"The Investigation of Bandwidth Efficient coding and modulation techniques"

NAG 5-1392: Final Report

NMSU Project Leader: Dr. William Osborne

NMSU Project Engineers:

Gerry Stolarczyk
Ted Wolcott
Brian Kopp

Maneul Lujan, Jr. Space Tele-Engineering Program
Department of Electrical Engineering
New Mexico State University
Las Cruces, NM
Telephone: 505-646-3012

Introduction

The NMSU Center for Space Telemetering and Telecommunications systems has been, and is currently, engaged in the investigation of trellis-coded modulation (TCM) communication systems [1] [2] [3] [4] [5] [6] [7] [8] [9] [10]. In particular, TCM utilizing M-ary phase shift keying is being studied. One particular aspect of MPSK TCM that is being evaluated is synchronization. Although proper synchronization is often assumed in theoretical discussions of MPSK TCM systems, obtaining it is a complicated process. In an MPSK TCM receiver several parameters that concern the timing of the signal must be recovered in order to begin decoding. In general there are three levels of synchronization in an MPSK TCM system. Carrier synchronization, symbol synchronization, and branch synchronization all must be obtained before any TCM data can be correctly decoded. The first of these, carrier synchronization, is generally considered to be the most difficult.

The study of carrier synchronization in an MPSK TCM environment, or in MPSK systems in general, has been one of the two main thrusts of this grant [5] [11]. This study has involved both theoretical modelling and software simulation of the carrier synchronization problem. The second area of concentration on the grant has also involved a carrier synchronization problem. However, it has encompassed a hardware project instead of a software one. To show that 8-PSK TCM can be used on an existing satellite channel, a NASA White Sands ground terminal (WSGT) high rate demodulator (HRD) has been modified. The HRD's original purpose was to support binary PSK (BPSK) and quadrature (QPSK) demodulation and carrier synchronization. The grant has supported the design and implementation of a modification to an HRD to allow it to perform 8-PSK demodulation and carrier synchronization. The modified HRD was then used on an existing tracking and data relay satellite (TDRS) channel in a demonstration of 8-PSK TCM.

The first part of this report is a discussion of the MPSK carrier synchronization problem. The available solutions are considered and the chosen solution, a decision-directed carrier tracking loop, is then described in depth. In the second part, the theoretical modelling and software simulations of the solution are presented. The theoretical studies that have been conducted have concentrated heavily on using mathematical models to analyze different aspects of a maximum a posteriori (MAP) MPSK carrier tracking loop, e.g., squaring loss. The carrier synchronization simulations have been conducted on a desktop workstation and have demonstrated some interesting characteristics of MPSK carrier tracking. The simulations have verified the carrier tracking models are accurate and confirmed that the mathematical approximation for MPSK squaring loss that is being used is accurate. The final part contains a discussion of the HRD hardware modification project. The original HRD mode of operation is reviewed and then the modifications are presented. And finally, the modified HRD's performance is examined. The report ends with some conclusions about the results of both projects and some areas of study that remain to be explored.

Part 1: MPSK Carrier Synchronization

Phase shift keying a sinusoid in order to transmit digital information results in a double-sideband suppressed carrier signal. The absence of the carrier in the spectrum makes coherent demodulation more difficult. There are several methods that are available to remedy this problem.

The first of the available methods is the "times N" loop. In a times N loop, shown in Figure 1, the MPSK signal is multiplied by itself N times. The N in the loop's name refers to the M in MPSK. By performing this multiplication the data is effectively multiplied out resulting in a spectral component at a frequency of N times the carrier that can be tracked. Unfortunately, as N, i.e., M, increases the frequency at which the carrier phase tracking loop must operate increases. For example, a 370 mega-Hertz (MHz) 16-PSK signal would require a "times 16" loop. The quiescent frequency in the loop would then be approximately 6 Gigahertz. Implementing a carrier tracking loop at such a frequency can be prohibitory because of the difficulties involved with working at such a high frequency.

Another method which can be used to achieve carrier synchronization is feed-forward synchronization [12]. The most noticeable feature of the feed-forward system is that it contains no phase-lock loops. The system estimates carrier phase by calculating its arctangent over 2N+1 samples where N is varied depending on the signal to noise ratio. The feed-forward system is shown in Figure 2. This system is useful in burst communication applications since the possibility of hang-up during acquisition can be significantly reduced. In evaluating the implementation requirements it is noted that there are four phase detectors required to receive the data. Further, the phase recovery unit requires a substantial amount of digital circuitry.

The third method that will be discussed is the decision-directed estimation loop. This loop has several variation. Certain types of Costas loops, demodulation/remodulation loops, reverse modulation loops, and MAP phase estimation loops are all forms of decision-directed loops. These loops recover some form of the transmitted data. This decision, about what was transmitted, is then used, in varying ways, to effectively remove the modulation.

One particular type of Costas loop, that was simulated during the course of the project is the Leclert and Vandamme variation [13]. It is shown in Figure 3. An error signal that drives a VCO is calculated from several hard-limiter processors, each related to the recovered data. The hard limiters provide the inputs to dual polar quantizers which estimate the transmitted phase. These estimates are then used along with the hard limiter outputs to calculate an error signal.

Two more types of decision-directed loops are the demodulation/remodulation loop and reverse modulation loop. In the former, the quadrature, baseband, data is recovered and then modulated onto the local VCO. This locally generated MPSK signal is then phase compared with the received signal. If good decisions are being made then the only phase difference between the two signals will be due to a carrier phase offset. In the reverse modulation loop recovered baseband data is "reverse" modulated on to the received signal. The resultant signal will have a spectral component at the carrier frequency that can be tracked. Further, in reverse modulation loops, like feed-forward systems, the chance of hang-up during acquisition can be avoided.

The fourth type of decision-directed loop that will be discussed is the MAP phase estimation loop. This loop uses an optimum a posteriori phase estimation process to determine what the most likely transmitted signal phase was. This estimate, utilized in rectilinear form, is then used to calculate an error signal to drive a local oscillator. The MAP phase estimation loop is shown in Figure 4.

In deciding which loop to use to study the MPSK carrier synchronization problem, several criteria were considered. First, the adaptability of the technique to MPSK systems was evaluated. Hardware requirements played a major roll in this evaluation. "Times N" loops are adaptable to MPSK environments but at very high costs. To support an M of 2,4,8, or 16 would, in effect, require four receivers if it were done in analog circuitry. A digital implementation would require substantial circuitry as well, and some form of wide bandwidth numerically controlled oscillator to achieve the range of from twice the carrier to 16 times the carrier.

The overall circuit complexity was considered as well. Although for modelling purposes this is perhaps not as important a consideration, the extension of this research into the construction of actual hardware made it pressing to consider overall complexity. The last loop described, the decision-directed MAP phase estimation loop has the simplest circuitry, relative to the other loops when considering an MPSK application. The MAP loop has two phase detectors, and one controlled oscillator. In a part-digital implementation of an MPSK MAP loop two programmable memories can be used to calculate the error signal for all M. An equivalent feed-forward system would require another, fixed, oscillator and four phase detectors. The feed-forward system would also require more digital circuitry to compute its estimation of the phase. Similarly, the Leclert and Vandamme Costas loop variation would require more digital circuitry with its dual polar quantizers while not increasing the analog circuitry required. The demodulation/remodulation loop and reverse modulation loops also substantially increase the amount of required hardware, particularly the analog hardware, in achieving an MPSK system.

At this stage of the evaluation it was decided that some form of decision-directed loop would be used. Because of the circuit complexity of the demodulation/remodulation loop (it contains two phase locked loops) and that of the reverse modulation loop it was further decided that either the Leclert and Vandamme or the MAP phase estimation loop would be used.

Preliminary simulations of the Leclert and Vandamme Costas variation showed that its performance did not exceed that of the MAP phase estimation loop. In fact, within a nearly negligible margin its simulation performance was inferior to that of the MAP estimation loop. This performance difference is attributable to the Leclert and Vandamme's all digital circuity which results in a loop that performs like a "bang-bang" servo control. It should be stressed that the difference in performance was minimal. The difference, however, in required digital circuitry was substantial enough to allow a choice to be made between the two final contenders. The MAP phase estimation loop was chosen for its performance, and ease of implementation. Its easier implementation also made analysis and modelling that much more obtainable.

To further understand the analysis and modelling of the MAP phase estimation loop in the next part of this report a brief discussion of the mechanics of the loop and of the application of MAP estimation to carrier phase is now presented. As shown in Figure 4, the MAP estimation of carrier phase uses a quadrature channel carrier recovery loop and a single polar phase estimator. Using the output of quadrature channel matched filters, the polar phase estimator makes a hard decision as to what modulation data was transmitted during the last symbol period. This estimate is then used in conjunction with the filter outputs to generate an error signal. The error signal is passed to a loop filter and voltage controlled oscillator (VCO) which generates the local carrier reference for demodulation.

The MAP estimator performs several functions in making its decision as to what was transmitted. First it obtains the phase angle that is conveyed with I and Q by taking the arctangent of the ratio Q/I. The angle is then compared with each possible modulation angle (e.g., in 8-PSK the modulation angles could be chosen as $\pi/16$, $3\pi/16$, $5\pi/16$, $7\pi/16$, $9\pi/16$, $11\pi/16$, $13\pi/16$, and $15\pi/16$). The modulation angle that is closest to the received angle is selected as the maximum a posteriori estimate to the transmitted angle. The cosine and sine of the estimate, $\hat{I}$ and $\hat{Q}$, are computed and used to generate an error signal.

To form the error signal the output of the I and Q matched filters are multiplied by the sine and cosine angle estimates, respectively. This is shown in Figure 4. The difference between the two products is the error signal. This is also shown, in Figure 4, as the input to the filter. In the absence of symbol errors this is the traditional phase-locked loop (PLL) tracking error quantity which occurs with a mixing phase detector and sinusoidal inputs. With the use of a filter whose Laplace transform is $1+a/s$ and in the absence of symbol errors this tracking system performs identically to a 2nd order PLL.

The above mentioned phase estimation process utilizes the MAP estimation technique but is actually an approximation to the exact MAP solution. To understand the reason for this approximation it is necessary to explain the MAP estimation process in detail. The explanation follows the consideration that the goal is to obtain an error signal which can be used to track phase errors between a received PSK signal and a local receiver VCO. To begin the explanation consider a received PSK signal written in the form:

$$R(t) = S[t,\theta_T(t),\theta_m(t)] + N(t) \qquad (1)$$

where S[•] is the PSK signal, t is time, $\theta_T(t)$ is the transmitter carrier phase referenced to the receiver VCO, $\theta_m(t)$ represents the PSK symbol, and N(t) is additive white Gaussian noise. A PSK decision-directed loop estimates the transmitter phase and uses this estimate to remove the modulation from the received carrier in the process generating an error signal which contains information regarding only the transmitter phase offset from the receiver VCO.

The optimum estimate of the transmitter phase, θ_T, is determined using MAP estimation. The estimate of this phase is denoted $\hat{\theta}(R(t))$ or just $\hat{\theta}(R)$. The a posteriori information is the received signal. Using this information, the estimate that maximizes the a posteriori conditional density, $P(\theta_T|R)$ is called the MAP estimate [14, pg. 57]. Using the monotonic nature of the natural logarithm, a necessary, but not sufficient, condition for the maximum can obtained by determining the estimate for which

$$\left.\frac{\partial \ln(P(\theta_T|R))}{\partial \theta_T}\right|_{\theta_T = \hat{\theta}_{MAP}(R)} = 0 \qquad (2).$$

Technically, to complete the test for maximizing $P(\theta|R)$, any estimate that satisfies (2) must also be evaluated to see if it is the absolute maximum.

Using Bayes' theorem in this problem

$$P(\theta_T|R) = \frac{P(R|\theta_T)\ P(\theta_T)}{P(R)} \qquad (3)$$

equation (2) can be written

$$\left.\frac{\partial \ln(P(R|\theta_T))}{\partial\theta_T}\right|_{\theta_T = \hat{\theta}_{MAP}(R)} + \left.\frac{\partial \ln(P(\theta_T))}{\partial\theta_T}\right|_{\theta_T = \hat{\theta}_{MAP}(R)} - \left.\frac{\partial \ln(P(R))}{\partial\theta_T}\right|_{\theta_T = \hat{\theta}_{MAP}(R)} = 0 \qquad (4).$$

The third partial derivative is zero. The second partial derivative will be zero since the transmitter phase is uniformly distributed in the interval $[0,2\pi]$. In that case (4) reduces to

$$\left.\frac{\partial \ln(P(R|\theta_T))}{\partial \theta_T}\right|_{\theta_T = \hat{\theta}_{MAP}(R)} = 0 \qquad (5).$$

From (5) it is seen that maximizing the a posteriori density function is equivalent to maximizing the transitional density function. This transitional density function is often referred to as the "likelihood function" [14, pg. 65]. Therefore, in this case the MAP estimate is equivalent to the maximum likelihood estimate.

Equation (5) represents an error signal that can be used to track phase. In analyzing (5) the type of modulation will be restricted to M-ary PSK with M = 8. From Simon [15] the left side of (5), before the substitution is conducted, can be stated

$$\frac{\partial \ln(P(R|\theta_T))}{\partial \theta_T} = \frac{\left[\sqrt{2}\cos(\frac{3\pi}{8})\,I\right]\tanh\left[\sqrt{2}\cos(\frac{\pi}{8})Q\right]}{1 + \frac{\cosh\left[\sqrt{2}\cos(\frac{3\pi}{8})Q\right]\cosh\left[\sqrt{2}\cos(\frac{\pi}{8})\,I\right]}{\cosh\left[\sqrt{2}\cos(\frac{\pi}{8})Q\right]\cosh\left[\sqrt{2}\cos(\frac{3\pi}{8})\,I\right]}} -$$

$$\frac{\left[\sqrt{2}\cos(\frac{\pi}{8})Q\right]\tanh\left[\sqrt{2}\cos(\frac{3\pi}{8})\,I\right]}{1 + \frac{\cosh\left[\sqrt{2}\cos(\frac{3\pi}{8})Q\right]\cosh\left[\sqrt{2}\cos(\frac{\pi}{8})\,I\right]}{\cosh\left[\sqrt{2}\cos(\frac{\pi}{8})Q\right]\cosh\left[\sqrt{2}\cos(\frac{3\pi}{8})\,I\right]}} + \frac{\left[\sqrt{2}\cos(\frac{\pi}{8})\,I\right]\tanh\left[\sqrt{2}\cos(\frac{3\pi}{8})Q\right]}{1 + \frac{\cosh\left[\sqrt{2}\cos(\frac{\pi}{8})Q\right]\cosh\left[\sqrt{2}\cos(\frac{3\pi}{8})\,I\right]}{\cosh\left[\sqrt{2}\cos(\frac{3\pi}{8})Q\right]\cosh\left[\sqrt{2}\cos(\frac{\pi}{8})\,I\right]}} -$$

$$\frac{\left[\sqrt{2}\cos(\frac{3\pi}{8})Q\right]\tanh\left[\sqrt{2}\cos(\frac{\pi}{8})\,I\right]}{1 + \frac{\cosh\left[\sqrt{2}\cos(\frac{\pi}{8})Q\right]\cosh\left[\sqrt{2}\cos(\frac{3\pi}{8})\,I\right]}{\cosh\left[\sqrt{2}\cos(\frac{3\pi}{8})Q\right]\cosh\left[\sqrt{2}\cos(\frac{\pi}{8})\,I\right]}} \qquad (6)$$

where

$$Q = K\int_0^T R(t)\sin(\omega_c t + \theta_o)\,dt$$

and $$I = K\int_0^T R(t)\cos(\omega_c t + \theta_o)\,dt \qquad (7)$$

are the quadrature arm correlator outputs in the receiver and θ_0 is the VCO phase. Generating such an error signal in hardware is difficult. Fortunately, approximations can be used to simplify the implementation.

For the case when high signal to noise ratios (SNR) are used tanh(x) can be approximated by sgn(x), the sign of x. Similarly cosh(x) can be approximated by 0.5exp(x). In the high SNR implementation the MAP estimation procedure has essentially been reduced to determining the best estimate of the transmitted data components. These are found by taking the arctangent of $\frac{Q}{I}$. This quantity is then compared with all eight of the possible transmitted phases. Which ever one of the eight possible transmitted phases is closed to the arctangent of $\frac{Q}{I}$ is selected as the best estimate of what was transmitted. The quadrature and inphase estimates are computed from this phase and passed to the cross-arm multipliers in the loop. As was mentioned previously, the error signal that results is $Q\hat{I} - I\hat{Q}$.

This approximation is appropriate for the SNR's being considered for MPSK operation. Now with an understanding of the decision-directed MAP carrier phase estimation process it is possible to analyze and simulate it, with respect to specific characteristics of interest. This the subject of the next part of this report.

Part 2: Modelling and Simulating MPSK Carrier Synchronization

The analytical investigation into using MAP estimation for the purpose of tracking the carrier phase of an MPSK system has concentrated on the development of a meaningful mathematical model which can be easily simulated for any M. Once a simulator was constructed it was used to study squaring loss, an important performance characteristic in the study of carrier synchronizers. The mathematical model is best understood by reconsidering the error signal mentioned in Part 1 to be a function of the phase detector characteristic. By constructing the error signal in this manner the model can be easily simulated. Consider the input to the MAP synchronizer in Figure 4 as

$$r(t) = s(t) + w(t) \tag{8}$$

where s(t) is the M-PSK transmitted signal

$$s(t) = \sqrt{\frac{2E_S}{T}}\cos(\omega_c t + \theta_m + \varphi_i) \tag{9}$$

and w(t) is white noise with power spectral density $\frac{N_O}{2}$. The energy per symbol is denoted as E_S and the symbol period is T. The carrier frequency is ω_c and the M-PSK symbol that is transmitted is designated as θ_m where

$$\theta_m = \frac{2\pi i}{M} \qquad i = 0,1,.....M-1 \qquad (10).$$

The term φ_i is the phase offset of the received signal. The corresponding integrator outputs are

$$I = g \int_0^T r(t) \cos(\omega_c t + \varphi_o) \quad \text{and}$$

$$Q = g \int_0^T r(t) \sin(\omega_c t + \varphi_o) \tag{11}$$

where φ_o is the phase offset of the reference VCO and g is the gain of the integrators. These two equations can be rewritten as

$$I = I_s + N_i \quad \text{and} \quad Q = Q_s + N_q \tag{12}$$

where N_i and N_q are zero mean Gaussian random variables with equal variances of $\sigma^2 = \frac{N_o}{2E_s}$. To evaluate I_s and Q_s we let $g = \sqrt{\frac{2}{E_sT}}$ and $\varphi = \varphi_i - \varphi_o$. Now performing the integration and inserting the values for g and φ we have

$$I_s = \cos(\theta_m + \varphi) \quad \text{and} \quad Q_s = \sin(\theta_m + \varphi) \tag{13}.$$

Since the signal components I_s and Q_s are separate from their corresponding noise terms N_i and N_q, the error signal

$$e(t) = Q\hat{I} - I\hat{Q} \tag{14}$$

becomes

$$e(t) = Q_s\hat{I} + N_q\hat{I} - I_s\hat{Q} - N_i\hat{Q} \qquad (15).$$

The tracking loop's error signal is comprised of two quantities of information. The phase detector characteristic and a noise term. The phase detector characteristic is the expected value of the error signal and since the noise terms N_i and N_q have expected values of 0

$$PD(\varphi) = E\left[e(t)\right] = E\left[Q_s\hat{I} - I_s\hat{Q}\right] \qquad (16).$$

The error signal can now be expressed as

$$e(t) = PD(\varphi) + N_e(t) \qquad (17)$$

where $\quad N_e(t) = N_q\hat{I} - N_i\hat{Q} \qquad (18).$

Having obtained an expression for the error function in terms of the phase detector characteristic the baseband model can now be constructed as shown in Figure 5(a). The phase offset from the VCO is subtracted from the received phase and this difference is what is passed to the phase detector. The noise term $N_e(t)$ is subsequently added and the result is passed to the loop filter.

Figure 5(b) shows the modification to the baseband model that is made to perform a simulation of the tracking loop. In effect, equation (15) is calculated to determine the error signal. The noise terms and the data are obtained using simple random number generators. The error is passed to a discrete time transfer function that performs the filtering and VCO integration. The resulting phase is retained for the following iteration. It should be noted that the phase detector gain at high SNR was fixed at 1.

The simulator that was constructed utilizes the C programming language. The code is broken up into several subroutines, each performing clearly defined functions in the tracking process. This made it possible to conduct simulation changes and variations efficiently. There are as many as 12 subroutines in a simulation depending on what features are required for a particular test. The simulation is a time simulation that has a starting point and end point determined by the number of samples to be taken. For example, in evaluating a phase-step response of the tracking loop 10,000 samples may be used, where as, in a phase error variance simulation where independent samples are needed for the variance calculation, upwards of 64,000 samples may be used. An example simulation code, that calculates phase error variance, is contained in Appendix 1.

To demonstrate that the simulations were accurate, normalized step responses of the carrier tracking loop simulator for BPSK, QPSK, 8PSK, and 16 PSK formats and random phase data were obtained. The step responses were compared to theoretical responses for second order control loops and determined to be accurate [16, pg. 49].

The random phase data collected with the simulator formed the second simulator test. The most important performance parameter associated with carrier tracking loops is the phase jitter in the loop versus SNR, since the jitter results in estimation performance loss. The variance is due to two factors. The first occurs in all PLL's and reflects the presence of noise in the incoming signal. The second factor is attributable to the use of a particular type of modulation and carrier tracking loop. This second factor is most often referred to as the squaring loss (the term was originally applied to BPSK squaring carrier loops). The analytical solution for BPSK and QPSK squaring loss is available in the current literature. The exact solution for the squaring loss in the 8 and 16 PSK loop is difficult to obtain and is, to date, unavailable. However, an approximation to the exact solution, which neglects self-noise, has proved adequate in analyzing the 8 and 16 PSK carrier tracking loop simulations. The phase error variance can be expressed as

$$\sigma_{\varphi}^2 = \frac{No}{2Es}\frac{BL}{SR} \bullet SL^{-1} \qquad (19)$$

where Es/No is the ratio of the energy per symbol to noise spectral density, SR is the symbol rate, BL is the loop noise bandwidth, and SL is the squaring loss of the phase detector. The squaring loss is the increase in phase jitter within the loop over a conventional PLL of the same bandwidth due to the nonlinearity involved in the phase detection process, i.e. the phase detector output PD is given by

$$PD(\varphi) = [Q\hat{I} - I\hat{Q}] \qquad (20)$$

where, I & Q are the analog outputs of the I & Q channel matched jitters, $\hat{I}$ & $\hat{Q}$ are the hard decision channel outputs and φ is the phase error. The squaring loss is generated from two physical actions and these are:

(1) The phase detector gain (even at constant signal levels) depends upon the SNR through I and Q and goes down as the error rate goes up. This increases the jitter in the loop because there is less signal to track at a given SNR.

(2) The variance of the equivalent noise term in the loop is also affected by the presence of errors. Generally, this effect lessens the phase error by a slight amount.

The affects upon the variance of noise are very secondary, as will be shown for a QPSK loop. It can be shown by linear loop analysis that the squaring loss neglecting the effects of errors on the noise term is given by

$$SL\,(SNR) = 1/G^2 \tag{21}$$

where G is the gain of the phase detector at zero phase error normalized to one at high SNR. The model used for evaluating phase detector gain is shown in Figure 6. With this model I & Q are ready shown to be independent Gaussian random variables with statistics given by:

$$m_I = \mathrm{Cos}(\theta m + \varphi) \qquad \sigma_I^2 = \frac{No}{2E_s}$$

$$m_Q = -\mathrm{Sin}(\theta m + \varphi) \qquad \sigma_Q^2 = \frac{No}{2E_s}\,.$$

The phase detector characteristic, PD(φ), is the expected value of the error signal shown in Figure 6, i.e.,

$$PD(\varphi) = E[I\mathrm{Sin}\theta_m - Q\,\mathrm{Cos}\theta_m] \tag{22}$$

To see how the phase detector works consider the no noise case and a static phase error of φ which is small compared to $2\pi/M$. Then

$$I = \mathrm{Cos}(\theta_m + \varphi)$$

$$Q = -\mathrm{Sin}(\theta_m + \varphi)$$

$$\hat{\theta} = \theta_m$$

and PD (φ) is given by

$$PD(\varphi) = \mathrm{Cos}(\theta_m + \varphi)\mathrm{Sin}\theta_m + \mathrm{Sin}(\theta_m + \varphi)\mathrm{Cos}\theta_m$$

or $\quad PD(\varphi) = \mathrm{Sin}(\varphi) \quad$ for no noise. $\quad$ (23)

Of course, if the phase error is larger than π/M then $\hat{\theta}$ will be equal to the value of modulation phase nearest to $\theta_m + \varphi$ and thus the phase detector characteristic is periodic in φ with period of $2\pi/M$. The phase detector characteristic in (22) can be evaluated by averaging over the noise and the data as follows

$$PD(\varphi) = E\{[\mathrm{Cos}(\theta_m + \varphi) + N_I]\mathrm{Sin}\hat{\theta} + [\mathrm{Sin}(\theta_m + \varphi) + N_Q]\,\mathrm{Cos}\hat{\theta}\}$$

or $$PD(\varphi) = E\{\mathrm{Sin}(\varphi - \frac{2\pi i}{M})\} \tag{24}$$

where i is the hard decision at the output of the polar estimator. It follows that

$$PD(\varphi) = \sum_{i=0}^{M-1} \mathrm{Pr}(i|0)\ \mathrm{Sin}\ (\varphi - \frac{2\pi i}{M}) \tag{25}$$

where we have fixed the transmitted data symbol at zero since PD(φ) is totally symmetrical with respect to transmitted symbols and Pr(i|0) is the probability of a hard decision on phase being in the i'th sector given $\theta_m = 0$. This expression can be evaluated for all forms of MPSK using the fact that the density function of phase is given by:

$$r(\psi) = \frac{1}{2\pi} e^{-Es/No} [1 + Z\sqrt{2\pi}\, e^{Z^2/2}\, Q(-Z)]$$

$$Z = \sqrt{\frac{2Es}{No}}\ \mathrm{Cos}\ (\psi) \tag{26}$$

and performing the indicated integration of (26) numerically to obtain Pr(i|0). A typical set of phase detector characteristics are shown in Figure 7 for an 8PSK loop. The periodicity and the gain reduction with decreasing SNR are apparent. The gain of the phase detector is obtained by calculating the slope of PD(φ) at

φ=0. The resulting squaring loss for QPSK is shown in Figure 8 for this technique and a more exact analysis from Hinedi and Lindsey [17]. As discussed previously, the squaring loss given by the approximation and the result including the effects of noise correlation are within .5dB Es/No of each other for all reasonable values of Es/No. The squaring loss calculated by this technique for all 4 loop types is shown in Figure 9. Note, the very large differences in squaring loss between the various schemes. At a SNR where squaring loss may be negligible for BPSK it can be prohibitory for 16 PSK. Operating at an E_S/N_O of 10 dB incurs no loss in loop performance due to squaring loss for BPSK but for 16 PSK there is a more than 40 dB loss in loop performance that must be considered.

The simulator was used to measure the variance in the phase error versus SNR and these results are shown in Figures 10 and 11. The first plot shows the variance in phase error for BPSK and QPSK. The theoretical approximations for the variances, using the approach of (19) and (21) are plotted as well. Figure 11 shows the variance in phase error for 8 PSK and 16 PSK. The quantity, E_S/N_O , present in the plots, is PSK symbol energy to noise spectral density.

One aspect of the simulation that is worth noting is the change in loop noise bandwidth that occurs in the simulator as the SNR changes with constant signal level (perfect automatic gain control (AGC)). Since the phase detector gain is a function of error rate and hence SNR, as the SNR drops the phase detector gain drops and the corresponding loop noise bandwidth gets smaller. This in effect lowers the amount of jitter that is present when compared to a similar calculation made with a fixed loop bandwidth. The theoretical calculations of phase jitter based upon squaring loss and a linear model must take into account the changing loop noise bandwidth that occurs in the simulator (and in most actual loops), if they are to be compared with the phase error variances of the simulator. This was done for the theoretical estimates in Figures 10 and 11. It is interesting to note that this aspect of the simulator more actually mimics practice than the theory does. As a rule, the phase detector gain of a carrier tracking loop is not modified on the fly to account for a changing SNR operating condition in an attempt to keep the loop noise bandwidth constant.

Part 3: An 8-PSK TCM High Rate Demodulator

To demonstrate 8-PSK TCM through a NASA satellite channel a suitable demodulator was required. This grant provided the support necessary to fulfill the requirement. Two options were available. A new demodulator could be constructed or an existing one could be modified. In making the selection several aspects of the problem were considered. First, the design of a new demodulator would require a significantly greater investment of time and funding when compared to the latter option. Second, spare demodulators, suitable for modification, were available at the nearby WSGT. And Third, since the WSGT is the optimum location for conducting the 8-PSK TCM tests, a demodulator which is already designed for the TDRS system presents a compatible environment in which to create an 8-PSK demodulator.

The selected demodulator, an HRD, performs carrier recovery and coherent demodulation over primary symbol rates of 10 to 100 mega-symbols per second (Msps) for BPSK and balanced and unbalanced QPSK. A block diagram of the HRD is shown in Figure 12. The first step in the project was to study the operation of the HRD. It is a complicated receiver and to consider the modifications required to turn the HRD into an 8-PSK demodulator a brief functional description is required. The description follows the block oriented presentation of the HRD in Figure 12. For a more detailed functional description, the complete technical report on the HRD modification is included in Appendix 2.

The satellite downlink intermediate frequency (IF) is 370 MHz and has a power level of -30 to 0 deci-Bel milliwatts (dBm). This signal enters the A1 card in Figure 12 from the left. This card contains the IF filtering, IF amplification, IF

AGC amplifier, and actual data recovery circuitry. The filtered and gain-controlled IF signal is passed from the A1 card to the A4 PLL detector card. The A4 card contains a modified Costas loop structure to perform demodulation. When BPSK is used, the same data is modulated on both the I and Q channel so the demodulation structure is the same. The term "modified" refers to the extra stage of quadrature mixtures in the loop. This allows the loop to avoid performing multiplications of the baseband data at direct-current (DC) levels. This is done because DC, analog, multiplications are difficult to implement. The IF signal enters the A4 card, from the A1 card, at three points. Two of these are mixed with the VCO signal, in quadrature. The two resultant signals are low-pass filtered and sent to the filter cards, A2 and A3, where they are further filtered. The third IF input to the A4 card is split and each component is multiplied by the quadrature data estimates computed in the hard-limiters, U10 and U11. These two components are then mixed with the VCO signal in quadrature form. After low-pass filtering, the resultant signals are summed to create the error signal.

The two above mentioned resultant signals that leave the A4 card and progress to the filter cards, A2 and A3, are low-pass filtered on these cards to optimize the signal to noise ratio for a particular data rate. They are then returned to the A4 card for calculation of the data estimates in the hard-limiters, through some amplifiers on the A6 card. The reason for passing these signals to the A6 card is that they are used to calculate two control signals there. The first of these control signals is the lock detect signal which is used to disable the sweep circuit when phase lock is obtained. The second is the AGC signal which is returned to the A1 card to control the AGC amplifier.

The error signal that leaves the A4 card is sent to the loop filter/VCXO card, A7, where it is filtered and then passed to a VCO. The local carrier is generated here and then, after amplification, sent to the A5 card for distribution. The distributed local carrier is relayed from the A5 card to both the A1 and A4 cards. On the A1 card it is used in quadrature form to perform the coherent demodulation of the received signal. The received data leaves the HRD from A1.

The actual modifications to the HRD progressed in several stages. A complete description of the steps involved are discussed in Appendix 2, so only a brief summary is given here. The obvious modification that was required was to replace the hard-limiters with multi-level data estimators. A second, initially required, modification was to account for the use of a lower symbol rate of 1 Msps. The original modification design involved a single board structure where a polar quantizer was implemented by conducting the four analog comparator tests: $I \lesseqgtr 0$, $Q \lesseqgtr 0$, $I \lesseqgtr Q$, $I \lesseqgtr -Q$. These tests were conducted using analog comparators. To get the inputs to the comparators, I and Q, the inputs to the A2 and A3 cards were rerouted to a new printed circuit board (PCB) where they were low-pass filtered and then fed through amplifiers to the comparators, both of which were also located on the new PCB. The amplified signals were also relayed to the A6 card to replace the signals from the A2 and A3 cards. The comparator's outputs were used to generate the appropriate data estimates through operational-amplifiers. These estimates were then amplified to a level appropriate for driving the U2 and U8 mixers on the A4 card and then relayed to the A4 card. This new PCB was constructed and mounted in a canopy housing that was constructed over the HRD assembly.

The first version of the modified HRD had two problems which resulted in it being revised. The first was that the lock detector circuitry on the A6 card utilized two full-wave rectifiers which did not respond appropriately to the multilevel I and Q signals. As a result, not only did the lock-detect circuitry fail to operate, the AGC signal was incorrect as well. It was decided that rather than modifying the A6 card to respond correctly, it would be disabled. During testing the operator can easily disable the sweep circuitry and the AGC can be controlled using a variable attenuator attached to the HRD IF input.

The second problem in the original modification design concerned the polar quantizer. The implementation of the polar quantizer caused the loop to oscillate between lock points and saddle points in the 8-PSK constellation. To circumvent this problem independent linear quantizers were substituted for the polar quantizer. Since this represented a substantial design change and since the data estimation process for each channel was now separate, it was decided to build separate PCB's and place them in the HRD card slots previously occupied by the A2 and A3 cards. The linear quantizers use comparators as well, to decide which of the four possible levels were transmitted. This receiver design is sub-optimum when compared to MAP estimation of the carrier phase but since carrier tracking was not being evaluated as part of the overall 8-PSK TCM test, the resulting HRD performance was acceptable.

This new modified HRD tracked carrier phase at higher signal to noise ratios but was unable to acquire and hold lock at lower ratios, particular those of interest in the 8-PSK TCM tests. A very simple solution was implemented to correct for this. Again keeping in mind that carrier synchronization at low SNR's was not an object of testing, it was decided that the received IF signal should maintain a very large SNR at the input to the HRD. This allows synchronization to occur accurately. However, for demodulation and 8-PSK TCM testing, the received IF signal is split before entering the HRD. This split component has calibrated noise added to it, creating a noisy signal suitable for testing. This noisy signal is fed into the A1 card replacing the received IF signal input to the coherent demodulator circuitry.

This version of the HRD is the current 8PSK receiver structure and is shown in Figure 13. It has been tested at the NMSU telemetry laboratory and at the WSGT. The 8-PSK receiver has been used to successfully receive 8-PSK TCM through several TDRS satellites and on both S-band and Ku-band channels. Figure 14 shows the results of the uncoded 8-PSK 1Msps tests before and after the final modification to allow the HRD to track on a strong signal. It is clear that at carrier-to-noise-power ratios below 78 dB it would not have been possible to acquire or stay locked without the final modification. Taking into account the implementation loss present in the HRD the modification performs very close to theory.

Conclusions

In the first project that was made possible by this grant a through investigation of carrier synchronization in MPSK TCM systems was conducted. A candidate method for performing carrier synchronization was selected using practical criteria. This method is the method of decision-directed MAP carrier-phase estimation. The method was then analyzed and simulated in an effort to understand its function and explore its applicability to MPSK TCM systems. The research into MPSK carrier synchronization has uncovered some interesting characteristics of decision-directed MAP carrier-phase estimation loops for M equal to 8 or 16. The squaring loss approximation and simulation data indicate a substantial loss of SNR in the loop for both 8 and 16-PSK. In particular, for 16-PSK, it could be argued that this form of synchronization is not adequate for TCM operation on a satellite channel where SNR's are realitively low. A modification, such as the use of some form of coding in the carrier synchronization loop, may be an optional way to gain loop SNR back if 8 or 16-PSK is to be employed.

The variance data on carrier phase also contains important information on cycle slip. The simulator failed to maintain lock at threshold SNR's that were much higher for 8 and 16-PSK than they were for QPSK and BPSK. The sequences of simulator data points in Figures 10 and 11 begin where lock could be maintained. Further, it was necessary to narrow the loop bandwidth to maintain lock on the 8 and 16-PSK signals at reasonably low SNR's. This suggests that it might be very useful to study lock thresholds, i.e., phase variance, as a function of SNR and normalized loop bandwidth.

There are several more, as yet unexplored, areas of study that have been considered for further research. First, the modelling and simulation data should be verified in a hardware study. The squaring loss data and lock threshold conditions could both be checked in this manner. Second, a solution for 8 and 16-PSK squaring loss should be found. It may be possible to find a numerical solution with minimal effort. The third area of possible study goes back to the discussion of implementing MAP estimation with the use of a high SNR approximation. While this approximation is valid for the SNR's that MPSK TCM may operate at, a comparison between the high and low SNR approximations and the actual error signal would quantify any loss in making the approximations. This could be checked by simulation and by hardware implementation, through the use of digital signal processing.

The second project completed under this grant was the modification of a NASA WSGT HRD for the purpose of sending 8-PSK TCM through the TDRS system. Originally the modifications that were thought to be required were minimal. However as the project progressed, many different modifications were required to complete the work. The final version of modification that was excepted allowed for very accurate carrier synchronization by allowing the loop to run at a very high SNR while not interfering with changing SNR bit-error-rate tests. This turned out to be useful from a testing standpoint when utilizing an actual TDRS channel. Since calibrated noise was injected at the demodulator to vary the SNR it was not necessary to spend time commanding the TDRS to vary its SNR or to spend time calibrating the downlink SNR each time a new test point was desired.

The modifications of the data estimators from hard-limiters to multilevel estimators took several iterations. The result, while sub-optimum at low SNR's, did not impede the testing in any way because of the benefit of the high SNR input to the HRD tracking loop.

Testing of the entire 8-PSK TCM system has been nearly completed and the results are very encouraging. With the help of the HRD modification project it has been shown that 8-PSK TCM can be efficiently transmitted through an existing TDRS channel.

References

[1] Carden, Frank, M. Ross, W. Osborne and B. Kopp, "Fast TCM Decoding: Phase Quantizing and Integer Weighing", IEEE Transactions on Communications Theory, Publication Date TBD.

[2] Ross, Michael, William P. Osborne, Frank Carden, and Jerry L. Stolarczyk, "Pragmatic Trellis Coded Modulation: A Hardware Implementation Using 24-sector 8-PSK," Supercomm/ICC '92.

[3] Whittington, Joel and William Osborne, "A Phase Ambiguity Resolution Technique for TCM," MILCOM 1992.

[4] Osborne, William, "Multi-Phase TCM Implementation Complexity", Consultative Committee on Space Data Systems, Annual Meeting, Orlando, FL., Sept., 1991.

[5] Osborne, William, F. Carden, B. Kopp and M. Ross, "Multi-Mode Modem/Codec Designs", AIAA - Communication Satellite System Conference, 1991.

[6] Ross, Michael, Frank Carden and William P. Osborne, "Pragmatic Trellis Coded Modulation: A Simulation Using 24-sector Quantized 8-PSK," International Phoenix Conference on Computers and Communications, 1991.

[7] Ross, Michael and Frank Carden, "64-State TCM for Spectrally Efficient Space Communications" Proceedings of the NAECON-91 Conference, Vol. 1 of 3, pp. 178-184.

[8] Kopp, Brian and Frank Carden, " A Quantized Euclidean Soft-Decision Maximum Likelihood Sequence Decoder for TCM", Proceedings of the 1988 IEEE Military Communications Conference, Vol. 2, 35.7

[9] Kopp, Brian, "A Four-State Trellis-Coded 8-PSK Modulation Computer Simulation", Proceedings of the 1988 International Telemetering Conference, pg 401.

[10] Osborne, William, "Advanced Modulation and Coding Studies", Proceedings of the Strategic Avionics and Technology C&T Panel Workshop, NASA Johnson Space Flight Center, Texas, August, 1991.

[11] Kopp, Brian and William Osborne, "Synchronization in a Multi-Mode PSK Modem", Proceeding of Supercomm/ICC-92.

[12] Cowley, W.G., "Synchronization Algorithms for Digital Modems", Proceeding of the ASSPA-1989, pps. 836-840.

[13] Leclert, A., and Vandamme, P., "Universal Carrier Recovery Loop for QASK and PSK Signal Sets", IEEE Trans. on Comm., Vol Com-31, No. 1, January 1983.

[14] Van Trees, H. L., "Detection, Estimation, and Modulation Theory, Part I", John Wiley and Sons, 1968.

[15] Simon, M.K., "Further Results on Optimum Receiver Structures For Digital Phase and Amplitude Modulated Signals", Proceeding of the ICC-1978.

[16] Gardner, F. M., "Phaselock Techniques", John Wiley and Sons, 1979.

[17] Hinedi, S., and Lindsey, W., "On the Self-Noise in QASK Decision-Feedback Carrier Tracking Loops", IEEE Trans. on Comm., Vol Com-37, No. 4, April 89.

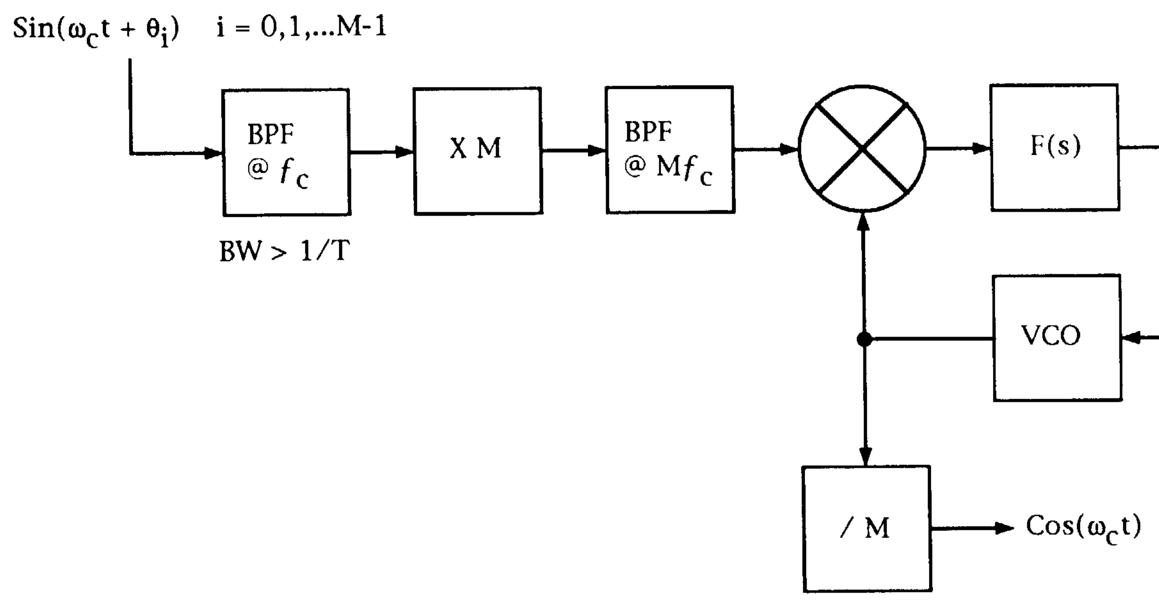

$\mathrm{Sin}(\omega_c t + \theta_i)$ $i = 0,1,\ldots M\text{-}1$
BPF @ f_c
BW > 1/T
X M
BPF @ Mf_c
F(s)
VCO
/ M
$\mathrm{Cos}(\omega_c t)$

Figure 1. "Times N" MPSK carrier recovery loop.

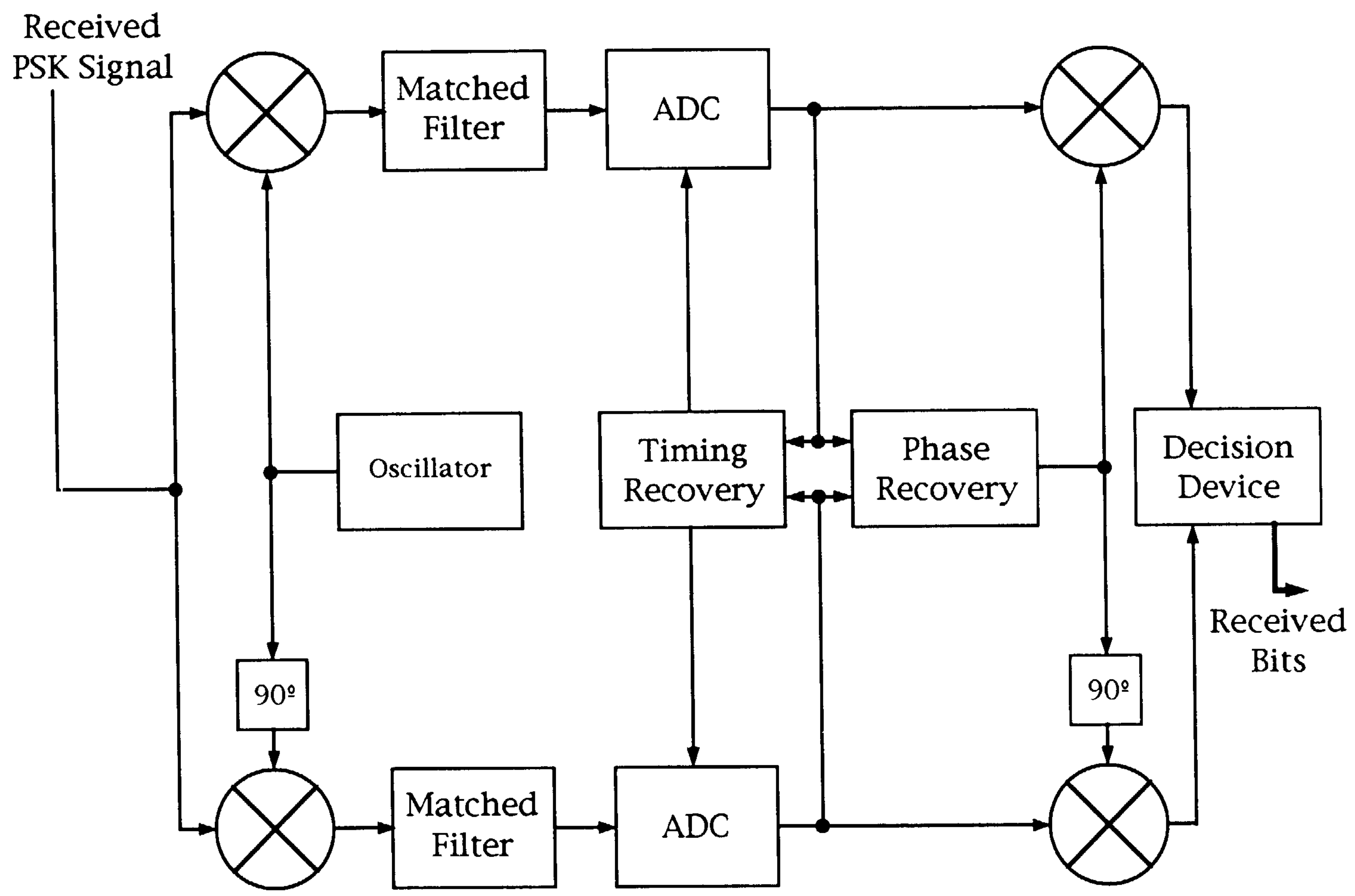


Figure 2. MPSK Feed-Forward Receiver.

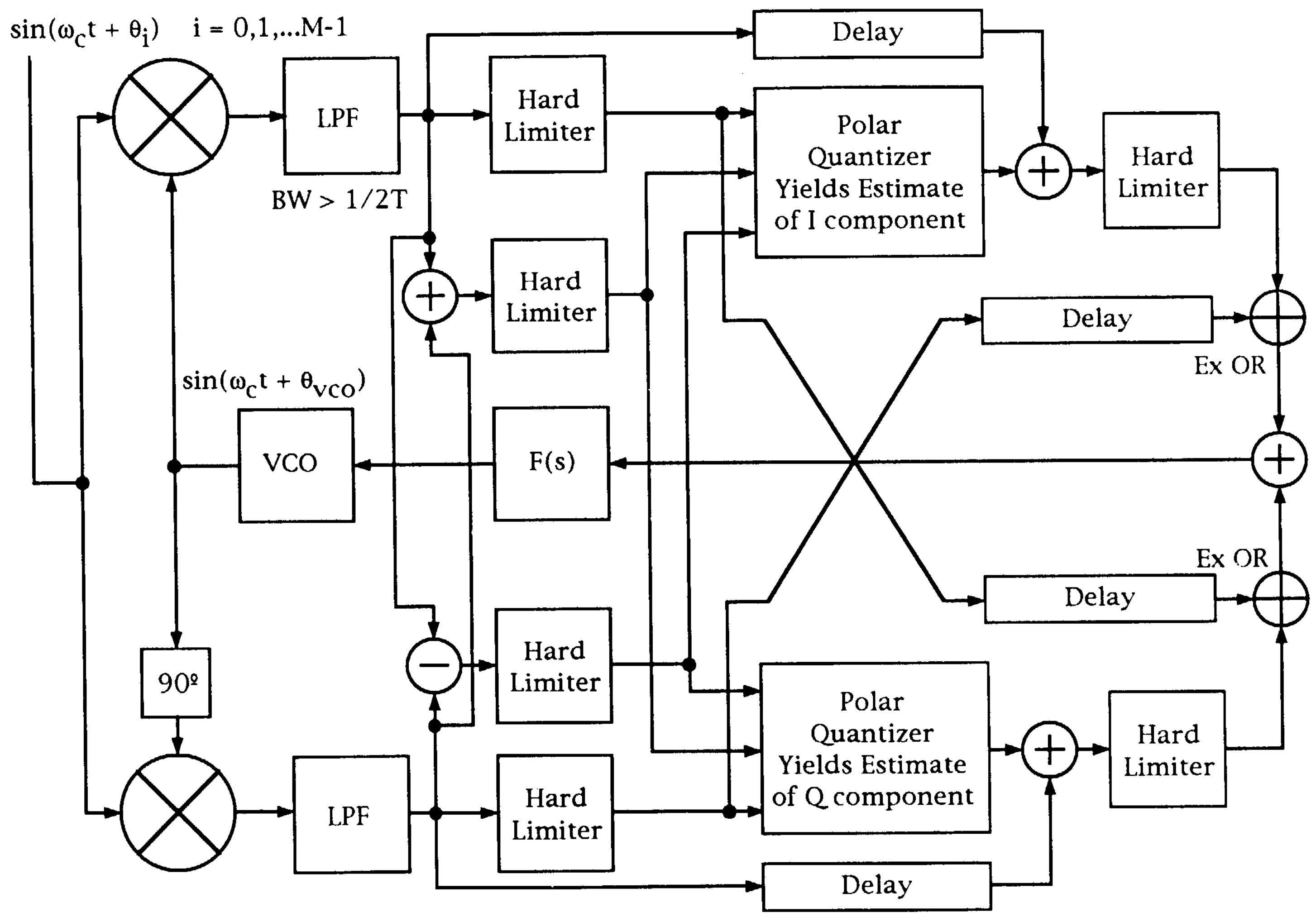

$\sin(\omega_c t + \theta_i)$ $i = 0,1,...M-1$
LPF
BW > 1/2T
Hard Limiter
Delay
Polar Quantizer Yields Estimate of I component
Hard Limiter
Hard Limiter
$\sin(\omega_c t + \theta_{VCO})$
VCO
F(s)
Delay
Ex OR
Ex OR
Delay
90º
Hard Limiter
Polar Quantizer Yields Estimate of Q component
Hard Limiter
LPF
Hard Limiter
Delay

Figure 3. 8-PSK Leclert and Vandamme variation.

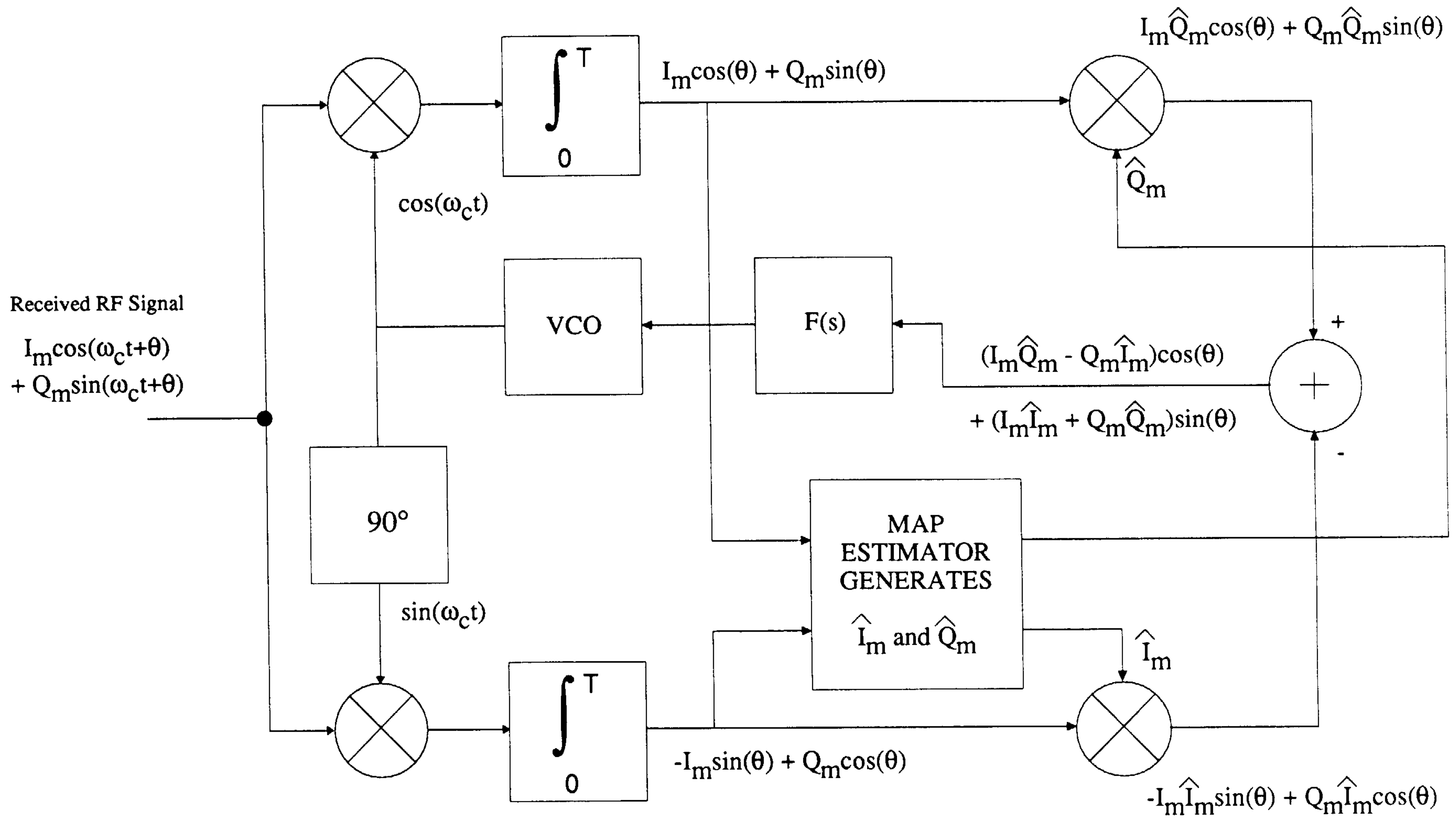

Received RF Signal
$I_m\cos(\omega_c t+\theta) + Q_m\sin(\omega_c t+\theta)$
$\cos(\omega_c t)$
$\sin(\omega_c t)$
90°
VCO
F(s)
$\int_0^T$
$I_m\cos(\theta) + Q_m\sin(\theta)$
$-I_m\sin(\theta) + Q_m\cos(\theta)$
MAP ESTIMATOR GENERATES $\hat{I}_m$ and $\hat{Q}_m$
$\hat{Q}_m$
$\hat{I}_m$
$I_m\hat{Q}_m\cos(\theta) + Q_m\hat{Q}_m\sin(\theta)$
$-I_m\hat{I}_m\sin(\theta) + Q_m\hat{I}_m\cos(\theta)$
+
-
$(I_m\hat{Q}_m - Q_m\hat{I}_m)\cos(\theta) + (I_m\hat{I}_m + Q_m\hat{Q}_m)\sin(\theta)$

Figure 4. Decision-Directed MAP Carrier Phase Estimation loop.

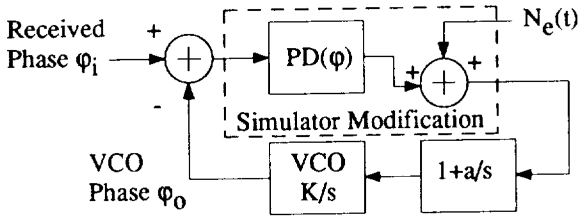


Figure 5(a). The Baseband Model.

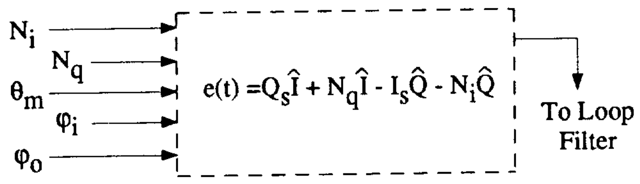
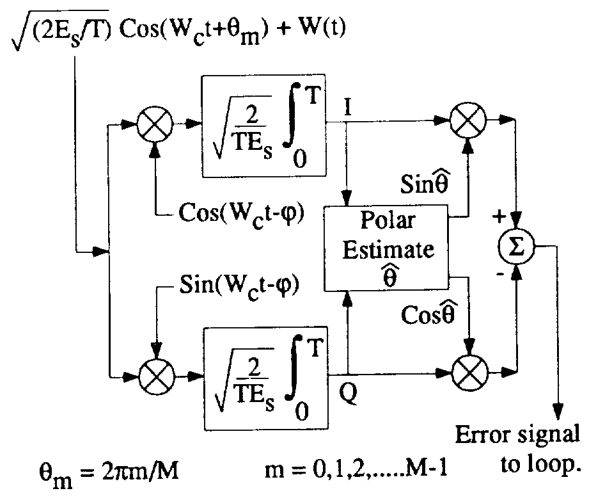


Figure 5(b). The Simulator Modification.



$\sqrt{(2E_s/T)}\ \mathrm{Cos}(W_c t+\theta_m) + W(t)$
$\sqrt{\frac{2}{TE_s}}\int_0^T$
I
Q
$\mathrm{Cos}(W_c t-\varphi)$
$\mathrm{Sin}(W_c t-\varphi)$
Polar Estimate $\hat{\theta}$
$\mathrm{Sin}\hat{\theta}$
$\mathrm{Cos}\hat{\theta}$
+
-
Σ
Error signal to loop.
$\theta_m = 2\pi m/M$
m = 0,1,2,.....M-1

Figure 6. Phase Detector Gain Evaluation Model.

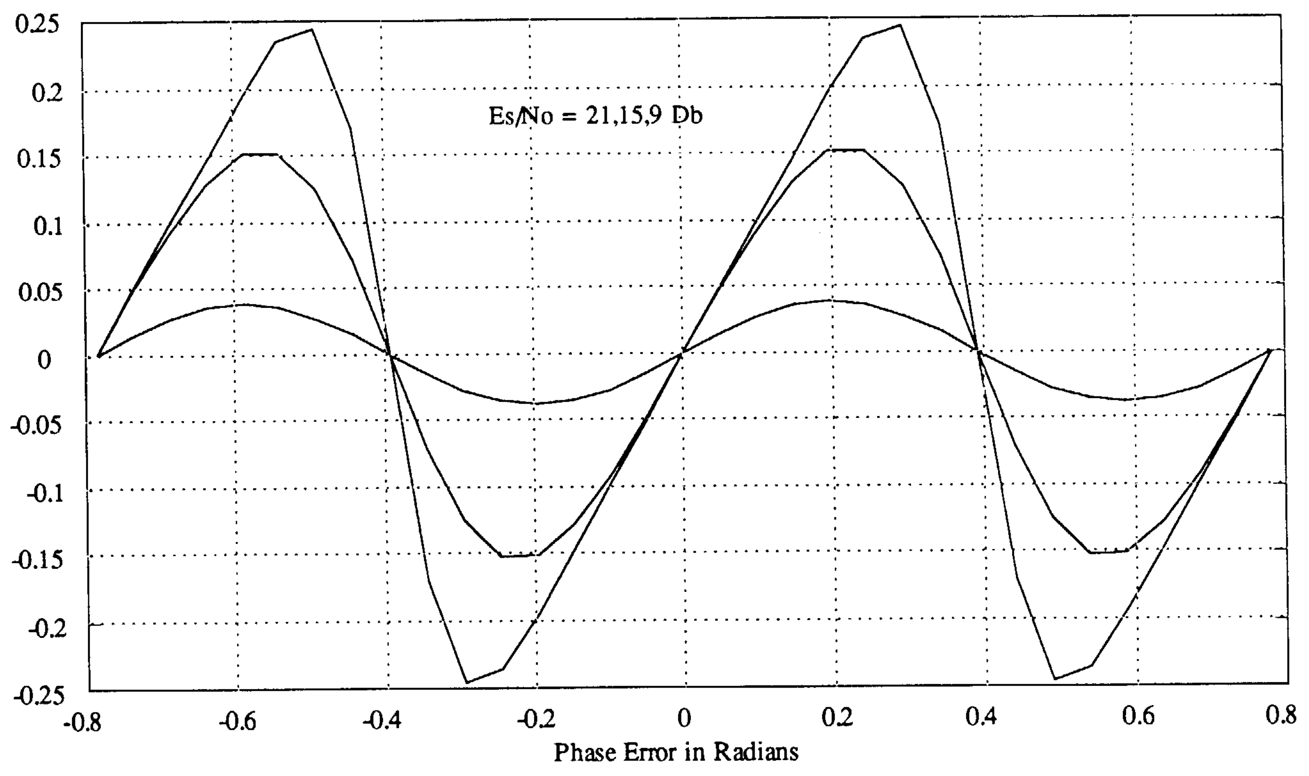


Figure 7. 8-PSK Phase Detector Characteristics.

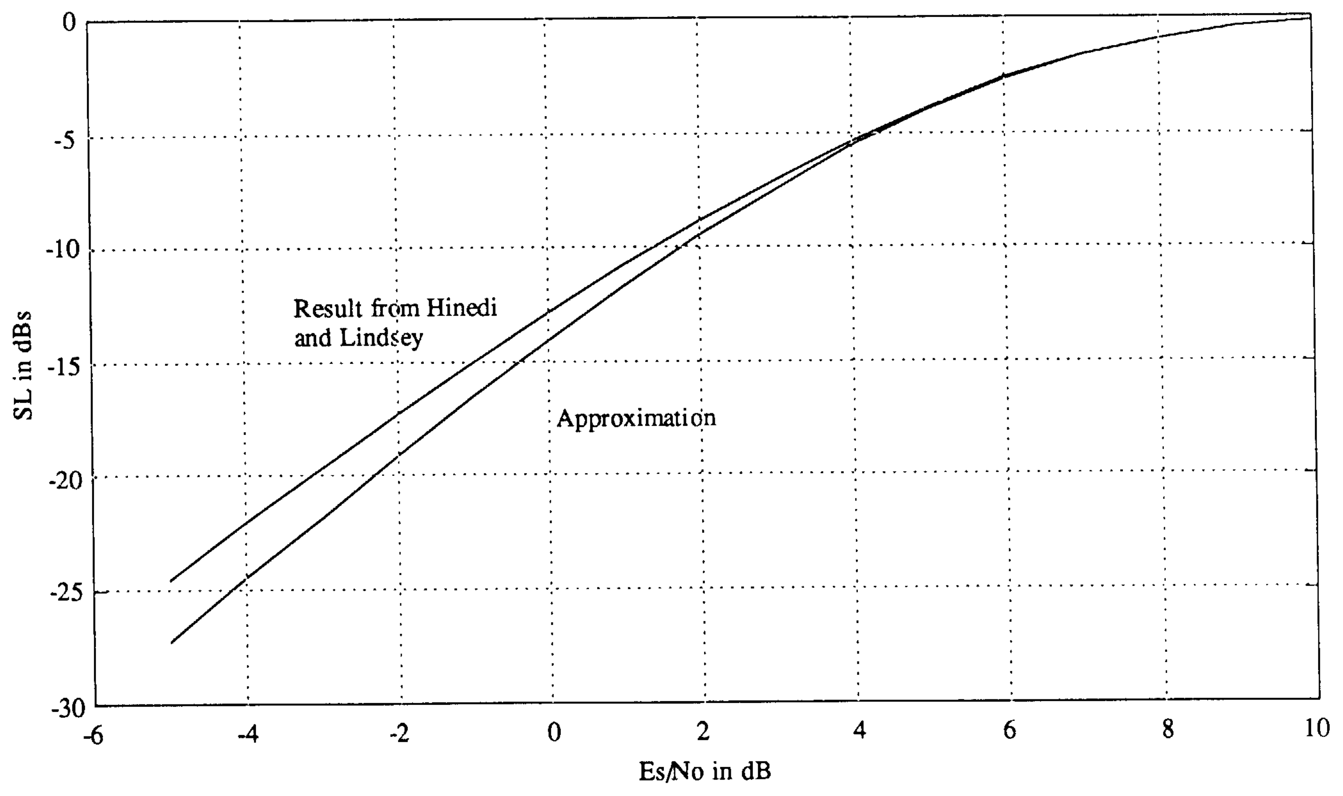


Figure 8. QPSK Squaring Loss.

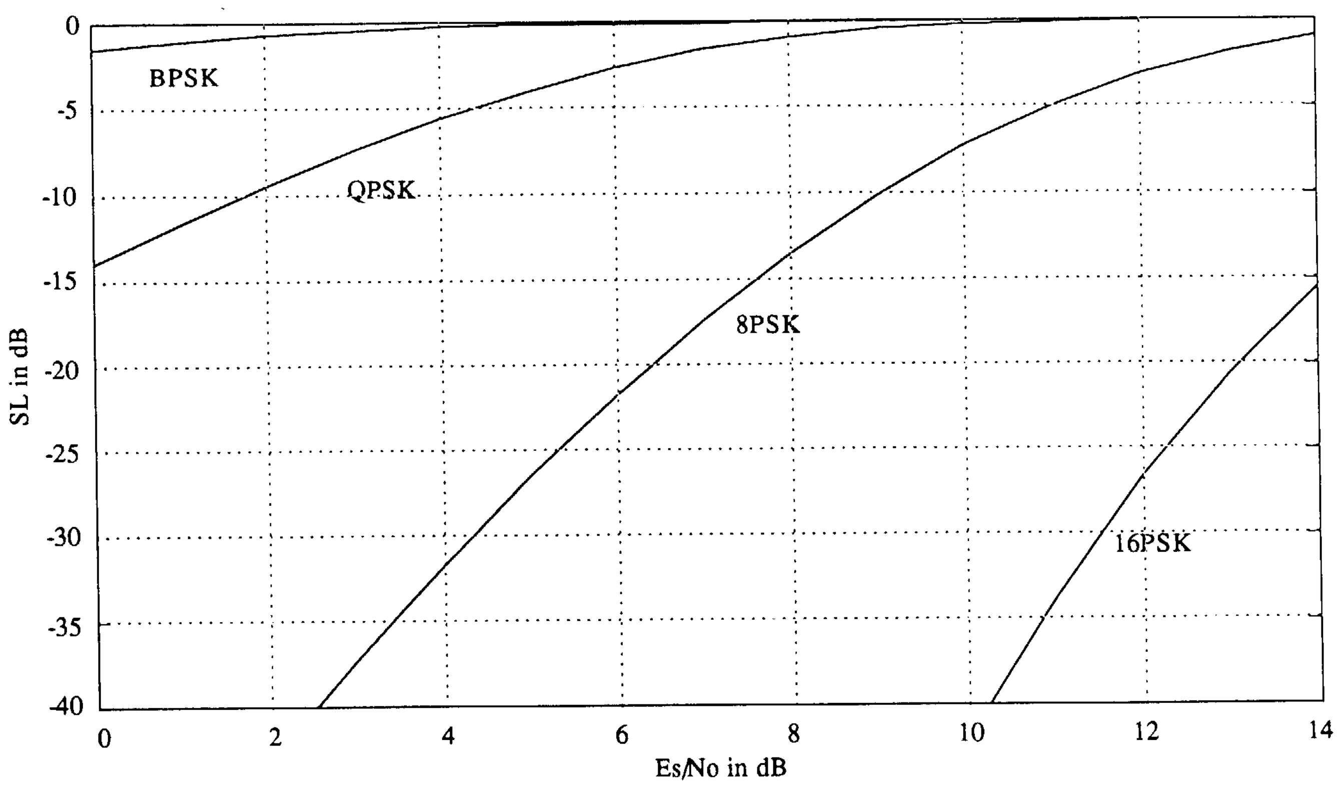


Figure 9. MPSK Squaring Loss.

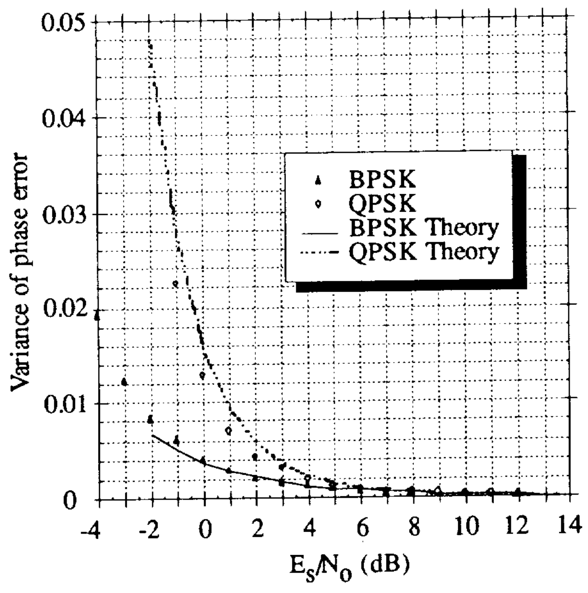


Figure 10. Phase Error Variance For BPSK and QPSK in High SNR Loop With Loop BW = 6250 HZ For a 1 MHZ Symbol Rate

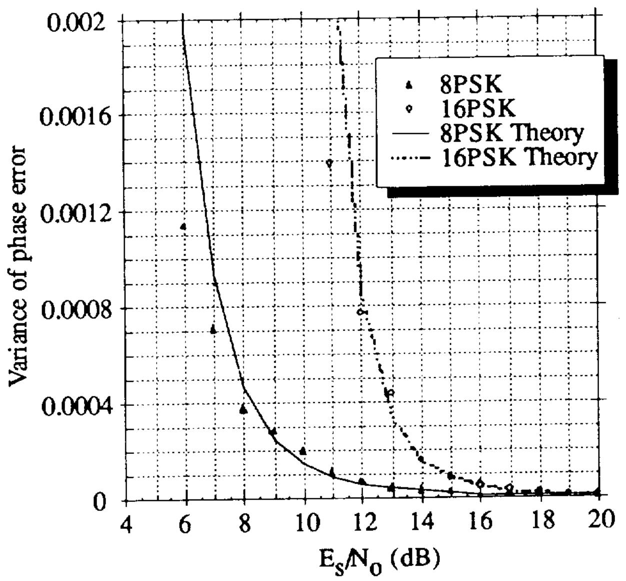


Figure 11. Phase Error Variance For 8PSK and 16PSK in High SNR Loop With Loop BW = 1250 HZ For a 1 MHZ Symbol Rate

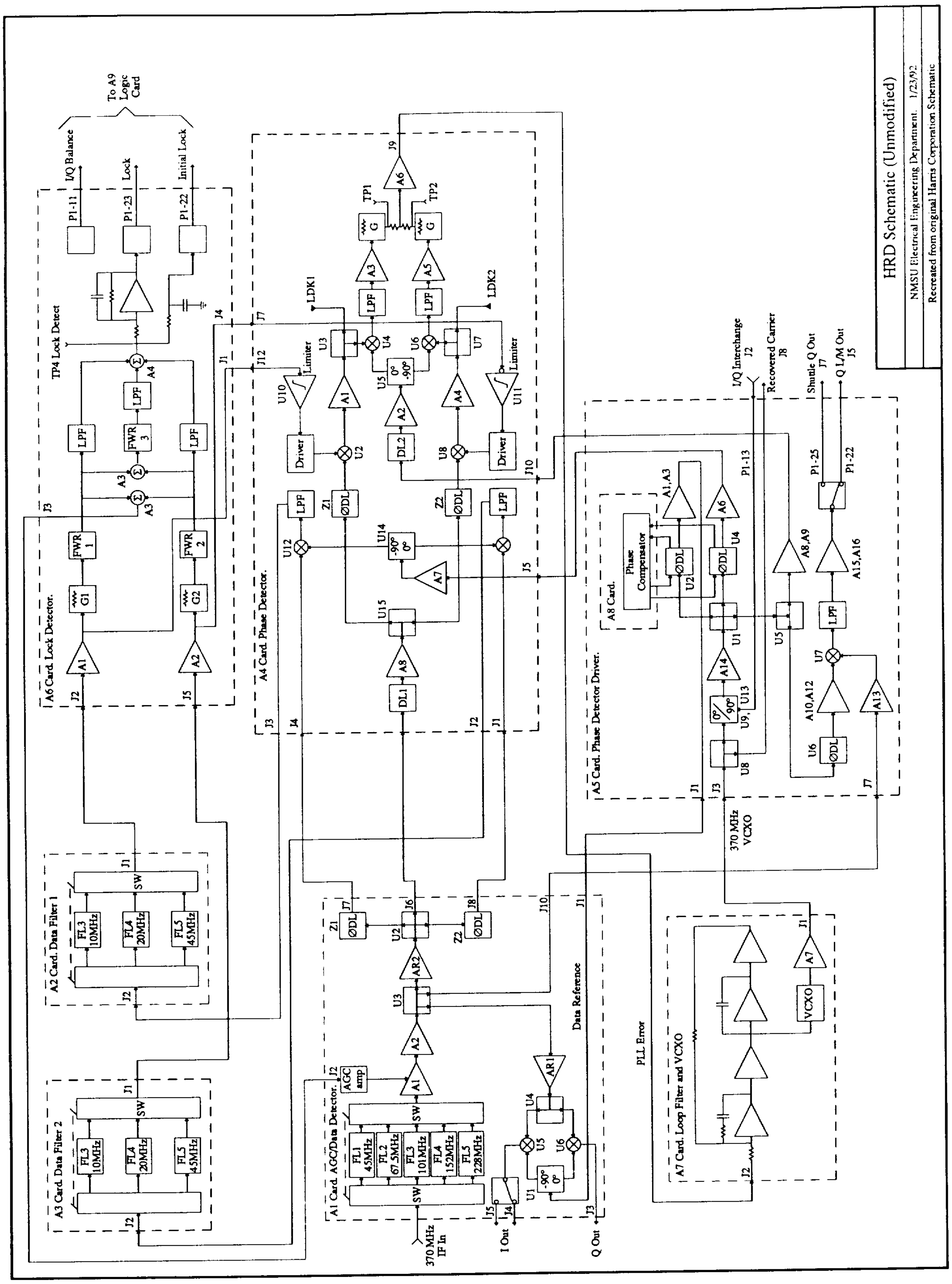


Figure 12. The HRD Carrier Tracking Loop.

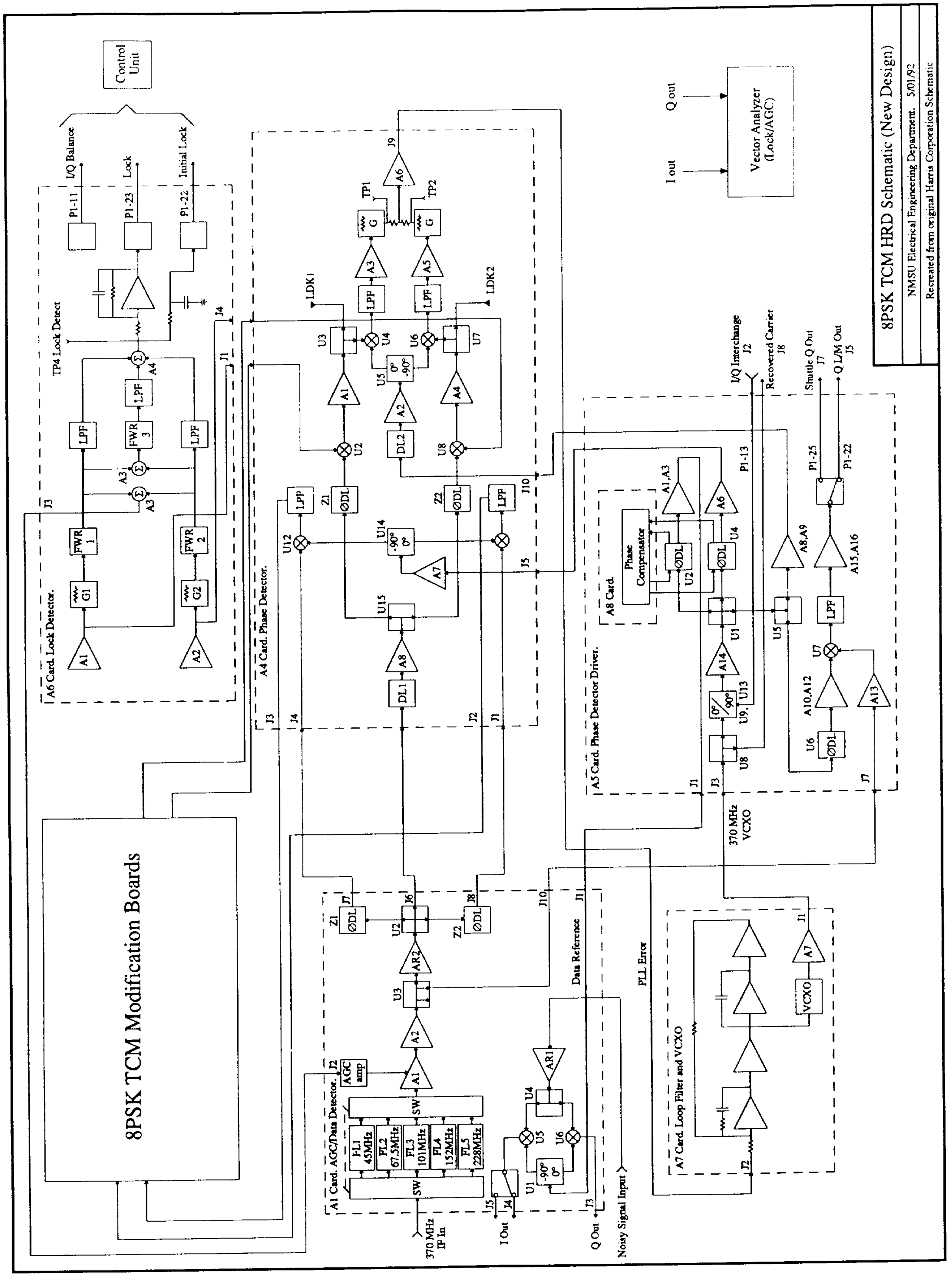


Figure 13. The Converted HRD Carrier Tracking Loop.

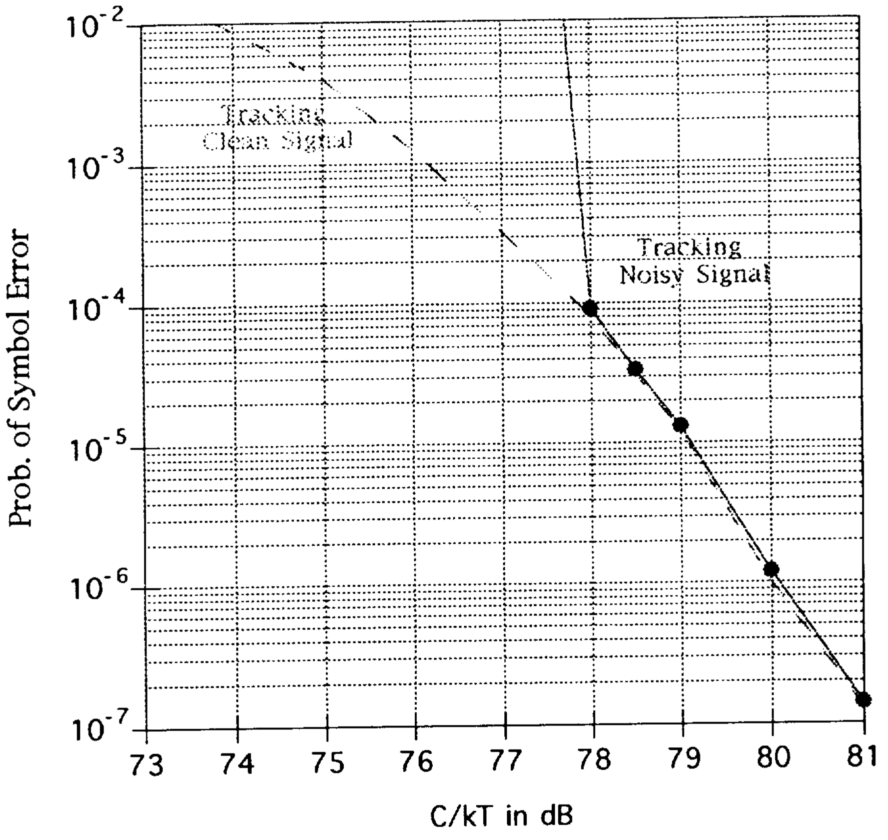


Figure 14. HRD Performance.

Appendix 1

C Simulation Source Code

MAP.c
Wednesday, September 2, 1992 10:25 AM

```
/*The following software project is a time simulation of a decision-directed MAP carrier phase
estimation loop. This particular simulator is a time simulation for computing squaring loss. Th
user inputs the loop parameters (natural freq. and damping factor) and the received Es/No. Th
user also selected M the PSK modulation type. 64000 samples are computed in a triple nested loo
shown below. The results are relayed to the screen and to an output file. This set of source code
files is divided into subroutines. There are 5 files or segments and a total of 11 subroutines. */

/*
Written by Brian Kopp, NMSU Telemetering Center
Phone: 5056464008
Email: bkopp@nmsu.edu
*/

#include <stdio.h>
#include <stdlib.h>
#include <math.h>

/*declare external variables for functions to use*/

FILE *outfile;          /*file name for data output*/
double a;               /*inverse of the one pole filter time constant (the 3dB break point)*/
double sigma;           /*noise standard deviation*/
double nik;             /*noise value for i*/
double nqk;             /*noise value for q*/
double k0;              /*phase detector gain*/
int k;                  /*discrete time variable counter*/
int n;                  /*discrete time variable counter*/
int m;                  /*discrete time variable counter*/
int loadhold;           /*test to only load every other data point*/
double vcoin;           /*input of VCO*/
double accum;           /*integrator hold in Loop filter*/
double vcoload;         /*input of VCO*/
double errk;            /*error signal out of phase detector at time k*/
double maxerr;          /*hold max error value*/
double maxvco;          /*hold max vco input value*/
double maxacc;          /*hold max filter integrator value*/
double phsk;            /*received signal phase at time k*/
double refk1;           /*phase of VCO at time k+1*/
double ik;              /*received value for I sample at time k*/
double imk;             /*generated I sample at time k*/
double qk;              /*received value for Q sample at time k*/
double qmk;             /*generated Q sample at time k*/
double inhold;          /*initial condition of filter integrator at time k*/
double t;               /*symbol period*/
double ihatk;           /*for this phase detector this is the output from the circular quantizer*
double qhatk;           /*for this phase detector this is the output from the circular quantizer*
int modtype;            /*holds the integer that represents the PSK constellation size M*/
char dptr[10];          /*string pointer for entering floating point numbers*/
char flnm[15];          /*user entered file name for output data*/
```

```
double wn;          /*user entered natural frequency of loop*/
double damp;        /*user entered damping factor*/
double bw;          /*loop bandwidth*/
double endb;        /*system Es/No*/
double i1k;         /*old Filter design. Holds filter integrator value*/
double i1k1;        /*old Filter design. Hold variable*/
double i2k;         /*old Filter design. VCO integrator*/


/*variables for squaring loss function*/

double sqrerr;
double sumerr;
int s;
double meanphs;
double varphs;
int counthold;

main()
{

/*run simulation setup functions*/

    userinput();
    sim_init();

/*start simulation*/

    for(m=0;m<2;++m)
        {
        for(k=0;k<320;++k)
            {
            for(n=1;n<=100;++n)
                {

                data_gen();
                noise_gen();
                baseband();
                phasedet();
                errorcalc();
                filter();
                sqrloss1();

                }
            }
        }
    sqrloss2();
    terminate();
}
```

```
/*This segment contains:

"userinput"
"sim_init"

*/

#include <stdio.h>
#include <stdlib.h>
#include <math.h>

/*declare external variables for functions to use*/

extern FILE *outfile;          /*file name for data output*/
extern double a;               /*inverse of the one pole filter
                               time constant (the 3dB break point)*/
extern double sigma;           /*noise standard deviation*/
extern double nik;             /*noise value for i*/
extern double nqk;             /*noise value for q*/
extern double k0;              /*phase detector gain*/
extern int k;                  /*discrete time variable counter*/
extern int n;                  /*discrete time variable counter*/
extern int loadhold;           /*test to only load every other data point*/
extern double vcoin;           /*input of VCO*/
extern double accum;           /*integrator hold in Loop filter*/
extern double vcoload;         /*input of VCO*/
extern double errk;            /*error signal out of phase detector at time k*/
extern double maxerr;          /*hold max error value*/
extern double maxvco;          /*hold max vco input value*/
extern double maxacc;          /*hold max filter integrator value*/
extern double phsk;            /*received signal phase at time k*/
extern double refk1;           /*phase of VCO at time k+1*/
extern double ik;              /*received value for I sample at time k*/
extern double imk;             /*generated I sample at time k*/
extern double qk;              /*received value for Q sample at time k*/
extern double qmk;             /*generated Q sample at time k*/
extern double inhold;          /*initial condition of filter integrator at time k*/
extern double t;               /*symbol period*/
extern double k0;              /*the VCO slope in Hertz per Volt*/
extern double ihatk;           /*for this phase detector this is the output
                               from the circular quantizer*/
extern double qhatk;           /*for this phase detector this is the output
                               from the circular quantizer*/
extern int modtype;            /*holds the integer that represents the PSK
                               constellation size M*/
extern char dptr[10];          /*string pointer for entering floating point numbers*/
extern char flnm[15];          /*user entered file name for output data*/
extern double wn;              /*user entered natural frequency of loop*/
extern double damp;            /*user entered damping factor*/
extern double bw;              /*loop bandwidth*/
extern double endb;            /*system Es/No*/
extern double i1k;             /*old Filter design. Holds filter integrator value*/
extern double i1k1;          /*old Filter design. Hold variable*/
extern double i2k;           /*old Filter design. VCO integrator*/

/*variables for squaring loss function*/

extern double sqrerr;
extern double sumerr;
extern int s;
extern double meanphs;
extern double varphs;
extern int counthold;

void userinput()
{

/*This routine is used for handling user input to set up a particular simulation.*/

/*declare  output file parameters*/

    char *modehold;
    char *nameholdl;

/*initialize  uniform RV seed for repeated tests*/

srand(69);

/*introduction*/

printf("This is a MAP estimation second order carrier loop synchronizer simulator\n");
printf("\n");
printf("\n");

/* get output file name*/

    printf("Please enter the output file name:\n");
    printf("\n");
    scanf("%s",flnm);

/*initialize file output parameters*/

    modehold = "w";
    nameholdl = flnm;
    outfile = fopen(nameholdl,modehold);

/*input natural frequency, damping factor, and Es/No for loop*/

printf("\n");
printf("\n");
printf("Please enter the PSK constellation size M:\n");
printf("\n");
scanf("%s",dptr);
modtype = atoi(dptr);
printf("Please enter the natural frequency of the loop transfer function:\n");
printf("\n");
scanf("%s",dptr);
wn = strtod(dptr,NULL);
printf("\n");
printf("Now please enter the loop damping factor:\n");
printf("\n");
scanf("%s",dptr);
printf("\n");
damp = strtod(dptr,NULL);
printf("\n");
printf("Please enter the system E(s)/N(o) in dB (a value of 100 simulates no noise):\n
printf("\n");
scanf("%s",dptr);
printf("\n");
endb = strtod(dptr,NULL);
printf("\n");

/*calculate some parameters*/

sigma = 1.0/sqrt(2.0*pow(10.0,endb/10.0));

if (endb == 100.0)
    sigma = 0.0;

bw = wn*(damp + 1.0/(4.0*damp));

a = wn/(2*damp);

k0 = 2.0*damp*wn;

printf("\n");

/*save user definable parameters*/

fprintf(outfile,"High SNR MAP estimator loop results\n");
fprintf(outfile,"\n");
fprintf(outfile,"Modulation Type is: %d PSK\n",modtype);
fprintf(outfile,"\n");
fprintf(outfile,"wn = %f\n",wn);
fprintf(outfile,"\n");
fprintf(outfile,"damping factor = %f\n",damp);
fprintf(outfile,"\n");
fprintf(outfile,"The loop noise bandwidth is:  %f  Hz\n",bw);
fprintf(outfile,"\n");
fprintf(outfile,"The loop filter time constant is:  %f  sec\n",a);
fprintf(outfile,"\n");
fprintf(outfile,"The test gain K0 is: %f\n",k0);
fprintf(outfile,"\n");
fprintf(outfile,"The test E(s)/N(o) is: %f\n",endb);
fprintf(outfile,"\n");
}

void sim_init()
```

```
{

/*This routine is used for initializing the simulation paramaters.*/

inhold = 0.0;
refk1 =0.0;           /*initialize a phase step*/
/*refk1 =0.0;*/       /*initialize a phase step*/
t =1;        /*For t = 1e-6 wn should be increased by 1,000,000*/
phsk = 0.0;
vcoin = 0.0;
accum = 0.0;
loadhold = -1;
maxerr = 0.0;
maxvco = 0.0;
maxacc = 0.0;
sqrerr = 0.0;
sumerr = 0.0;
s = 0;
counthold = 0;
i1k = 0.0;
i1k1 = 0.0;
i2k = refk1;

fprintf(outfile,"Initial phase step = %f\n",refk1);
fprintf(outfile,"\n");
fprintf(outfile,"The symbol period is:  %f  seconds\n",t);
fprintf(outfile,"\n");
fprintf(outfile,"\n");
fprintf(outfile,"sample  refk1    errk    accum   vcoin");
fprintf(outfile,"\n");


}
```

MAPdata_gen.c
Wednesday, September 2, 1992 10:31 AM

```
/*this segment contains:
A function "data_gen" that generates the simulation data.
A function "noise_gen" that generates the noise.
A function "baseband" that creates the baseband data ik and qk.
*/

#include <stdio.h>
#include <stdlib.h>
#include <math.h>

/*declare external variables for functions to use*/

extern FILE *outfile;         /*file name for data output*/
extern double a;              /*inverse of the one pole filter
                              time constant (the 3dB break point)*/
extern double sigma;          /*noise standard deviation*/
extern double nik;            /*noise value for i*/
extern double nqk;            /*noise value for q*/
extern double k0;             /*phase detector gain*/
extern int k;                 /*discrete time variable counter*/
extern int n;                 /*discrete time variable counter*/
extern int loadhold;          /*test to only load every other data point*/
extern double vcoin;          /*input of VCO*/
extern double accum;          /*integrator hold in Loop filter*/
extern double vcoload;        /*input of VCO*/
extern double errk;           /*error signal out of phase detector at time k*/
extern double maxerr;         /*hold max error value*/
extern double maxvco;         /*hold max vco input value*/
extern double maxacc;         /*hold max filter integrator value*/
extern double phsk;           /*received signal phase at time k*/
extern double refk1;          /*phase of VCO at time k+1*/
extern double ik;             /*received value for I sample at time k*/
extern double imk;            /*generated I sample at time k*/
extern double qk;             /*received value for Q sample at time k*/
extern double qmk;            /*generated Q sample at time k*/
extern double inhold;         /*initial condition of filter integrator at time k*/
extern double t;              /*symbol period*/
extern double k0;             /*the VCO slope in Hertz per Volt*/
extern double ihatk;          /*for this phase detector this is the output
                              from the circular quantizer*/
extern double qhatk;          /*for this phase detector this is the output
                              from the circular quantizer*/
extern int modtype;           /*holds the integer that represents the PSK
                              constellation size M*/
extern char dptr[10];         /*string pointer for entering floating point numbers*/
extern char flnm[15];         /*user entered file name for output data*/
extern double wn;             /*user entered natural frequency of loop*/
extern double damp;           /*user entered damping factor*/
extern double bw;             /*loop bandwidth*/
extern double endb;           /*system Es/No*/
extern double i1k;            /*old Filter design. Holds filter integrator value*/
extern double i1k1;           /*old Filter design. Hold variable*/
extern double i2k;            /*old Filter design. VCO integrator*/
```

```
void data_gen()
{

/*This subroutine generates a random sample MPSK data point*/

/*declarations*/

int d;

    d = rand(); /*d is now divided into M regions each representing
                 a possible xmitted symbol. M is set by the integer modtype*/

if (modtype == 2)
{
    if(d  < 16384)
        {
        imk = 1.0;
        qmk = 0.0;
        }
    else
        {
        imk = -1.0;
        qmk = 0.;
        }
}

else if (modtype == 4)
{
    if(d  < 8192)
        {
        imk = 0.707107;
        qmk = 0.707107;
        }
    else if((d  >= 8192) && (d < 16384))
        {
        imk = -0.707107;
        qmk = 0.707107;
        }
    else if((d  >= 16384) && (d < 24576))
        {
        imk = -0.707107;
        qmk = -0.707107;
        }
    else
        {
        imk = 0.707107;
        qmk = -0.707107;
        }
}

else if (modtype == 8)
{
    if(d  < 4096)
        {
        imk = 0.923879;
        qmk = 0.382683;
        }
    else if((d  >= 4096) && (d < 8192))
        {
        imk = 0.382683;
        qmk = 0.923879;
        }
    else if((d  >= 8192) && (d < 12288))
        {
        imk = -0.382683;
        qmk = 0.923879;
        }
    else if((d  >= 12288) && (d < 16384))
        {
        imk = -0.923879;
        qmk = 0.382683;
        }
    else if((d  >= 16384) && (d < 20480))
        {
        imk = -0.923879;
        qmk = -0.382683;
        }
    else if((d  >= 20480) && (d < 24576))
        {
        imk = -0.382683;
        qmk = -0.923879;
        }
    else if((d  >= 24576) && (d < 28672))
        {
        imk = 0.382683;
        qmk = -0.923879;
        }
    else
        {
        imk = 0.923879;
        qmk = -0.382683;
        }
}

else if (modtype == 16)
{
    if(d  < 2048)
        {
        imk = 0.980785;
        qmk = 0.195090;
        }
    else if((d  >= 2048) && (d < 4096))
        {
        imk = 0.831470;
        qmk = 0.555570;
        }
    else if((d  >= 4096) && (d < 6144))
        {
        imk = 0.555570;
        qmk = 0.831470;
        }
    else if((d  >= 6144) && (d < 8192))
        {
        imk = 0.195090;
        qmk = 0.980785;
        }
    else if((d  >= 8192) && (d < 1024))
        {
        imk = -0.195090;
        qmk = 0.980785;
        }
    else if((d  >= 1024) && (d < 12288))
        {
        imk = -0.555570;
        qmk = 0.831470;
        }
    else if((d  >= 12288) && (d < 14336))
        {
        imk = -0.831470;
        qmk = 0.555570;
        }
    else if((d  >= 14336) && (d < 16384))
        {
        imk = -0.980785;
        qmk = 0.195090;
        }
    else if((d  >= 16384) && (d < 18432))
        {
        imk = -0.980785;
        qmk = -0.195090;
        }
    else if((d  >= 18432) && (d < 20480))
        {
        imk = -0.831470;
        qmk = -0.555570;
        }
    else if((d  >= 20480) && (d < 22528))
        {
        imk = -0.555570;
        qmk = -0.831470;
        }
    else if((d  >= 22528) && (d < 24576))
        {
        imk = -0.195090;
        qmk = -0.980785;
        }
    else if((d  >= 24576) && (d < 26624))
        {
```

```
        imk = 0.195090;
        qmk = -0.980785;
        }
    else if((d  >= 26624) && (d < 28672))
        {
        imk = 0.555570;
        qmk = -0.831470;
        }
    else if((d  >= 28672) && (d < 30720))
        {
        imk = 0.831470;
        qmk = -0.555570;
        }
    else
        {
        imk = 0.980785;
        qmk = -0.195090;
        }
}

}

void noise_gen()
{

/*Two gaussian noise samples are generated here. They are functions of the Es/No*/

/*declarations*/

long x;
long y;
double sample1;
double sample2;

x = rand();
y = rand();
sample1 = (x/32767.0)*0.999999;
sample2 = (y/32767.0)*0.999999;

nik = sigma*sqrt(-2.0*log(1.0-sample1))*cos(2.0*3.14159265*sample2);
nqk = sigma*sqrt(-2.0*log(1.0-sample1))*sin(2.0*3.14159265*sample2);

}

void baseband()
{

/*This subroutine calculates the baseband phase difference and received I and Q*/

double pdifk;          /*phase difference between received signal and VCO at time k*/


/*The received phase is a ramp or step*/
```

```
/*  phsk = -0.001*(k*100+n);*/

    phsk = 0.0;

/*subtract the reference phase from the received phase*/

pdifk = phsk - refk1;

/*calculate the received symbol values for I and Q based on the generated data and noi
    the phase difference pdifk*/

ik = imk*cos(pdifk) - qmk*sin(pdifk) + nik;
qk = imk*sin(pdifk) + qmk*cos(pdifk) + nqk;

}
```

MAPphs_det.c
Wednesday, September 2, 1992 10:41 AM

```
/*this segment contains:

"phasedet", a function that determines the phase of ik and qk.
"errorcalc", a function that calculates the error value of
            that is sent to the filter.

*/

#include <stdio.h>
#include <stdlib.h>
#include <math.h>

/*declare external variables for functions to use*/

extern FILE *outfile;       /*file name for data output*/
extern double a;            /*inverse of the one pole filter
                            time constant (the 3dB break point)*/
extern double sigma;        /*noise standard deviation*/
extern double nik;          /*noise value for i*/
extern double nqk;          /*noise value for q*/
extern double k0;           /*phase detector gain*/
extern int k;               /*discrete time variable counter*/
extern int n;               /*discrete time variable counter*/
extern int loadhold;        /*test to only load every other data point*/
extern double vcoin;        /*input of VCO*/
extern double accum;        /*integrator hold in Loop filter*/
extern double vcoload;      /*input of VCO*/
extern double errk;         /*error signal out of phase detector at time k*/
extern double maxerr;       /*hold max error value*/
extern double maxvco;       /*hold max vco input value*/
extern double maxacc;       /*hold max filter integrator value*/
extern double phsk;         /*received signal phase at time k*/
extern double refk1;        /*phase of VCO at time k+1*/
extern double ik;           /*received value for I sample at time k*/
extern double imk;          /*generated I sample at time k*/
extern double qk;           /*received value for Q sample at time k*/
extern double qmk;          /*generated Q sample at time k*/
extern double inhold;       /*initial condition of filter integrator at time k*/
extern double t;            /*symbol period*/
extern double k0;           /*the VCO slope in Hertz per Volt*/
extern double ihatk;        /*for this phase detector this is the output
                            from the circular quantizer*/
extern double qhatk;        /*for this phase detector this is the output
                            from the circular quantizer*/
extern int modtype;         /*holds the integer that represents the PSK
                            constellation size M*/
extern char dptr[10];       /*string pointer for entering floating point numbers*/
extern char flnm[15];       /*user entered file name for output data*/
extern double wn;           /*user entered natural frequency of loop*/
extern double damp;         /*user entered damping factor*/
extern double bw;           /*loop bandwidth*/
extern double endb;         /*system Es/No*/
extern double i1k;          /*old Filter design. Holds filter integrator value*/
extern double i1k1;          /*old Filter design. Hold variable*/
extern double i2k;           /*old Filter design. VCO integrator*/

void phasedet()
{

/*This subroutine determines the MAP estimate of what was transmitted for MPSK with
any M.*/

double rcvdp;       /*the phase of the received vectort whose components are ik and qk

rcvdp = atan2(qk,ik);

if (modtype == 2)
{
    if ((rcvdp >= 0.0) && (rcvdp < 1.570796))
        {
        ihatk = 1.0;
        qhatk = 0.0;
        }
    else if ((rcvdp < 0.0) && (rcvdp >= -1.570796))
        {
        ihatk = 1.0;
        qhatk = 0.0;
        }
    else if ((rcvdp < -1.570796) && (rcvdp >= -3.141593))
        {
        ihatk = -1.0;
        qhatk = 0.0;
        }
    else if ((rcvdp < 3.141593) && (rcvdp >= 1.570796))
        {
        ihatk = -1.0;
        qhatk = 0.0;
        }
}

else if (modtype == 4)
{
    if ((rcvdp >= 0.0) && (rcvdp < 1.570796))
        {
        ihatk = 0.707107;
        qhatk = 0.707107;
        }
    else if ((rcvdp >= 1.570796) && (rcvdp < 3.141593))
        {
        ihatk = -0.707107;
        qhatk = 0.707107;
        }
    else if ((rcvdp < -1.570796) && (rcvdp >= -3.141593))
        {
        ihatk = -0.707107;
        qhatk = -0.707107;
        }
    else if ((rcvdp >= -1.570796) && (rcvdp < 0.0))
        {
        ihatk = 0.707107;
        qhatk = -0.707107;
        }
}

else if (modtype == 8)
{
    if ((rcvdp >= 0.0) && (rcvdp < 0.785398))
        {
        ihatk = 0.923879;
        qhatk = 0.382683;
        }
    else if ((rcvdp >= 0.785398) && (rcvdp < 1.570796))
        {
        ihatk = 0.382683;
        qhatk = 0.923879;
        }
    else if ((rcvdp >= 1.570796) && (rcvdp < 2.356194))
        {
        ihatk = -0.382683;
        qhatk = 0.923879;
        }
    else if ((rcvdp >= 2.356194) && (rcvdp < 3.141593))
        {
        ihatk = -0.923879;
        qhatk = 0.382683;
        }
    else if ((rcvdp < -2.356194) && (rcvdp >= -3.141593))
        {
        ihatk = -0.923879;
        qhatk = -0.382683;
        }
    else if ((rcvdp >=  -2.356194) && (rcvdp < -1.570796))
        {
        ihatk = -0.382683;
        qhatk = -0.923879;
        }
    else if ((rcvdp >= -1.570796) && (rcvdp < -0.785398))
        {
        ihatk = 0.382683;
        qhatk = -0.923879;
        }
    else if ((rcvdp >= -0.785398) && (rcvdp < 0.0))
        {
        ihatk = 0.923879;
        qhatk = -0.382683;
        }
}

else if (modtype == 16)
```

```
{
    if ((rcvdp >= 0.0) && (rcvdp < 0.392699))
        {
        ihatk = 0.980785;
        qhatk = 0.195090;
        }
    else if ((rcvdp >= 0.392699) && (rcvdp < 0.785398))
        {
        ihatk = 0.831470;
        qhatk = 0.555570;
        }
    else if ((rcvdp >= 0.785398) && (rcvdp < 1.178097))
        {
        ihatk = 0.555570;
        qhatk = 0.831470;
        }
    else if ((rcvdp >= 1.178097) && (rcvdp < 1.570796))
        {
        ihatk = 0.195090;
        qhatk = 0.980785;
        }
    else if ((rcvdp >= 1.570796) && (rcvdp < 1.963495))
        {
        ihatk = -0.195090;
        qhatk = 0.980785;
        }
    else if ((rcvdp >= 1.963495) && (rcvdp < 2.356194))
        {
        ihatk = -0.555570;
        qhatk = 0.831470;
        }
    else if ((rcvdp >= 2.356194) && (rcvdp < 2.748894))
        {
        ihatk = -0.831470;
        qhatk = 0.555570;
        }
    else if ((rcvdp >= 2.748894) && (rcvdp < 3.141593))
        {
        ihatk = -0.980785;
        qhatk = 0.195090;
        }
    else if ((rcvdp < -2.748894) && (rcvdp >= -3.141593))
        {
        ihatk = -0.980785;
        qhatk = -0.195090;
        }
    else if ((rcvdp < -2.356194) && (rcvdp >= -2.748894))
        {
        ihatk = -0.831470;
        qhatk = -0.555570;
        }
    else if ((rcvdp >=  -2.356194) && (rcvdp < -1.963495))
        {
```

```
        ihatk = -0.555570;
        qhatk = -0.831470;
        }
    else if ((rcvdp >=  -1.963495) && (rcvdp < -1.570796))
        {
        ihatk = -0.195090;
        qhatk = -0.980785;
        }
    else if ((rcvdp >= -1.570796) && (rcvdp < -1.178097))
        {
        ihatk = 0.195090;
        qhatk = -0.980785;
        }
    else if ((rcvdp >= -1.178097) && (rcvdp < -0.785398))
        {
        ihatk = 0.555570;
        qhatk = -0.831470;
        }
    else if ((rcvdp >= -0.785398) && (rcvdp < -0.392699))
        {
        ihatk = 0.831470;
        qhatk = -0.555570;
        }
    else if ((rcvdp >= -0.392699) && (rcvdp < 0.0))
        {
        ihatk = 0.980785;
        qhatk = -0.195090;
        }
}
}

void errorcalc()
{

/*calculate the phase error errk as a function of of the received data and MAP
estimates of what was transmitted; ik, qk, ihatk, and qhatk.*/

    errk =( -ik*qhatk + qk*ihatk);

}
```

```
/*this segment contains:

"filter", a function that simulates the loop filter and VCO.
"terminate", a function to print results and close output streams.
"sqrloss1", this routine keeps track of mean and variance data about the phase.
"sqrloss2", makes final variance calculations before terminating program.

*/

#include <stdio.h>
#include <stdlib.h>
#include <math.h>

/*declare external variables for functions to use*/

extern FILE *outfile;          /*file name for data output*/
extern double a;               /*inverse of the one pole filter
                               time constant (the 3dB break point)*/
extern double sigma;           /*noise standard deviation*/
extern double nik;             /*noise value for i*/
extern double nqk;             /*noise value for q*/
extern double k0;              /*phase detector gain*/
extern int k;                  /*discrete time variable counter*/
extern int n;                  /*discrete time variable counter*/
extern int m;                  /*discrete time variable counter*/
extern int loadhold;           /*test to only load every other data point*/
extern double vcoin;           /*input of VCO*/
extern double accum;           /*integrator hold in Loop filter*/
extern double vcoload;         /*input of VCO*/
extern double errk;            /*error signal out of phase detector at time k*/
extern double maxerr;          /*hold max error value*/
extern double maxvco;          /*hold max vco input value*/
extern double maxacc;          /*hold max filter integrator value*/
extern double phsk;            /*received signal phase at time k*/
extern double refk1;           /*phase of VCO at time k+1*/
extern double ik;              /*received value for I sample at time k*/
extern double imk;             /*generated I sample at time k*/
extern double qk;              /*received value for Q sample at time k*/
extern double qmk;             /*generated Q sample at time k*/
extern double inhold;          /*initial condition of filter integrator at time k*/
extern double t;               /*symbol period*/
extern double k0;              /*the VCO slope in Hertz per Volt*/
extern double ihatk;           /*for this phase detector this is the output
                               from the circular quantizer*/
extern double qhatk;           /*for this phase detector this is the output
                               from the circular quantizer*/
extern int modtype;            /*holds the integer that represents the PSK
                               constellation size M*/
extern char dptr[10];          /*string pointer for entering floating point numbers*/
extern char flnm[15];          /*user entered file name for output data*/
extern double wn;              /*user entered natural frequency of loop*/
extern double damp;            /*user entered damping factor*/
extern double bw;              /*loop bandwidth*/
extern double endb;          /*system Es/No*/
extern double i1k;           /*old Filter design. Holds filter integrator value*/
extern double i1k1;          /*old Filter design. Hold variable*/
extern double i2k;           /*old Filter design. VCO integrator*/

/*variables for squaring loss function*/

extern double sqrerr;
extern double sumerr;
extern int s;
extern double meanphs;
extern double varphs;
extern int counthold;

/*variables for lock detect*/

extern double coserr;        /*cosine of the phase error*/

void filter()
{

/* This subroutine calculates the loop filters response.*/

int temp;

/********************** Filter # 1 ****************************************/
/*This is the original filter that WPO and I worked out*/

    refk1 = refk1 + k0*t*errk + inhold*k0*t + a*k0*t*t*errk/2.0;
    inhold = inhold + a*t*errk;

/************************************************************************/

/*print data to output file. The intialize function prints a heading for this data.*/

/*if (k ==0)
    fprintf(outfile,"%d  %15.11f  %15.11f  %15.11f  %15.11f\n",k*100+n,refk1,errk,accu
*/
/*retain maximum values*/

if (fabs(refk1) > maxerr)
    maxerr = fabs(refk1);

}

void terminate()
{
    fprintf(outfile,"Maximum phase deviation:    %30.22f\n",maxerr);
    printf("Maximum phase deviation value:    %30.22f\n",maxerr);

    fclose(outfile);
}
```

```
void sqrloss1()
{


/*test to see if enough samples have occurred to uncorrelate
statistical calculations. If they have then increment mean and
variance variables, and their counter.*/

if ((k*100+n) >= 0)
    {
    if ((k*100+n) == (1000 + counthold) || (k*100+n == 1))
        {
        s =s + 1;
        counthold = k*100+n;
        sqrerr = sqrerr + refkl*refkl;
        sumerr = sumerr + refkl;
        }
    }
}

void sqrloss2()
{

/*Make final calculations and then print results before terminating simulation.*/

double sqrls;

meanphs = sumerr/s;

varphs = sqrerr/s - meanphs*meanphs;

sqrls = bw*t/(2.0*pow(10.0,endb/10.0))/varphs;

fprintf(outfile,"Mean of error = %10.7f, variance of error = %15.11f\n",meanphs,varphs
fprintf(outfile,"\n");
fprintf(outfile,"The squaring loss equals = %18.11f\n",10*log10(sqrls));
fprintf(outfile,"\n");
fprintf(outfile,"The number of samples used in the previous statistical analysis is: %
fprintf(outfile,"\n");
printf("\n");
printf("\n");
printf("Mean of error = %10.7f, variance of error = %15.11f\n",meanphs,varphs);
printf("\n");
printf("The squaring loss equals = %18.11f\n",10*log10(sqrls));
printf("\n");
printf("The number of samples used in the previous statistical analysis is: %d\n",s);
}
```

Appendix 2

HRD Modification Technical Report

Modification of the TDRSS High Rate Demodulator for 8-PSK Operation

William P. Osborne
Gerald L. Stolarczyk

NMSU-ECE-92-010 May 1992

Modification of the TDRSS High Rate Demodulator for 8-PSK Operation

BY

Gerald L. Stolarczyk

A technical report in partial fulfillment

of the requirements for the Degree

Master of Science in Electrical Engineering

New Mexico State University

Las Cruces, New Mexico

May 1992

ABSTRACT

Modification of the TDRSS

High Rate Demodulator

for 8-PSK Operation

Master of Science in Electrical Engineering

New Mexico State University

Las Cruces, New Mexico, 1992

Dr. William P. Osborne, Chair

In the late 1990's, NASA plans to launch into orbit the Space Station Freedom. The space station will require communication links back to earth that are capable of handling large quantities of data. The expence incurred by the building of the space station has elliminated the effort to build new space communications equipement. It is the current plan of NASA to use the already existant TDRSS communication satelite system to download data from the space station to the earth. However, the current TDRSS system is not capable of handling the data amounts that will be required by the space station.

The current TDRSS system needs modification in order to handle this increase in data through put. The TDRSS system is to be modified in such a way that data communications schemes such as Trellis Coded Modulation (TCM) can be handled by TDRSS.

Research is currently being persude by the Telemetering Team at New Mexico State University into the required TDRSS system modifications. In particular, the improved TDRSS system will require hardware capable of handling a TCM 8-PSK modulation scheme. This requires new hardware and modifications to existing TDRSS hardware.

This paper will focus on one aspect of the modification process. Namely, the modification needed to the TDRSS High Rate Demodulator (HRD). The current HRD is only capable of handling QPSK modulation schemes. In order to have coding gains through the TDRSS system, an 8-PSK modulation scheme is needed. This paper discusses the current state of the HRD as well as the method used in the modification processes to convert the current QPSK HRD to an 8-PSK HRD.

Table of Contents:

1.0 Introduction

The New Mexico State University Telemetering Team under the leadership skills of Dr. Frank Carden and Dr. Bill Osborne is currently undertaking a monumental research project. This project with funding from NASA involves research into the coding gains available from an 8-PSK Trellis Coded Modulation (TCM) scheme. What makes this project unique is that not only is the theoretical research being investigated but actual TCM hardware is being developed and implemented. In particular, the project covers the modification of the TDRSS ground station to make it capable of handling this TCM scheme. The project involves the development of a new modulator, TCM codecs, symbol synchronizers, modifying an existing demodulator, and developing other support equipment.

This paper will focus on one aspect of this project; the modification of the existing TDRSS high rate demodulator (HRD) to make it capable of demodulating an 8-PSK modulated signal.

This project is both unique and difficult in its' own right. The difficult part stems from the fact that the HRD was originally developed in the mid 1970's. Some of the technology used in the HRD is outdated. Trying to incorporate new technology into the old can be a complicated task. It is also difficult because the HRD was designed for the sole task of demodulating data for the TDRSS system. This means the HRD is a one of a kind piece of hardware. So, the modifications needed are limited by the uniqueness of the machinery. Which in turn means that the modifications must be made to fit the HRD and not force the HRD to fit the modifications.

This project is unique because no one has ever completed this type of work before. Converting a working QPSK demodulator to a demodulator capable of handling 8-PSK data is a first. However, the real trick is to make the modifications work without disrupting other viable parts of the HRD. Great care is needed to analyze and then re-analyze any affect that a particular modification might have on the remainder of the system.

This paper presents an overview of the modification process. In particular, chapter 2 gives a synopsis stating the current functionality of the HRD. In chapter 3, a review of the modifications needed for the HRD will be presented. Chapter 4 discusses the actual modification design. And, finally, chapter 5 gives the results of the modifications described in this paper to date.

2.0 High Rate Demodulator Functional Description

The High Rate Demodulator (HRD) performs, as expected, carrier recovery and coherent demodulation over primary symbol rates of 10-100 Msps for BPSK and balanced or unbalanced QPSK. The resulting demodulated baseband data is provided to bit synchronizers farther down the receiver chain.

The placement of the HRD in the receiver network is shown in figure 2.0 on page 4.

A block diagram of the HRD is shown in figure 2.1 on page 5. There are a total of nine circuit cards in the HRD, each performing a specific function.

The following describes the major circuit contents of each card, the function of each card, and the relationships of each card, one to another.

2.1 AGC/Data Detector Card (A1 card)

The AGC/Data Detect card (A1 card) contains the IF filtering, IF amplification, AGC leveling of the IF, and actual baseband data recovery from the IF.

The A1 card accepts an IF input signal at a frequency of 370MHz with a level of -40 to -20 dBm. It then provides four bandlimited and level controlled 370MHz outputs. Three IF outputs go to the PLL Detector card (A4 card) and one output to the Detector Driver card (A5 card).

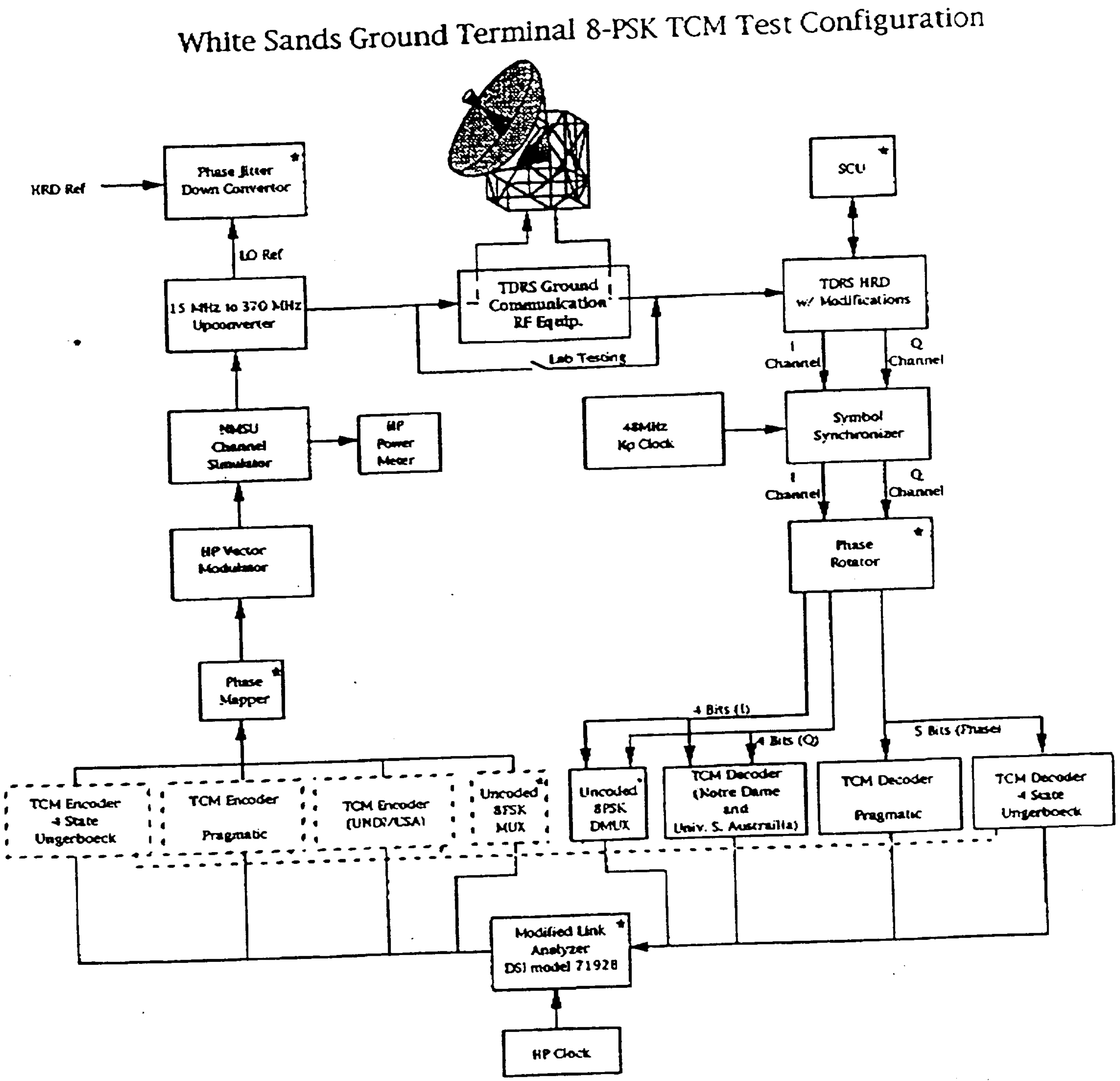

White Sands Ground Terminal 8-PSK TCM Test Configuration
HRD Ref
Phase Jitter Down Convertor
LO Ref
15 MHz to 370 MHz Upconverter
TDRS Ground Communication RF Equip.
Lab Testing
SCU
TDRS HRD w/ Modifications
I Channel
Q Channel
NMSU Channel Simulator
HP Power Meter
48MHz Kp Clock
Symbol Synchronizer
HP Vector Modulator
Phase Rotator
Phase Mapper
4 Bits (I)
4 Bits (Q)
5 Bits (Phase)
TCM Encoder 4 State Ungerboeck
TCM Encoder Pragmatic
TCM Encoder (UND/USA)
Uncoded 8PSK MUX
Uncoded 8PSK DMUX
TCM Decoder (Notre Dame and Univ. S. Australia)
TCM Decoder Pragmatic
TCM Decoder 4 State Ungerboeck
Modified Link Analyzer DSI model 71928
HP Clock

Figure 2.0: TDRSS system block diagram.

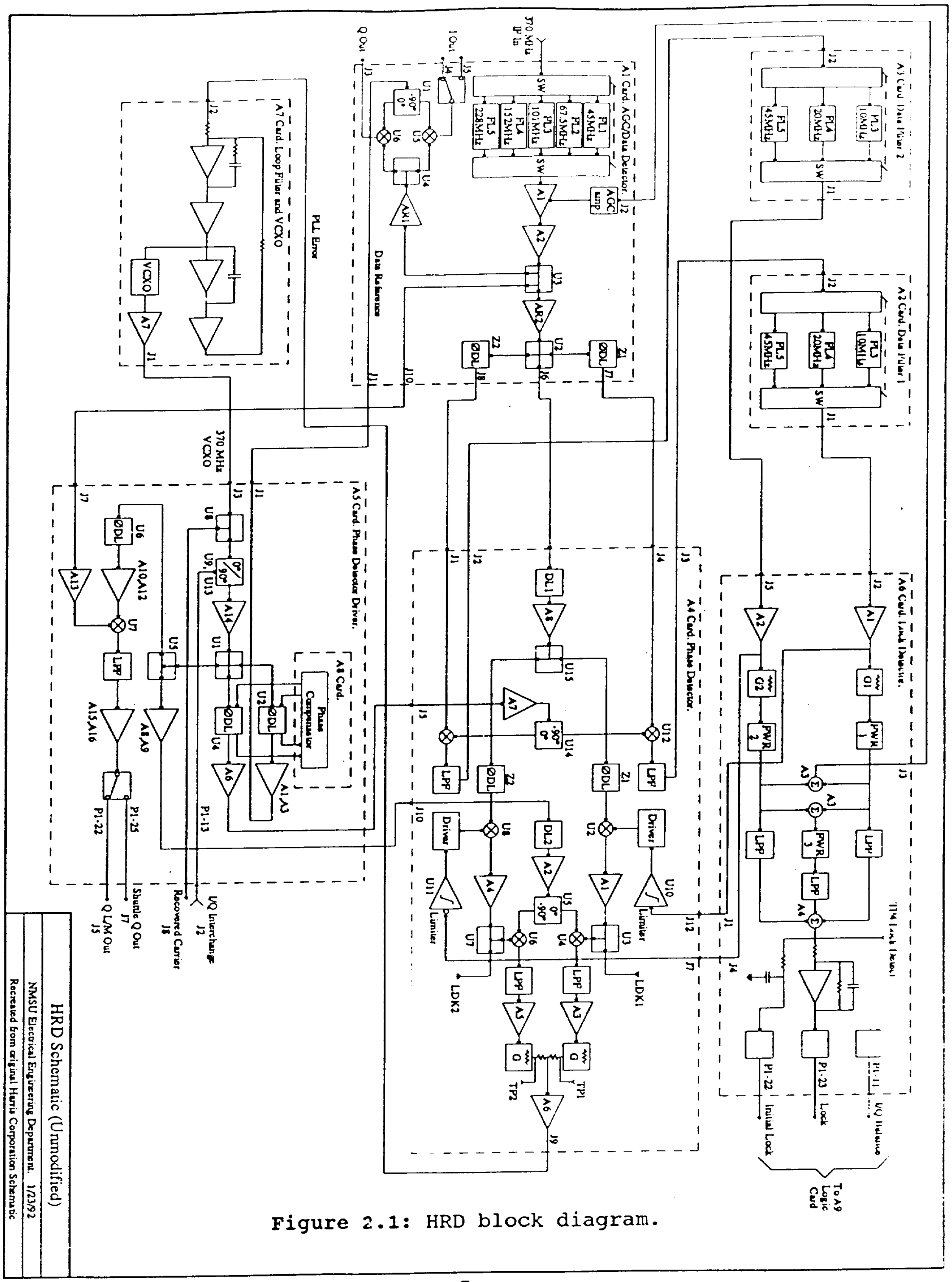


Figure 2.1: HRD block diagram.

Upon entering the A1 card, the IF signal is filtered via one of the five front end bandpass filters. The filter used selected based on the baseband data rate (i.e. the IF is filtered at the approximate 95% power point of the spectrum).

The AGC is used to maintain a constant signal plus noise power level to the A4 card, A5 card, and to the data detection circuits.

The bandlimited IF with a constant level is then split 4-ways. Three of these split outputs go to the A4 card through connectors J7, J6, and J8. The other goes to the A5 card through connector J10. Another split, bandlimited IF signal is routed to the data detection circuitry. The data detection circuitry receives the Data Reference signal from the A5 card and mixes it with the IF at the mixers labeled U5 and U6. This provides the baseband I and Q data streams to the bit synchronizer (I Out and Q Out).

2.2 Data Filter Cards (A2 and A3 cards)

The Data Filter cards (A2 and A3 cards) are identical. Each card contains three time delay matched low pass filters. These low pass filters are the I and Q channel arm filters for the modified Costas Carrier Recovery Loop of the PLL Detector card (A4 card). The filter used is selected based upon the rate of the baseband data. The filter cards receive baseband data from the loop detector circuits located on the A4 card. The filtered baseband data is sent to the Lock Detector card (A6 card) where it is amplified and then processed for AGC and lock detection functions.The filtered, amplified data is then returned to the A4 card for I and Q cross correlation.

2.3 PLL Detector Card (A4 card)

The A4 card is the heart of the HRD. This card is a modified version of the Costas Loop which is used to phase lock the recovered carrier to the IF.

The A4 card receives three equal amplitude and phase signals from the A1 card. Two of these signals are mixed with a reference signal to produce the baseband data. The results of this quadrature detection are then sent to the Data Filter Cards (one for the I channel and one for the Q channel) through the connectors labeled J2 and J3. The Data Filter outputs are then brought back onto the A4 card through connectors J7 and J12. Here, the filtered data is recovered by ECL comparators which in turn drive the multipliers U2 and U8. The ECL comparators are essentially hard limiters that perform bit decisions. The 370MHz filtered IF that is multiplied at the mixers U2 and U8 is delayed through the phase delay line, DL1, such that the IF and recovered data are multiplied in the same time interval. The result of this remodulation of the carrier is then quadrature detected to baseband through the mixers labeled U4 and U6. The resulting signals are then subtracted to generate the error signal for the carrier recovery loop. This error signal is routed to the Loop Filter card (A7 card) through connector J9.

Since this card is so relevant to the HRD and to the intended modifications, it should warrant a finer understanding of the PLL

Detector. To accomplish this, a mathematical analysis of the Modified Costas Loop follows.

Figure 2.2 on page 9 shows an expanded block diagram view of the A4 card including the data filters.The circuitry here comprises the Modified Costas Loop. This loop type is used here because of it's ability to need mixing at the IF only and not at baseband.

Let the received signal, s(t), be represented by,

$$s(t) = I_m \cos(\omega_c t + \theta) + Q_m \sin(\omega_c t + \theta) \tag{1}$$

where: 1) I_m is the I channel data stream.

2) Q_m is the Q channel data stream.

3) $\omega_c = 2\pi f_c$, f_c is the carrier frequency.

4) θ is the reference phase of the received signal.

The received signal is first mixed with quadrature components of the recovered carrier in the mixers U12 and U13. So, before lowpass filtering we have

$$\begin{aligned} I_{comp} &= s(t)\cos(\omega_c t) \\ &= I_m\cos(\omega_c t + \theta)\cos(\omega_c t) + Q_m\sin(\omega_c t + \theta)\cos(\omega_c t) \qquad (2) \end{aligned}$$

$$\begin{aligned} Q_{comp} &= s(t)\sin(\omega_c t) \\ &= I_m\cos(\omega_c t + \theta)\sin(\omega_c t) + Q_m\sin(\omega_c t + \theta)\sin(\omega_c t) \qquad (3) \end{aligned}$$

Using the following trigonometric relationships, we can solve the equations (2) and (3):

$$\cos A \cos B = 1/2[\cos(A+B) + \cos(A-B)]$$

$$\sin A \cos B = 1/2[\sin(A+B) + \sin(A-B)]$$

$$\sin A \sin B = 1/2[\cos(A-B) - \cos(A+B)].$$

Using the above identities and taking into account the lowpass filtering to remove the higher order frequency terms, we have at

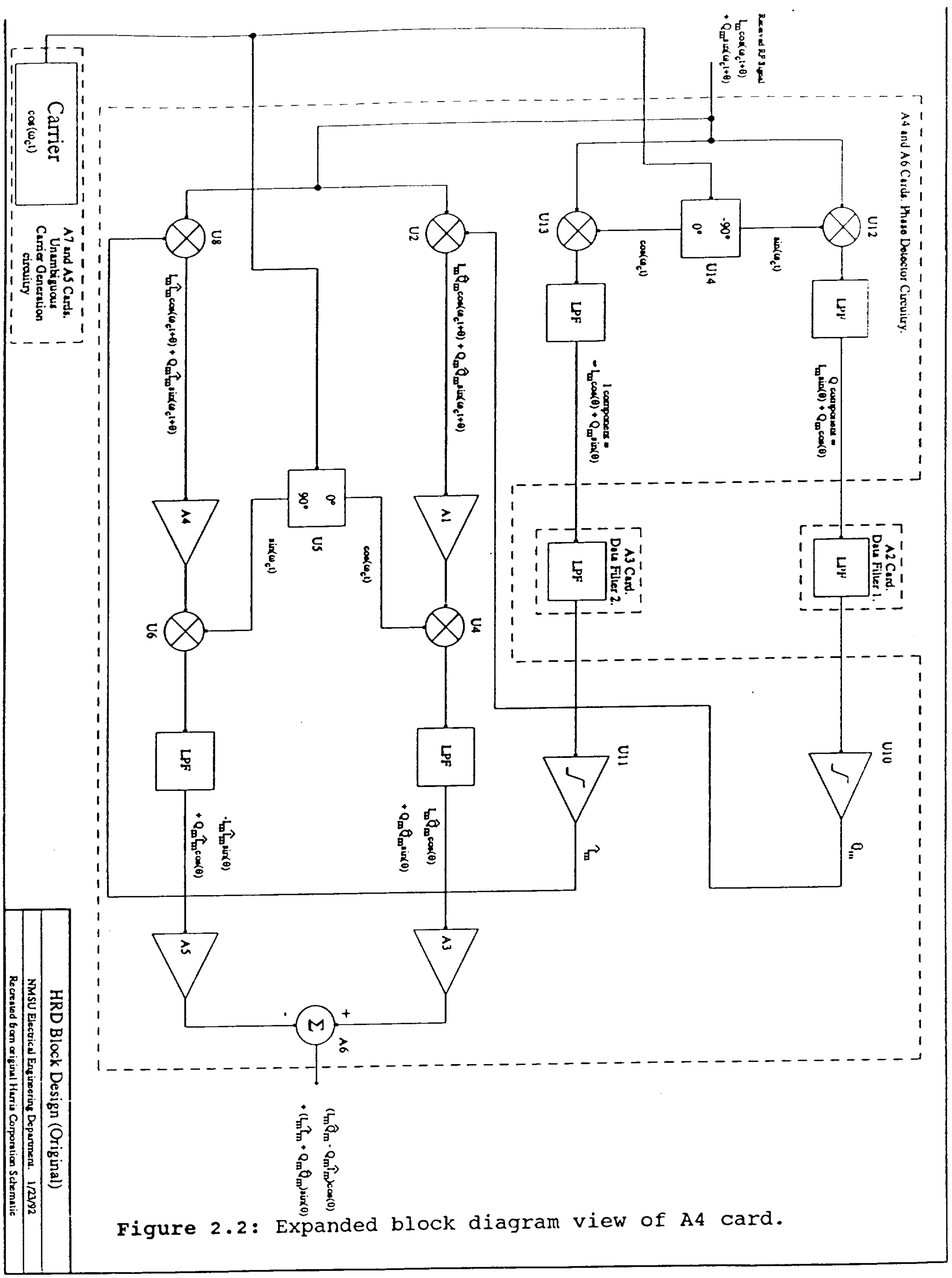

Received RF Signal
$I_m\cos(\omega_c t+\theta) + Q_m\sin(\omega_c t+\theta)$
A4 and A6 Cards Phase Detector Circuitry.
U12
U13
U14
-90°
0°
$\sin(\omega_c t)$
$\cos(\omega_c t)$
LPF
Q component = $I_m\sin(\theta) + Q_m\cos(\theta)$
I component = $I_m\cos(\theta) + Q_m\sin(\theta)$
A2 Card. Data Filter 1.
A3 Card. Data Filter 2.
U10
U11
$\hat{Q}_m$
$\hat{I}_m$
Carrier
$\cos(\omega_c t)$
A7 and A5 Cards. Unambiguous Carrier Generation circuitry
U2
U8
U5
U4
U6
A1
A3
A4
A5
A6
$I_m\hat{Q}_m\cos(\omega_c t+\theta) + Q_m\hat{Q}_m\sin(\omega_c t+\theta)$
$I_m\hat{I}_m\cos(\omega_c t+\theta) + Q_m\hat{I}_m\sin(\omega_c t+\theta)$
$I_m\hat{Q}_m\cos(\theta) + Q_m\hat{Q}_m\sin(\theta)$
$-I_m\hat{I}_m\sin(\theta) + Q_m\hat{I}_m\cos(\theta)$
$(I_m\hat{Q}_m - Q_m\hat{I}_m)\cos(\theta) + (I_m\hat{I}_m + Q_m\hat{Q}_m)\sin(\theta)$
HRD Block Design (Original)
NMSU Electrical Engineering Department. 1/23/92
Recreated from original Harris Corporation Schematic

Figure 2.2: Expanded block diagram view of A4 card.

the output of the lowpass filter,

$$I_{comp} = I_m\cos(\theta) + Q_m\sin(\theta) \quad (4)$$

and, $$Q_{comp} = -I_m\sin(\theta) + Q_m\cos(\theta). \quad (5)$$

In the next steps, the quadrature detected signals are sent through the data filters and then to the hard limiters. At this point, we have what should be a good estimate of the original baseband data. Denote these estimates as $I^{\sim}_m$ and $Q^{\sim}_m$.

These baseband data estimates are then remodulated with the carrier at the multipliers U2 and U8. The carrier has been delayed by the delay line DL1 so that there is not a reference time problem.

The remodulation process produces the two channel terms,

$$I_{rm}(t) = s(t)(Q^{\sim}_m)=I_mQ^{\sim}_m\cos(\omega_c t + \theta) + Q_mQ^{\sim}_m\sin(\omega_c t + \theta) \quad (6)$$

$$Q_{rm}(t) = s(t)I^{\sim}_m = I_mI^{\sim}_m\cos(\omega_c t + \theta) + Q_mQ^{\sim}_m\sin(\omega_c t + \theta). \quad (7)$$

The results I_{rm} and Q_{rm} are then quadrature detected again with the recovered carrier reference. So, from the U4 mixer we have:

$$I_{rm,comp} = I_{rm}(t)\cos(\omega_c t)$$
$$=I_mQ^{\sim}_m\cos(\omega_c t+\theta)\cos(\omega_c t) + Q_mQ^{\sim}_m\sin(\omega_c t+\theta)\cos(\omega_c t). \quad (8)$$

And, from the U6 mixer we have:

$$Q_{rm,comp} = Q_{rm}(t)\sin(\omega_c t)$$
$$= I_mI^{\sim}_m\cos(\omega_c t+\theta)\sin(\omega_c t) - Q_mI^{\sim}_m\sin(\omega_c t+\theta)\sin(\omega_c t). \quad (9)$$

Thus, using the previously presented trigonometric identities and taking into account the lowpass filtering, we have at the outputs of the lowpass filters:

$$I_{rm,comp} = I_mQ^{\sim}_m\cos(\theta) + Q_mQ^{\sim}_m\sin(\theta) \quad (10)$$

and, $$Q_{rm,comp} =-I_mI^{\sim}_m\sin(\theta) + Q_mI^{\sim}_m\cos(\theta). \quad (11)$$

The components in equations (10) and (11) are then subtracted to produce the error signal given by,

$$e_r = I_{rm,comp} - Q_{rm,comp}$$
$$= I_m\tilde{Q}_m\cos(\theta) + Q_m\tilde{Q}_m\sin(\theta) + I_m\tilde{I}_m\sin(\theta) - Q_m\tilde{I}_m\cos(\theta)$$
$$= (I_m\tilde{Q}_m - Q_m\tilde{I}_m)\cos(\theta) + (I_m\tilde{I}_m + Q_m\tilde{Q}_m)\sin(\theta). \qquad (12)$$

Finally, if the estimates of the baseband data are good estimates, then we have

$$\tilde{I}_m = I_m$$

and,
$$\tilde{Q}_m = Q_m.$$

These relationships then reduce equation (12) to

$$e_r = (I^2 + Q^2)\sin(\theta).$$

Since $I^2 + Q^2 = 1$, we have

$$e_r = \sin(\theta). \qquad (13)$$

Thus, the error signal is strictly dependant on the phase of the carrier. This result will become important later when the modifications are discussed (see chapter 3.3).

2.4 Detector Driver Card (A5 card)

Refer again to figure 2.1.

The Detector Driver card receives the VCXO 370MHz signal from the Loop Filter card. It then generates the recovered carrier output, the various reference signals required by the carrier recovery loop and the data detector circuitry. On this card is a selective phase shifter that rotates the phase of the recovered carrier to 0 or 90 degrees. This interchanges the I and Q channels under backwards lock conditions.

2.5 Lock Detector Card (A6 card)

The Lock Detector card provides control signals to the Loop Filter card (A7 card) such that the sweep and loop bandwidth of the filter are automatically controlled during acquisition.

The Lock Detector card receives the I and Q channel data filter outputs, amplifies them, and then splits them to drive both the PLL Detector card multipliers and further lock detection circuitry.

Figure 2.3 on page 13 shows a block diagram of the HRD with emphasis on the lock detector circuitry.

When the loop is in lock, both I and Q low pass filter outputs will consist of noisy baseband data.

The full wave rectifiers (FWR) in the I and Q arms will have outputs with DC values proportional to the signal level. The mean value will correspond to the signal level, but will be corrupted by noise.

The difference of the I and Q FWRs is taken at the summing node 2 ($\Sigma2$). Here, the average DC levels cancel because they are equal, and a noisy signal remains.

The output of FWR3 is a noisy DC level whose average value is proportional to the noise level. The lowpass filtered outputs of FWR1, FWR2, and FWR3 are combined at summing node 3 ($\Sigma3$) resulting in a DC value whose level is a good estimate of the signal level. This entire process is depicted in figure 2.4 on page .

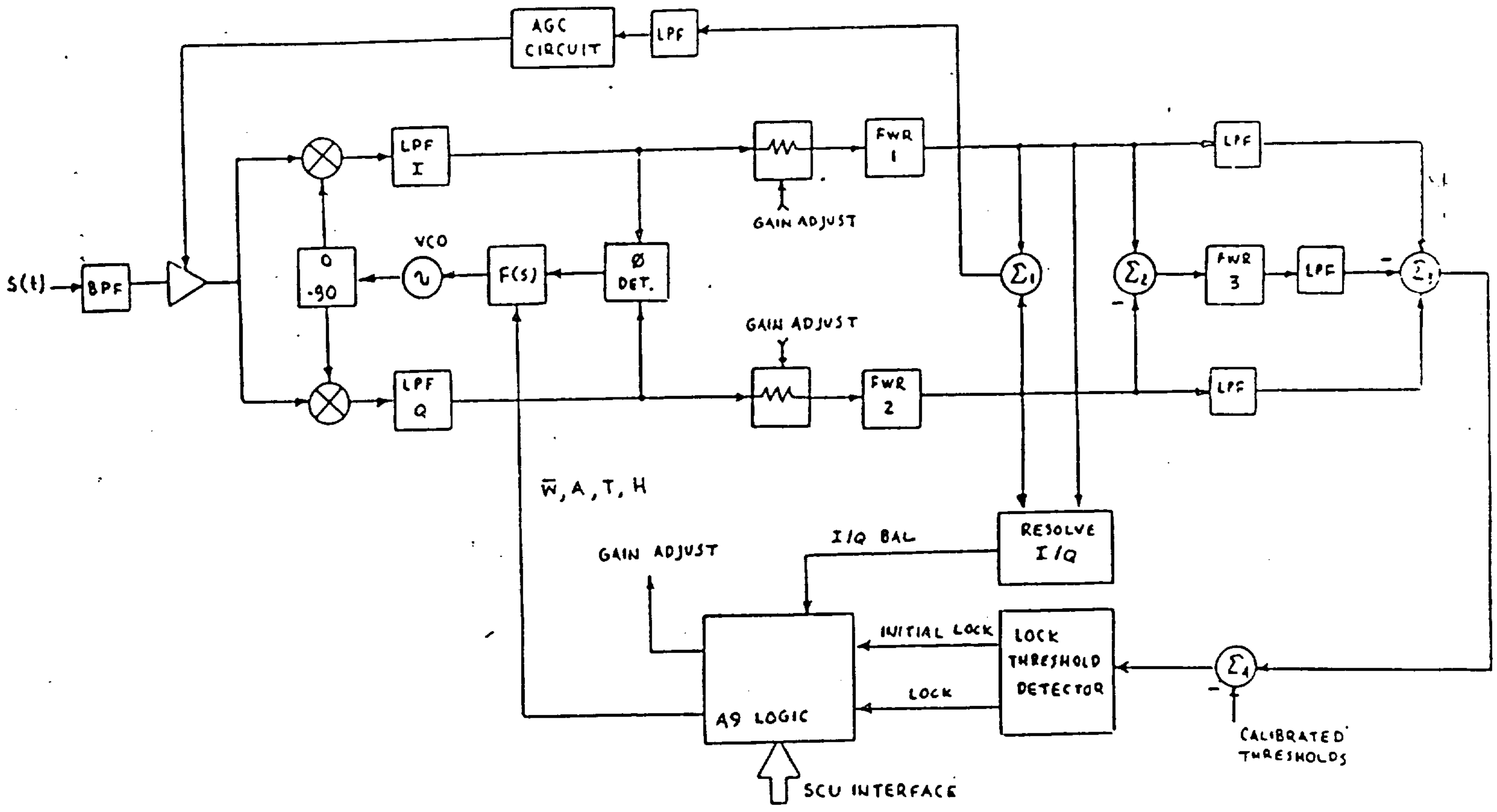


Figure 2.3: Block diagram of the HRD with emphasis on lock detect

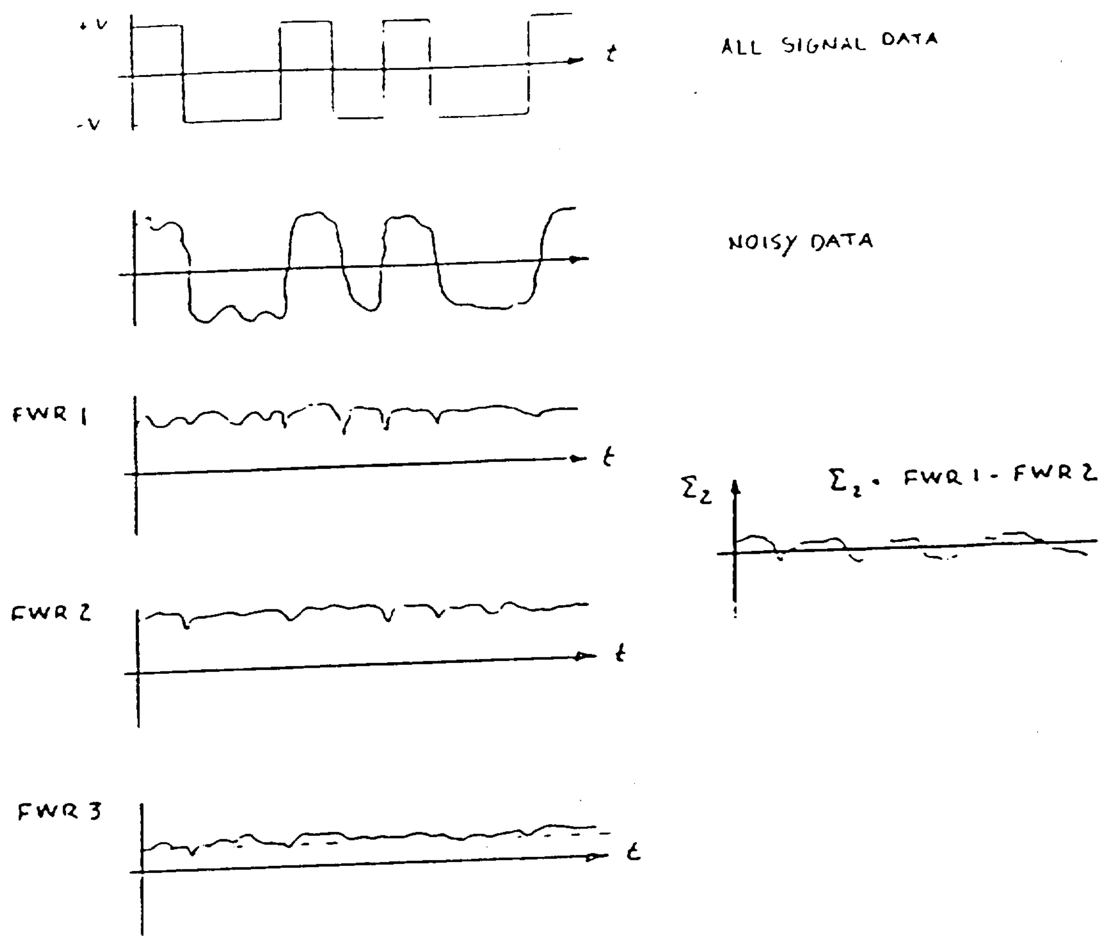


Figure 2.4: Graphical representation of Lock Detector.

The signal estimate is then compared to a calibrated threshold and lock is maintained if this threshold is exceeded. When the loop is not locked, the I and Q filter arm outputs will not have equal signal levels. A beat note with frequency equal to the carrier offset will be present in the lock detector. The loop filter will drive the VCO in a direction such that the beat note will cross zero frequency. At that point the proper cancellations will occur and the lock detector will cross the threshold, achieving lock.

2.6 Loop Filter Card (A7 card)

The phase error signal from the PLL Detector card is filtered by the loop filter circuitry. This happens before driving the VCXO to obtain the proper loop characteristics for both acquisition and tracking modes. A ramp signal is generated from the closed loop sweep circuitry and is combined with the PLL error signal during acquisition. Provision is made in the sweep circuitry for full ±508kHz acquisition or for breaking lock. The VCXO 370MHz output drives the Detector Driver card.

2.7 Phase Compensator Card (A8 card)

The Phase Compensator card contains temperature compensation components used to maintain proper input voltages to the voltage controlled phase shifters located on the Detector Driver card.

2.8 Logic Card (A9 card)

The Logic card is used to decode the System Control Unit (SCU) commands into mode, bandwidth, and gain controls appropriate for each of the circuit cards. Additionally, located on the Logic card are window detectors for the Lock Detector card outputs to provide false lock rejection and automatic signal acquisition sequencing of the loop filter sweep and bandwidth controls. It also includes the power balance decision logic to prevent backwards lock for the ±6dB I/Q power modes and also the A/D converter which generates the signal strength word to be returned to the SCU.

Reference:

The preceding information was obtained from:

Harris Corp. Document TM6100-004, Vol. II, Chapter 5J, "The Operation and Maintenance for Receive Components", pp. 5J-1-1 to 5J-2-14 and appendix "Theory of Operation", pp. 3 - 22.

3.0 <u>Required Modifications</u>

The ultimate goal of this work was to modify the existing HRD, which is limited in it's operation to QPSK modulation, in such a fashion that it could be capable of operating in an 8-PSK environment. The reasons for such modifications are twofold: 1) In order to recognize a TCM coding gain in the entire system, it is required that an 8-PSK modulation process be supported in the overall communications system. 2) The TDRSS system is a highly complex entity. The HRD was itself designed for the specific task of demodulating data in this unique environment. Therefore, it was decided for time constraint reasons, as well as financial considerations, the task would be best approached by modifying the existing HRD to operate in the 8-PSK world rather than to build a new modulator from the ground up.

When the redesign of the HRD was first initiated, it was recognized that there were a few basic differences between the existing HRD and our proposed modified HRD. First, in the 8-PSK system, the data rates are a magnitude slower than the current data rates of the existing system (1Mbps for the 8-PSK system versus 10Mbps minimum for the TDRSS system). Therefore, our system bandwidth requirements will be much smaller. In the HRD, this means that the existing filters should be replaced with filters that have smaller bandwidths.

Secondly, QPSK data can take on one of two values (in a noise free case). Either the data is +V or -V in amplitude. In the 8-PSK case, the data can take on one of four different values. These values can be represented as ±aV or ±bV (see figure 3.0).

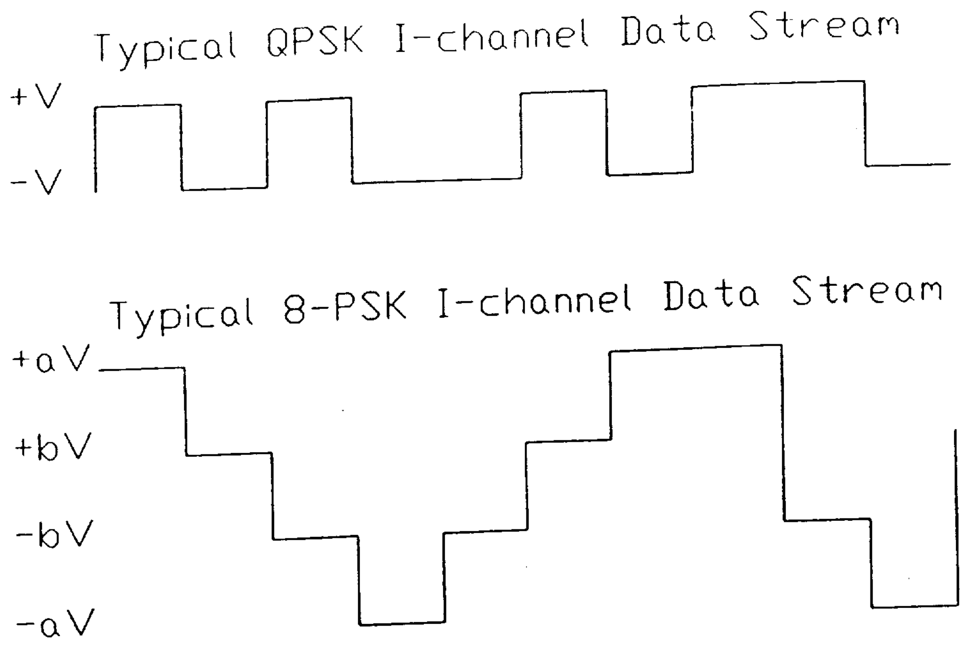

Typical QPSK I-channel Data Stream
+V
-V
Typical 8-PSK I-channel Data Stream
+aV
+bV
-bV
-aV

Figure 3.0: 8-PSK data representation.

For this reason, any bit decision circuitry in the original HRD must be replaced with hard decision circuits that can provide the proper levels for 8-PSK data streams.

Finally, the HRD resolves phase ambiguity problems (backwards lock) by phase shifting the recovered carrier through 0 or 90 degrees. Phase ambiguity is a greater problem in the 8-PSK world and a method for resolving this issue is of importance. The implications of this will be discussed later.

In order to properly facilitate a discussion of the needed modifications, the design changes made to the HRD will be discussed for each card separately.

3.1 AGC/Data Detector Card (A1 card)

The AGC/DATA card as part of it's responsibilities, performs the front end filtering of the IF. Since our bandwidth constraints are smaller for the 8-PSK application, a filter with a smaller bandwidth is needed here. The current filter FL1 has the smallest bandwidth of 45MHz. This is still too large for the final application. The extra bandwidth would be overkill and could introduce additional undesired noise into the system. Thus, for this particular application,a filter centered at 370MHz with a 3dB bandwidth of 10MHz is to be positioned on the external IF connector into the box to provide front end filtering.

The remaining function of this card is to amplify and split the IF signal. Neither of these processes will affect the 8-PSK modulated IF. So, no further modifications are needed on this board.

3.2 Data Filter Cards (A2 and A3 cards)

The situation here is the same as in the AGC/DATA Detector card. Since our data rates are slower, data filters with smaller bandwidths are required. In this case, we have elected to add our own lowpass data filters. The data filters we have selected are lowpass Bessel filters with a 3dB bandwidth of 2MHz. The placement of these filters is discussed in chapter 4.

3.3 PLL Detector Card (A4 card)

The major modification to the HRD was completed on the PLL Detector card. As discussed earlier, any part of the HRD that makes a two level hard bit decision will have to be replaced with a hard bit decision circuit capable of generating 8-PSK levels. The A4 card contains to hard limiters used to make the QPSK bit decisions. These hard limiters are to be replaced with circuitry that can make the bit decisions on four levels. Since this is the bulk of the HRD modifications, a complete discussion of these circuits, their development and implementation, are covered in chapter 4.

There are no other modifications needed to the A4 card. The Modified Costas Loop is not dependant on baseband data levels. It only relies on the fact that the data estimates from the hard bit decision circuits are good estimates of the original data streams in each channel. The analysis set forward in chapter 2.3 verifies this assertion. As shown in chapter 2.3, when good estimates of the baseband data are available, the resulting error signal will only be a function of the phase of the IF signal. Therefore, it is not dependant on the actual levels of the data, whether they be two level or four level data types. So, no other modifications to the A4 card are warranted.

3.4 Detector Driver Card (A5 card)

This card is only affected by the error signal derived on the A4 card and it will not be affected by switching to an 8-PSK format. However, one "modification" does need to take place. As stated earlier, if the HRD locks on a wrong QPSK vector, the lock point can be changed by shifting the recovered carrier 0 or 90 degrees. Phase ambiguity resolution is not as simple in the 8-PSK world. So, here the I/Q Interchange line must be "deactivated" to keep the phase of the recovered carrier from shifting 0 or 90 degrees. This can be accomplished by pulling the I/Q Interchange line high.

The question now arises, how will phase ambiguity resolution be resolved? In the context of 8-PSk demodulation, the demodulator will be locked on the wrong signal vector if it locks on any symbol other than the 000 symbol. Figure 3.1 shows the signal constellation for QPSK and 8-PSK (naturally mapped) and the location of the correct locking point.

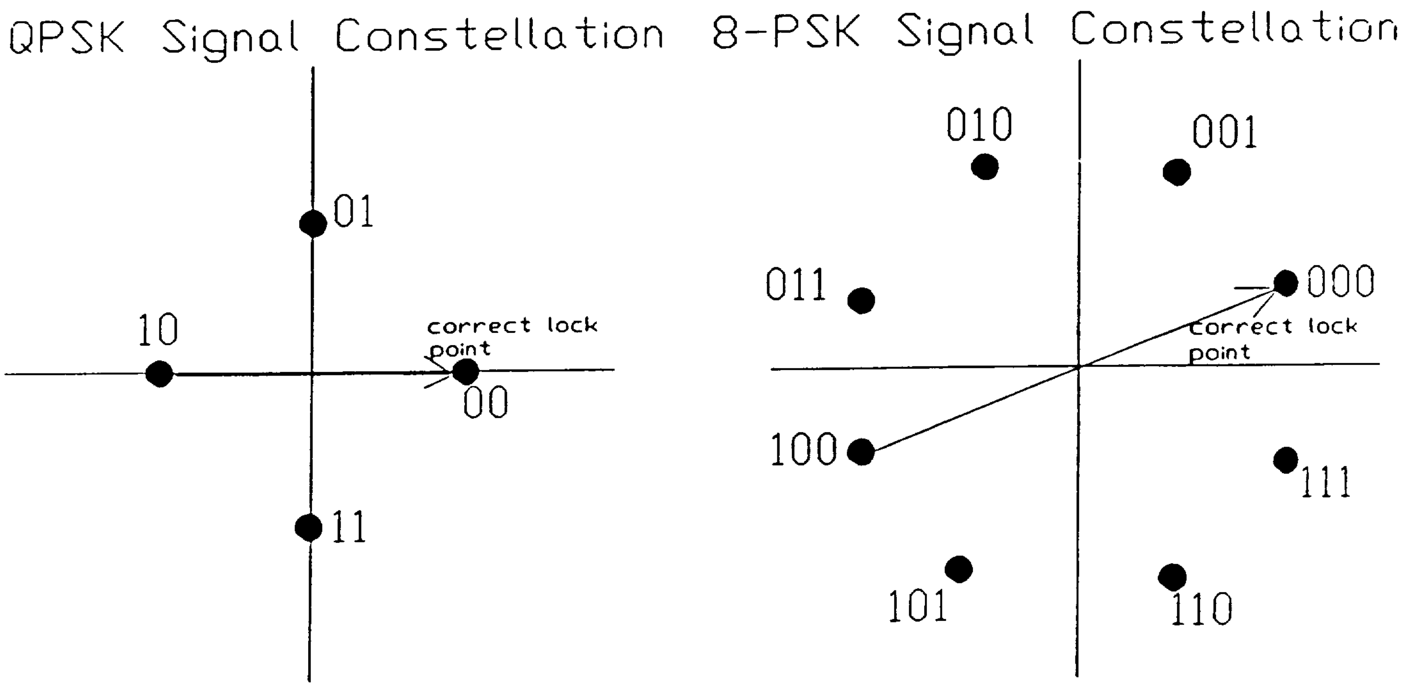


Figure 3.1: QPSK and 8-PSK signal constellations.

For QPSK, if the HRD locks anywhere other than the 00 symbol, a simple rotation of the recovered carrier by 90 degrees will bring it back to the proper locking position. However, for 8-PSk if the HRD locks on any symbol other than the 000 symbol then the HRD is incorrectly locked and a phase rotation of the recovered carrier by 90 degrees will not solve this locking condition (unless the HRD locks 90 degrees away from the 000 symbol).

This problem is not easily solved inside the HRD. So, it was decided that this phase ambiguity problem would be resolved in the symbol synchronizer farther down the receiver chain. Since the phase rotation is now not a function of the HRD, it will not be further discussed here.

3.5 Lock Detector Card (A6 card)

Originally it was believed that the lock detector would work with 8-PSK data. However, testing and analysis showed otherwise. The lock detector works fine for data that comprises no more than two levels. This becomes a problem when using data that has four distinct levels. The problem is that the lock detector uses full wave rectifiers (FWR) to obtain the lock condition. These FWRs are based on the assumption that the data has only two levels. In this manner, the two channels can be full wave rectified and then subtracted from each other to give the noise component of the data (see chapter 2.5 for the discussion on the Lock Detector card). If this method were used for 8-PSk data, the result would not be the noise component of the data. Rather, it would be a square wave as shown in figure 3.2 (compare this to figure 2.4 in chapter 2.5).

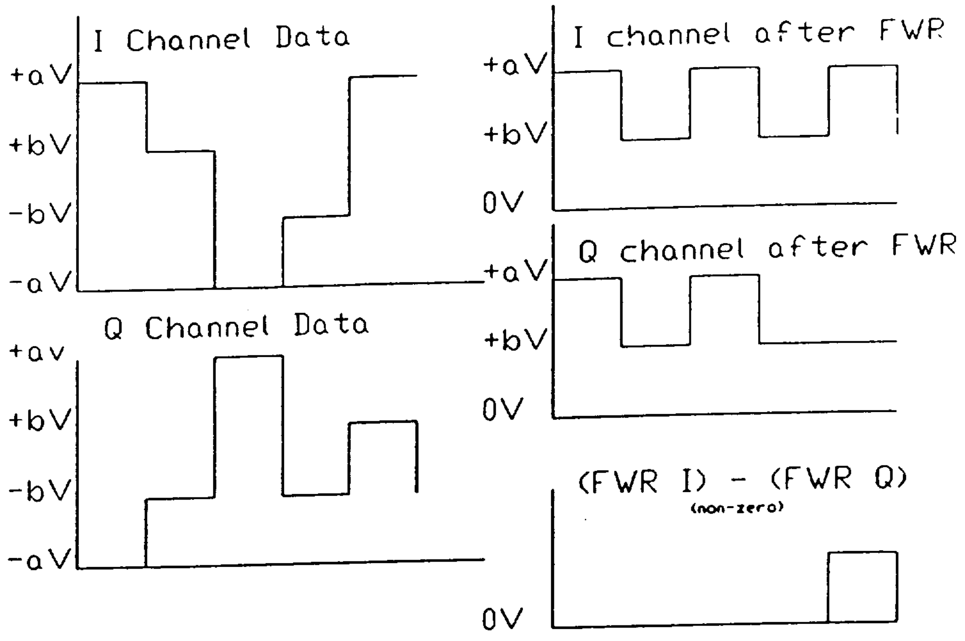


Figure 3.2: Lock detector operating on 8-PSK data.

Obviously, this will not be acceptable for lock detection purposes. The solution in this case is to force the HRD into lock. The method for doing this is not within the scope of this report and instead will be pursued by other members of the project team.

The AGC is also a part of this card. However, since the current lock detector is not usable, the AGC circuits will need to be forced to do their intended job. Once again, this is out of the scope of this report and will be pursued by other project team members.

3.6 Loop Filter Card (A7 card)

The only modification needed to this card is to narrow the loop bandwidth. This modification was determined to be needed from HRD test results. This change is also far beyond the scope of this report and is instead being completed by other project team members.

3.7 Phase Compensator Card (A8 card) and Logic Card (A9 card)

No modifications are needed to either of these two cards.

4.0 <u>HRD 8-PSK TCM Modification Board Design</u>

As aiscussed earlier, the majority of tne work in modifying the HRD was in the replacement of the hard limiters on the A4 card with multilevel bit decision circuitry. The basic concept here was to develop circuits that could take the baseband data from the A4 card, filter it using the new filters, and then make a hard decision on the data to one of four levels. The components needing modification are shown as the cross hatched areas in figure 4.0 on page 26.

For this application, it was decided that the old Data Filter cards (A2 and A3 cards) would be removed from the box. The new filters and data estimate circuitry would then be placed on the same card and installed in the old Data Filter card positions. Remember, there is an I channel of data and a Q channel of data. So, we need two boards, one for each channel. The functionality of each board is identical. However, the alignment of the two cards is different and will be discussed later.

Figure 4.1 on page 27 shows a modified block diagram of the HRD with the modifications.In this figure, we have combined the functionality of the two new boards to indicate only how the changes affect the HRD layout. In this scheme, the baseband data is removed from the A4 card at connectors J2 and J3. The data is filtered via the new data filters. The data signals are then amplified to match the levels required by the lock detector. Next, the data signals are split with one line driving the lock detector and the other driving the hard decision circuitry. Once the

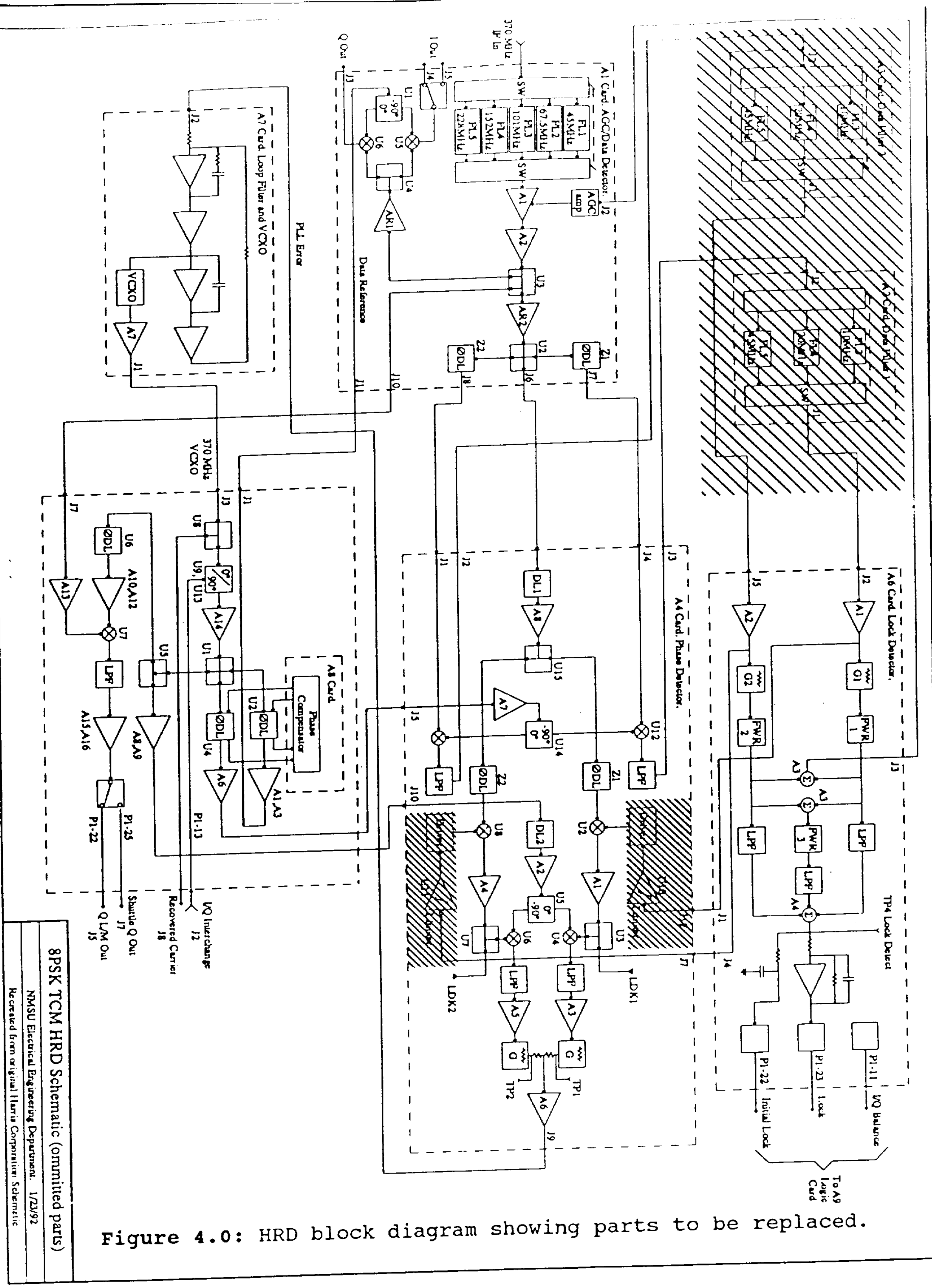


Figure 4.0: HRD block diagram showing parts to be replaced.

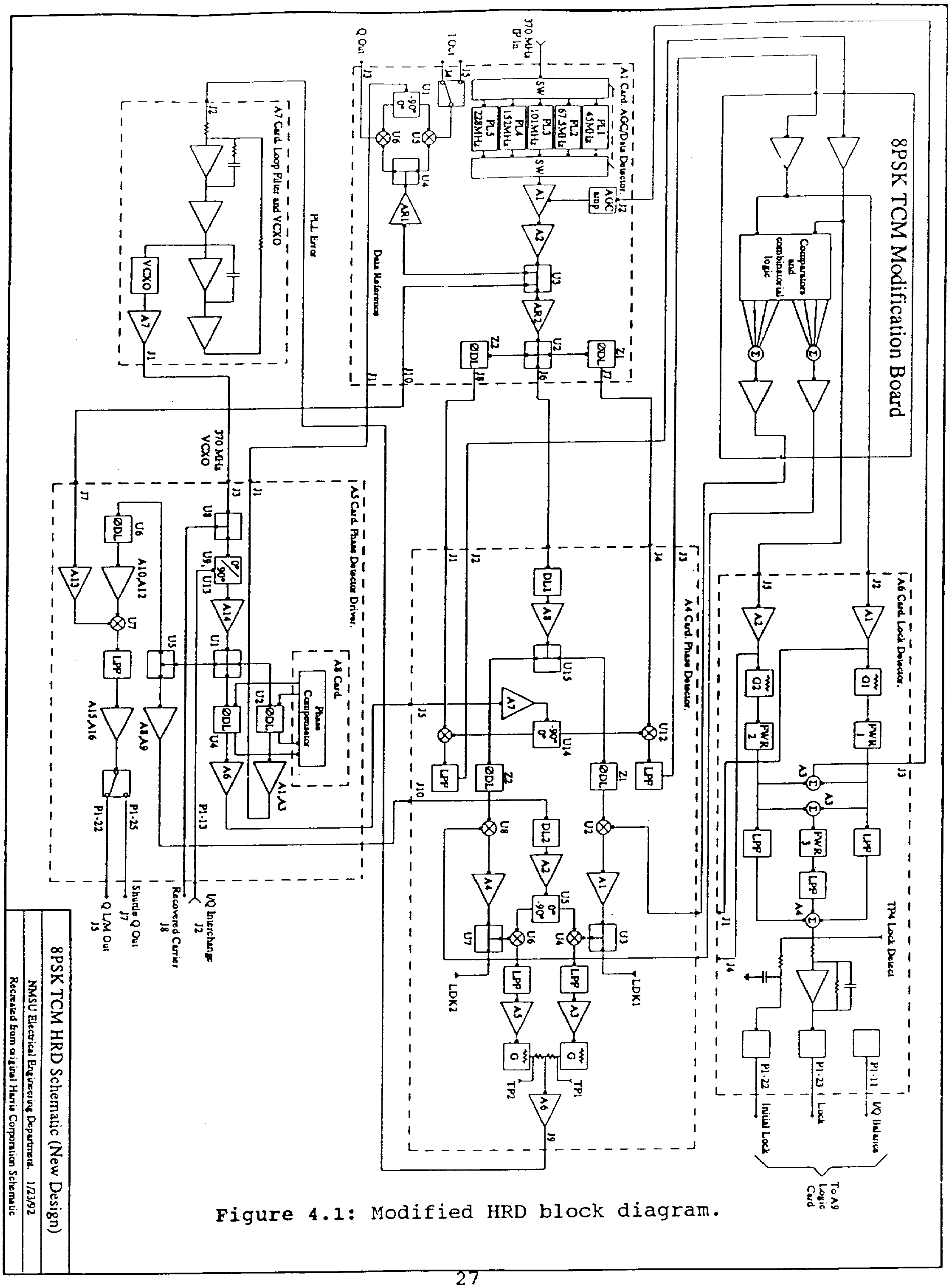


Figure 4.1: Modified HRD block diagram.

cicuits have made a hard decision on the data, the data estimates are given enough gain to directly drive the multipliers U2 and U8 on the A4 card. Note that the original limiters and drivers have been removed from the A4 card. In the old version of the HRD, the lock detector drove the inputs to the hard limiters. These connectors have been removed from the lock detector card at connectors J1 and J4 and terminated in 50Ω.

Figure 4.2 on page 29 shows an expanded block diagram of the A4 card.The cross hatched area details the changes to be made on the card.

Figure 4.3 on page 30 shows how these changes are incorporated into the new boards.

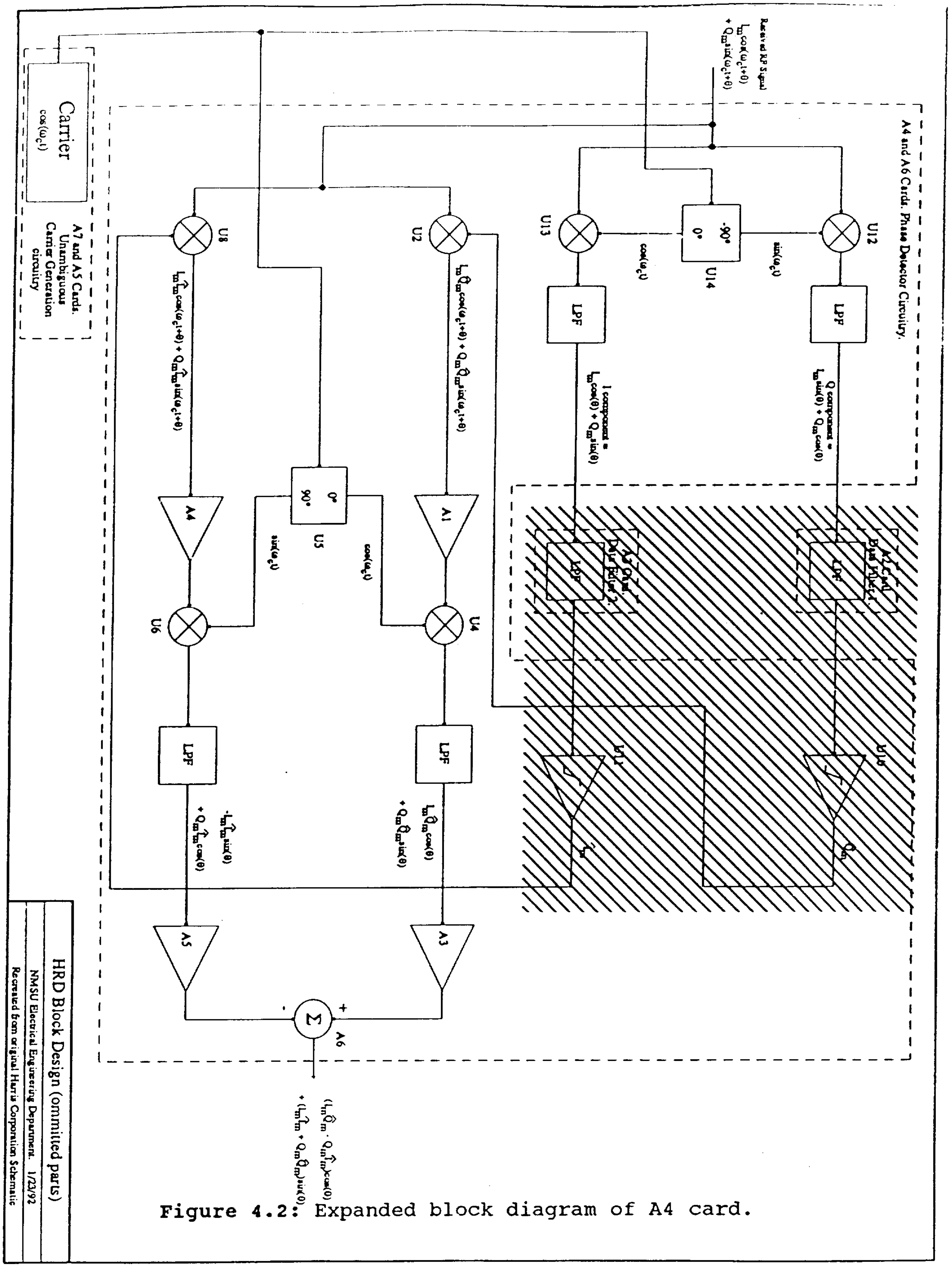
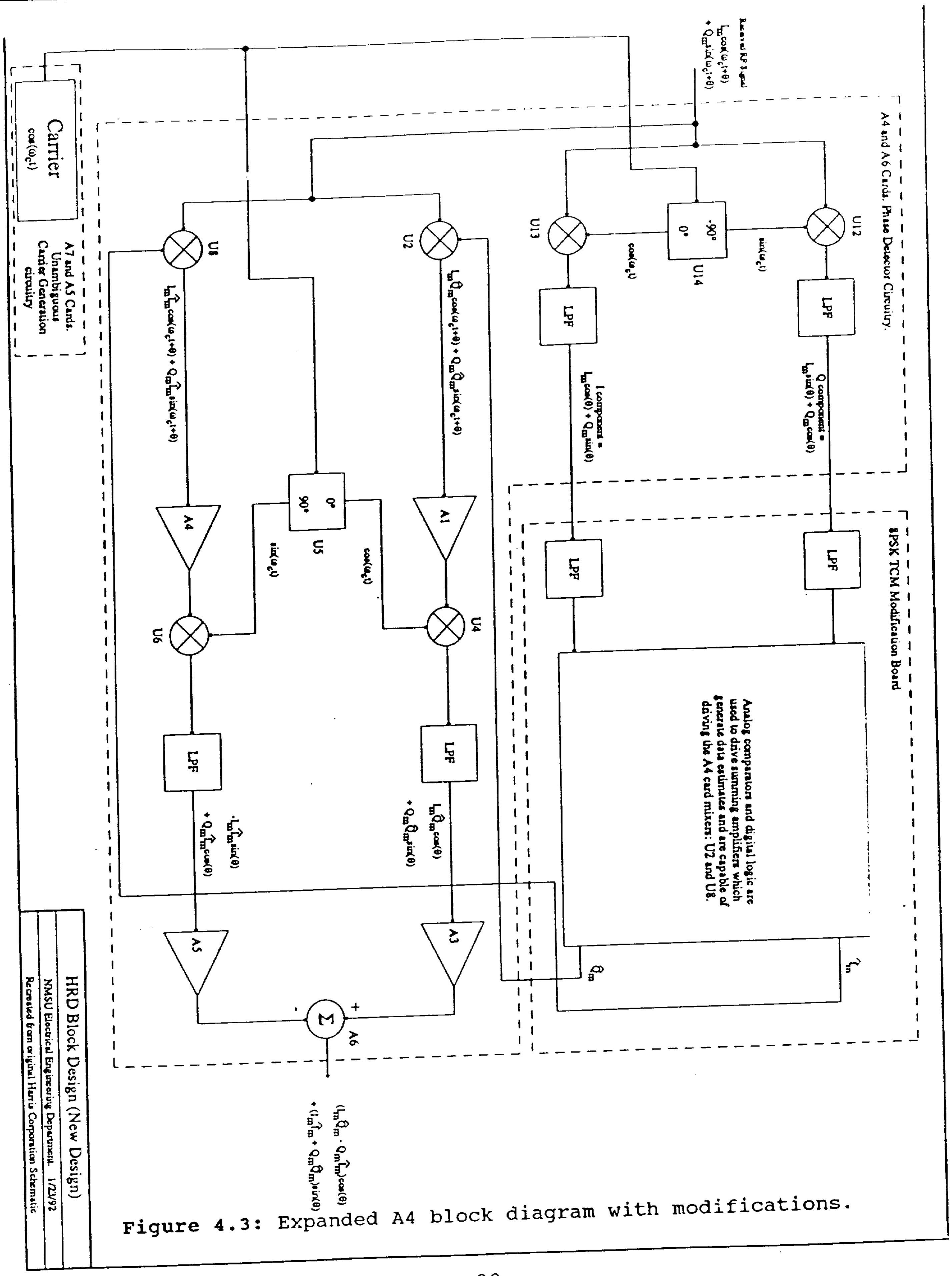

Received RF Signal
A4 and A6 Cards. Phase Detector Circuitry.
Carrier
cos(ω_c t)
A7 and A5 Cards. Unambiguous Carrier Generation circuitry
U2
U4
U5
U6
U8
U12
U13
U14
U16
LPF
A1
A3
A4
A5
A6
I component
Q component
HRD Block Design (ommitted parts)
NMSU Electrical Engineering Department. 1/23/92
Recreated from original Harris Corporation Schematic

Figure 4.2: Expanded block diagram of A4 card.



Received RF Signal
A4 and A6 Cards. Phase Detector Circuitry.
U12
U13
U14
-90°
0°
LPF
Q component
I component
8PSK TCM Modification Board
Analog comparators and digital logic are used to drive summing amplifiers which generate data estimates and are capable of driving the A4 card mixers: U2 and U8.
Carrier
A7 and A5 Cards. Unambiguous Carrier Generation circuitry
U2
U8
U5
U4
U6
A1
A4
A3
A5
A6
HRD Block Design (New Design)
NMSU Electrical Engineering Department. 1/23/92
Recreated from original Harris Corporation Schematic

Figure 4.3: Expanded A4 block diagram with modifications.

4.1 Development of the 8-PSK Hard Decision Bit Slicer

In this section, an attempt is made to explain the development of the circuitry used to make the hard bit decisions.

It is best to begin with an exact overview of the nature of the problem. Figure 4.4 shows the constellation for the 8-PSK system.

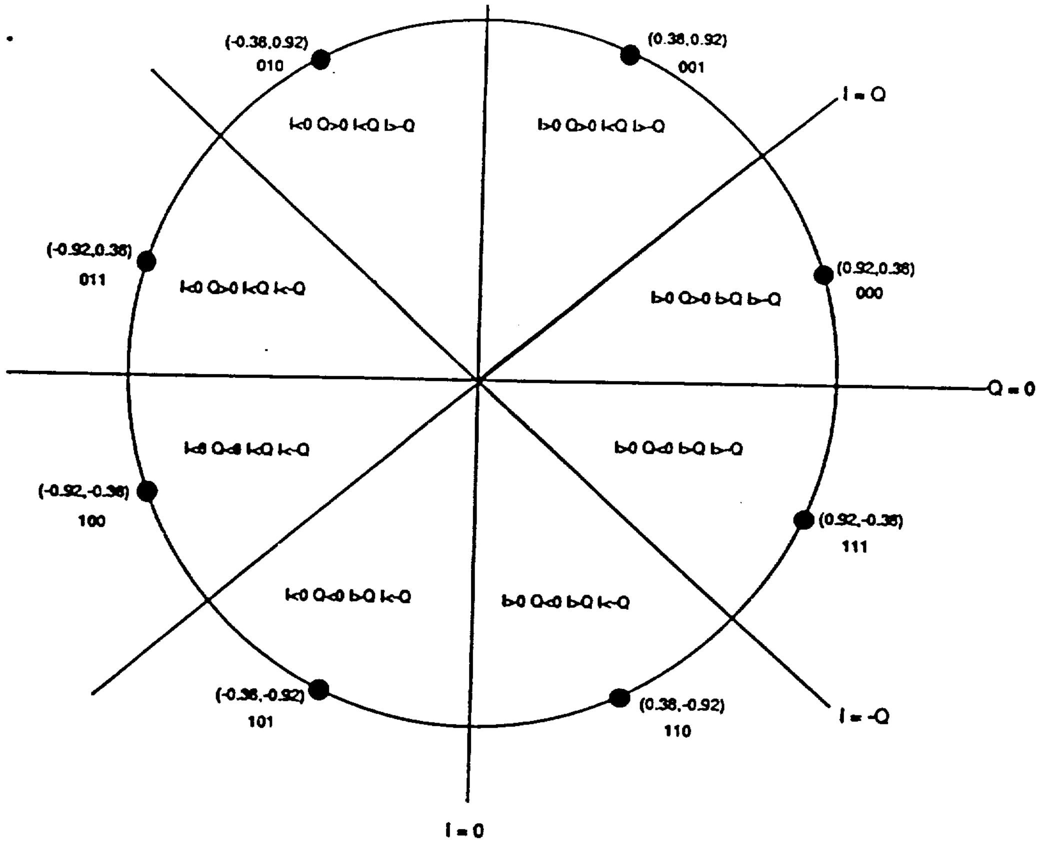


Figure 4.4: An 8-PSK naturally mapped constellation.

This particular constellation is naturally mapped with the vectors offset from the axes by 22.5 degrees. From this figure, we can understand the arrangement of the 8-PSK symbols. Without noise, the data points of the constellation would be mapped as shown in figure 4.4. That is to say, if the symbol 000 were sent, then the I channel data would take on the value 0.92V and the Q channel data would take on the value of 0.38V. These voltage values are derived from the fact that the symbols lie on a unit circle. So, the voltage values of the symbol components (I and Q) are simply the cosine and sine of the symbol angle respectively. In this example, the cosine and sine of 22.5 degrees.The same can be said for the other symbol points in the constellation.

However, when the symbol is corrupted by noise, the received symbol can move anywhere in this constellation depending on the noise power in the signal. The additive noise on the signal constellation symbols is precisely the reason why bit decisions must be made.

To facilitate making the bit decisions, the 8-PSK constellation is broken into 8 different sectors. These sectors are the pie looking lines shown in figure 4.4. The sector regions are bounded by the lines I=0, Q=0, I=Q, and I=-Q.

When a symbol corrupted by noise is received, it is broken down into it's I and Q components and then laid down on the signal constellation. The symbol is then tested to see exactly where it lies on the constellation. The received symbol can be tested by using it's I and Q components in the following manner. The symbol components are tested against the following conditions:

a) Is I greater that 0V (I>0) ?

b) Is Q greater than 0V (Q>0) ?

c) Is I greater than Q (I>Q) ?

d) Is I greater than -Q (I>-Q) ?

Using these tests, the received symbol can be placed into one of the eight sectors on the constellation.

The noise free, naturally mapped symbol that belongs in the sector the received symbol was mapped into is deemed the correct symbol. That is, had the received symbol not been corrupted by noise, then it would have been received as the noise free symbol that occupies the sector. This noise free symbol becomes the hard decision symbol.

As an example, assume that a symbol corrupted by noise is received and it is broken down into it's I and Q components. It is found to have an I component value of 0.23V and a Q component value of -0.75 volts. Applying the testing conditions above (i.e. is I>0, Q>0, I>Q, I>-Q ?), we find that the point lies in sector 7 as shown in figure 4.5.

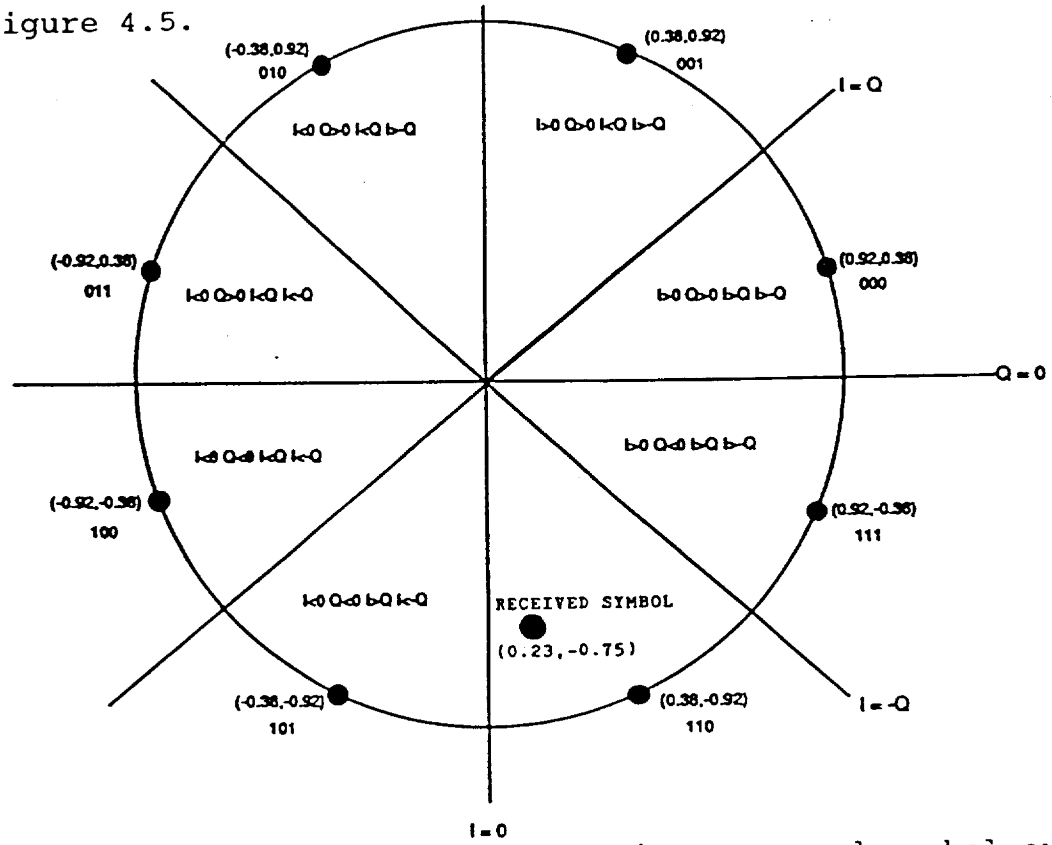


Figure 4.5: 8-PSK constellation showing received symbol example.

Since the received symbol is located in sector 7, it is decided that the original symbol sent must be the 110 symbol. Thus, the I channel data is bit decided to 0.38V and the Q channel data is bit decided to a value of -0.92V.

Hard bit decision circuitry can be built based upon the preceding discussion. In fact, our original 8-PSK modifications used a technique similar to this. The process was to take the I and Q data streams from the A4 card and using the testing process above, map the symbols to one of eight sectors. The mapping indicated what noise free symbol the received symbol originally was. Comparators were used to test for the conditions I>0, Q>0, I>Q, and I>-Q. The outputs of these comparators drove logic gates that made the sector determination. The result was then converted to the appropriate analog voltage levels which represented the estimates of the actual data on the I and Q channels.

The problem with this first generation scheme was twofold. First, the circuit realizations were rather large. This was in itself not a major problem. However, the second point was of some concern. In this method of hard bit decision, the values of the received I and Q data are dependant variables. This means that the I and Q data streams must come onto the same board and then interact with each other. This method also meant using several logic gates to determine sector locations. Due to the interactions of I and Q and the large number of TTL and analog devices on the board, cross talk and instability became a real issue. Using this method, I and Q could cross through zero volt levels at the same time. When this occurred, the comparators used for sector testing osсillated and injected large amounts of noise into the system. This method of hard bit decision was retired for one in which I and Q could be independently evaluated. This first generation board schematic is shown in figure 4.6 below.

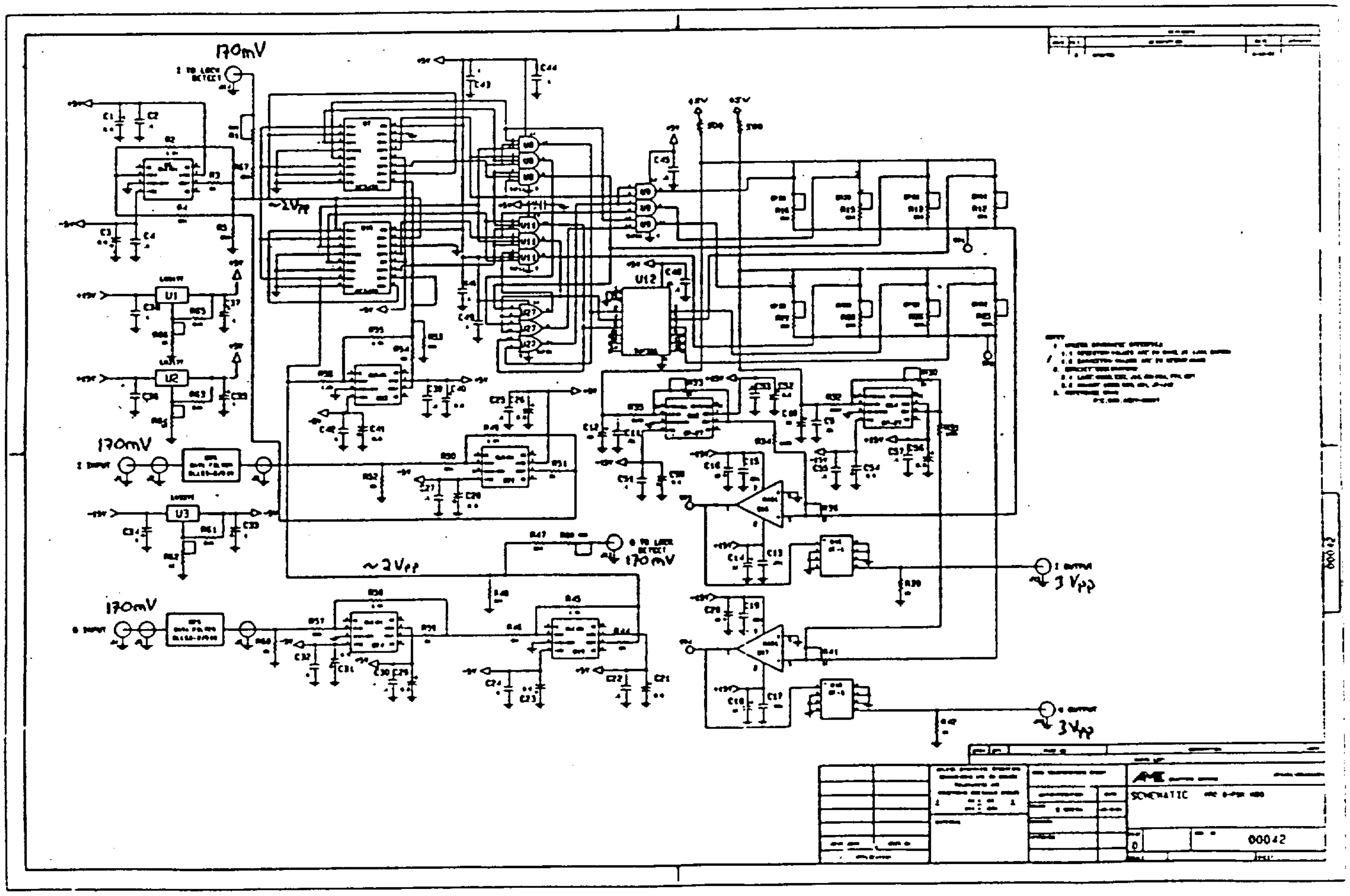


Figure 4.6: Schematic of first generation Mod board.

In the second method, a need still exists to determine the sector location of the received symbol. However, it can be accomplished in such a way that I and Q are independent variables rather than dependent ones. The bit slicing method is far more improved because it decreases the number of components needed to determine bit levels and leaves the I channel and Q channel isolated.

This method is illustrated in Figure 4.7.

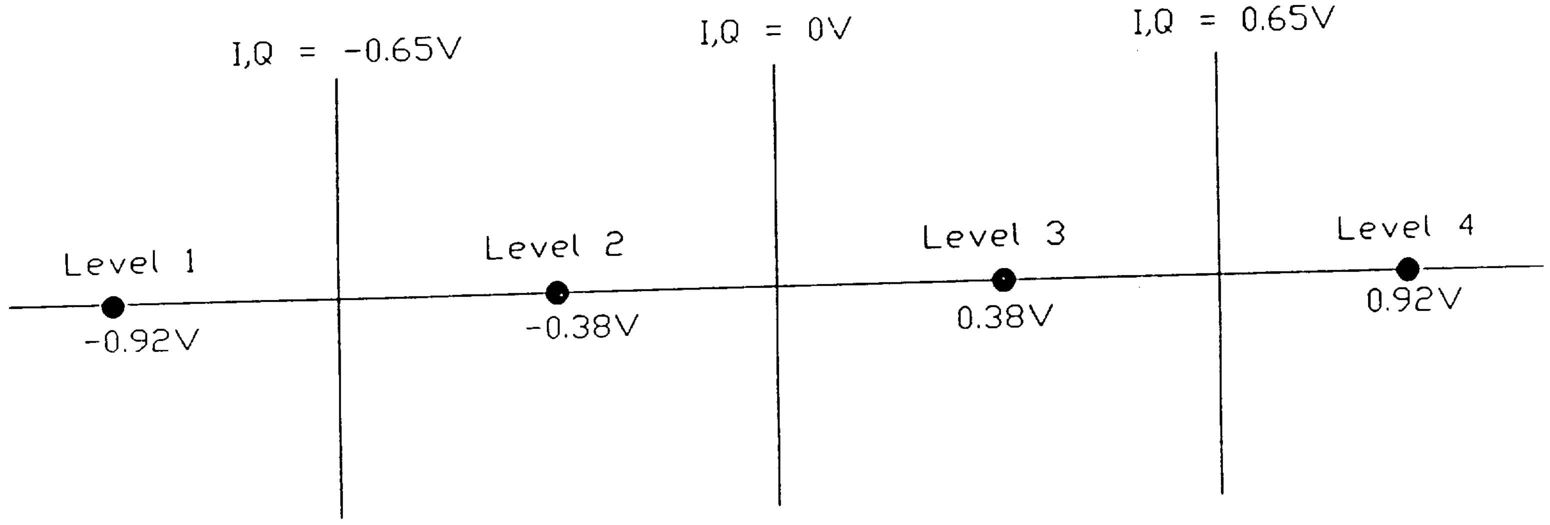


Figure 4.7: Quantizing method for hard bit decisions.

Here, the I channel and Q channel data is individually compared to the threshold levels. These threshold levels are at 0V, +0.65V, and -0.65V. They are set up in such a way that the threshold value is midway between two symbol points (i.e. the mid point between the 0.38V level and 0.92V level is 0.65V). When this is actually implanted, the values will change. The reasons are discussed later.

The seperate data channels are introduced to a comparator that compares the data level in the channel to the threshold levels. The results of these comparisons determine the level that should be selected. For example, if the channel data is tested and found that it is greater than -0.65V and greater than 0V, but less than 0.65V, then the assumed result is that the original data level was 0.38V (level 3). If, however, the channel data level was found less than -0.65V, then the result would be a level 1 of -0.92V as the estimate for the original channel data.

Another way to view this process is in terms of quantizing. Essentially, the received channel data is being quantized into 1 of four levels.

4.2 Board Design

Figure 4.8 below shows a block diagram of the required modifications.

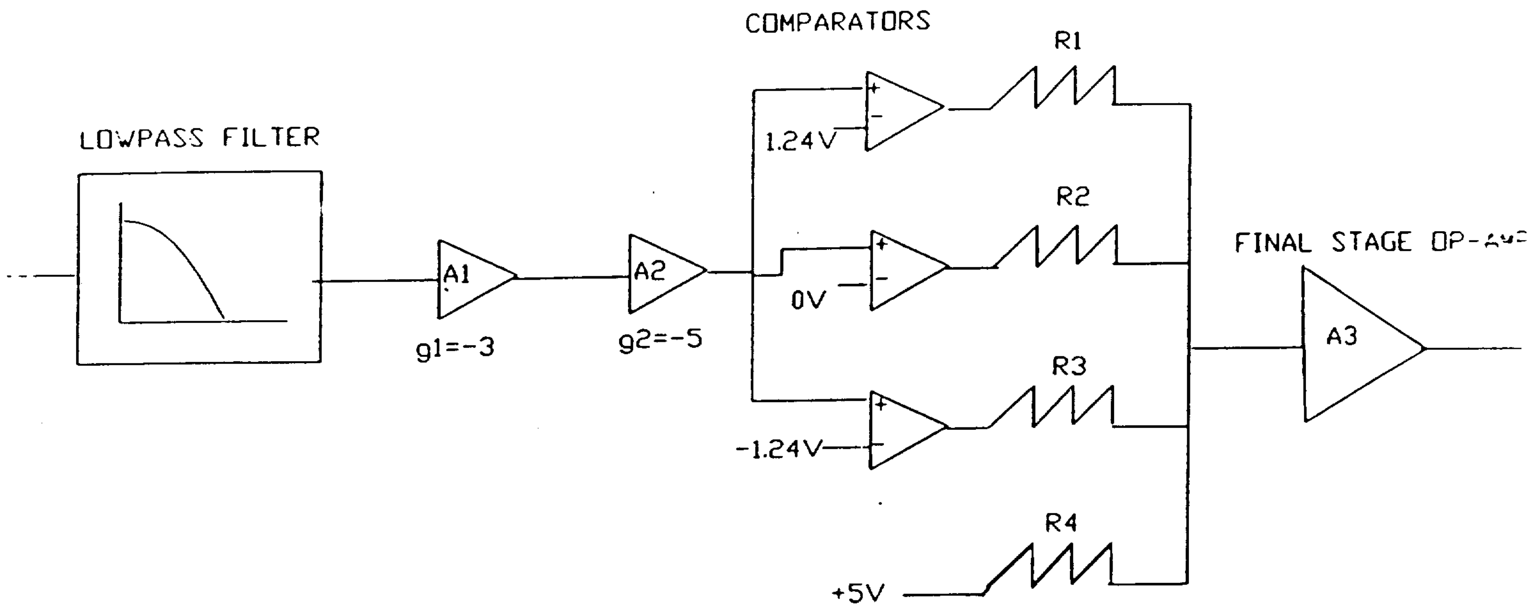


Figure 4.8: Block diagram of the 8-PSK Mod board.

Recall now that the block diagram represents only one channel. The remainder of this discussion will focus on the board development. Bare in mind that the final modifications will require two of these boards; one for the I channel and one for the Q channel. During the ensuing discussion, where there are differences between the two boards, they will be noted. Otherwise, it is fair to assume that the functionality of the two boards is identical. The board development discussion will begin with the input to the data filters and end with the output circuitry.

The quadrature detected baseband data comes from the A4 card and enters the modification (MOD) board at the data filter. These filters reside on the MOD board to make efficient use of space. The data filter is a lowpass, Bessel filter with 3dB bandwidth of 2MHz and has a five pole roll-off. These data filters were custom made by K&L Microwave.

From the filter, the data enters the first stage amplifiers. Figure 4.9 below shows a picture of this waveform.

Figure 4.9: Picture of eye pattern at input to first stage amp. Notice the eye pattern in the waveform indicating that the HRD is indeed locked. The level of the data here is approximately 230mVpp. The first stage op amp delivers a gain of -3 to the signal and the second stage op and grants a gain of -5. We

incorporate a cascaded op amp structure for two reasons. First, cascaded amps are more stable than a single amp that delivers a one shot gain of +15. Secondly, for our applications, we added op amps that had a large bandwidth gain product and fast-slow rate. The amps we chose are op amps manufactured by Comlinear, Inc. They have a large gain bandwidth product and fast-slow rate, 4000uV/S. However, they are only stable in an inverting op amp mode. Hence, we had to cascade two of them to obtain a positive gain and, thus, force our need to split the gains. The second stage op amp has circuitry to eliminate any DC offset. Elimination of the DC is important because the next stage is the comparators and any DC component will affect the comparator process. The output level of the second stage OP amp is 3.5Vpp. The comparators perform better with larger signal levels.

The next stage in the chain is the comparator stage. Three comparators are used to perform the bit slicing mentioned earlier. Recall that the thresholds were calculated for a signal whose peak amplitude was 0.92V. In this case, we are driving the comparators with a 3.5Vpp signal. So, our threshold (quantizing) levels must reflect this change.

Also, the I channel and Q channel are different at this point. Refer again to Figure 2.1. In this block diagram view, note the bubble on the hard limiter U11 on the A4 card. This hard limiter is an inverting hard limiter. Thus, the data through our MOD board must reflect this inversion. The channel that needs the inversion is the Q channel board.

We must jump ahead for one moment. The final stage amplifier on the MOD board is an inverting type op-amp. So, without doing anything else to the data, we can invert the Q channel data using the final stage op amp. However, this means that the I channel data will also be inverted. To correct the situation, we have inverted the I channel data using the comparators. That is to say, we test the Q channel data at the comparators for values greater than the thresholds, and test the I channel data at the comparators for values <u>less than</u> the thresholds. This gives us our inversion in the I data channel to offset the final state amplifiers inversion.

Figure 4.10 illustrates the quantization diagram for both the I and Q channel data, as well as, illustrating how the comparators are arrayed in each of the channels.

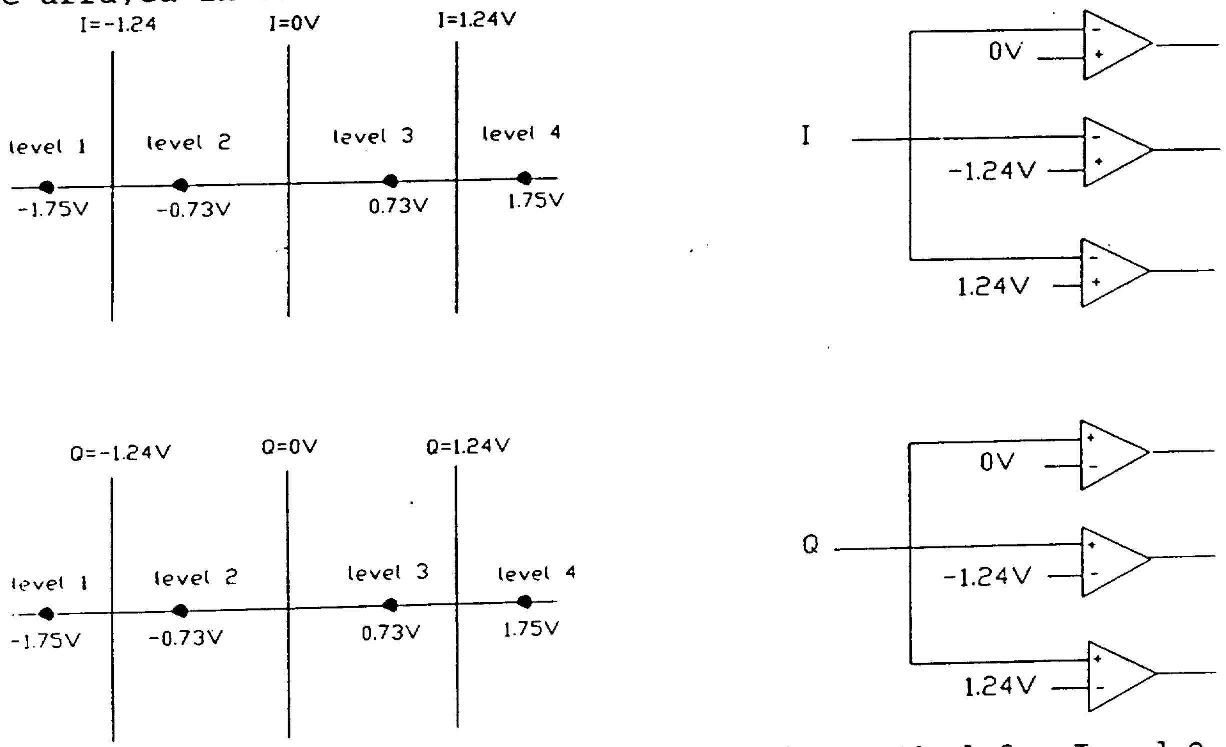


Figure 4.10: Quantizing levels and quantizing method for I and Q.

The threshold values were calculated in the following manner: The signal levels entering the comparators are 3.5Vpp or 1.75Vpeak. In chapter 4.1, the signal peak was assumed to be 0.92Vpeak (derived from the cosine of the angle of the phase offset of the constellation). The ratio of the original constellation level values were:

$$\cos(22.5^\circ)/\sin(22.5^\circ) = 0.924/0.383 = 2.412.$$

This ratio must be upheld when scaling the constellation values to 1.75Vpeak. Thus, the level 1 and level 4 quantizing points are -1.75V and +1.75V respectively. To find the level 2 and level 3 quantizing points, we apply the previously calculated ratio. So,

$$\text{level } 2, \text{ level } 3 = \pm 1.75\text{V}/2.412 = \pm 0.726.$$

Quantized Level 2 is at -0.73V and level 3 is at +0.73V. these points are indicated in Figure 4.10.

The threshold levels are found by obtaining the midpoints between two consecutive quantizing levels. The midpoint between -0.73V and +0.73V is 0V, meaning, one threshold is at 0V. The midpoint between ±0.73V and ±1.75V is ±1.24V. So, the other two thresholds are set at ±1.24V.

Thus, the comparators quantize the channel data to one of the four levels. The Q channel tests for points greater than the thresholds and the I channel tests for points less than the thresholds.

The comparators drive variable resistors that make up a summing node to the final stage op amp. The levels out of the comparators are TTL. The variable resistors can then be used to set the four levels out of the final stage amp to any value. Resistor R9C sets the level 3 value in the I channel and level 1 value in the Q channel. Resistor R9B sets the level 2 value in the I channel and the level 2 value in the Q channel. Resistor R9A sets the level 1 value in the I channel and the level 3 value in the Q channel. The fourth resistor (R9D) is a DC offset pot and is used to set the level 4 quantizing point in both channels.

The final stage op amp has the sole purpose of giving the output enough power to drive the multipliers on the A4 card. The multipliers require +9dBm of drive. The op amps used for the final stage are high power op amps that have good gain-bandwidth product and are quite capable of delivering the +9dBm required.

The final problem to solve is the selection of output level values. That is, what do the output levels need to be in order to provide +9dBm to the multipliers? In the QPSK HRD, the multipliers required signals with levels of 2.5Vpp. This level provided the multiplier with the appropriate power. However, this is for QPSK with only two levels.

For 8-PSK, we must match power requirements, but instead of two levels, we have four levels. In order to derive the output levels we must recognize that our four level system must provide the same amount of drive as the two level system.

Figure 4.11 shows a typical 8-PSK channel waveform and illustrates the four levels.

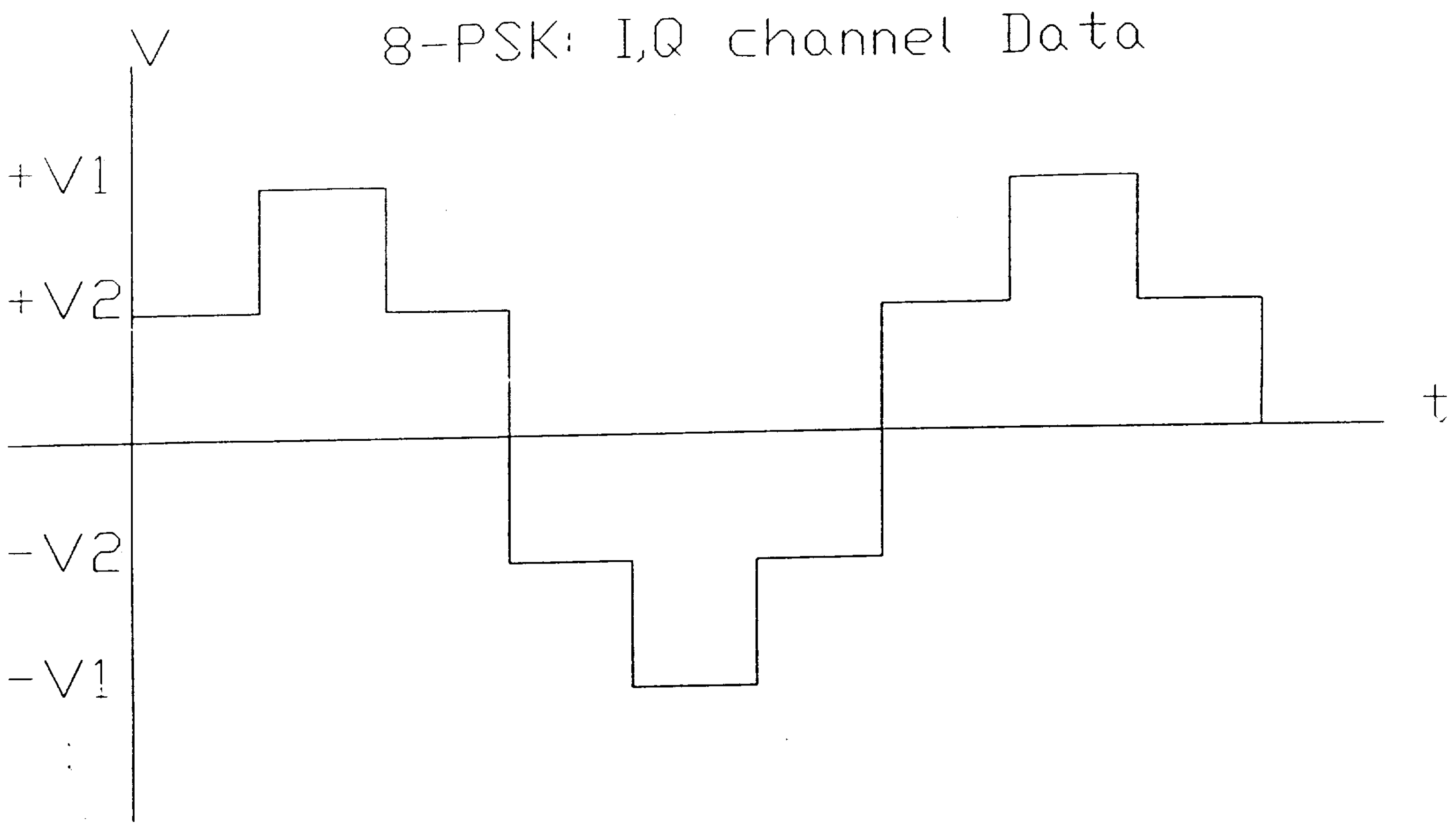


Figure 4.11: Typical 8-PSK channel waveform.

Let the upper level be represented by V_1 and the second level by V_2. Note the other levels can be represented by $-V_1$, and $-V_2$. We can write the power in this waveform (into a 1Ω resistor) as:

$$P_o = V^2_1/2 + V^2_2/2. \qquad (1)$$

The power required by the multiplier is:

$$P_m = (2.5/2)^2 = (1.25)^2. \qquad (2)$$

From our previous relationship, we know that the original level ratio must be maintained (i.e. $r = 0.92/0.38 = 2.412$). So, it follows that:

$$V_1/V_2 = r = 2.412$$

or,

$$V_1 = 2.412\ V_2. \qquad (3)$$

Now, equating equations (1) and (2) and subsituting equation (3) into the result gives

$$1/2[2.412\ V_2]^2 + 1/2\ V^2_2 = (1.25)^2.$$

Solving yields, $V_2 = 0.68V$. Using equation (3) to solve for V_1 gives $V_1 = 1.64V$.

So, the output levels would be as shown in Fiqure 4.12.

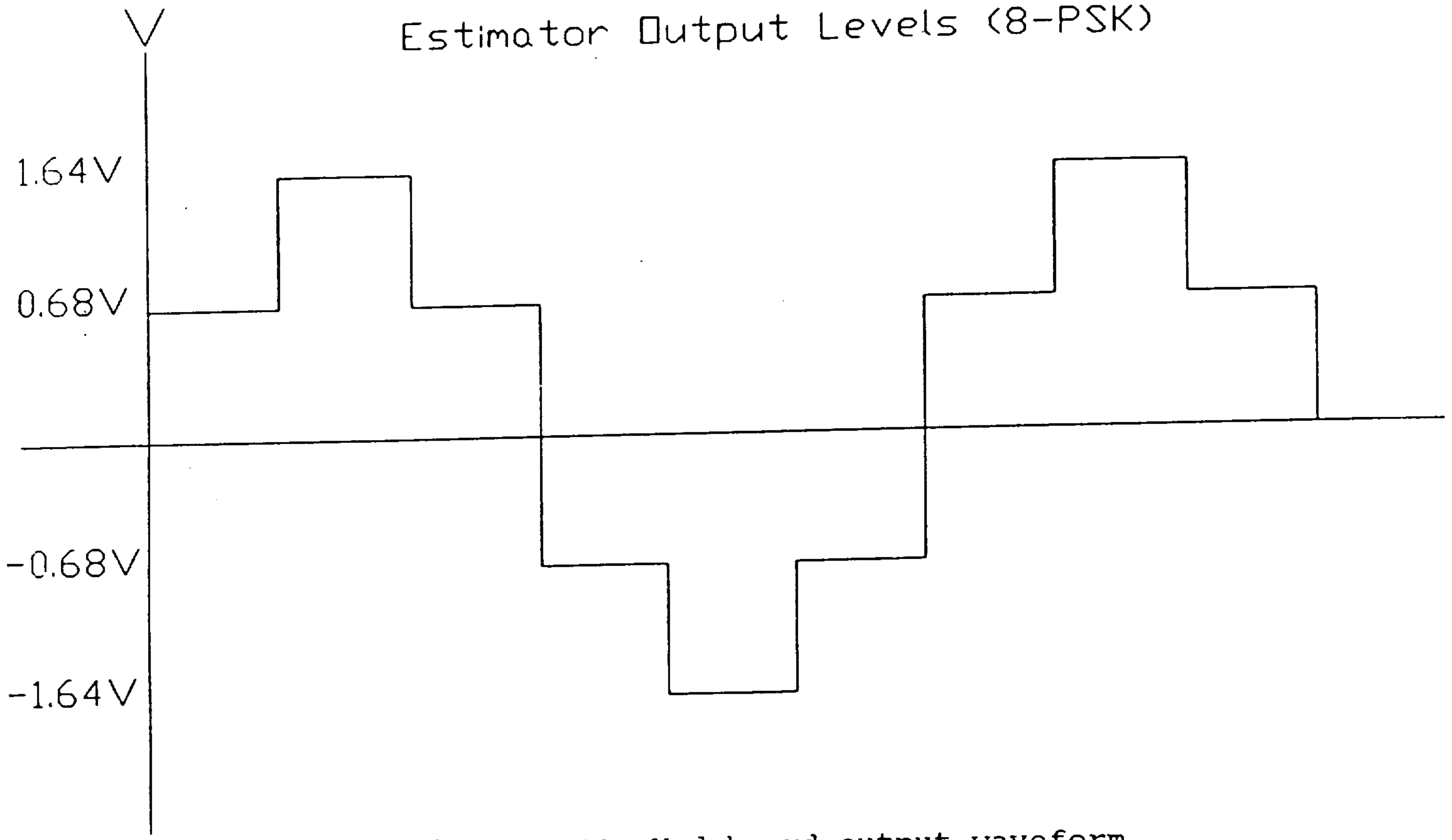


Figure 4.12: Mod board output waveform.

These levels can be established at the output by adjusting the R9 resistors, as described earlier.

Figure 4.13 shows an actual picture of the output of the final stage amplifier.

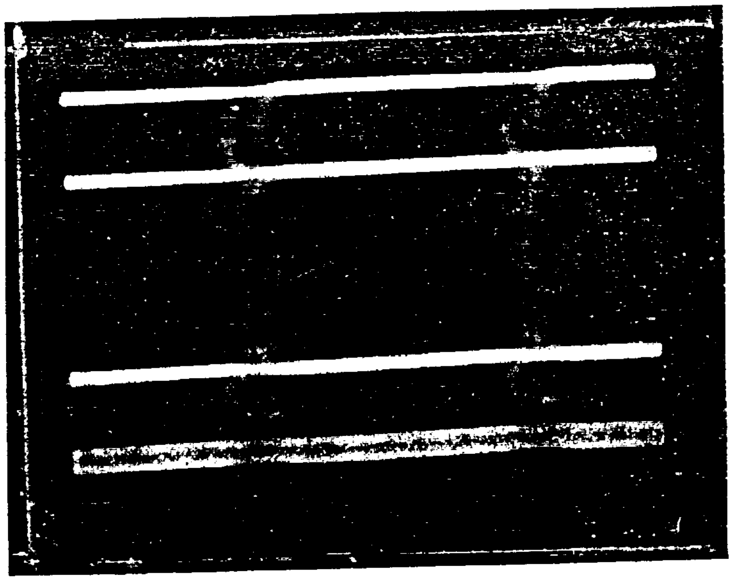


Figure 4.13: Eye pattern of the output waveform.

For this photo, the HRD is locked. Notice the clean eye pattern compared to the eye pattern of Figure 4.9.

Figure 4.14 on page 47 shows the schematic of the HRD 8-PSK MOD board.There are a few items worth noting. The circuit uses +5V and -5V, as well as, +15V and -15V. The voltages +15V and -15V are obtained from the HRD power supplies and then regulated to give +5V and -5V. The MOD boards were designed to replace the data filter cards (A2 and A3 cards). Figure 4.15 on page 48 shows a top down view of the inside of the HRD.

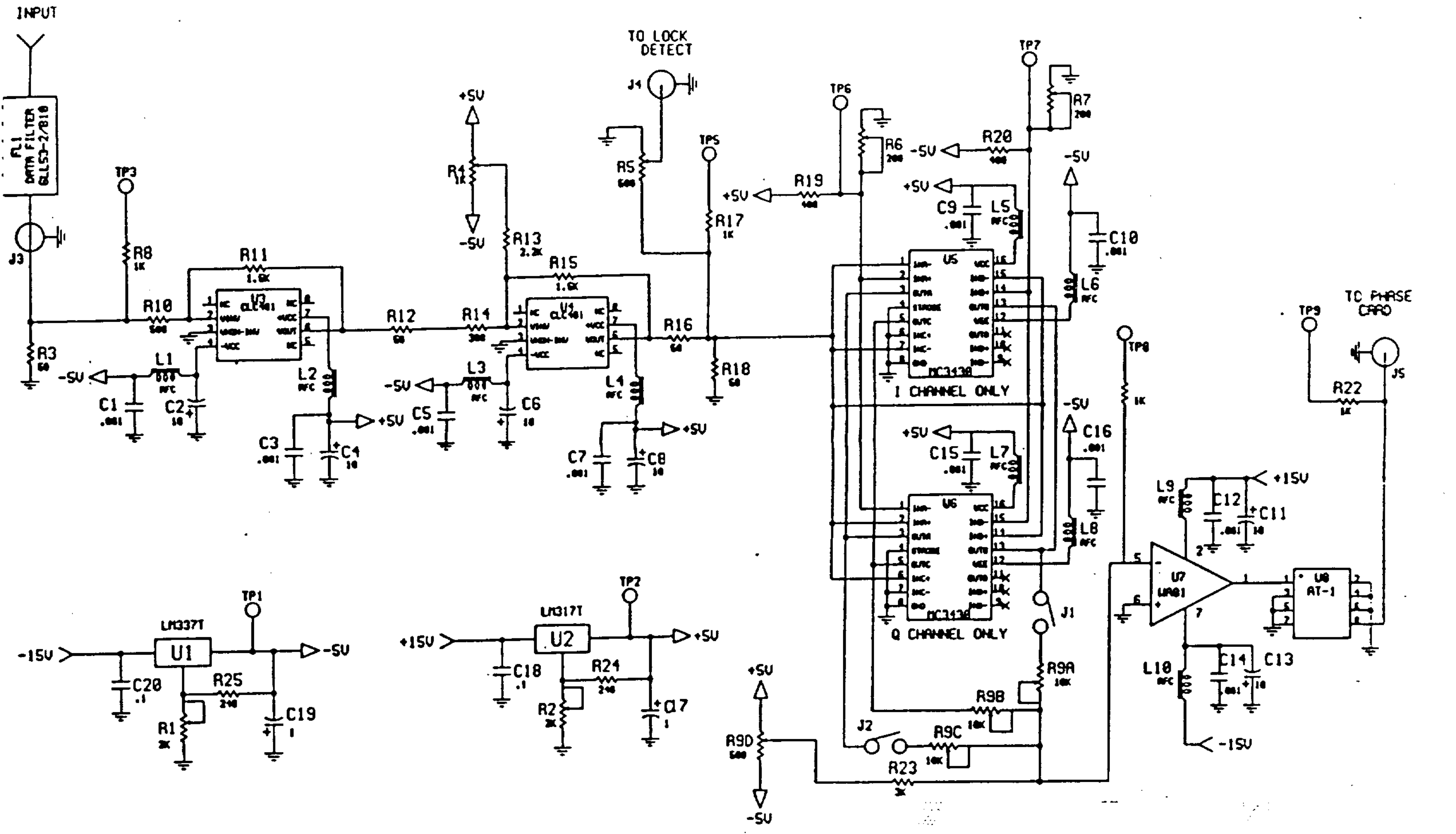


Figure 4.14: HRD 8-PSK mod board schematic.

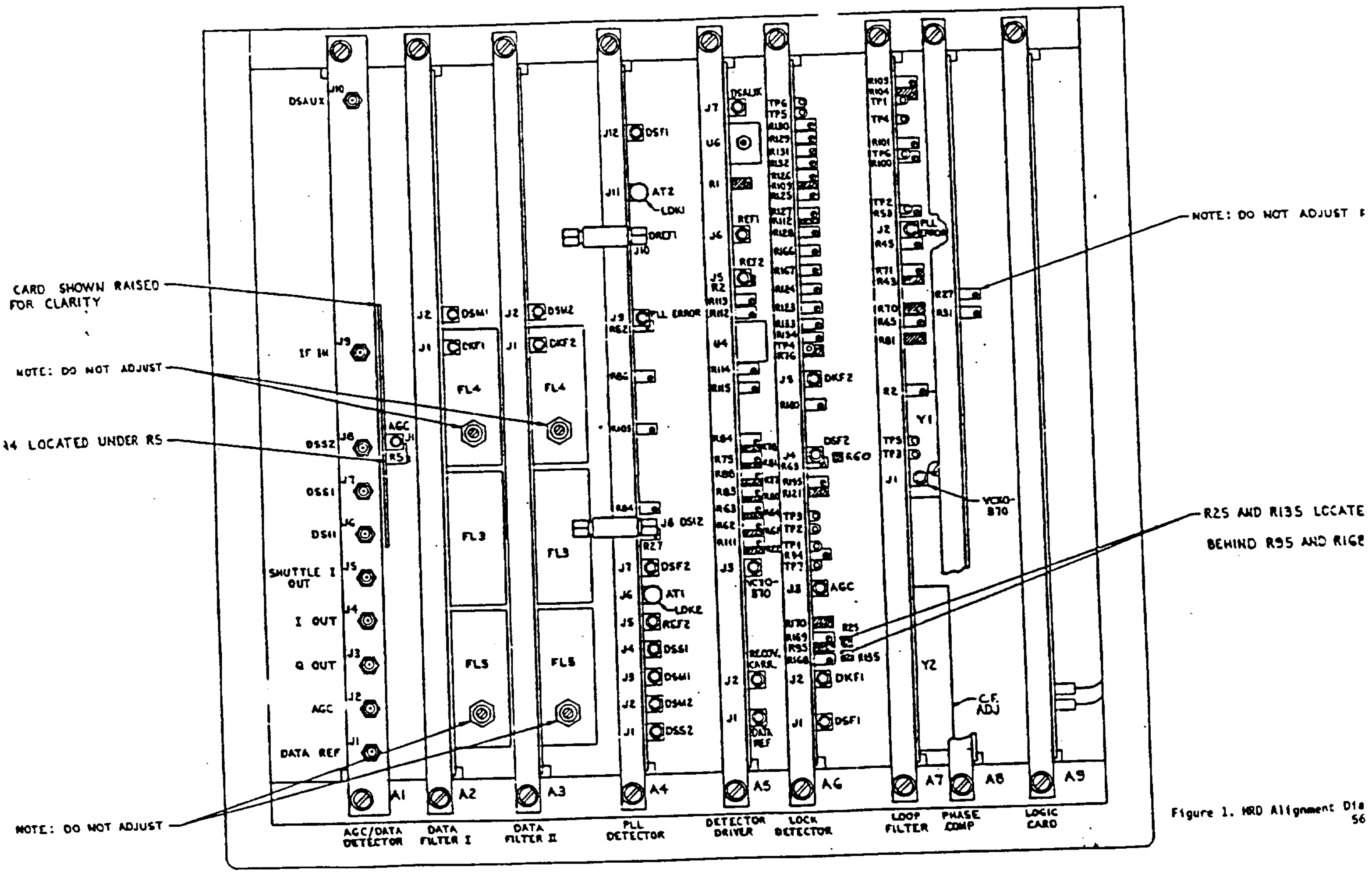


Figure 4.15: HRD card placement (Top View).

Note the cards vertical placements. The I channel and Q channel MOD boards are to replace the cards marked A2 and A3 respectively.

Figure 4.16 on page 50 shows the component placement and dimensionality of the new board design.All the potentiometers and test points have been placed near the top of the board to facilitate adjusting the boards for the required levels. The appendix to this paper includes the procedures for aligning the board as well as necessary resistor functions.

Note on the mod board that there are positions for two comaprators. However, only one is used depending on the board type. The board was designed to be able to operate in either channel. To make the board a dedicated I channel board, the comparator labeled I channel only is installed and the Q channel only comparator is left off. Likewise for a dedicated Q channel board.

One added feature to this board is the ability to use the board in a QPSK mode. Refer to Figure 4.14. If the jumpers indicated by J9 and J2 are removed, this effectively eliminates two levels from the output waveform. Thus, it allows the board to operate in a QPSK mode. Essentially, this board can remain in place for either operation. All that is required is to remove the header block labeled J1 J2, shown in Figure 4.16, to allow QPSK operation.

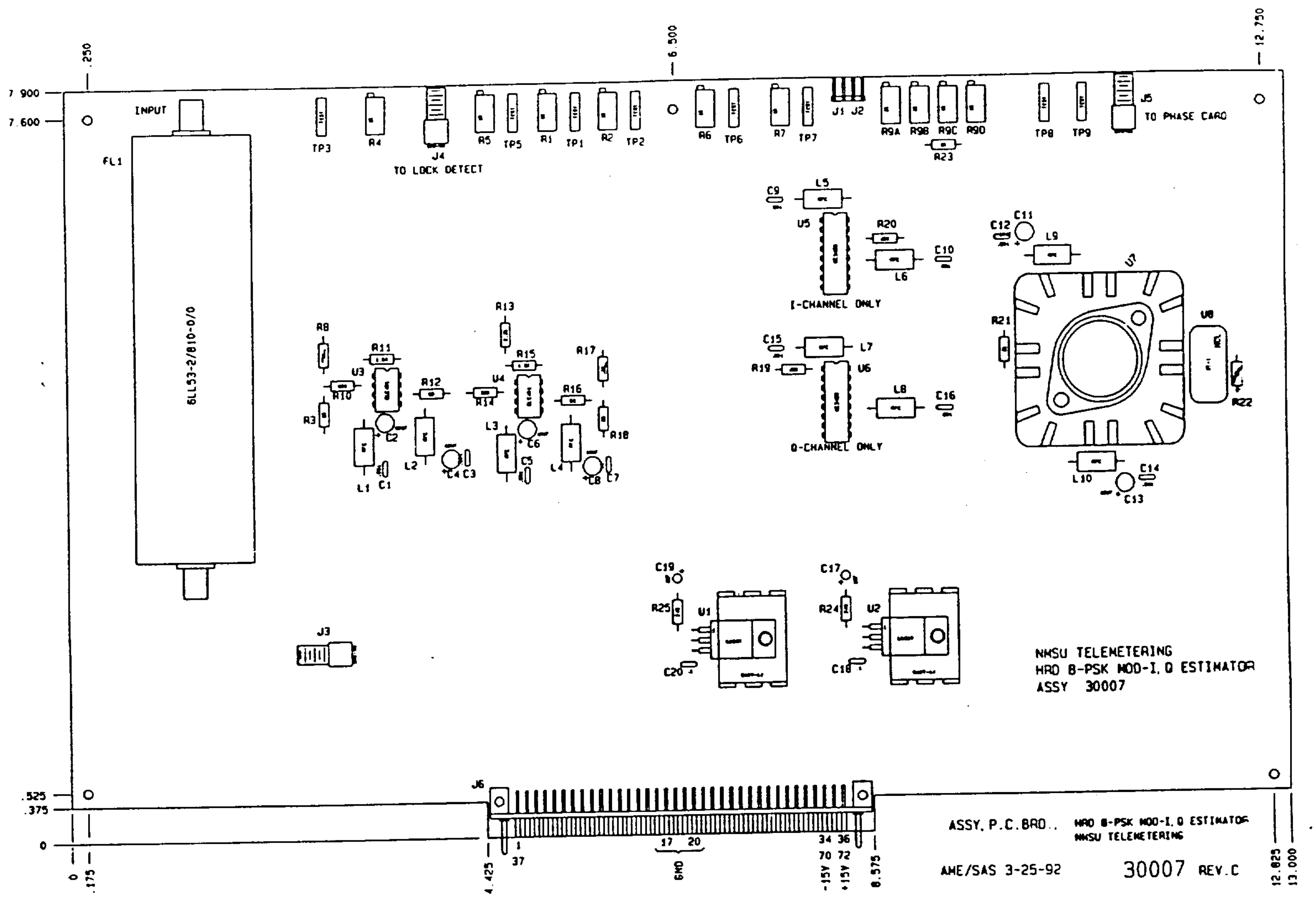


Figure 4.16: Assembly drawing of the 8-PSK Mod board.

5.0 Test Results

The 8-PSK MOD boards were assembled and incorporated into the HRD. The I channel board was inserted in the A2 card slot and the Q channel board was inserted in the A3 card slot. The 8-PSK MOD boards were connected to the HRD. The HRD was then incorporated into the test set-up as shown in Figure 5.0.

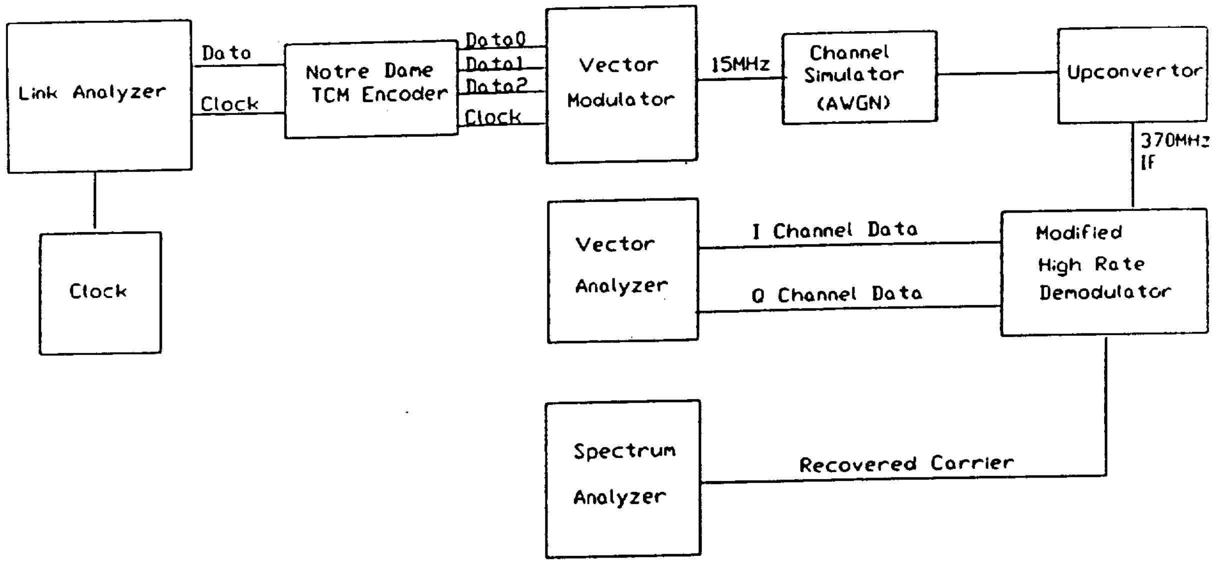


Figure 5.0: Test setup for testing HRD modifications.

The setup was configured to run a 1Mbps data rate. The link analyzer generated a random serial data stream which was fed to the TCM encoder. The Notre Dame TCM encoder generated the 8-PSK symbol set from the serial data stream. Next, the 3 bit 8-PSK symbols were given to the Vector Modulator. The Vector Modulator used the three bits to modulate a 15MHz carrier. The carrier had noise added to it in the channel simulator and the result was upconverted to 370MHz and sent into the HRD. The I and Q channel data was from the HRD to the Vector Analyzer where the 8-PSK constellation was viewed.

Figure 5.1 is a picture of the demodulated HRD output (I and Q signal constellation) before the HRD was locked.

Figure 5.1: HRD output in unlocked state.

Note how the circle is a single trace with 8 signal points barely visible. Figure 5.2 is a picture of the signal constellation with the HRD locked.

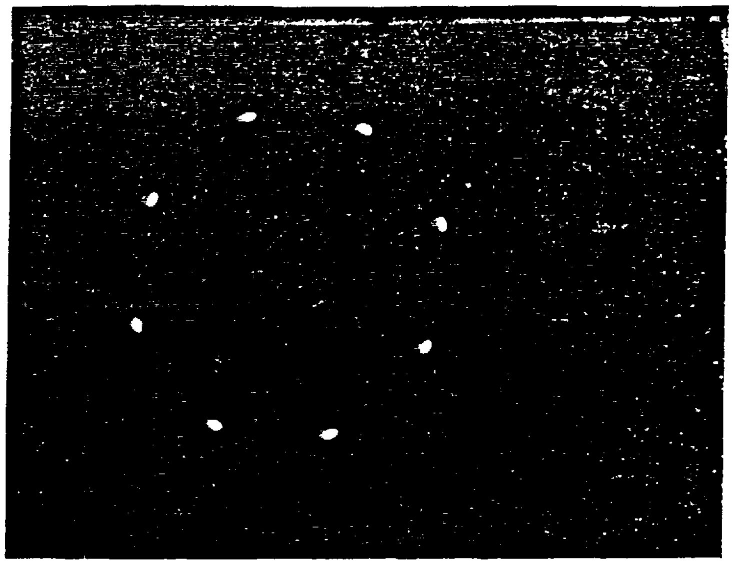


Figure 5.2: Locked HRD with no noise.

The constellation now has 8 distinct symbol points. This picture is for a signal with no noise added. Figure 5.3 shows a picture of the signal constellation when noise is added to the signal.

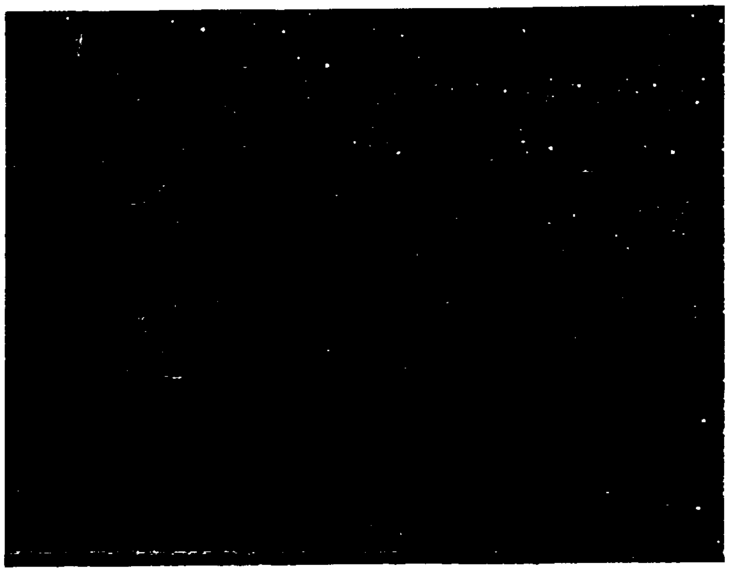


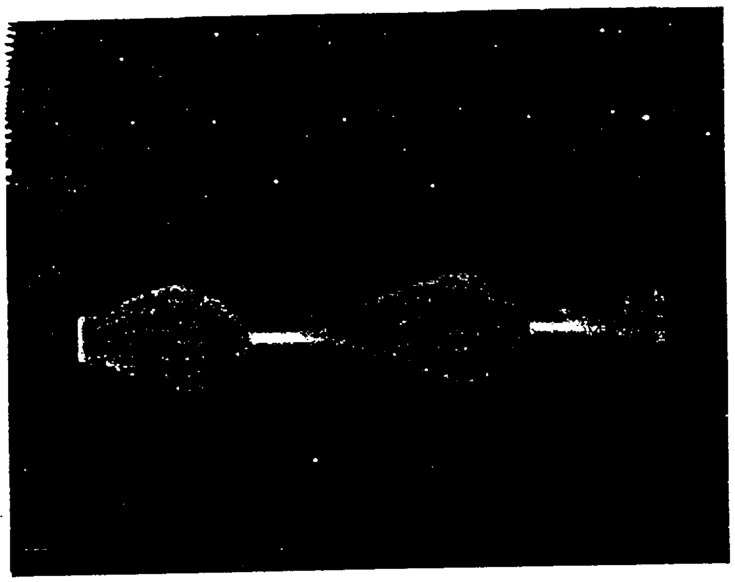
Figure 5.3: Locked HRD output with noise added to signal.

Notice the mountains of probability surrounding each symbol.

Finally, figure 5.4 shows the output of the A4 card. This picture is the error signal generated by the PLL. The error signal goes to zero as it should. The non-zero zones of the error signal are due to the inter symbol interference (ISI) at the bit transitions.

As of this writing, no bit error rate test had been conducted using the modified HRD. These tests are to be conducted sometime in the future.

Still, the fact that the HRD demodulated 8-PSK data is encouraging. Of course, the HRD had to be tricked into locking, but none-the-less, the modifications performed exactly as expected. In fact, there is some data that might support the fact that the modifications improved the HRD's performance in the QPSK mode.



6.0 Conclusions

This paper presented the analysis and design of modifications needed to convert a QPSK demodulator into a demodulator capable of handling an 8-PSK modulated carrier.

Our results showed that we could successfully modify the existing HRD to work in the 8-PSK mode. Albeit, our success was dependent on our being able to force the HRD into the lock state. Still, we recovered good data from the HRD and accomplished what we intended to do.

The steps to undertake in the future are basic from our point of view. First, the lock detector needs to be reworked, if the modified HRD is to ever be used in real time applications. Secondly, the data rates that the 8-PSK MOD boards were designed to handle are to slow for real time applications. The modifications will have to be reworked to allow higher data rates. This would involve replacing the current filters with filters of larger bandwidth. The comparators might also need upgrading if the data rates are pushed above 10Mbps. Finally, bit error rates must be tested with the modified HRD before it is incorporated into the final system. If it performs worse than the original HRD did in the QPSK mode, then there is no need for further investigation. Bit error rates have not been currently investigated due to a lack of hardware.

APPENDIX

The information contained in the Appendix is designed to aid in the installation and alignment of the HRD 8-PSK modification boards.

The boards must first be built with the proper components. If an I channel board is to be built, be sure the comparator, U5, is installed in the board. Leave the comparator, U6, off the board. If Q channel board is to be built, be sure to install the U6 comparator and leave the U5 comparator off the board. Note: a channel 1 board is an I channel board and a channel 2 board is a Q channel board.

Insert the HRD 8-PSK MOD board into the proper slot. If the board is an I channel board, insert it into the A2 card slot.

Place a 50 ohm SMA terminator on the J5 output connector. On the input inject a 500KHz sine wave with magnitude 230mVpp.

Turn the power switch on the HRD to ON. Place a fluke meter positive probe into TP9. Adjust R1 until the voltage at TP1 reads -5.2VDC on the Fluke. Next, place the positive probe of the Fluke meter into TP2. Adjust R2 until the Fluke reads +5.2VDC.

The next step is to place the positive probe of the Fluke meter into TP6. Adjust R6 until the Fluke reads 1.24VDC. Place Fluke probe into TP7. Adjust R7 until the Fluke reads -1.24VDC.

The following steps require the use of an oscilloscope.

Place the scope probe on TP5. Adjust R4 until there is minimum

DC offset.

Now, place the scope probe on TP5. You should measure a signal with 3.5Vpp. If you do not, adjust the input signal level until 3.5Vpp is reached at TP5. With this completed, place the scope probe on TP9. Following one of the following alignment strategies, depending on what board you are working on:
(See figure A1)

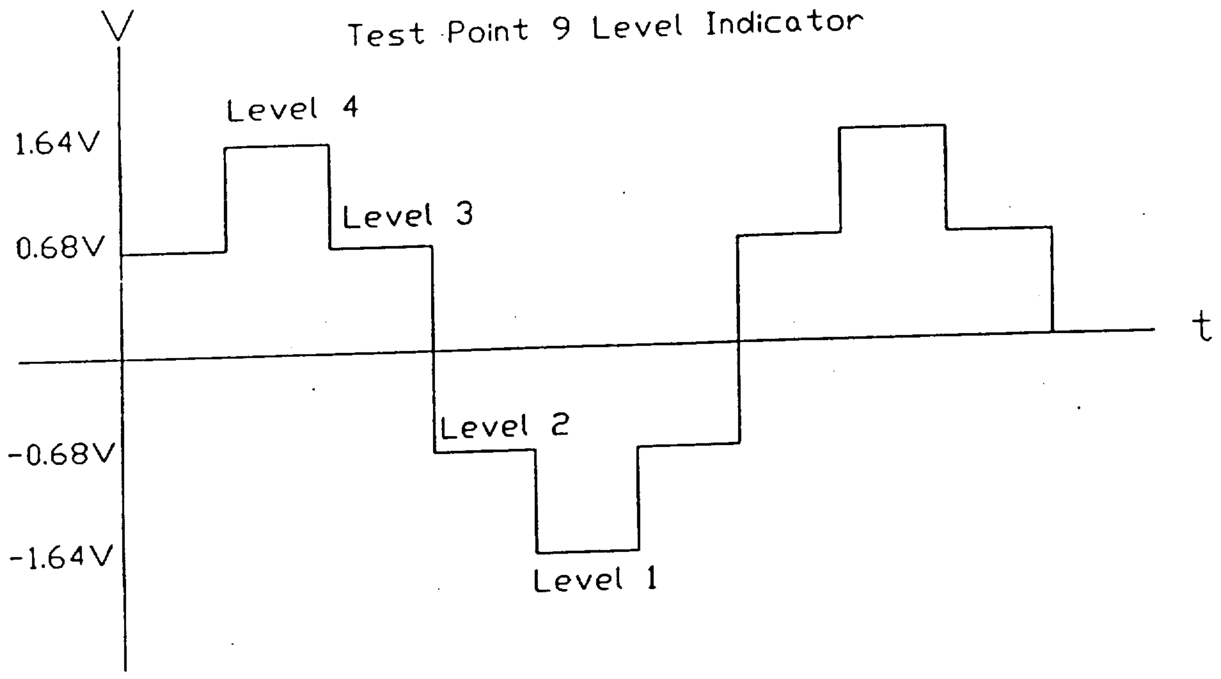


Figure A1: Level indication in output signal.

1.) FOR THE I CHANNEL BOARD:

a) adjust R9D until level 4 is at 1.62V

b) adjust R9C until level 3 is at 0.68V

c) adjust R9B until level 2 is at -0.68V

d) adjust r(a until level 1 is at -1.62V

2.) FOR THE Q CHANNEL BOARD:

a) adjust R9D until level 4 is at 1.62V

b) adjust r9A until level 3 is at 0.62V

c) adjust R9B until level 2 is at -0.62V

d) adjust R9C until level 1 is at -1.62V

NOW, turn the power off to the HRD. Connect the cables inside the box to the MOD cards in the following fashion:

1) connect A4J3 to channel 1 J4

2) connect A4J12 to channel 1 J5

3) connect A4J2 to channel 2 J4

4) connect A4J7 to channel 2 J5

Note that A4J12 and A4J7 are the two black SMA cables soldered directly onto the A4 card. The cable nearest the A4J12 connector is the A4J12 line.

Finally, turn the power back on to the HRD. With the scope probe still connected to TP9, adjust the R9 resistors again to achieve the best output level alignments following the steps given above for the I channel card and for the Q channel card.