero, but rather are random ±1's (as must be the case in a binary system), then correlation "sidelobes" can arise from a partial match with the random data. The longer the unique word, the less is the probability of matching the random data, and thus the performance is improved. A list of all known Barker codes is given in Table 3.2, from which we see that the longest Barker code is 13 bits.* Codes not possessing the ideal autocorrelation properties of Barker codes, but of greater length, may discriminate better against random data. For example, a 100-bit sequence with $C_k \leq 2$, k = 1, 2, ..., 99, would provide better discrimination against random data than the longest-known 13-bit Barker code [20].

Neuman and Hofman [20] conducted an exhaustive computer search for sequences with good correlation properties for lengths up to 24 bits. The Neuman-Hofman codes are listed in Table 3.3. The maximum sidelobe of the autocorrelations of those codes is $C_k \leq 2$ for all codes except the first 24-bit code for which $C_k = 3$. The magnitudes of the autocorrelations, which are of interest if a sign ambiguity exists in the received data, are listed in Table 3.4 in the same order as the codes are listed in Table 3.3.

Massey [21] has investigated the probability of erroneous synchronization for optimum and correlation detection of Barker and Neuman-Hofman codes imbedded in binary data transmitted over the additive white Gaussian noise channel. His results, obtained by computer simulation, are summarized in Table 3.5.

In choosing a unique word for use in the spectral data transfer system, we desire the shortest preamble which will give adequate

*Proof exists that no Barker codes exists for length $14 < N < 6083$, and it is conjectured that Barker codes longer than N = 13 exist [20].

TABLE 3.2

BARKER CODES

LENGTH	CODE SEQUENCE*
1	+1
2	+1 +1
	+1 -1
3	+1 +1 -1
4	+1 +1 +1 -1
	+1 +1 -1 +1
5	+1 +1 +1 -1 +1
7	+1 +1 +1 -1 -1 +1 -1
11	+1 +1 +1 -1 -1 -1 +1 -1 -1 +1 -1
13	+1 +1 +1 +1 +1 -1 -1 +1 +1 -1 +1 -1 +1

*The complement code, the reflected code (reverse order), and the complement reflected code also are Barker codes.

TABLE 3.3

NEUMAN-HOFMAN CODES

LENGTH	CODE*
7**	+1 +1 +1 -1 -1 +1 -1
8	+1 +1 +1 +1 -1 -1 +1 -1
8	+1 +1 +1 -1 -1 -1 +1 -1
9	+1 +1 +1 +1 -1 -1 -1 +1 -1
9	+1 +1 -1 -1 -1 -1 -1 +1 -1
10	+1 +1 +1 +1 +1 -1 -1 +1 -1 +1
10	+1 +1 +1 +1 -1 -1 +1 -1 +1 -1
11	+1 +1 +1 +1 -1 +1 +1 +1 -1 -1 +1
11**	+1 +1 +1 -1 -1 -1 +1 -1 -1 +1 -1
12	+1 +1 +1 -1 -1 -1 -1 +1 -1 -1 +1 -1
12	+1 +1 -1 -1 +1 +1 +1 +1 +1 -1 +1 -1
13	+1 +1 +1 +1 +1 +1 -1 -1 +1 +1 -1 +1 -1
13	+1 +1 +1 +1 +1 -1 +1 -1 -1 +1 +1 -1 -1
14	+1 +1 +1 +1 -1 -1 +1 +1 -1 -1 +1 -1 +1 -1
14	+1 +1 -1 -1 +1 +1 -1 -1 -1 -1 -1 +1 -1 +1
15	+1 +1 -1 -1 -1 -1 -1 +1 +1 -1 -1 +1 -1 +1 -1
15	+1 +1 +1 +1 -1 -1 +1 +1 -1 +1 +1 -1 +1 -1 +1
16	+1 +1 +1 +1 +1 -1 -1 +1 +1 -1 -1 +1 -1 +1 -1 -1
16	+1 +1 +1 +1 -1 -1 -1 +1 -1 -1 -1 +1 -1 -1 +1 -1
17	+1 +1 +1 +1 -1 +1 -1 -1 +1 +1 -1 -1 -1 +1 -1 +1 -1
17	+1 +1 +1 +1 -1 -1 -1 -1 +1 -1 -1 +1 -1 -1 -1 +1 -1
18	+1 +1 +1 +1 -1 +1 -1 +1 -1 -1 +1 -1 -1 +1 +1 -1 -1 -1
18	+1 +1 -1 -1 +1 +1 -1 -1 -1 -1 -1 +1 -1 +1 +1 -1 +1 -1
19	+1 +1 +1 +1 -1 -1 -1 +1 +1 +1 -1 +1 +1 +1 -1 +1 +1 -1 +1
19	+1 +1 +1 -1 -1 -1 +1 -1 -1 -1 +1 -1 -1 +1 -1 -1 +1 -1 +1
20	+1 +1 +1 +1 +1 -1 +1 +1 -1 -1 +1 -1 +1 -1 +1 +1 -1 -1 -1 +1
20	+1 +1 +1 -1 +1 +1 +1 -1 -1 -1 -1 -1 +1 +1 -1 +1 -1 -1 +1 -1
21	+1 +1 +1 +1 +1 +1 -1 +1 -1 -1 -1 +1 -1 +1 +1 -1 -1 -1 +1 +1 -1
21	+1 +1 -1 -1 +1 -1 -1 +1 +1 +1 +1 -1 +1 +1 +1 +1 -1 +1 -1 +1 -1
22	+1 +1 1 -1 +1 +1 +1 -1 -1 -1 -1 -1 +1 +1 -1 -1 +1 -1 -1 +1 -1 +1
23	+1 +1 +1 +1 +1 +1 -1 +1 -1 +1 -1 -1 +1 +1 -1 -1 +1 -1 +1 +1 -1 -1 -1
23	+1 +1 +1 +1 +1 +1 -1 -1 -1 -1 +1 +1 -1 -1 +1 +1 -1 +1 +1 -1 +1 -1 +1
24	+1 +1 +1 -1 -1 -1 -1 -1 -1 +1 +1 -1 +1 +1 +1 +1 -1 -1 +1 +1 -1 +1 -1 +1
24	+1 +1 +1 +1 +1 -1 -1 -1 +1 +1 -1 -1 -1 +1 -1 +1 -1 +1 -1 -1 +1 -1 -1 +1

*The complement code, the reflected code, and the complement reflected code are also Neuman-Hofman codes.

**This is also a Barker code.

TABLE 3.4

AUTOCORRELATION SIDELOBES OF

NEUMAN-HOFMAN CODES

CODE LENGTH	MAXIMUM SIDELOBE	
	C_k	$\lvert C_k \rvert$
7	0	1
8	1	2
8	1	3
9	2	2
9	2	2
10	2	2
10	2	2
11	2	2
11	0	1
12	1	4
12	2	2
13	2	2
13	2	2
14	2	2
14	2	2
15	2	2
15	2	2
16	2	2
16	2	2
17	1	4
17	2	2
18	1	3
18	1	2
19	2	2
19	2	2
20	2	2
20	1	4
21	2	2
21	2	2
22	1	3
23	2	5
23	2	3
24	3	9
24	1	4

Codes are in same order as Table 3.3.

TABLE 3.5

FALSE-SYNCHRONIZATION PROBABILITY FOR

BARKER AND NEUMAN-HOFMAN CODES

CONDITIONS	FALSE-SYNCHRONIZATION PROBABILITY FOR		
	13-BIT BARKER CODE	13-BIT NEUMAN-HOFMAN CODE	7-BIT BARKER CODE
NO SIGN AMBIGUITY			
OPTIMUM DETECTOR			
$E/N_0 = 1/2$	0.31	0.28	0.40
1	0.09	0.07	0.21
2	0.00	0.00	0.09
CORRELATION DETECTOR			
$E/N_0 = 1/2$	0.42	0.32	0.45
1	0.19	0.18	0.32
2	0.08	0.07	0.22
BPSK WITH SIGN AMBIGUITY			
OPTIMUM DETECTOR			
$E/N_0 = 1/2$	0.39	0.39	0.63
1	0.14	0.14	0.37
2	0.00	0.00	0.21
CORRELATION DETECTOR			
$E/N_0 = 1/2$	0.47	0.49	0.63
1	0.27	0.24	0.46
2	0.12	0.13	0.40

performance so that prime power is not wasted on unnecessary transmission, as well as maximizing the amount of useful data which can be relayed through the satellite. Looking at the sync performance in Table 3.5, we can conclude that a 13-bit Barker sequence would be a reasonable choice for the unique word.

The receiver can detect the unique word by using correlation techniques, either active or passive. An example of the receiver's unique word detector using a passive correlator is shown in Figure 3.42. When the received unique word is in the register, the comparator recognizes the coincidence of the received unique word with the local replica of the 13-bit Barker code and outputs a signal. This signal can be used to enable the clock to the receiver's PN generators, to enable the PN sync recovery circuits, and to provide a reference epoch to the demultiplexer.

3.4.2 Structure of the Preamble

From the discussion in Section 3.4.1, we arrive at the structure of the preamble, which is illustrated in Figure 3.43. We see that the preamble consists of 3 segments, sent time-sequentially. The first segment is an unmodulated carrier for carrier-sync recovery. This segment lasts for T_0 seconds. It is followed by a bit-sync pattern (alternating 1's and 0's or all-1's, depending upon the baseband coding format, as discussed in Section 3.4.1.2) which lasts for T_1 seconds. The bit-sync pattern is followed immediately by the unique word for frame synchronization. The unique word has a duration of T_3 seconds, after which time the data frames are sent.

3.4.3 Generation of the Preamble

A block diagram of a circuit to generate the preamble is shown in Figure 3.44. Although this circuit and the subsequent operational description are in terms of a specialized hardware implementation, it

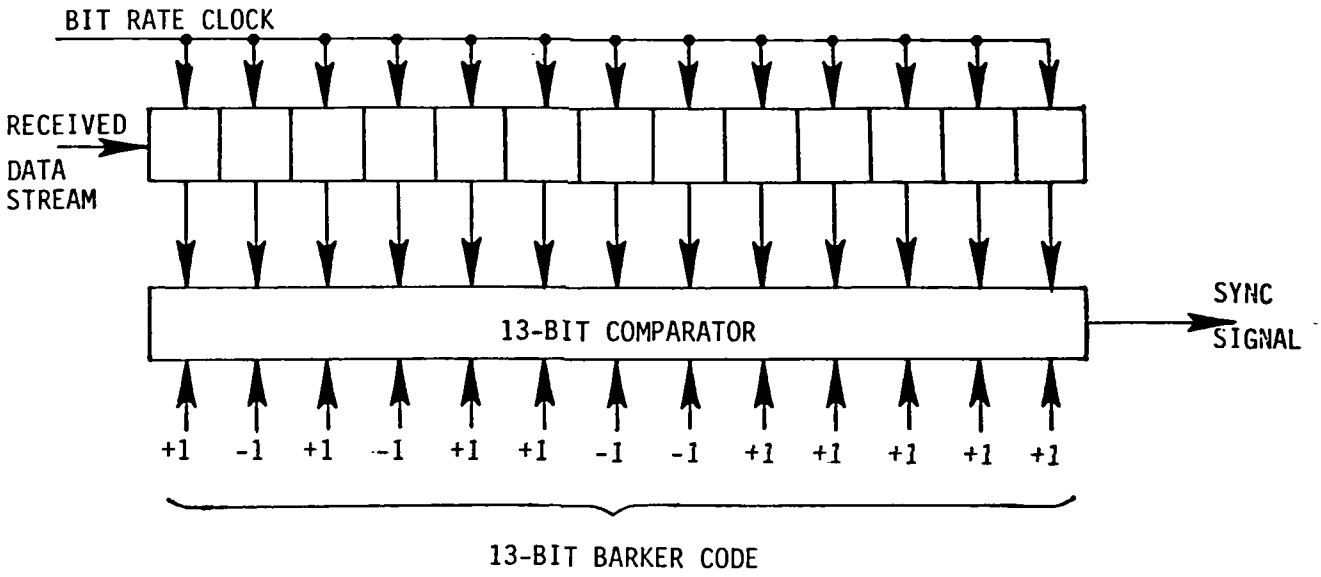


Figure 3.42 PASSIVE CORRELATOR FOR UNIQUE-WORD DETECTION

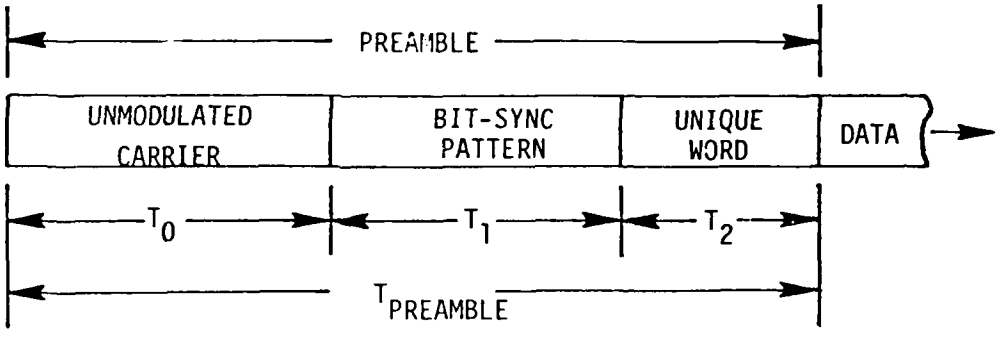


Figure 3.43 PREAMBLE STRUCTURE

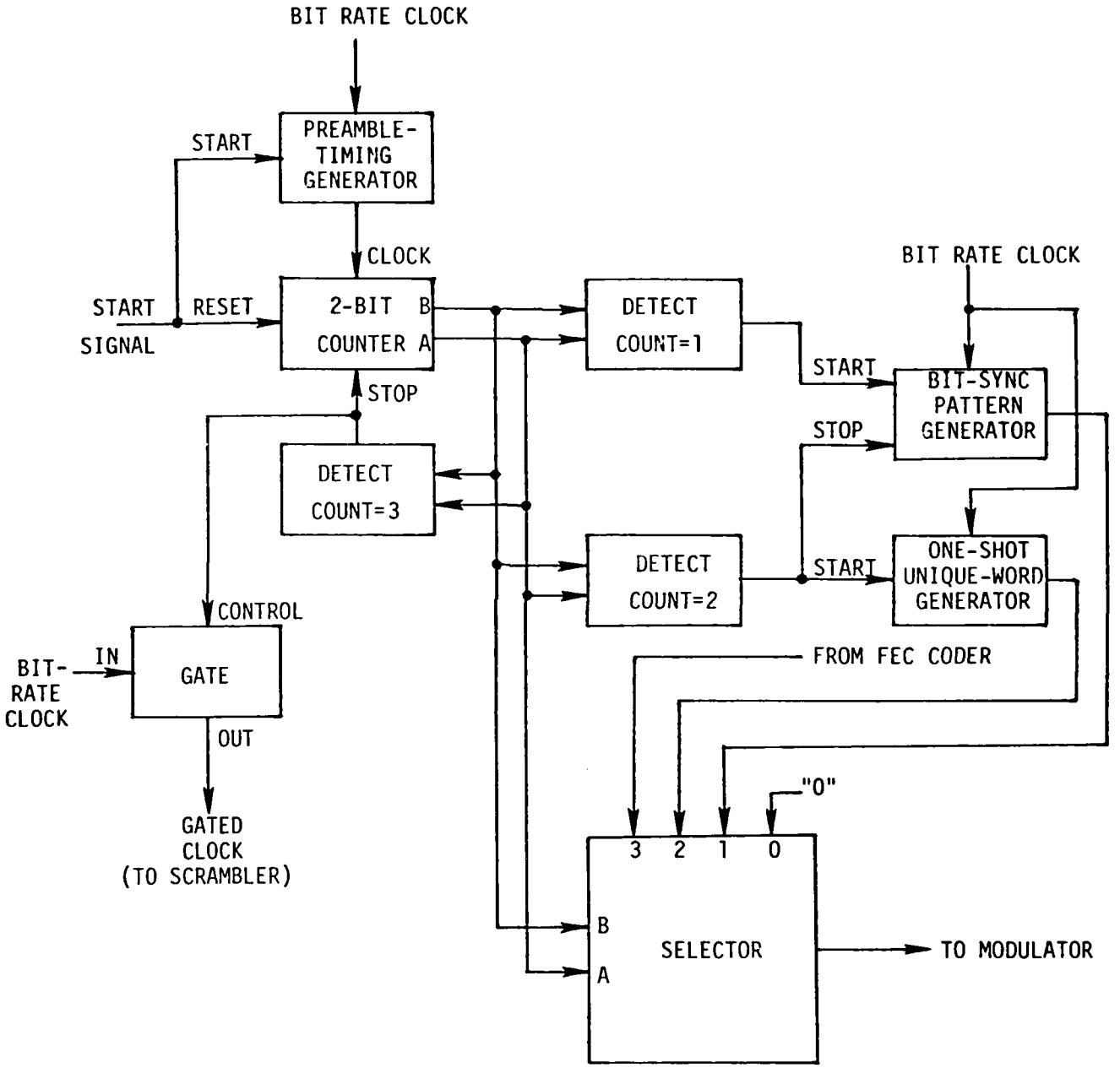


Figure 3.44 PREAMBLE GENERATION

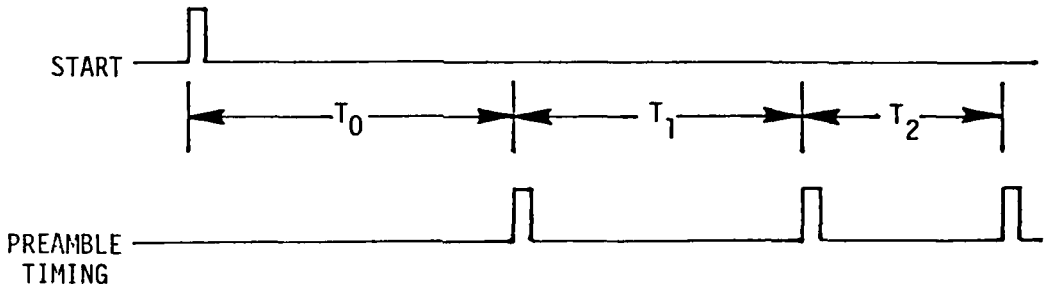
must be borne in mind that a software/firmware implementation using a microprocessor is a viable alternative.

The implementation example shown in Figure 3.44 assumes the use of a binary DPSK modulator. If other modulations with higher-order alphabets (e.g. QPSK) are used, then the details would differ in order to generate the proper symbol patterns.

The principle of the preamble generator is to select sequentially inputs to the modulator which create the three segments of the preamble. After the completion of the preamble, the FEC-coder outputs are connected to the modulator so that the FEC-coded scrambled data may be transmitted. The stepping of the selector from one source to another is controlled by a timing circuit in accordance with the preamble design. In addition, the bit-rate clock to the scrambler is inhibited during the preamble transmission so that the proper initial load is present when scrambled data is first sent to the modulator via an FEC coder.

The generation of the preamble is initiated by an external START signal which is applied to a 2-bit counter and the preamble-timing generator. (In addition, this START signal would turn on the RF transmitter; but this is external to the generation of the preamble.) The assertion of the START signal causes the 2-bit counter to be reset to the state 00 and initiates the first timing interval of the preamble-timing generator which provides clock pulses to the 2-bit counter.

The preamble-timing generator is a device which outputs three pulses in responses to a START input pulse. The timing of the output pulses is shown in Figure 3.45. The design is such that the leading edge of each pulse terminates a segment of the preamble.



T_0 = Carrier-Sync duration

T_1 = Bit-sync pattern duration

T_2 = Unique-word duration

Figure 3.45 PREAMBLE TIMING SIGNAL

The outputs of the 2-bit counter are fed to the control inputs of a 4-line-to-1-line selector and also to three count-decoding circuits. These decoding circuits are designed to decode counts equal to 01, 10, and 11, respectively, and provide a signal when the appropriate count is recognized. (In practice the three decoders could be merged into one 2-line-to-4-line decoder.)

The selector operates in accordance with the truth table shown in Table 3.6. The output of the selector agrees with input 0, 1, 2, or 3 as the control lines B and A are 00, 01, 10, or 11, respectively. Thus the selector can be considered as a four-position switch. The selector controls the signal going to the modulator.

Upon the receipt of a START pulse, the outputs of the 2-bit counter are reset to 00 and the first timing interval of the preamble-timing generator is initiated. The outputs of the three count-detection circuits all remain 0 since none of them are configured to detect count 00. The selector selects input 0, which is hard-wired to a logic-0 level. This produces a constant logic-0 input to the modulator, and hence yields a constant-phase carrier, which constitutes the first segment of the preamble.

After a time period of T_0 seconds, the preamble-timing generator outputs a clock pulse to the 2-bit counter, causing the counter to count up to state 01. The 01 count is decoded by the COUNT=1 detector, which outputs a signal to start the bit-sync pattern generator. The presence of the 01 count also causes the selector to select its number 1 input which is connected to the output of the bit-sync pattern generator; hence the bit-sync pattern is applied to the modulator. If the modulator employs DPSK, the bit-sync pattern generator is designed to put out all-1's data; if BPSK is employed, then the bit-sync pattern generator is designed to put out alternating 1's and 0's.

TABLE 3.6

TRUTH TABLE FOR SELECTOR GATE

CONTROL B A	INPUTS 0 1 2 3	OUTPUT
0 0	0 X X X	0
0 0	1 X X X	1
0 1	X 0 X X	0
0 1	X 1 X X	1
1 0	X X 0 X	0
1 0	X X 1 X	1
1 1	X X X 0	0
1 1	X X X 1	1

X = Don't Care

After a time period of T_1 seconds, the preamble-timing generator outputs a clock pulse to the 2-bit counter, causing the counter to count up to state 10. The 10 count is decoded by the COUNT=2 detector, which outputs a signal to start the one-shot unique-word generator. This signal also serves to stop the bit-sync pattern generator. Upon being triggered by the output of the count-detection logic, the one-shot unique-word generator outputs the chosen unique word exactly once, then returns to a quiescent state. The presence of the 10 count also causes the selector to select its number 2 input, which is connected to the output of the one-shot unique-word generator, and hence sends the unique word to the modulator.

After a time period of T_2 seconds, the preamble-timing generator outputs a clock pulse to the 2-bit counter, causing the counter to count up to state 11. The 11 count is decoded by the COUNT=3 detector, which outputs a signal to enable a gate, allowing the bit-rate clock to be applied to the data-scrambling circuits. This same signal also stops the 2-bit counter and inhibits further counting until a new start signal is applied. The presence of the 11 count also causes the selector to select its number 3 inputs, which is connected to the output of the FEC coder which processes the scrambled data, thus sending the FEC-coded scrambled data to the modulator.

At this point the preamble generation has been completed and the system placed into the run mode. The system will continue to operate until shut down by a source external to the preamble generator.

4.0 TRANSMISSION OF DPSK SIGNAL THROUGH SATELLITE TRANSPONDER

The information collected by the spectral detector must be transmitted to the processor for analysis. Typically, a satellite relay communications system would be used to provide remote processing. A modem employing differentially encoded phase-shift-keying (DPSK) signal transmission with differentially coherent demodulation is an appropriate choice when the circuit simplicity and the accompanying cost-effectiveness is the over-riding considerations in the data communication system considered here. We consider such a DPSK system operating over the hard-limiting satellite channel.

The performance of the communications link depends upon the transmitter and antenna on the platform, the satellite, and the receiving shore-based terminal. In order to give a quantitative performance evaluation, we must have available the pertinent specifications of these elements of the link. The platform containing the spectral detector could be a buoy in a real situation.

4.1 Buoy-to-Shore Link

4.1.1 Constraints Imposed by the Buoy Environment

A small buoy does not permit the use of directive antennas, since it is an unstable platform whose location is time-varying as the buoy drifts in the ocean. Therefore, the buoy's antenna is considered to be, at best, a hemispheric pattern with at most 3 dB gain over isotropic. A more realistic assumption is a 0-dB antenna, because of the varying ground-plane formed by the ocean surface, feed line losses, antenna imperfections, etc.

Since the buoy's antenna can not be used to provide gain, the actual power out of the final amplifier, less feed losses, will be the EIRP. Furthermore, the buoy has a limited prime power capability, which places limitations on the feasible transmitter peak power of

around 50 Watts maximum. These considerations motivate the system design to use the lowest practical carrier frequency, in order to minimize the free-space loss [22, p. 34-3] L_{fs}. which is given by

$$L_{fs} = 36.58 + 20 \log_{10} f + 20 \log_{10} d \qquad (4\text{-}1)$$

where

f is the frequency in MHz, and

d is the range in statutue miles.

The choice of satellite is constrained by the operating frequency. Most commercial communications satellites operate in the 4 GHz/6 GHz band, while the military DSCS satellites operate in the 7 GHz/8 GHz band. At these frequencies the free-space losses are excessive for buoy applications. This leaves only FLTSAT [23], and similar satellites such as GAPSAT [24, pp. 57-63], which operates at the UHF (225-400 MHz) band. Typical parameters for a UHF satellite are given in Table 4.1. Assuming a 10^0 elevation angle to the satellite, the slant range would be 25267 statute miles. Using (4-1), we then find the uplink free-space loss to be 174.14 dB and the downlink free-space loss, 172.86 dB.

The ground station which receives the signal is assumed to be of low-to-moderate cost. A typical station would use a helix antenna of moderate size and a terminal available in the Navy's inventory, such as an AN/ARC-143B. Such a terminal and antenna combination typically has a G/T ≈ -14.8 dB/K.

4.1.2 Link Performance Analysis

The performance of DPSK transmission over a hard-limiting channel has been evaluated for the practical case of correlated noise and SNR imbalance at the phase detector [25]. The results of [25] express the bit error probability as a function of uplink SNR (R_u^2), downlink SNR (R_d^2), and SNR imbalance at the phase detector (λ^2). Performance curves for typical values of R_u^2, R_d^2, and λ^2 are shown in Figures 4.1 and 4.2. It was found [25] that the noise correlation did not effect the results when

TABLE 4.1

Typical Communications System Parameters
for a UHF Communications Satellite

Carrier Frequency	
Uplink	299 MHz
Downlink	258 MHz
Satellite Receiver G/T	-16.7 dB/K
Satellite Transmitter EIRP	28 dBW
Orbit	*Geosynchronous*
Transponder Bandwidth	25 kHz
Transponder Type	Hard limiting
Access Method	Single carrier per transponder

the a priori symbol probabilities are equal. Since we are considering a system employing a data scrambler, this assumption is valid (the scrambler tends to "randomize" the data).

It is interesting to note the asymptotic behavior of the curves in Figures 4.1 and 4.2. For low R_d^2, i.e. the downlink-limited region ($R_d^2 << R_u^2$), the curves for a constant λ^2 approach the same limit regardless of R_u^2. At high R_d^2, i.e. the uplink-limited region ($R_d^2 >> R_u^2$), the curves for a constant R_u^2 approach a constant asymptote regardless of the imbalance λ^2. If R_d^2 tends to infinity, the concept of an SNR imbalance becomes meaningless, and the performance depends only on uplink SNR, as seen in Figure 4.2.

The uplink and downlink SNRs, R_u^2 and R_d^2, in Figures 4.1 and 4.2 are ratios of signal power to noise power. Thus the receiver bandwidths of the satellite and the earth station implicitly enter the calculations through these parameters. For the uplink, the appropriate bandwidth is the noise bandwidth of the transponder; for the downlink, the receiver bandwidth is commonly equal to the bit rate. Under these assumptions, we find that R_u^2 = (E - 6.22) dB where E is the EIRP of the buoy's transmitter in dBW and R_d^2 = (68.94 - B) where B=10 $\log_{10}$(bit rate). If the power output of the buoy's transmitter is in the range of 1 Watt to 50 Watts, then $-6.22 \leq R_u^2 \leq 10.77$ dB. Also, for a bit rate of 100 b/s, R_d^2 = 48.94 dB and for a bit rate of 2.4 kb/s, R_d^2 = 35.14 dB. It is quite clear that the uplink will be the controlling factor. Furthermore, for these high values of R_d^2 the effects of SNR imbalance at the phase detector will be negligible and the curve in

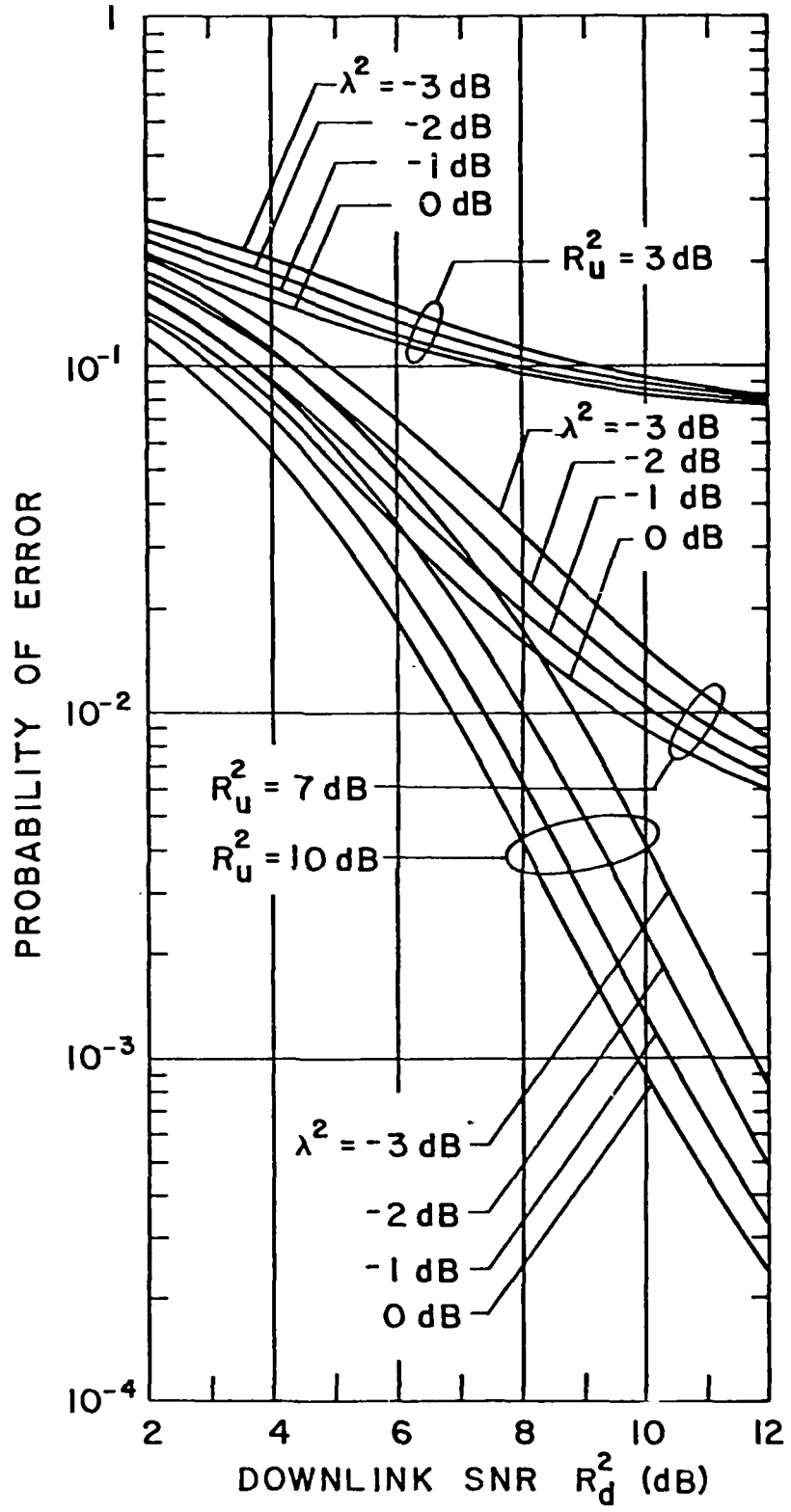


Figure 4.1 TOTAL ERROR PROBABILITY FOR DPSK AS A FUNCTION OF DOWNLINK SNR WITH UPLINK SNR AND POWER IMBALANCE AS PARAMETERS

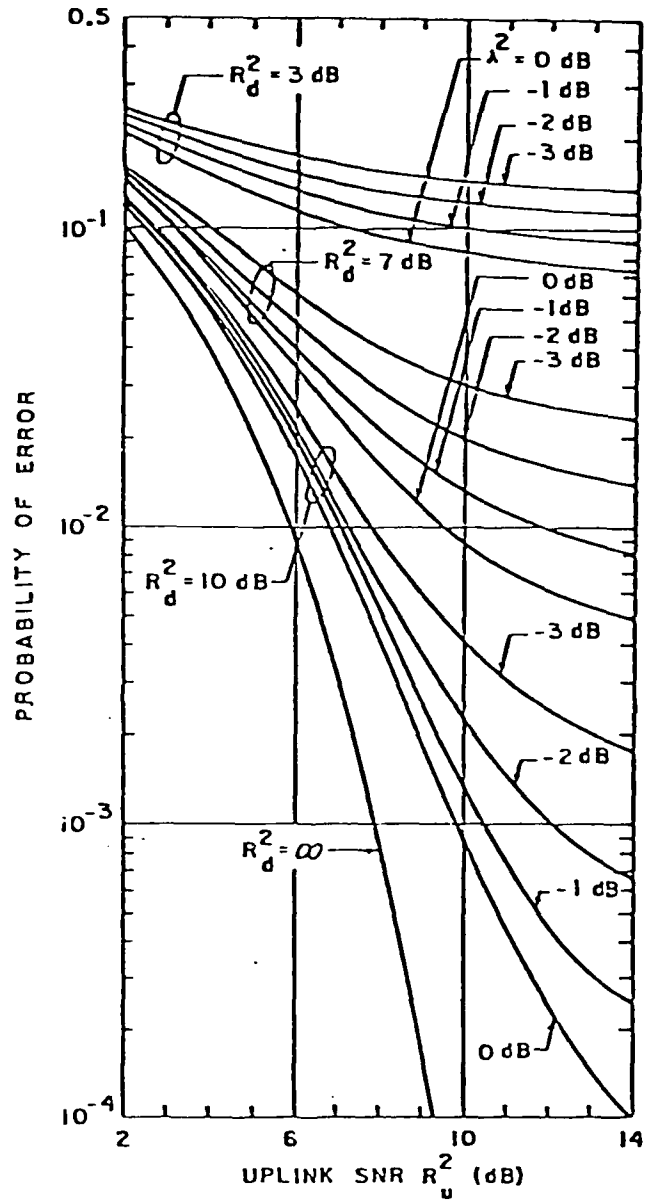


Figure 4.2 TOTAL ERROR PROBABILITY FOR DPSK AS A FUNCTION OF UPLINK SNR WITH DOWNLINK SNR AND POWER IMBALANCE AS PARAMETERS

Figure 4.2 labelled $R_d^2 = \infty$ applies; thus we may use the "classical" analysis dicussed below to a good approximation.

When the effects of SNR imbalance on the link performance are negligible, the performance of a single-carrier-per-transponder link (i.e. no intermodulation noise) can be evaluated by use of the equation [26]

$$\left(\frac{C}{N_0}\right)_{TOTAL} = \left[\left(\frac{C}{N_0}\right)_{UP}^{-1} + \left(\frac{C}{N_0}\right)_{DOWN}^{-1}\right]^{-1} \qquad (4\text{-}2)$$

where $\left(\frac{C}{N_0}\right)_{UP}$ is the uplink carrier power-to-noise density ratio at the output of the hard limiter. If $(C/N_0)_{DOWN}$ is large compared to $(C/N_0)_{UP}$, then $(C/N_0)_{TOTAL} \simeq (C/N_0)_{UP}$. It may be related to the carrier power-to-noise density ratio at the input to the limiter [27] by the factor shown in Figure 4.3. Finally the total C/N_0 and the bit rate are used to calculate E_b/N_0 from the relation, in decibels,

$$\frac{E_b}{N_0} = \frac{C}{N_0} - 10 \log_{10}(\text{bit rate}) \qquad (4\text{-}3)$$

which may be used with the "classical" DPSK performance curves [13, p. 303], [12, p. 385] to assess link performance. Using (4-2), (4-3) and Figure 4.3 we have plotted in Figure 4.4 the E_b/N_0 at the shore receiver as a function of the buoy's EIRP for data rates of 100 b/s, 1200 b/s, and 2400 b/s.

The "classical" DPSK performance in terms of bit-error probability is given by [12, p. 384]

$$P_e = \frac{1}{2} \exp(-E_b/N_0). \qquad (4\text{-}4)$$

By applying (4-4) to the E_b/N_0 values plotted in Figure 4.4, we can plot

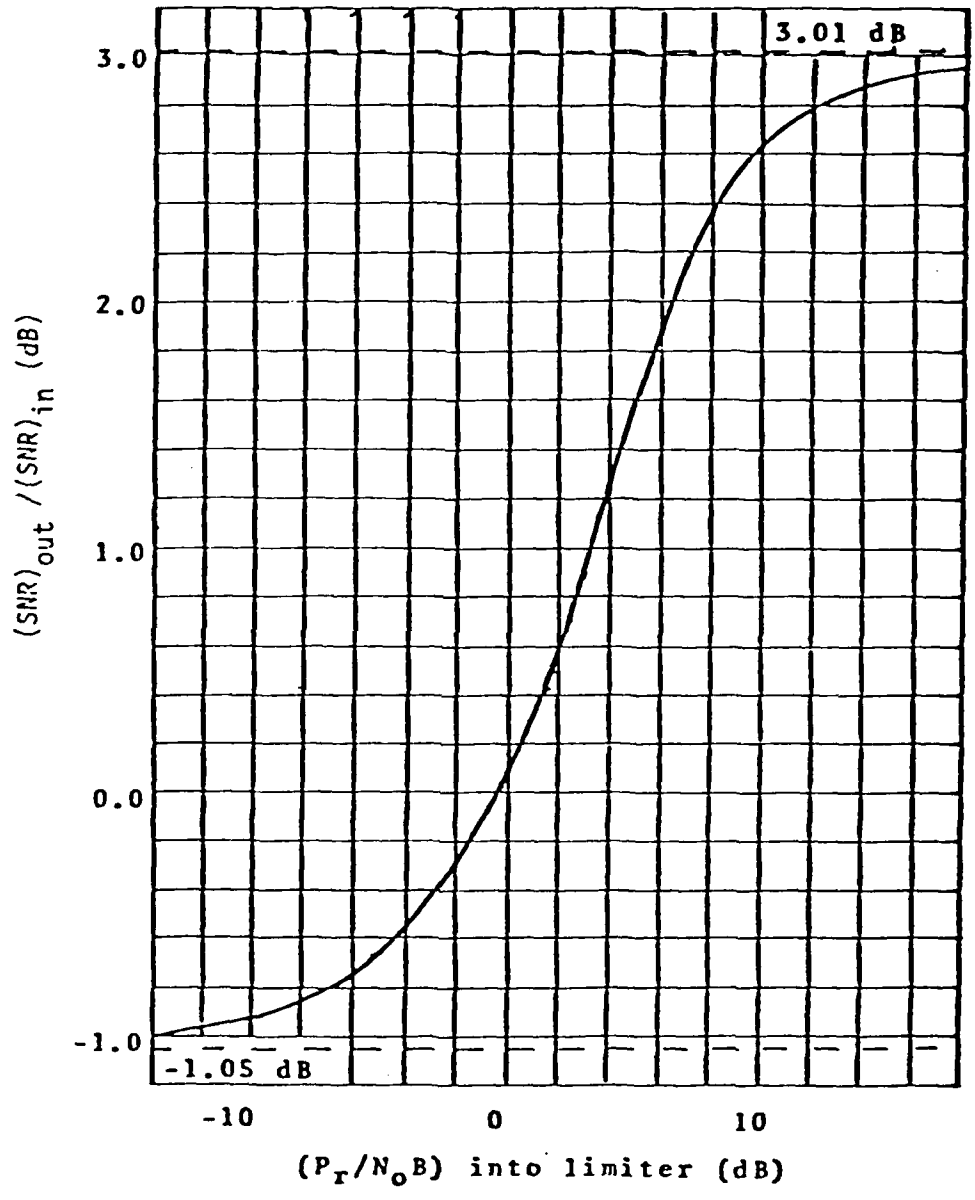


Figure 4.3 EFFECT OF HARD LIMITER ON SINGLE-CARRIER SIGNAL-TO-NOISE RATIO

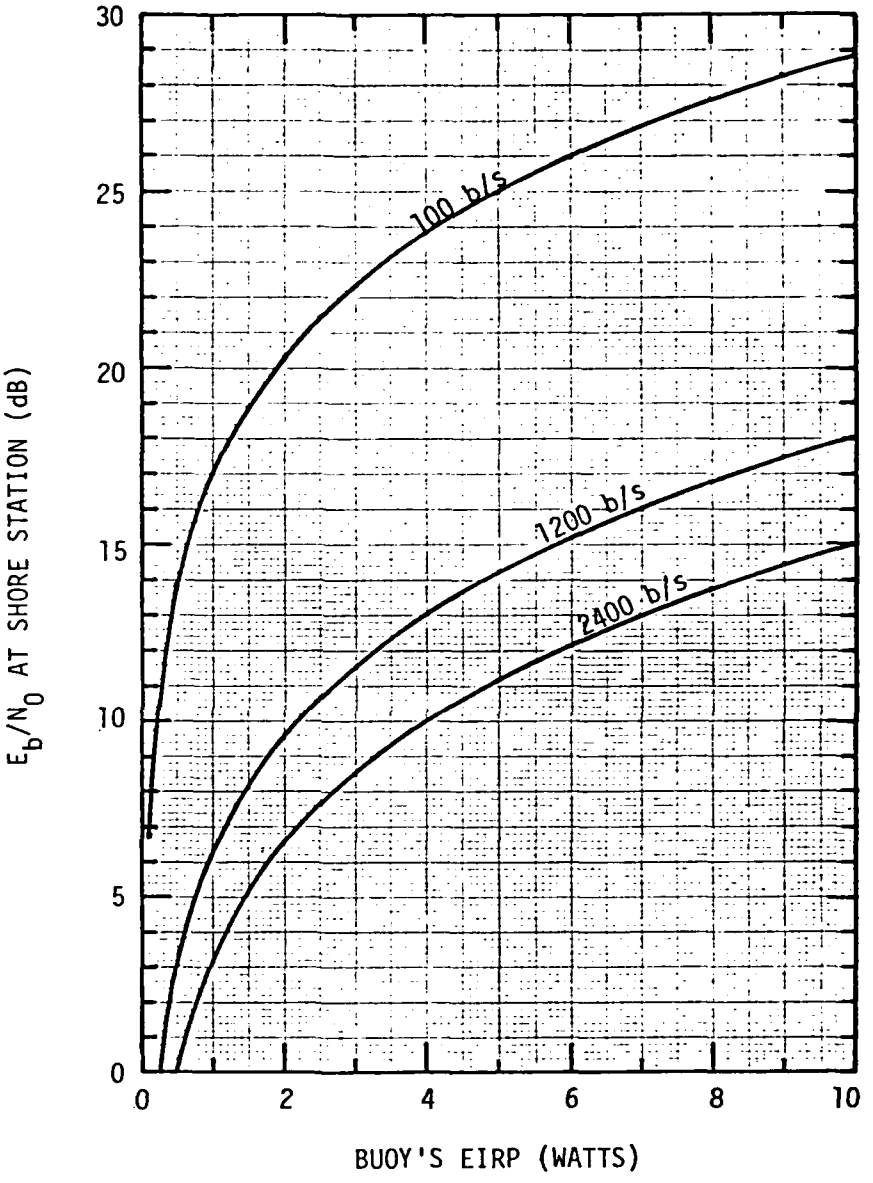


Figure 4.4 EXAMPLE OF TOTAL COMMUNICATION-LINK PERFORMANCE (BUOY-TO-SHORE)

the bit error probability as a function of the buoy's EIRP, as is done in Figure 4.5.

It must be borne in mind that the performance given by (4-4) and in Figure 4.5 assumes no detuning and perfect bit synchronization. The effects of Doppler shift, oscillator drift, noise in the bit-rate clock tracking loop (bit synchronizer), propagation disturbances, and other deviations of a practical system from the "ideal" theoretical system must be taken into account in addition to the bit error probability given by (4-4) or Figure 4.5. These effects require that a "margin" be added to the required buoy transmitter power to accommodate fading and the imperfections inherent in a real system. This margin can amount to several decibels, and will be a function of the required link performance.

4.2 Shore-to-Buoy Link

We now turn our attention to the link from the shore to the buoy. This link may be required for several reasons, such as a need to command the buoy to different processing modes or polling in a TDMA system. We assume this link will be supported by the same satellite as the buoy-to-shore link.

The shore transmitter is assumed to be similar in capability to an AN/ARC-143B used with a helix antenna. The shore station's EIRP is assumed to be 36 dBW. The satellite parameters are as given in Table 4.1. The G/T of the buoy receiver is taken as a parameter, since it will depend upon the antenna structure on the buoy and the noise figure of the receiver's input stage.

For an elevation angle of 10^{o}, the uplink and downlink free-space losses were found previously to be 174.14 dB and 172.86 dB, respectively. The uplink C/N_0 is 73.76 dBHz. For a 25-kHz transponder, the input P_r/N_0B

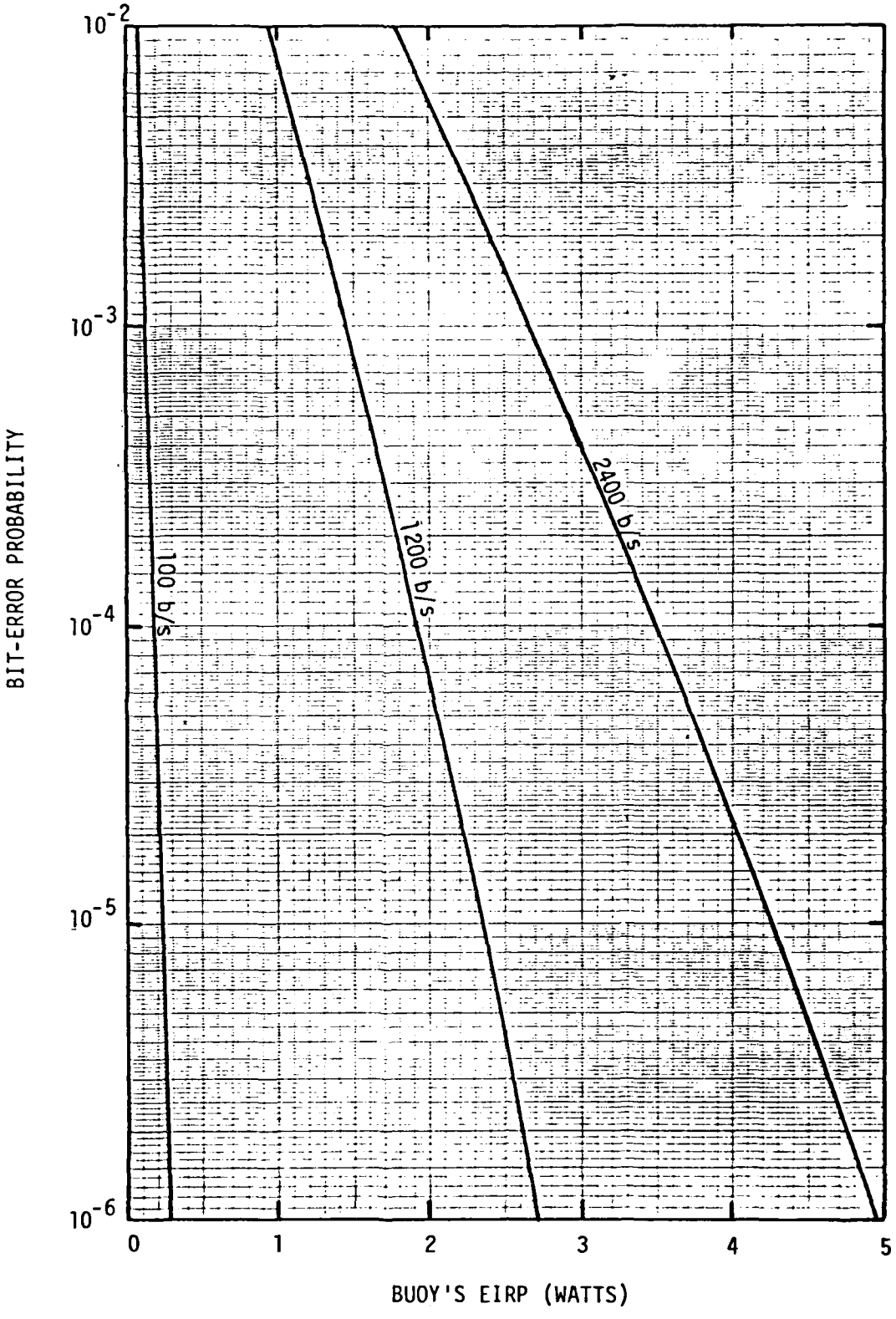


Figure 4.5 BIT-ERROR RATE PERFORMANCE OF EXAMPLE SYSTEM (BUOY TO SHORE)

to the transponder is 29.78 dB, and Figure 4.3 shows the limiter factor to be 3 dB. Thus $(C/N_0)_{UP}$ in (4-2) is 76.76 dBHz.

For the downlink to the buoy, $(C/N_0)_{DOWN}$ = (83.74 + G/T)dBHz where G/T is the figure of merit of the buoy's receiving system. Since a simple omnidirectional antenna is used and since the receiver must be low cost, a reasonable range for G/T would be -40 dB/K $\leq$ G/T $\leq$ -20 dB/K. This gives a range of 43.74 dBHz $\leq (C/N_0)_{DOWN} \leq$ 63.74 dBHz for use in (4-2). For this range of $(C/N_0)_{DOWN}$, $R_d^2 \geq$ 16.75 dB for bit rates of 500 bits/second or less. Again, this shows that the "classical" analysis is a good approximation to the link performance. Therefore (4-2) applies to the shore-to-buoy link. The results are shown in Figure 4.6, which also indicates the link margin for a bit-error probability of 10^{-6}. Because of the low data rates on the shore-to-buoy link, the margin is large and the link will not be a constraint on system operations.

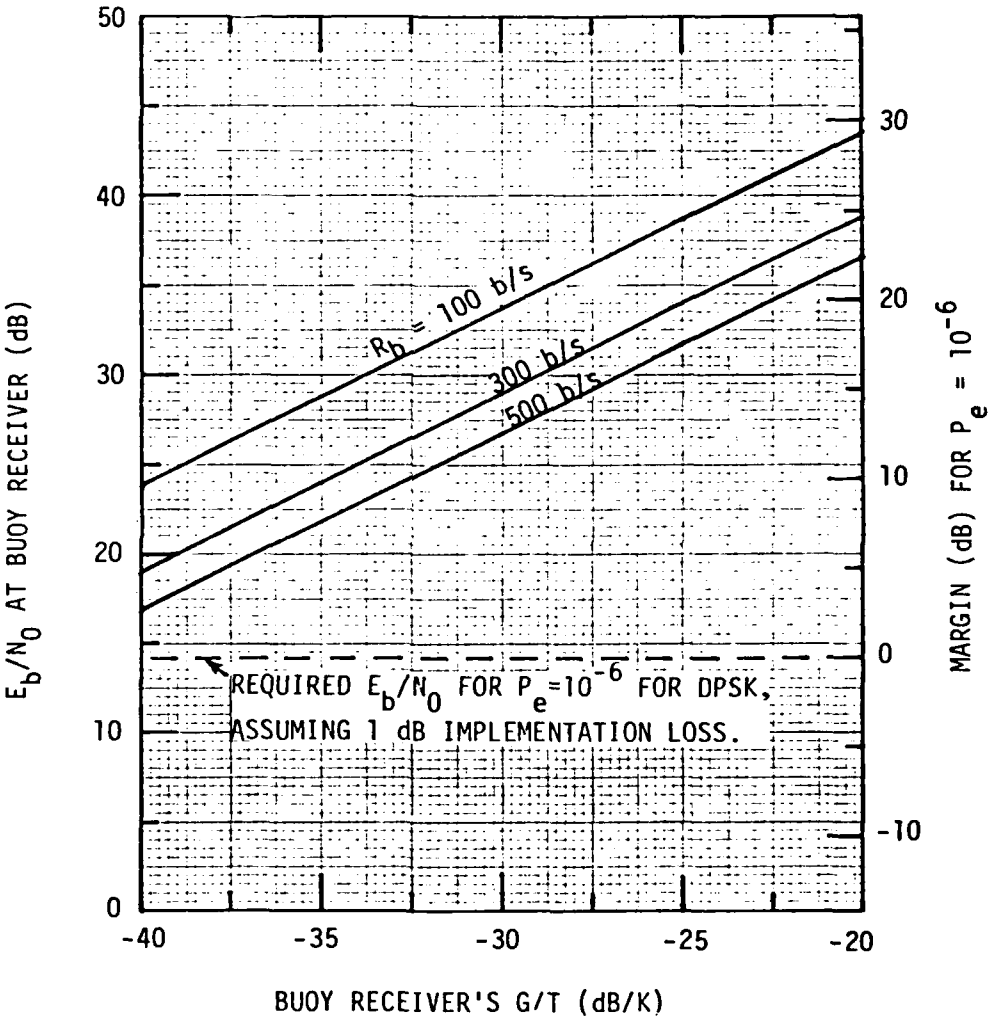


Figure 4.6 EXAMPLE OF TOTAL COMMUNICATION-LINK PERFORMANCE (SHORE-TO-BUOY)

5.0 FORWARD ERROR CONTROL SCHEME

As discussed in Section 4, the command link (shore-to-buoy) is not a constraint on the system performance since a large margin in signal-to-noise energy ratio (E_b/N_0) is allowed as seen in Figure 4.6. However, as we can see from Figure 4.5, the data link (buoy-to-shore) may be very likely subject to the power or bit-rate constraint. For example, in order to support a link with rate 100 b/s at a bit-error rate of 10^{-6}, the required buoy's EIRP is only 0.3 watts, while for a bit rate of 2400 b/s, it is about 5 watts. In other words, reliable data transmission is highly dependent upon the transmitter's (buoy's) power and the bit rate used. Under such circumstances, coding techniques can play an important role in achieving the required reliability of the data transmission by controlling channel error through tradeoff between power and bandwidth.

There are two general classes of error control techniques. When the channel is two way (not necessarily full duplex), the transmitter can store the data until a verification or a request for repeat is received for a data block. This system is called "Automatic Request for Retransmission" (ARQ). It needs some redundancy for error detection, and requires storage at the transmitter. The satellite channel has a substantial time delay on the order of 0.25 second. This makes ARQ costly if not impractical.

The second class is "Forward Error Correcting" (FEC) codes. In this system a redundancy is built into the data to provide the decoder at the receiver with error-correcting capability. When FEC coding is used, the signal-to-noise ratio (E_b/N_0) required to achieve a given performance can be reduced through error correction, by as much as 5 or 6 dB in practice. Thus, the signal power required to sustain a given error rate can be reduced

significantly. This reduction in required received power level is termed "coding gain."

There are two types of coding techniques most commonly used in FEC scheme: block coding technique and convolutional coding technique. In what follows we will briefly describe these two coding techniques and show performance (coding gain) curves of some candidate codes which are applicable to the Spectral Data Transfer System.

5.1 Block Codes

A block encoder stores k data bits, encodes them into n code-bits (n>k), and sends them to the modulator; the digits in a block of n code-bits are independent of those in either the preceding blocks or the succeeding blocks. Block codes have rigid structure based on mathematical theory, particularly, the theory of finite field.

The concept of (Hamming) distance is useful in discussing the error-correcting capability of codes. The "(Hamming) distance" between two codewords is defined to be the number of positions in which the words differ. Thus, minimum (Hamming) distance is a measure of error-correcting capability. For example, if the minimum distance of a code is d, then it is possible to correct all error patterns of t or fewer errors if and only if

$$d \geq 2t + 1.$$

The code rate R of a block code is defined to be the ratio of the number of data bits k to the block length n, k/n, and thus indicate the efficiency of a code.

A class of codes known as the "BCH codes" have powerful error-correcting properties and known decoding algorithms, which are conceptually complicated but relatively simple in terms of decoding time and required circuitry in implementation. In these codes, long codes can be constructed

with both code efficiency (code rate) and error-correcting capability. On the other hand, short block codes are attractive from implementation point of view. Consistent with the requirement of simple implementation for the Spectral Data Transfer System, we shall show in subsection 5.5 the performance curves of two relatively short codes, (15, 7) BCH code and (24, 12) Golay code, together with those of convolutional codes.

5.2 Convolutional Codes

Convolutional codes differ from block codes in that the digits generated by the encoder in a particular time unit depend not only on the information digits within that time unit, but also on the information digits within a previous span of time units. At any time unit, a block of k_0 information digits is fed into the encoder, and a block of n_0 code digits is generated at the output of the encoder, where $k_0<n_0$. The n_0-digit output block not only depends on the k_0-digit information block of the same time unit but also depends on the previous K-1 information blocks. The ratio k_0/n_0 is the code rate and Kk_0 is the constraint length.

The error-correcting capability of a convolutional code is determined by its minimum distance. However, unlike block codes, convolutional codes do not have mathematical structure; there is no analytic method to construct convolutional codes with maximum minimum distance. Some progress has been made in developing algorithms for finding convolutional codes of moderate length with good distance properties [28] [29], and computer-generated codes have been tabulated by several researchers [5, p. 411].

There are three well-known decoding techniques for convolutional codes: threshold decoding, sequential decoding, and Viterbi decoding. Threshold decoding is an extremely simple technique applicable to many short codes correcting a few errors, and easily extendable to correct bursts

of errors. Its efficiency diminishes as the number of errors to be corrected becomes large. Sequential decoding is the best performing practical technique for achieving very low probability of error. For a number of reasons such as buffer size requirements, computation speed, and metric sensitivity, sequential decoding is not suited for high-speed operations and is more complex than the other decoding scheme, notably the Viterbi decoding. Viterbi decoding is particularly desirable for efficient communication where low probability of error is not required. The crossover point above which Viterbi decoding is preferable to sequential decoding occurs at values of probability of error somewhere between 10^{-3} and 10^{-5} [30] depending on the transmitted data rate. As the data rate increases, the probability of error crossover point decreases.

5.3 Location of the FEC Encoder

Forward error correcting coding can be implemented in three different positions in the system as shown in Figure 5.1. In position I, the data is encoded before scrambled while in position II, the data is scrambled first then encoded; in both cases, the Y-sequence is encoded separately. Hence in both position I and position II, two separate Encoder/Decoder pairs are required. In position III, the Y-sequence and the scrambled data sequence are combined before being encoded; a single Encoder/Decoder is required but it operates at double speed. The proposed system operates at a transmission rate in the kbps range, well within the start of the art; hence from equipment complexity point of view, position III is a better choice.

5.4 Selection of Error Correcting Codes

We have reviewed two important coding techniques which are applicable to the spectral data transfer system. The purpose of using error correcting codes in the data transfer system is to achieve an acceptable reliability in

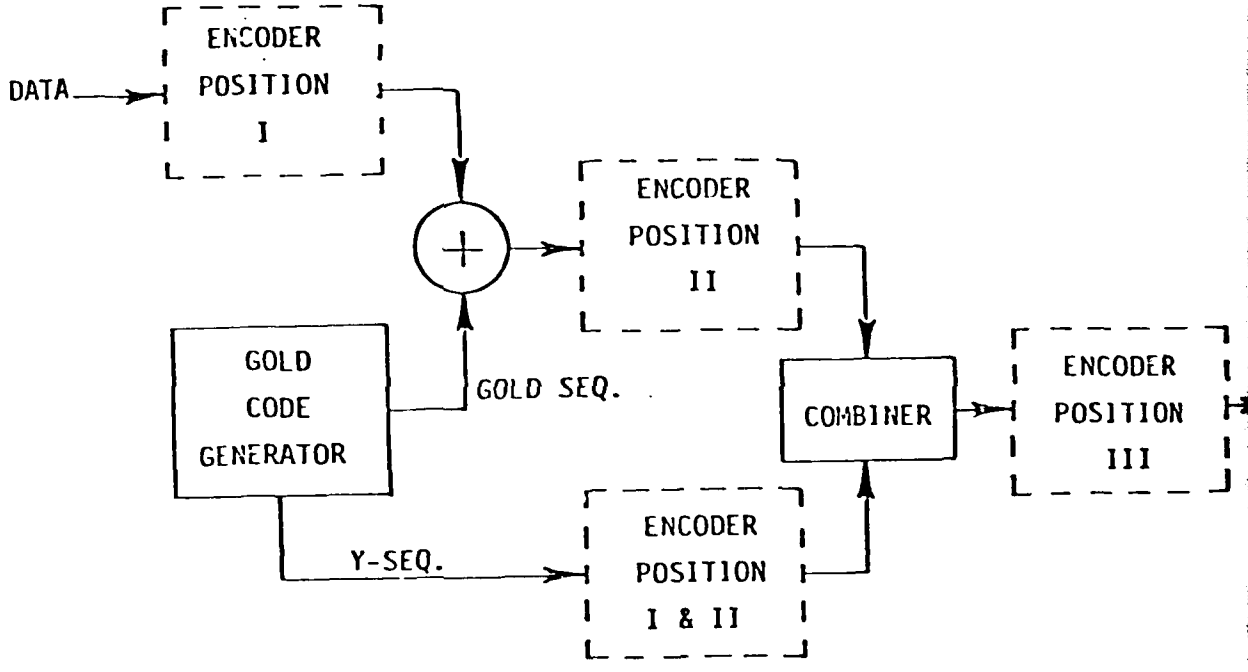


Figure 5.1 Location of the FEC Encoder

face of power limitation, not to enhance the ultimate channel capacity; most short codes with simple decoding procedure could be proper candidates. We will show in the following subsection the performance curves of various short codes (block codes and convolutional codes) for which encoders and decoders are relatively simple to implement.

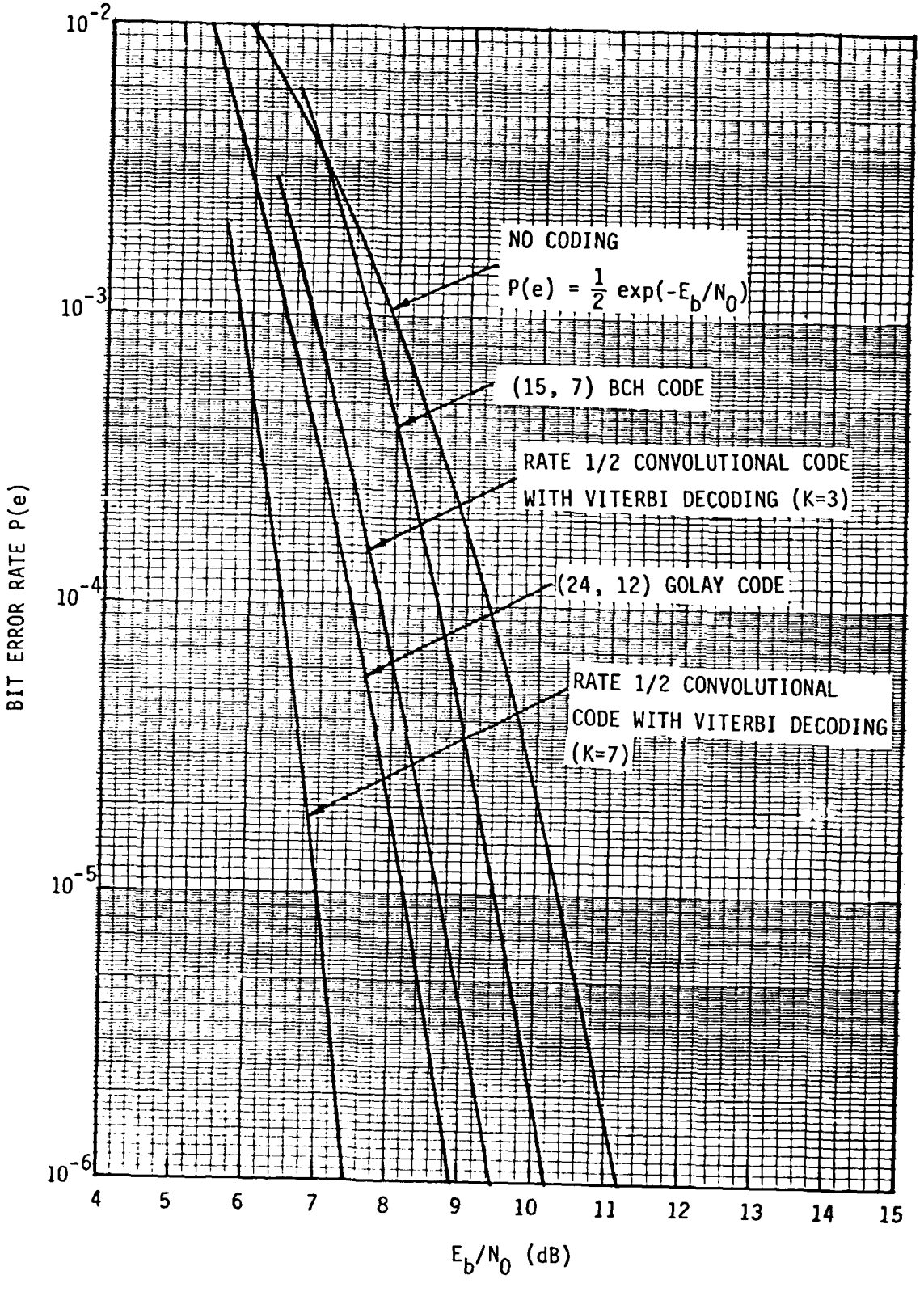
In comparing block codes with convolutional codes, it is widely claimed that convolutional codes give better performance on the space channel than block codes of the same order of complexity [30], [31]. With regard to decoding techniques, Viterbi decoding is a proven concept, widely applied in satellite communication systems, and does not require major technical and hardware breakthroughs. Sequential decoding may be ruled out because of its complexity.

We will thus consider convolutional codes with Viterbi decoding as the candidate scheme for the Spectral Data Transfer System, and will show more detailed performance of this scheme.

5.5 Performance Comparison of Various Coding Schemes

We now present the performance curves of several short codes of interest. Figure 5.2 shows the bit error rate P(e) vs. E_b/N_0 curves for (15, 7) BCH code, (24, 12) extended Golay code, and rate-1/2 convolutional codes of constraint length Kk_0=k=3 and 7 with Viterbi decoding. The curve for the case of no coding is plotted from the DPSK performance (4-4): $P(e) = \frac{1}{2} \exp(-E_b/N_0)$.

As mentioned earlier, coding gain is the reduction of E_b/N_0 acquired by coding in achieving a given P(e). For example, at $P(e) = 10^{-5}$, we obtain from Figure 5.2 coding gains of 0.9 dB, 1.6 dB, 2.2 dB, and 3.3 dB for



ure 5.2 BIT ERROR RATE VS. E_b/N_0 FOR VARIOUS CODING SCHEMES (HARD DECISION - Q = 2)

(15, 7) BCH code, rate 1/2 convolutional code of K=3, (24, 12) Golay code, and rate 1/2 convolutional codes of K=7 respectively.*

Convolutional codes, in connection with the simplicity of their encoders and decoders for short constraint lengths, out perform block codes of the same order of complexity. Note that the performance of the convolutional code of K=3 is quite close to that of (24, 12) Golay code whose decoding is known to be quite messy. For this reason, we consider convolutional codes of short constraint lengths with Viterbi decoding as the candidate error controlling sheme in the Spectral Data Transfer System. More discussion on the performance of this scheme is thus in order.

The coding gain of a convolutional code is a function of various parameters such as number of quantization levels (Q), constraint length (K), code rate (R), and decoding delay (or path history length, L). Figures 5.3 through 5.7 are typical examples of the effects of these parameters on achievable coding gain obtained by the Viterbi decoder. Figures 5.3 and 5.4 indicate the performance improvements that can be realized by using hard decision (Q=2) and soft decision (several quantization levels) as compared to the uncoded case (no coding) for K=3 and 5 respectively. The effect of constraint length K on the coding gain is shown in Figures 5.4 and 5.5 for Q=2 and 8 respectively. One can readily see that there is an increase of about 0.3 dB for each increment in K at a bit error rate of 10^{-4}. In order

*The coding gains presented here are obtained from the classical results available in the literature [32], [33], [34], [35], which are all based on binary antipodal signalling or PSK modulation with coherent demodulation. These results may not be the same as those of a DPSK system. In fact, the performance curves of PSK and DPSK systems for the uncoded case are quite different. However, it can easily be shown that the "coding gains" are approximately the same for both cases over the range of interest [P(e) 10-3 to 10-6]. Thus we may use the results of the PSK system directly on the DPSK system. The net effect is to shift all the curves of the PSK case by a certain amount of E_b/N_0 due to the difference between $Q(\sqrt{2E_b/N_0})$ where $Q(\alpha) = \frac{1}{\sqrt{2\pi}}\int_{\alpha}^{\infty} \exp(-x^2/2)\, dx$ and $\frac{1}{2}\exp(-E_b/N_0)$.

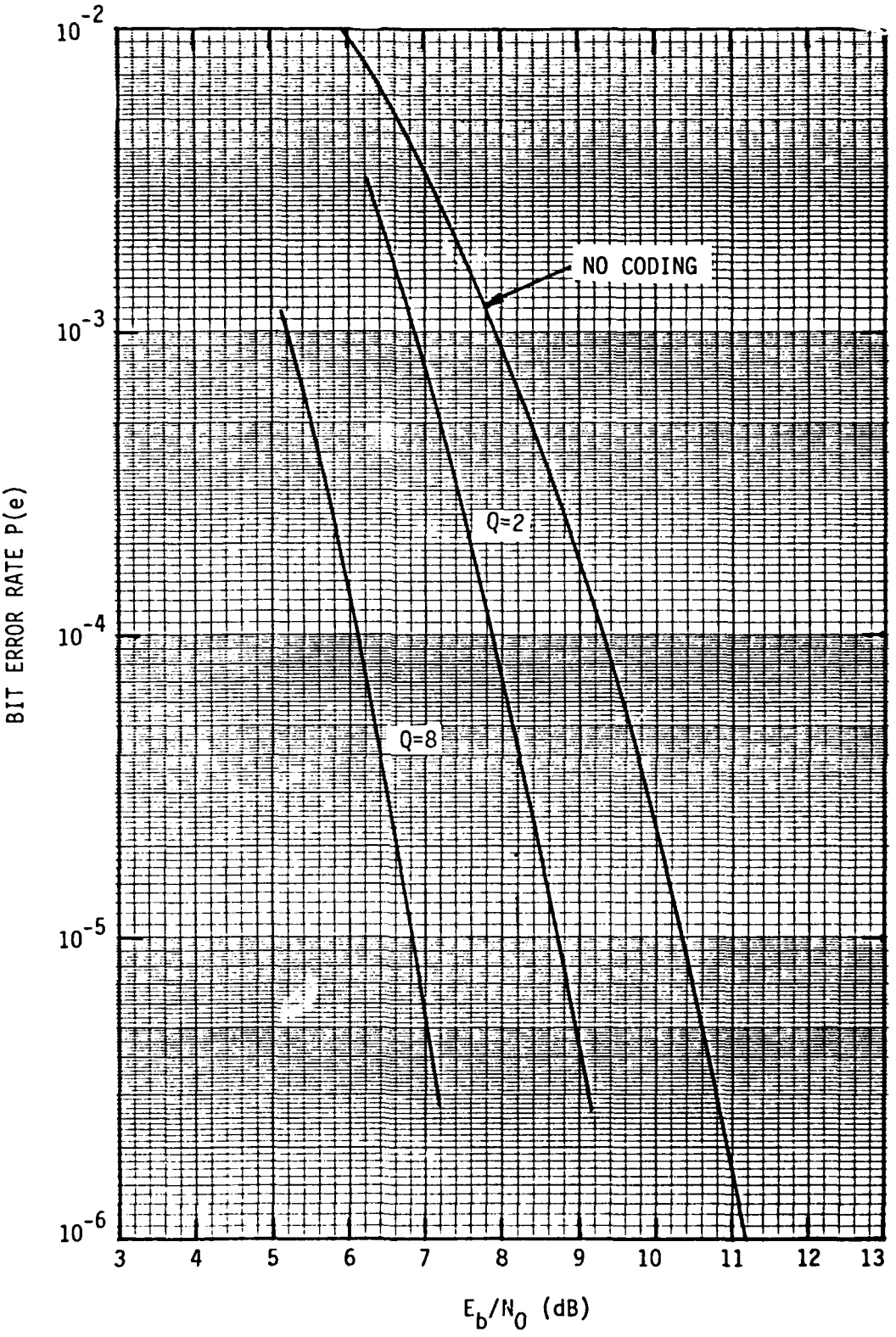


Figure 5.3 PERFORMANCE COMPARISON OF VITERBI DECODING USING RATE 1/2, k=3 CONVOLUTIONAL CODE WITH Q=2 AND 8 LEVEL QUANTIZATION (PATH HISTORY LENGTH = 32 BITS)

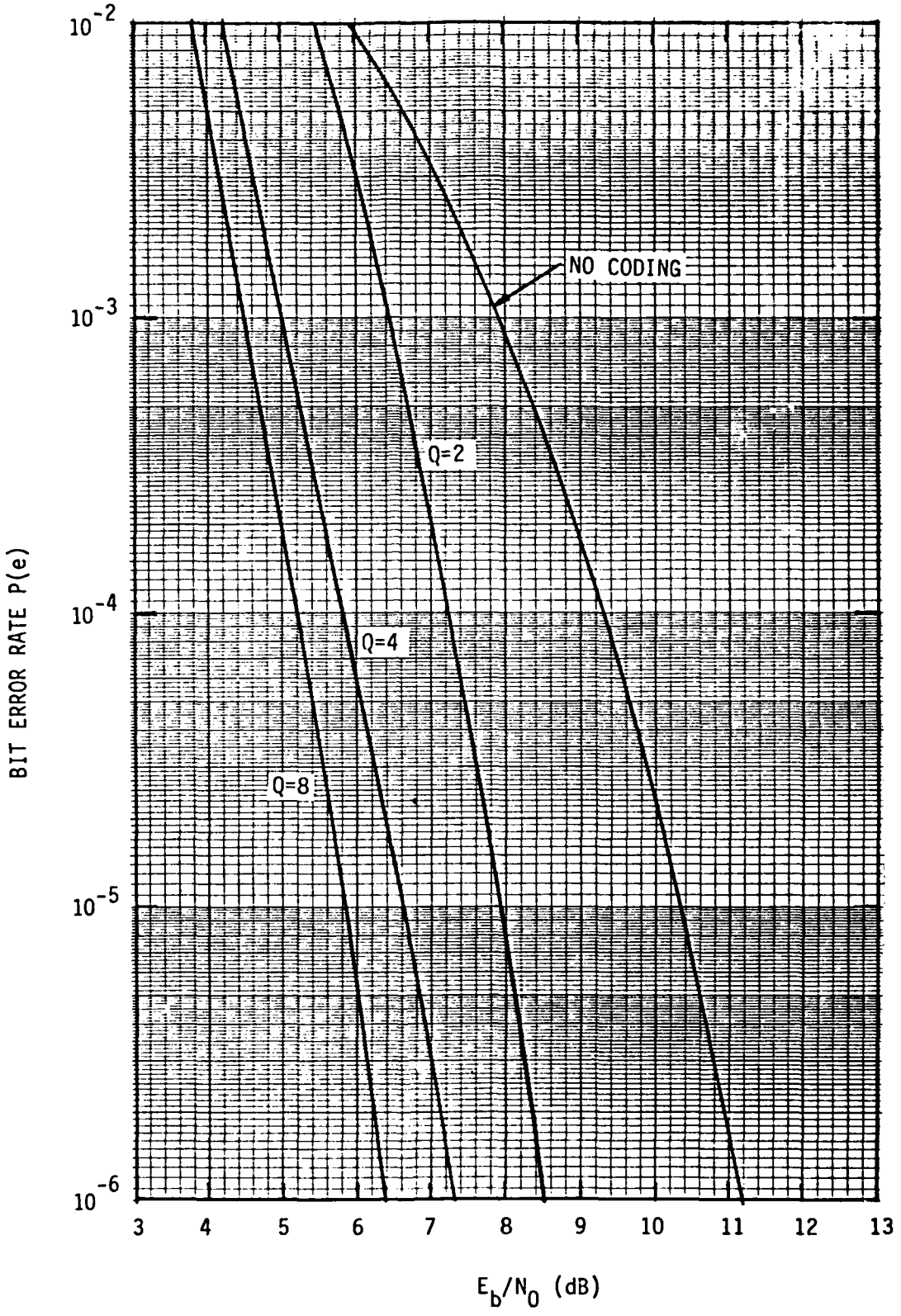


Figure 5.4 PERFORMANCE COMPARISON OF VITERBI DECODING USING RATE 1/2, k=5 CODE WITH Q=2, 4, AND 8 LEVEL QUANTIZATION (PATH HISTORY LENGTH = 32 BITS)

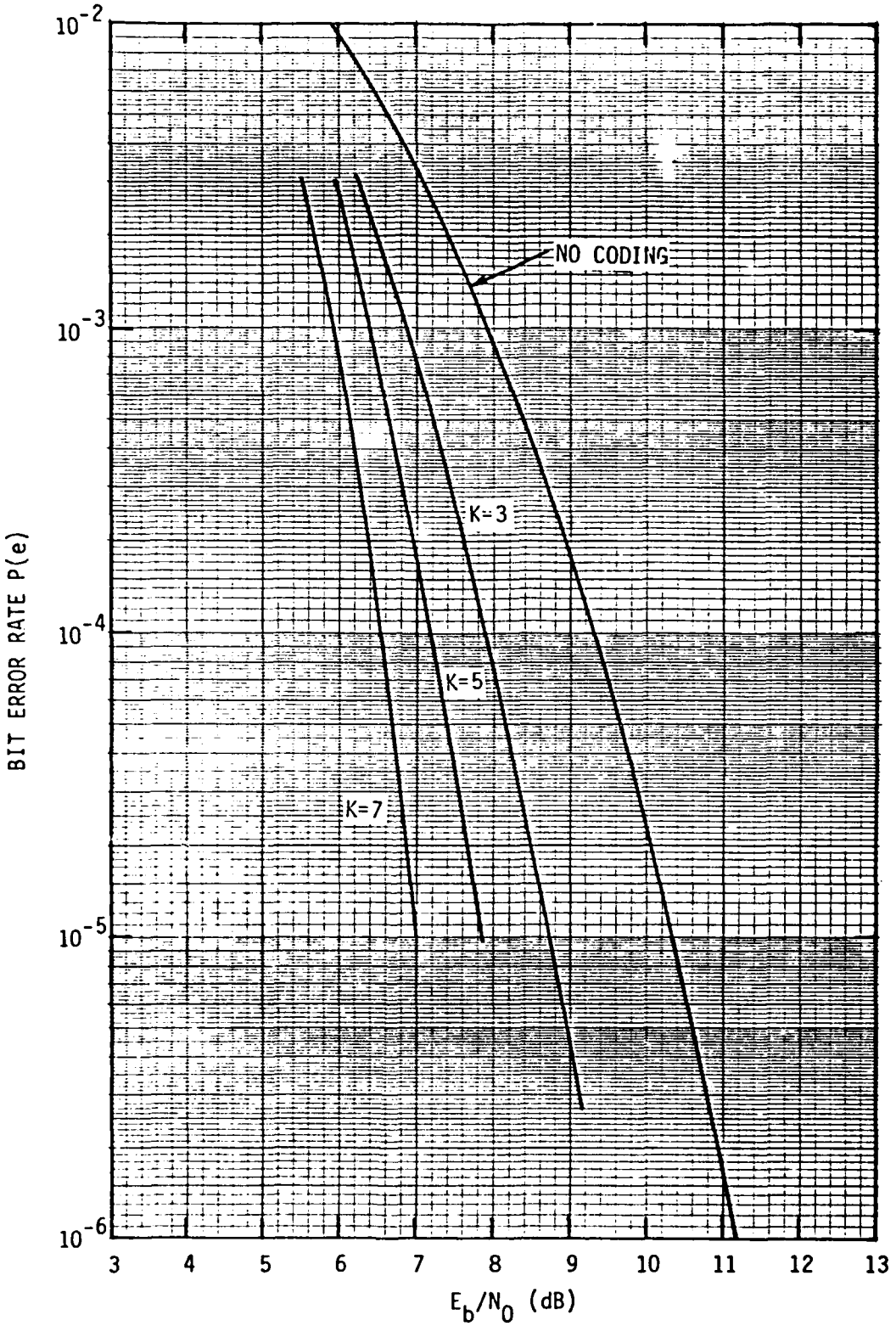


Figure 5.5 BIT ERROR RATE VS. E_b/N_0 FOR RATE 1/2 VITERBI DECODING, HARD QUANTIZED RECEIVED DATA (Q=2). (PATH HISTORY LENGTH =32 BITS), K=3,5,7

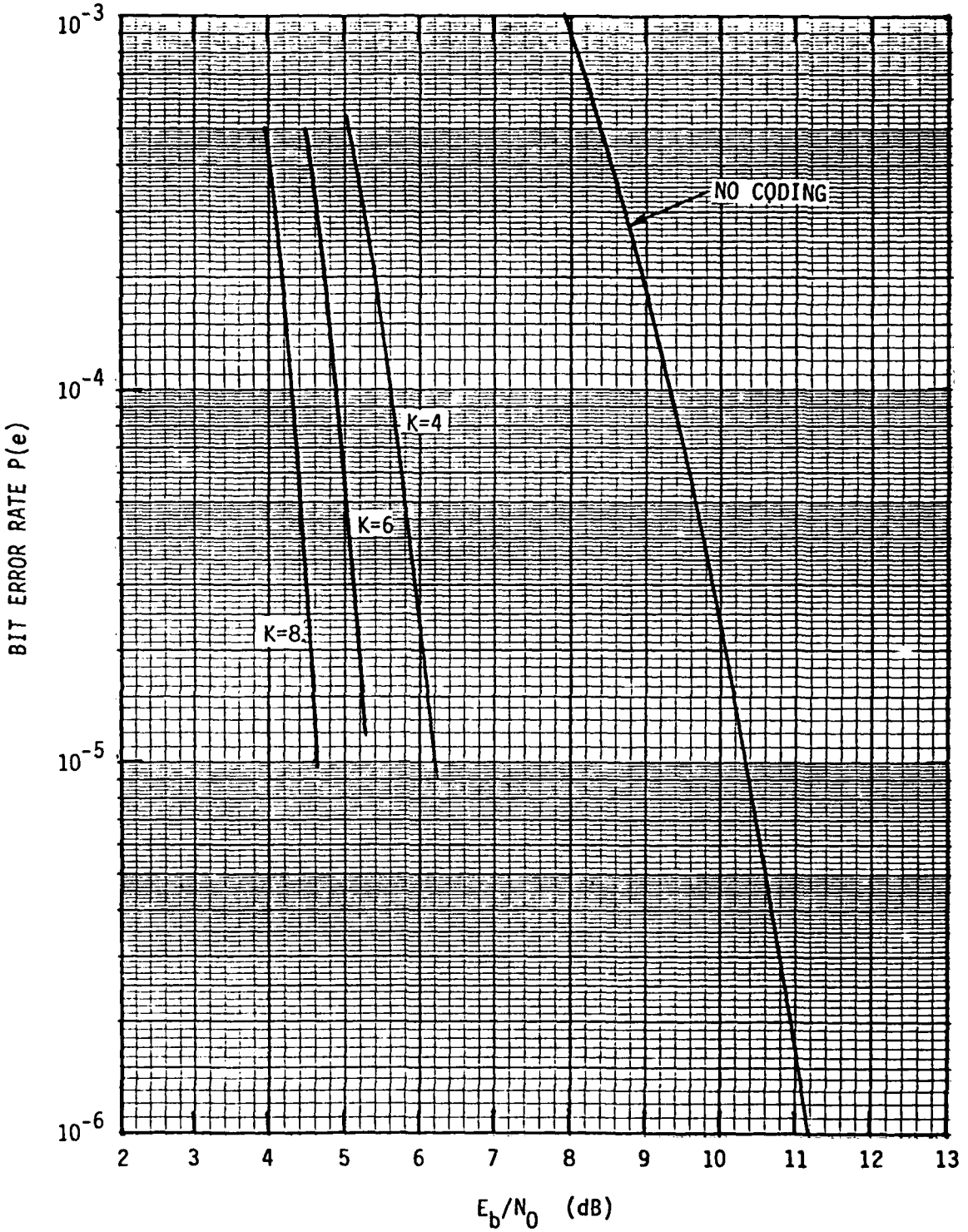


Figure 5.6 BIT ERROR RATE VS. E_b/N_0 FOR RATE 1/2 VITERBI DECODING, 8-LEVEL QUANTIZATION (Q=8) (PATH HISTORY LENGTH = 32 BITS), K=4, 6, 8.

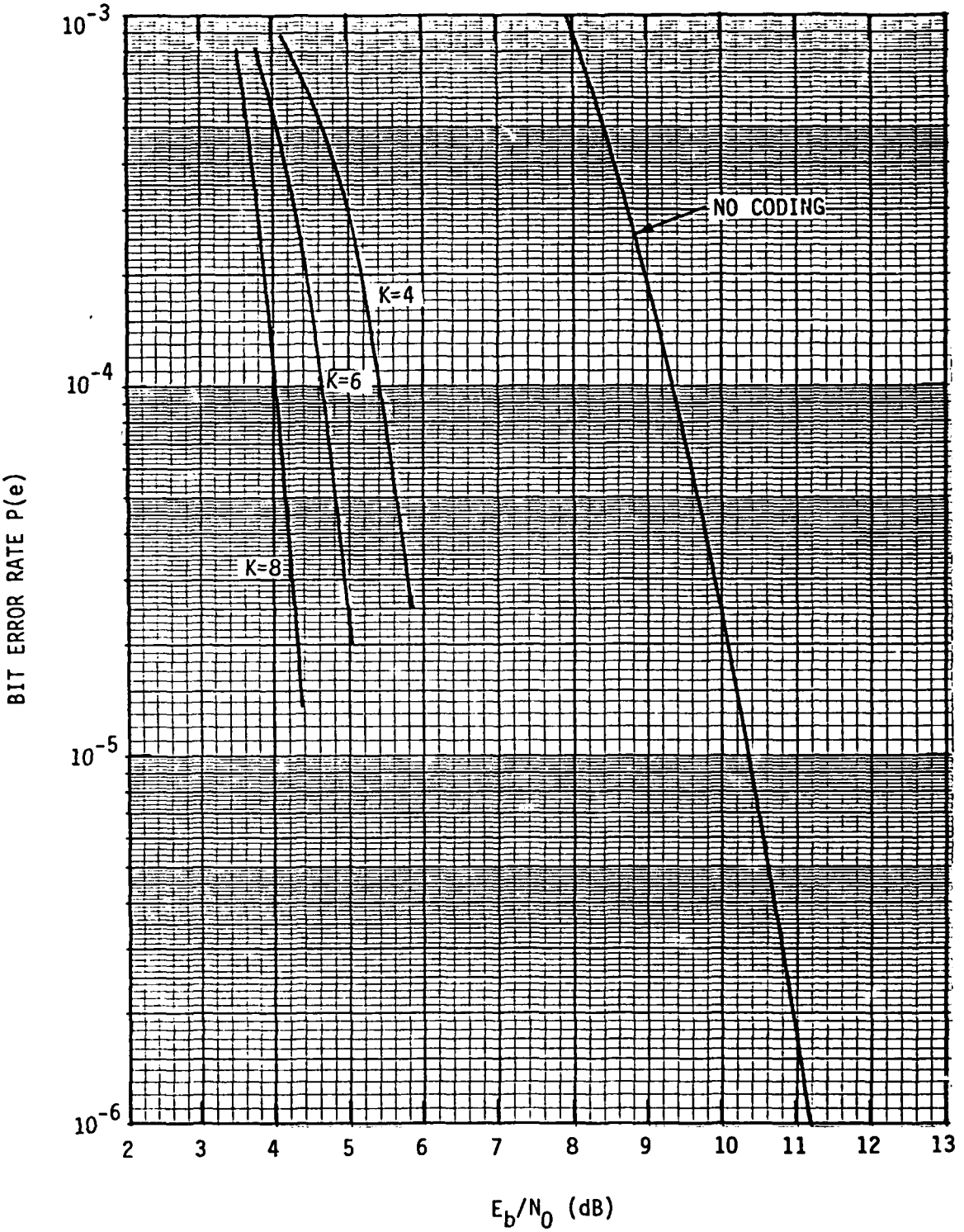


Figure 5.7 BIT ERROR RATE VS. E_b/N_0 FOR RATE 1/3 VITERBI DECODING, 8-LEVEL QUANTIZATION (Q=8) (PATH HISTORY LENGTH = 32 BITS), K=4, 6, 8

to see the effect of code rate R, we present in Figure 5.7 the performance curves of a rate 1/3 code. Comparing Figure 5.6 with Figure 5.7 we can see that the latter offers a 0.3 to 0.5 dB improvement over the former for fixed K in the range shown. The "improvements" mentioned here are obtained, of course, at the expense of some other quantity. Note that rate-1/3 coding requires 3 times the bandwidth of that of an uncoded system while rate 1/2 requires only twice the original uncoded system bandwidth. The larger constraint length means higher cost, and thus the systems evaluator must consider several factors, knowing the individual parameter influence, when making the recommendations. It seems that constraint length K=3 to 5 may suffice for the Spectral Data Transfer System.

In Figures 5.3 through 5.7, the path history length (L) is 32 bits. It should be the engineer's knowledge that a value of L of 4 or 5 times the code constraint length is sufficient for negligible degradation from optimum decoder performance.

Appendix A

We have described the correlator output as

$$y(T) = \mu(T) + n(T), \tag{A-1}$$

in which T is understood to be the interval over which the output is observed. Because of the integrating effect of the lowpass filter--indeed, one model of the lowpass filter which has been used is a T-second integrator--it is possible to assert that the noise term n(T) is nearly Gaussian. Since n(T) is defined to have a zero mean value, we need only its variance to describe its statistical behavior.

It matters how large $\sigma_y (\equiv \sigma_{n(T)})$ is compared to the quantization level q_μ. The natural interpretation of q_μ assumes, for example, that $\sigma_y << q_\mu$.

For moderate values of $W_N T$ and CNR, it can be shown that

$$\frac{\sigma_y^2}{K_1^2 K_2^2 (2\pi W_N)^4} \approx \frac{1}{2\pi W_N T}\left\{\frac{4\gamma^2}{\varepsilon}\sqrt{\frac{\pi}{2}} + \frac{8\delta^2\gamma}{\varepsilon}\sqrt{\pi} + \frac{4\gamma\sqrt{\pi}}{CNR\sqrt{1+\varepsilon^2}}\left(\frac{\varepsilon^2}{1+\varepsilon^2}\right)\right.$$

$$\left. + \frac{2\sqrt{\pi}}{(CNR)^2}\left(\frac{2\gamma}{\sqrt{2+\varepsilon^2}}\right) + \delta^2\sqrt{2}\right\}$$

$$- \frac{1}{(2\pi W_N T)^2}\left\{\frac{4\gamma^2}{\varepsilon^2} + \frac{16\delta^2\gamma}{\varepsilon^2} + \frac{8}{CNR}\left(\frac{\gamma}{1+\varepsilon^2} - \frac{2\gamma}{(1+\varepsilon^2)^2} - \delta^2\right)\right. \tag{A-2}$$

$$\left. + \frac{4}{(CNR)^2}\left(\frac{2\gamma}{2+\varepsilon^2} + \delta^2\right)\right\}$$

where

$$\gamma = \frac{P_m}{(2\pi W_N)^2}, \quad \delta = \frac{D}{2\pi W_N} = \frac{D}{D_{max}}, \quad \varepsilon = \frac{W_m}{W_N} \approx 1. \tag{A-3}$$

For $\gamma = 0$

$$\frac{\sigma_y^2}{\mu_{max}^2} = \frac{\delta^2}{2\pi W_N T}\left\{\frac{2\sqrt{2\pi}}{(CNR)^2}\right\} + \frac{4\delta^2}{(2\pi W_N T)^2}\left\{\frac{2}{CNR} - \frac{1}{(CNR)^2}\right\} \qquad \text{(A-4)}$$

These expressions are for the no-offset configuration. The offset case is handled by replacing δ by $\frac{1}{2}(1+\delta)$.

The required WT and CNR to insure that $\sigma_y < \frac{1}{6} q_\mu$ can be found from these expressions, and are given in the following tables.

Table A-1 describes the quantization of the FM correlator output when no offset is used. For an input resolution whose normalized value is 1/20, as previously mentioned there need to be output (uniform) quantization levels whose normalized values are 1/800. However, Table A-1 shows that an effective nonuniform quantization scheme can be implemented by grouping numbers of output cells to correspond to single input cells. For example, 20 output cells (41-60) correspond to input cell 5.

Because the effective size of these groups of output cells increases in the same proportion as the noise σ_y, the mininum WT to insure that $q_\mu > 6\sigma_y$ remains constant.

Fewer cells would be required if the input range is restricted. If we desired to limit the number of output cells to 512, for example, we would simply modify the table as follows:

Input Cells	Input Range	Nominal	Effective Q	Output Cells
16	$.775 < \lvert\delta\rvert < .7996$	0.7937	15/200 (30)	481-511
17	$\lvert\delta\rvert > .7996$	-	- -	"OVERFLOW"

TABLE A-1 QUANTIZATION SCHEME FOR FM CORRELATOR OUTPUT, NO OFFSET

Input Cells	Input Range	Nominal $\|\delta\|$	Effective Q_μ, Output cells size	Output Cells	At CNR=10, Minimum WT for $q_\mu > 6\sigma_y$
0	$0<\|\delta\|<.025$	0.00	1/800 (1)	0	-
1	$.025<\|\delta\|<.075$	0.05	1/200 (4)	1-4	31
2	$.075<\|\delta\|<.125$	0.10	1/100 (8)	5-12	31
3	$.125<\|\delta\|<.175$	0.15	3/200 (12)	13-24	31
4	$.175<\|\delta\|<.225$	0.20	1/50 (16)	25-40	31
5	$.225<\|\delta\|<.275$	0.25	1/40 (20)	41-60	31
6	$.275<\|\delta\|<.324$	0.30	3/100 (24)	61-84	31
7	$.325<\|\delta\|<.375$	0.35	7/200 (28)	85-112	31
8	$.375<\|\delta\|<.425$	0.40	1/25 (32)	113-144	31
9	$.425<\|\delta\|<.475$	0.45	9/200 (36)	145-180	31
10	$.475<\|\delta\|<.525$	0.50	1/20 (40)	181-220	31
11	$.525<\|\delta\|<.575$	0.55	11/200 (44)	221-264	31
12	$.575<\|\delta\|<.625$	0.60	3/50 (48)	265-312	31
13	$.625<\|\delta\|<.675$	0.65	13/200 (52)	313-364	31
14	$.675<\|\delta\|<.725$	0.70	7/100 (56)	365-420	31
15	$.725<\|\delta\|<.775$	0.75	3/40 (60)	421-480	31
16	$.775<\|\delta\|<.825$	0.80	2/25 (64)	481-544	31
17	$.825<\|\delta\|<.875$	0.85	17/200 (68)	545-612	31
18	$.875<\|\delta\|<.925$	0.90	9/100 (72)	612-684	31
19	$.925<\|\delta\|<.975$	0.95	19/200 (76)	685-760	31
20	$.975<\|\delta\|<1.025$	1.00	1/10 (80)	761-840	31

This nice compatibility between uniform input and uniform output quantizations does not hold when an offset of $\Omega = D_{max}$ is used. Table A-2 details a compromise that results from designing a quantization scheme to involve 128 output cells by limiting the range over which quantization is performed. The numbers in the table are based on the following considerations. If the range over which A/D conversion is performed is limited to $100\Delta\%$, this corresponds to an input range of $|\delta|<a$. Now to require that the number of quantization levels be a power of 2 is to require that

$$\frac{\text{range}(\mu)}{Q_\mu} = \frac{2a}{(1-a)Q_D} = 2^m - 1 \qquad \text{(A-5)}$$

or

$$a = \frac{(2^m-1)Q_D}{2 + (2^m-1)Q_D} \quad . \qquad \text{(A-6)}$$

Thus when $Q_D = 1/20$ and seven bits of quantization have been selected, the resulting value of a is 127/167 and Table A-2 gives the quantization intervals. The implementation of this scheme proceeds as in Figure 2.2 of the text in which it is shown that the correlator output μ must be conditioned by the operations

$$u = G(\mu - \mu_{min}). \qquad \text{(A-7)}$$

These operations are performed so that the appropriate range of μ matches the input range of the quantizer (expressed as zero to full scale = F.S.).

It should be noted also that the equivalent input quantization cells shown in Table A-2 are not uniform in size. Therefore, the $W_N T$ product required for insuring small correlator output noise will vary from cell to cell as shown in the table. The design choice here would be the maximum required value of $W_N T$.

TABLE A-2 QUANTIZATION SCHEME FOR FM CORRELATOR OUTPUT USING OFFSET AND LIMITED QUANTIZER RANGE

Input Cells	Input Range	Nominal δ	Effective Q_μ, Output cell size	Output Cells	At CNR=10 Minimun WT for $q_\mu > 6\sigma_y$
-16	$\delta < -.78694$	-	-	"OVERFLOW(-)"	-
-15	$-.78694 < \delta < -.73666$	-.76048	1/167	0	118
-14	$-.73666 < \delta < -.69455$	-.71483	"	1	166
-13	$-.69455 < \delta < -.62423$	-.65758	2/167	2,3	62
-12	$-.62423 < \delta < -.56514$	-.59361	"	4,5	86
-11	$-.56514 < \delta < -.51316$	-.53842	"	6,7	110
-10	$-.51316 < \delta < -.46623$	-.48915	"	8,9	134
-9	$-.46623 < \delta < -.42309$	-.44424	"	10,11	158
-8	$-.42309 < \delta < -.38297$	-.40270	"	12,13	182
-7	$-.38297 < \delta < -.32726$	-.35451	3/167	14-16	96
-6	$-.32726 < \delta < -.27582$	-.30107	"	17-19	112
-5	$-.27582 < \delta < -.22780$	-.25143	"	18-20	128
-4	$-.22780 < \delta < -.18260$	-.20488	"	21-23	144
-3	$-.18260 < \delta < -.12595$	-.15380	4/167	24-27	92
-2	$-.12595 < \delta < -.07276$	-.09897	"	28-31	104
-1	$-.07276 < \delta < -.02247$	-.04728	"	32-35	116
0	$.02247 < \delta < .02537$	.00174	"	36-39	128
1	$.02537 < \delta < .07107$	.04847	"	40-43	140
2	$.07107 < \delta < .12559$	.09867	"	44-48	100
3	$.12559 < \delta < .17759$	.15188	"	49-53	109
4	$.17759 < \delta < .22738$	.20274	"	54-58	119
5	$.22738 < \delta < .27524$	.25154	"	59-63	128
6	$.27524 < \delta < .32136$	.29850	"	64-68	138
7	$.32136 < \delta < .37467$	.34828	6/167	69-74	104
8	$.37467 < \delta < .42598$	.40056	"	75-80	112
9	$.42598 < \delta < .47551$	.45096	"	81-86	120
10	$.47551 < \delta < .52343$	.49966	"	87-92	128
11	$.52343 < \delta < .56989$	.54684	"	93-98	136
12	$.56989 < \delta < .62241$	.59637	7/167	99-105	107
13	$.62241 < \delta < .67329$	.64805	"	106-112	114
14	$.67329 < \delta < .72266$	.69815	"	113-119	121
15	$.72266 < \delta < .76388$	.74339	6/167	120-127	172
16	$\delta > .76388$	-	-	"OVERFLOW(+)"	-

REFERENCES

[1] J. S. Lee and L. E. Miller, "Detection Performance of an FM Correlator," JTR-79-03, J. S. Lee Associates, Inc., Arlington, Va., March 1979 (AD-A067751).

[2] J. Klapper and E. J. A. Kratt, "A New Family of Low-Delay FM Detectors," IEEE Trans. on Communications, Vol. COM-27, No. 2, February 1979, pp. 419-429.

[3] N. Abramson, "Bandwidth and Spectra of Phase-and-Frequency-Modulated Waves," IEEE Trans. on Communications Systems, Vol. CS-11, No. 6, December 1963, pp. 407-414.

[4] L. E. Miller, "Computing R.O.C. for Quadratic Detectors," Naval Surface Weapons Center, Technical Report TR-76-148, Silver Spring, Md., October 1976.

[5] W. W. Peterson and E. J. Weldon, Error-Correcting Codes, 2nd Edition. Cambridge, Mass.: MIT Press, 1972.

[6] R. Gold, "Optimal Binary Sequences for Spread Spectrum Multiplexing," IEEE Trans. on Information Theory, Vol. IT-13, No. 4, October 1967, pp. 619-621.

[7] B. Arazi, "Self Synchronizing Digital Scramblers," IEEE Trans. on Communications, Vol. COM-25, No. 12, December 1977, pp. 1505-1507.

[8] James J. Spilker, Jr., Digital Communications by Satellite. Englewood Cliffs: Prentice-Hall, Inc., 1977.

[9] Andrew J. Viterbi, Principles of Coherent Communications. New York: McGraw-Hill Book Company, 1966.

[10] William C. Lindsey and Marvin K. Simon, Telecommunication Systems Engineering. Englewood Cliffs: Prentics-Hall, Inc., 1973.

[11] Floyd M. Gardner, Phaselock Techniques, New York: John Wiley & Sons, Inc., 1966.

[12] Herbert Taub and Donald L. Schilling, Principles of Communications Systems. New York: McGraw-Hill Book Company, 1971.

[13] Robert M. Gagliardi, Introduction to Communications Engineering. New York: John Wiley & Sons, Inc., 1978.

[14] S. L. Goldman, "Second-Order Phase-Lock-Loop Acquisition Time in the Presence of Narrow-Band Gaussian Noise, IEEE Trans. on Comm., Vol. COM-21, No. 4, April 1973, pp. 297-300.

[15] Walter J. Gill, "A Comparison of Binary Delay-Lock Tracking-Loop Implementations" IEEE Trans. on Aerosp. and Elect. Sys., Vol. AES-2, No. 4, July 1966, pp.415-424.

AD-A092 940 LEE (J S) ASSOCIATES INC ARLINGTON VA F/G 17/2
FM CORRELATOR SPECTRAL DATA TRANSFER BY SCRAMBLED TRANSMISSION --ETC(U)
OCT 80 J S LEE, S TSAI, L E MILLER N00014-80-C-0129
UNCLASSIFIED JTR-80-03 NL

[16] Charles R. Cahn, "Comparison of Phase Tracking Schemes for PSK," MX-TM-3111-71, Magnavox Research Labs, Torrance, California, 15 July 1971 (the body of this report is published in Proc. International Telemetering Conf. 1971).

[17] Donald R. Hummels, "Some Simulation Results for the Time to Indicate Phase Lock," IEEE Trans. on Commun., Vol. COM-20, No. 1, February 1972, pp. 37-43.

[18] Jhong S. Lee and James H. Hughen, "An Optimum Phase Synchronizer in a Partially Coherent Receiver," IEEE Trans. on Aerosp. and Elect. Sys., Vol. AES-7, No. 4, July 1971, pp. 652-661.

[19] C. A. Filippi and R. H. French, "Study of Optimum Performance Characteristics of PCM Bit Synchronization Systems," Final Report on Contract NAS5-20264, Dec. 1973, Magnavox, Silver Spring, Md.

[20] Frank Neuman and Larry Hofman, "New Pulse Sequences with Desirable Correlation Properties," NTC '71 Conference Record, pp. 277-282.

[21] James L. Massey, "Optimum Frame Synchronization," IEEE Trans. on Comm., Vol. COM-20, No. 2, April 1972, pp. 115-119.

[22] Reference Data for Radio Engineers, 5th Ed., Indianapolis: Howard W. Sams & Co., Inc., 1968.

[23] F. S. McCartney and E. K. Heist, "FLTSATCOM Program Review: Requirements, Design and Performance," EASCON '78 Conference Record, pp. 442-452.

[24] Leo M. Keane and David W. Lipke, "Maritime Satellite Communications," in R. G. Gould and Y. F. Lum (eds.), Communications Satellite Systems: An Overview of the Technology. New York: IEEE Press, 1976.

[25] Jhong S. Lee, Robert H. French, and Yoon K. Hong, "Error Performance of Differentially Coherent Detection of Binary DPSK Data Transmission on the Hard-Limiting Satellite Channel," Final Report on Contract N00014-77-C-0056, Report No. JTR-79-06, J. S. Lee Associates, Inc., August 1979 (AD-A073377).

[26] Wilbur L. Pritchard, "Satellite Communication--An Overview of the Problems and Programs," Proc. IEEE, Vol. 65, No. 3, March 1977, pp. 294-307.

[27] W. B. Davenport, Jr., "Signal-to-Noise Ratios in Band-Pass Limiters," Journal of Applied Physics, Vol. 24, No. 6, June 1953, pp. 720-727.

[28] S. Lin, An Introduction to Error-Correcting Codes. Englewood Cliffs: Prentice-Hall, 1970.

[29] J. L. Massey, Threshold Decoding. Cambridge, Mass.: MIT Press, 1963.

[30] A. J. Viterbi, "Convolutional Codes and Their Performance in Communication Systems," IEEE Transactions on Communication Technology, Vol. COM-19, No. 5, October 1971, pp. 751-772.

[31] G. D. Forney, "Coding and Its Application in Space Communications," IEEE Spectrum, Vol. 7, June 1970, pp. 47-58.

[32] M. E. Mitchell, "Performance of Error-Correcting Codes," IRE Transactions on Communications Systems, Vol. CS-10, No. 1, March 1962, pp. 72-85.

[33] D. Chase, "A Class of Algorithms for Decoding Block Codes with Channel Measurement Information," IEEE Transactions on Information Theory, Vol. IT-18, No. 1, January 1972, pp. 170-182.

[34] Harris Corporation, "Coding for Navy UHF Satellite Communications (Phase I)", Final Report, April 1977.

[35] J. A. Heller and I. M. Jacobs, "Viterbi Decoding for Satellite and Space Communication," IEEE Transactions on Communication Technology, Vol. COM-19, No. 5, October 1971, pp. 835-848.


FILMED
8