BARBA, JOSEPH

A MODIFIED ADAPTIVE DELTA MODULATOR FOR VIDEO DATA COMPRESSION

City University of New York PH.D. 1981

PLEASE NOTE:

In all cases this material has been filmed in the best possible way from the available copy. Problems encountered with this document have been identified here with a check mark __✓__.

1. Glossy photographs or pages _____
2. Colored illustrations, paper or print _____
3. Photographs with dark background __✓__
4. Illustrations are poor copy _____
5. Pages with black marks, not original copy _____
6. Print shows through as there is text on both sides of page _____
7. Indistinct, broken or small print on several pages _____
8. Print exceeds margin requirements _____
9. Tightly bound copy with print lost in spine _____
10. Computer printout pages with indistinct print _____
11. Page(s) _____ lacking when material received, and not available from school or author.
12. Page(s) _____ seem to be missing in numbering only as text follows.
13. Two pages numbered _____. Text follows.
14. Curling and wrinkled pages _____
15. Other _____

A MODIFIED ADAPTIVE DELTA MODULATOR

FOR VIDEO DATA COMPRESSION

by

Joseph Barba

A dissertation submitted to the Graduate Faculty in Engineering in partial fulfillment of the requirements for the degree of Doctor of Philosophy, The City University of New York.

1981

This manuscript has been read and accepted for the Graduate Faculty in Engineering in satisfaction of the dissertation requirement for the degree of Doctor of Philosophy

9/14/81
date

Chairman of Examining Committe

9/14/81
date

Executive Officer

Prof. D.L.Schilling (mentor)

Prof. N.Scheinberg

Prof. H.Taub

Prof. T.Apelewicz

Prof. T.Saadawi

Dr. J.Garodnick

Supervisory Committee

The City University of New York

ABSTRACT

A MODIFIED ADAPTIVE DELTA MODULATOR
FOR VIDEO DATA COMPRESSION

by

Joseph Barba

Adviser: Professor D.L.Schilling

This paper presents the results of a study in data compression of adaptive delta modulated video signals. The Song mode ADM is first investigated and shown not to produce enough redundancy to warrant entropy encoding. We then present a modified adaptive delta modulator algorithm that produces sufficient redundancy to yield a 40-50% data compression by using a simple 4-bit block code. Other techniques such as field interpolation and direct substitution are shown to further increase the possible data compression without noticeable degradation in the two input images used in this investigation. A Frame Freeze unit is presented in appendix A that can be used in

conjunction with the MADM to capture a frame a video for transmission over a computer network.

The effects of channel error in the transmission of packet video over a computer network are considered in the last chapter. A leaky integrator is used to reduce the effects of channel errors in the data bits. We show that the effects of channel errors on the block code header bits can be reduced by field interpolating those packet that can be determined to contain error on header bits.

To my parents

ACKNOWLEDGEMENT

I would like to thank my advisor, Professor Donald L. Schilling for his help, encouragement and valuable guidance during the course of this research. The many technical suggestions and his constructive criticism are well appreciated.

I would like to thank Dr. N. Scheinberg for his help in simulating the ADM hardware and his many technical suggestions.

I would also like to thank all of my Doctoral committe members and Professor David Cheng for their many accomodations extended to me in times of need.

I would also like to express my thanks to Kathi Koh for her help, understanding and morale support.

TABLE OF CONTENTS

TABLES

FIGURES

1.0.0 INTRODUCTION

With the developement of integrated digital circuits and the sophistication of digital systems, communcications techniques have become more digital. Digital communication systems offer certain advantages over their analog counter-part. Such advantages include an ability to operate at lower signal to noise ratios, less channel errors, smaller size and less power requirements, less complex circuits, and lower cost. The one major disadvantage of digital systems is the required channel bandwidth necessary for transmission. An analog video signal requires an 4 MHz. channel bandwidth while the digital encoded video signal might require 64 MHz., an increase in channel bandwidth requirement by a factor of sixteen. The need to reduce the bandwidth requirements of digital imagery is there because of the continuously increasing demand on channel bandwidth imposed by the ever increasing number of users.

Data compression is an attempt to reduce the required amount of binary data that must be transmitted in order to achieve one of two goals: 1) reduce the time required to transmit the data in the same bandwidth, 2)

reduce the channel bandwidth required to transmit the compressed data in the same amount of time. A general approach is to develope source coding algorithms to encode the video signal. One method to accomplish this is image transform coding while another is to use predictive coding algorithms.

1.1.0 PULSE CODE MODULATION (PCM) [11,19,26,30]

All digital video systems require that the analog video signal be converted into digital form. If no further processing is done on the digital samples prior to transmission the resulting form of modulation is called pulse code modulation. A pulse code modulation system is shown in fig. 1.1 The input signal is low pass filtered, sampled and quantized. The low pass filter prevents any aliasing errors which may result from sampling. The quantizer must have a minimum of 64 levels which corresponds to 6 bits per sample. The minimum sampling rate must be twice the maximum frequency (approx. 8 MHz.). The PCM bit rate is in bit per second and is given by:

1.1-1 $B=M*N*K*F$

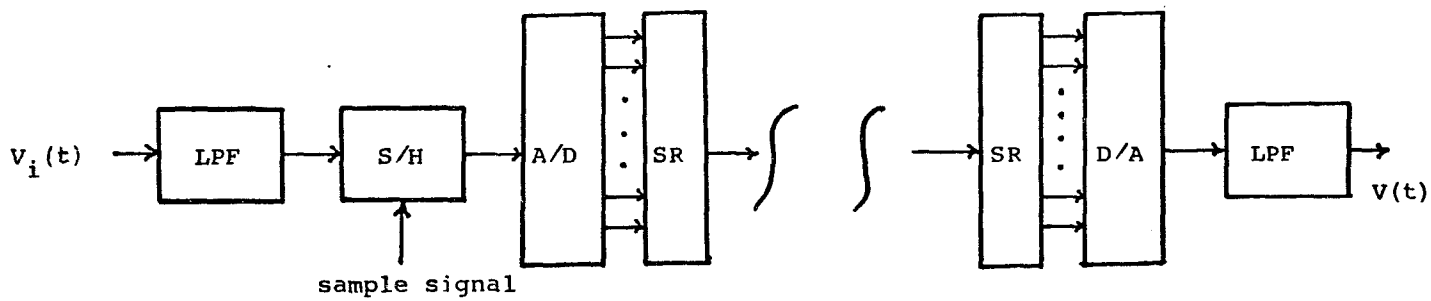


Fig. 1.1 Pulse code modulation system (PCM)

If M = 512 pixels/line

N =525 lines/frame

K = 6 bits/pixel

F = 30 frames/sec

then the PCM bit rate required to transmit a NTSC video signal is at least 48 MBPS.

In the transmission process noise is added to the transmitted signal. Even though the errors are random (any bit can be in error) the significance of the errors varies from bit to bit. An error in the most significant bit (MSB) will be over 60 times greater than an error in the least significant bit (LSB). the signal to noise ratio of a PCM system is given by:

$$1.1\text{-}2 \qquad (S/N) = \frac{2^{2K}}{1+2^{2(K+1)}P_e}$$

where: K = number of bits/pixel.

P_e = probability of error.

Because the human eye does not tolerate this type of error, PCM systems are operated in environments where the error rate is 10^{-5} or less.

A PCM system can transmit any of the $64^{512*512}$ possible television images because it does not make use of any pixel correlation. Human observers find meaningful only those images where there is a high degree of pixel correlation. Data compression can be achieved by taking advantage of the correlation that exists in a picture. There are two techniques generally used to achieve data compression; image transform coding and predictors.

1.2.0 IMAGE TRANSFORM CODING

Transform coding is one approach to accomplish data compression by utilizing some aspect of both statistical and psychovisual coding. The image transformation process is illustrated in Fig. 1.2. Briefly, the transform image coding process involves the following operations:

A) The image to be transformed is sampled and digitized to at least 6 bits per sample.
B) The samples are then arranged in a matrix such that the location of the sampled pixel corresponds to its location in the matrix, called blocks.
C) A block of picture elements is converted through a well defined transformation into a block of transform coefficients.

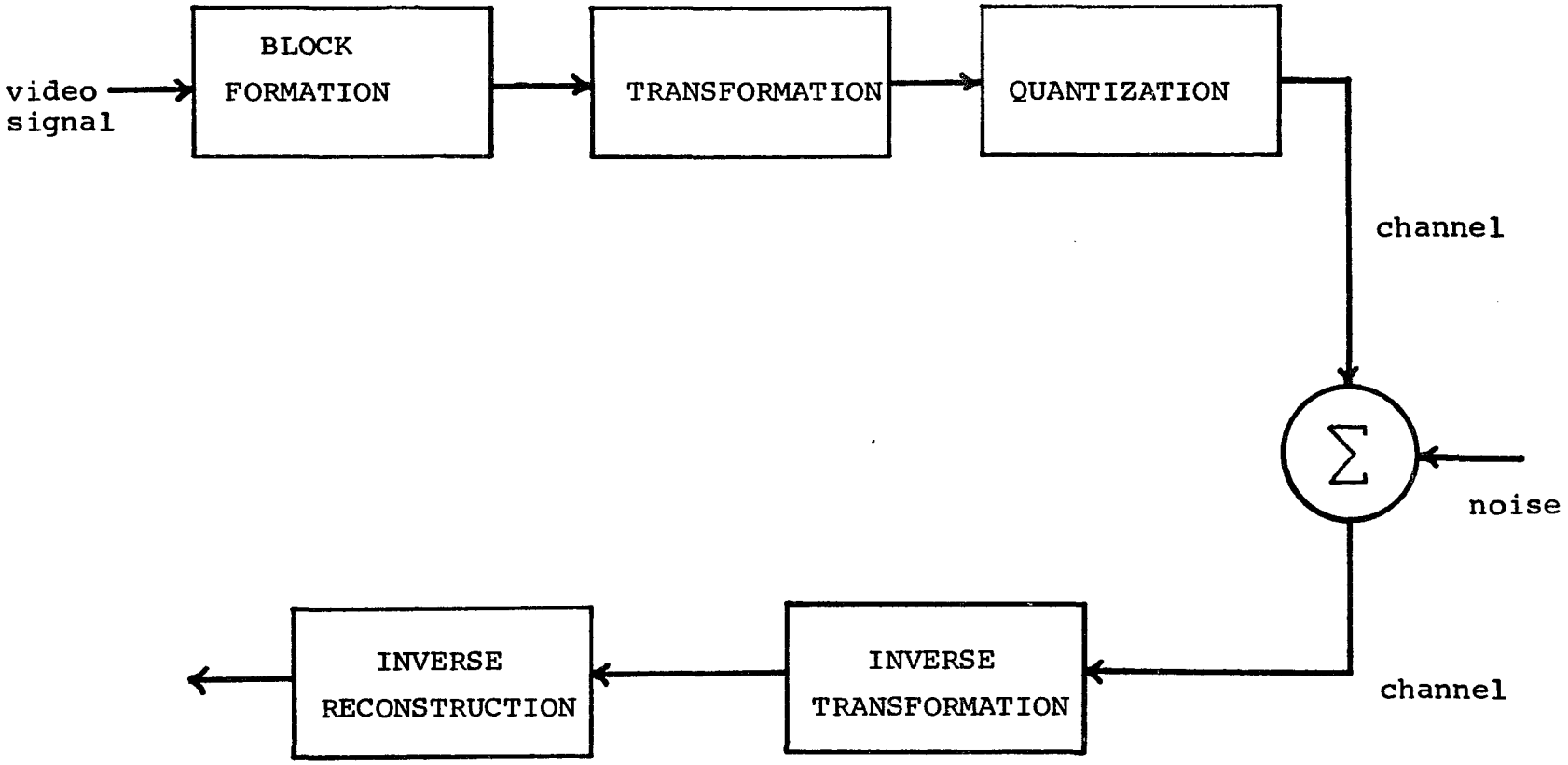


Fig. 1.2 Block diagram of a Transform Image Transmission System.

D) The transformed coefficients are then quantized for transmission.

E) An estimate of the original image is recovered at the receiving end via an inverse transformation.

Since image data contains inherent structure, the picture elements (pixels) within a local region are generally highly correlated. The transform is chosen so that the transform coefficients are less correlated and can be treated independently. If the transformation is chosen properly, some of the transformed coefficients can be disregarded and others can be quantized coarsely. The quantized transformed coefficients can then be transmitted using fewer bits than would be required if the original image samples were to be transmitted. If the transform is chosen properly and no important transformed coefficients are disregarded, the inverse transformation at the receiver should produce a good estimate of the original image.

1.2.1 BLOCK FORMATION

The image to be transform encoded is first divided into small size blocks of locally adjacent pixels. Each block is treated as an entity of data to be transformed, quantized, and transmitted. The use of small data blocks

reduces the computational complexity, takes advantage of the fact that pixels in a small region are highly correlated, and reduce the effects of channel errors by containing their effects to a small area.

A block can be of one, two or three dimension. A one dimension block of size M contains M adjacent pixels within the same line of a video image. A two dimensional block of size MxN contains M adjacent pixel within N lines of the same frame of video. A three dimensional block of size MxNxK contains pixel among K adjacent video frames. The block size and dimensionality directly effects the implementation complexity.

1.2.2 TRANSFORMATION

The transform process involves the computation of the transform coefficients for the blocks of pixel samples. Generally, the transforms selected are linear orthogonal transforms which are characterized by a matrix having an inverse. In addition, the transforms selected have seperable kernel functions which allows the computation of a higher dimensional transform to be partitioned into successive one dimensional transforms. The order of the transform corresponds to the number of pixel samples in the block.

An effective interpretation of the image transform process is obtained by using the concept of basis patterns. The number of basis patterns required equals the order of the transform. A particular coefficient is the result of correlating the pixel samples in a block to the samples of the corresponding basis pattern. The block of pixel samples can then be viewed as a weighted sum of a set of known basis pattern used for the specific transform where the transform coefficients are equivalent to the weights. The coefficients instead of the original pixel samples are quantized for transmission. The inverse transformation at the recevier is then equivalent to recomputing the block of pixel samples using the set of quantized coefficients and the set of known basis patterns.

If we consider a 1-dimensional N^{th} order transform, let $X=(X_1,X_2,..,X_N)$ be the set of pixel samples. Let $(P_1,P_2,...P_N)$ be a set of N known basis patterns of the given N^{th} order transform, where each $P_j,j=1,2,..,N$ is a vector of N elements.

$$P_j = \begin{bmatrix} P_{1j} \\ P_{2j} \\ . \\ . \\ P_{Nj} \end{bmatrix} \qquad 1.2\text{-}1$$

The transform coefficients are obtained by correlating the pixel samples of a given block with the set of basis patterns.

$$1.2\text{-}2 \qquad Y = \begin{bmatrix} Y_1 \\ Y_2 \\ . \\ Y_N \end{bmatrix} = \frac{1}{N} \sum_{j=1}^{N} X_j P_j$$

where:

$$1.2\text{-}3 \qquad Y_j = \frac{1}{N} \sum_{i=1}^{N} X_i P_{ij} = \frac{1}{N} \{X_1 P_{1j} + \ldots + X_N P_{Nj}\}$$

The pixel samples can be expressed in terms of the basis patterns and the transform coefficients.

$$1.2\text{-}4 \qquad X = \begin{bmatrix} X_1 \\ X_2 \\ . \\ X_N \end{bmatrix} = \sum_{i=1}^{N} Y_j P_j$$

where

$$1.2\text{-}5 \qquad X_j = \sum_{i=1}^{N} Y_i P_{ij} = \{Y_1 P_{1j} + \ldots . Y_N P_{Nj}\}$$

1.2.3 1-DIMENSIONAL DISCRETE FOURIER TRANSFORM (DFT)

A familiar 1-dimensional transform is the discrete Fourier transform (DFT). The M^{th} order 1-dimensional transform is defined as:

$$1.2\text{-}6 \qquad Y_w = \frac{1}{N} \sum_{s=1}^{N} X_s \exp(-2\pi jws/M)$$

$$1.2\text{-}7 \qquad X_s = \sum_{w=1}^{N} Y_w \exp(+2\pi jws/M)$$

where: M = number of pixel samples in the block.

X_s = value of the s^{th} pixel sample in the block.

Y_w = value of the w^{th} coefficient.

$j = (-1)^{1/2}$

$0 \leq w, s \leq M-1$

We notice from equation 1.2-6 that the basis patterns for the DFT transform are harmonically related sinusoids, where w is the frequency.

1.2-8 $P_0 = \exp(-2\pi j 0 s/M) = 1$

1.2-9 $P_1 = \exp(-2\pi j s/M)$

1.2-10 $P_2 = \exp(-4\pi j s/M)$

.

.

1.2-11 $P_N = \exp(-2N\pi j s/M)$

Equations 1.2-6 and 1.2-7 can be written in matrix form as:

1.2-12 $$Y = \frac{1}{M}[T]X$$

1.2-13 $$X = [T]^{-1}Y$$

where: X = a vector whose elements are the pixel samples.

Y = a vector whose elements are the transform coefficients.

T = an NxM matrix with elements $T_{ik}=\exp(-2\ jik/M)$.

1.2.4 2-DIMENSIONAL DFT

The 2-dimensional DFT is defined as:

$$1.2\text{-}14 \quad Y_{v,w} = \frac{1}{MN} \sum_{s=1}^{M} \sum_{t=1}^{N} X_{t,s} \exp[-2\pi j(vt/N + ws/M)]$$

$$1.2\text{-}15 \quad X_{t,s} = \sum_{w=1}^{M} \sum_{v=1}^{N} Y_{u,w} \exp[+2\pi j(vt/N + ws/M)]$$

where: M = number of pixel samples per line in the block.

N = number of lines in the block.

$X_{t,s}$ = value of the s^{th} pixel sample in the t^{th} line.

$Y_{v,w}$ = value of the $(vw)^{th}$ coefficient.

We notice from the above equations that the transform kernal is separable, i.e

$$1.2\text{-}16 \quad \exp[-2\pi j(vt/N + ws/M)] = \exp(-2\pi jvt/N)\exp(-2\pi jws/M)$$

The ability to seperate the kernal allows the computation of higher dimension transforms by performing successive 1-dimension transform. Thus, the 2-dimension DFT can be written as:

$$1.2\text{-}17 \quad Y_{v,w} = \frac{1}{N} \sum_{t=1}^{M} \exp\{-2\pi jvt/N\} \left[\frac{1}{M} \sum_{s=1}^{N} X_{t,s} \exp(-2\pi jws/M) \right]$$

Thus, a 2-dimensional DFT can be obtained by first performing a 1-dimensional transform horizontally on the 2-dimensional block, then a second 1-dimensional transform is performed vertically on the results of the first transform.

The basis patterns for a 2-dimensional DFT transform consists of MN harmonically related sinusoids. Each sinusoid contains MN elements which are the values of the particular sinusoid that are correlated to the MN pixel samples. The basis pattern for $P_{v,w}$ consists of the following values:

$$1.2\text{-}18 \quad P_{v,w} = \begin{bmatrix} P_{v,w,0,0} \cdots\cdots\cdots P_{v,w,0,M-1} \\ \vdots \qquad\qquad\qquad \vdots \\ \vdots \qquad\qquad\qquad \vdots \\ \vdots \qquad\qquad\qquad \vdots \\ P_{v,w,N-1,0} \cdots\cdots P_{v,w,N-1,M-1} \end{bmatrix}$$

From equation 1.2-17 we see that the 2-dimensional DFT can be put in matrix form as:

1.2-19 $$[Y] = [T_N][X][T_M]$$

where: Y = NxM matrix of coefficients.

X = NxM matrix of pixel samples.

T_N = NxN square matrix to perform 1-dimensional transform in the vertical direction.

T_M = MxM square matrix to perform 1-dimensional transform in the horizontal direction.

1.2.5 COMPUTATION

From equations 1.2-17 and 1.2-19 we see that a 2-dimensional DFT can be performed as two 1-dimensional DFTS. The elements of $[T_N]$ and $[T_M]$ are complex numbers which are fixed values that depend on their position within the matrix. Thus, the elements of $[T_N]$ and $[T_M]$ can be precalculated and stored. Because the elements of the coefficient matrix are complex quantities they consist of 2NM coefficients (real and complex). The matrix has the following conjugate symmetric properties:

1.2-20 $$Y_{v,w} = Y^*_{N-v,M-w} \qquad \begin{matrix} v=1,..N/2-1,N/2+1,..N-1 \\ w=1,..M/2-1,M/2+1,..M-1 \end{matrix}$$

1.2-21 $Y_{N/2,M/2}, Y_{0,0}, Y_{M/2,0}, Y_{N/2,0}$ are real

1.2-22 $Y_{0,w} = Y^*_{0,M-w}$

1.2-23 $Y_{v,0} = Y^*_{N-v,0}$

Thus, only NM coefficients are needed to specify the coefficient matrix.

1.3.0 PREDICTORS

Another class of encoders, called predictive encoders, can also be used to achieve data compression. In particular, we are interested in linear predictive encoders because of their simplicity and relative ease in implementation.

A linear predictive encoder will transmit information about the difference between the input signal and the internally generated estimate. The equations describing a linear predictive encoder are given below:

1.3-1 $E(N) = S(N)-X(N)$

1.3-2 $X(N) = a_1S(N-1)+a_2S(N-2)+\ldots+a_kS(N-k)$

where $S(N)$ = current input signal.

$X(N)$ = current encoder estimate.

$E(N)$ = current difference.

a_i = weighs on past input samples.

We see from the above equations that the current estimate is predicted based on the weighted sum of the previous k input samples. The weights placed on the previous input samples are chosen to minimize the mean square error between the input signal and the encoder's estimate if one assumes that this represents a satisfactory measure of the resulting image quality. The mean square error is given by:

$$1.3\text{-}3 \quad E\{[S(N)-X(N)]^2\}=E\{[S(N)-[a_1S(N-1)+\ldots a_kS(N-k)]]^2\}$$

where $E[\ ]$ represents expected value.

To minimize the mean square error we take the partial derivative of $E\{[S(N)-X(N)]^2\}$ with respect to each of the a_i and set them equal to zero, i,e.

$$1.4\text{-}3 \quad \frac{dE\{[S(N)-X(N)]^2\}}{da_i}=0 \qquad i=1,2,..,k$$

$$1.3\text{-}5 \quad -2E\{S(N-i)[S(N)-[a_1S(N-1)+..+a_kS(N-k)]]\}=0$$

Equation 1.3-5 can be written in matrix form as:

$$1.3\text{-}6 \qquad \begin{vmatrix} E[S(N)S(N-1)] \\ \cdot \\ \cdot \\ E[S(N)S(N-k)] \end{vmatrix} = \begin{vmatrix} E[S^2(N-1)] \ldots . E[S(N-k)S(N-1)] \\ \cdot \qquad\qquad\qquad\qquad \cdot \\ \cdot \qquad\qquad\qquad\qquad \cdot \\ E[S(N-1)S(N-k)] \ldots . E[S^2(N-k)] \end{vmatrix} \begin{vmatrix} a_1 \\ \cdot \\ \cdot \\ a_k \end{vmatrix}$$

The above equations can be solved for the weights if the covariance matrix of the input signal is known. Some predictors are described below.

1.3.1 DELTA PCM(DPCM) [1,4,11,19]

A DPCM system is shown in fig. 1.3. The predictive coder will use the past N samples (for an Nth order predictor) to generate an estimate of the next pixel by a weighted summation.

$$1.3\text{-}7 \qquad S_m = \sum_{i=1}^{N} A_i * S_{m-i}$$

The estimate is then compared to the incoming pixel and the difference is encoded into a fewer number of bits and transmitted. Since the number of bits used to encode the difference is less than the number of bits needed to

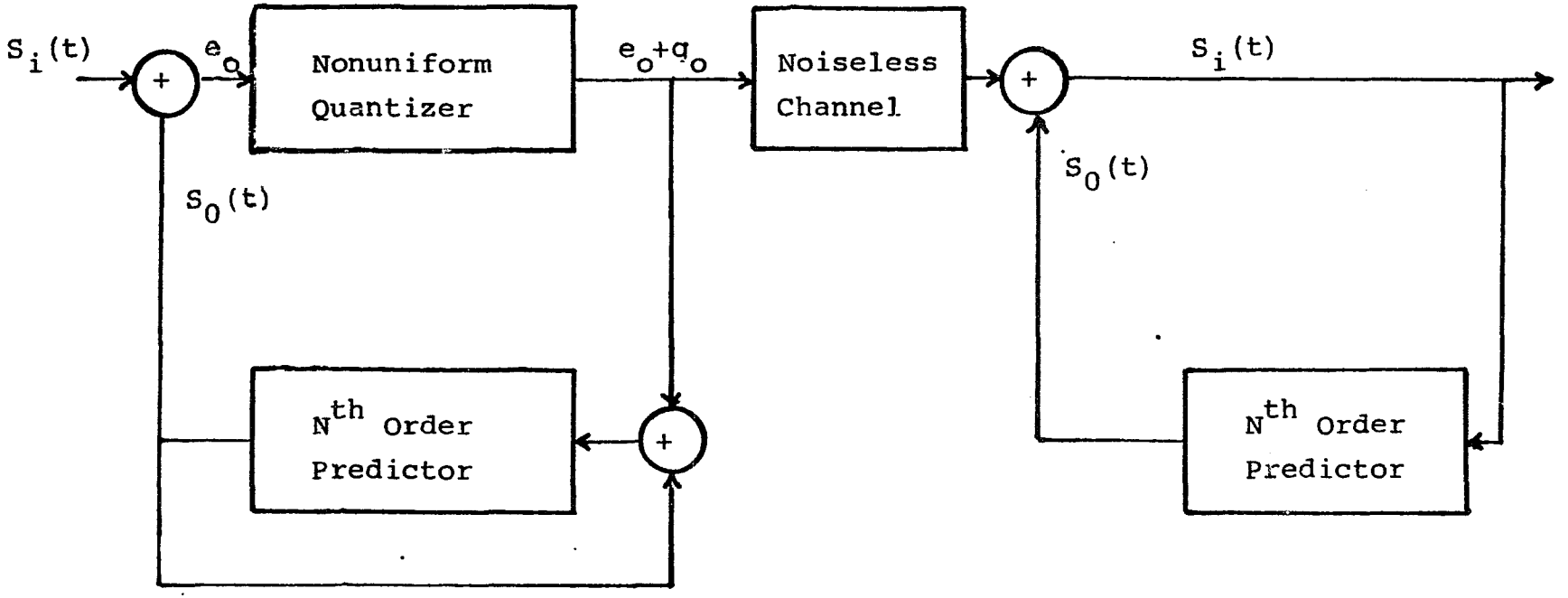


Fig. 1.3 Delta Pulse Code Modulation System (DPCM)

quantize a pixel, fewer bits are transmitted per pixel and good quality video can still be obtained.

The coefficents A_i are chosen to minimize the variance of the error signal. A DPCM system will perform well as long as the input video signal maintains the statistics for which the A_i's have been calculated. Since the input signal's statistics can vary widely, an adaptive scheme must be considered if high quality performance is to be maintained.

1.3.2 DPCM WITH ADAPTIVE PREDICTORS [2,4,11,19,26-28]

In designing a DPCM system one must either use a predictor with variable parameters such that the parameters would change with the variations in the input signal or one can use a fixed predictor with a variable quantizer to accommodate the resulting nonstationary differential signal.

A) DPCM With Adaptive Linear Predictor

In a DPCM system with an adaptive linear predictor, the weights on the adjacent samples used in predicting an incoming sample can change according to variations in the input signal value. One way in which these signal variations can be accounted for is to include a delay

during which the incoming samples are stored in an input buffer and used to obtain an estimate of the signal covariance matrix. The measured covariance matrix can be used to obtain a set of weightings for the predictor. These values are then used for processing the stored signals. The updated values of the predictor coefficients need to be transmitted periodically to the receiver.

B) DPCM Systems With Adaptive Quantizers

A DPCM system with a fixed predictor will have a nonstationary differential signal for nonstationary data. Using a fixed quantizer, nonstationary differential signals could often cause saturation or a frequent utilization of the smallest level in the quantizer. To remedy this situation, the threshold and the reconstruction levels of the quantizer must be made variable to expand and contract according to the input signal statistics. Adaptation of the quantizer to signal statistics is accomplished using various approaches. One such approach stores K samples of the differential signal to obtain an estimate for the local standard deviation of the signal. Then the stored signal is normalized by the estimated standard deviation and is quantized using a fixed quantizer. Naturally, the scaling coefficient must be transmitted once for every K samples for receiver synchronization. In a similar approach,

called Block-Adaptive DPCM, a block of M samples is stored and is normalized by N possible constants. The total distortion for all M samples using each normalizing constant is calculated at the encoder. The normalizing constant giving the smallest distortion is used to scale the samples in the block prior to their quantization and transmission. The system requires $(\log_2 N)/M$ binary digits per sample overhead information for receiver synchronization.

Still another approach could utilize a variable set of thresholds and reconstruction levels. This is the self-synchronizing approach used in adaptive delta modulators where the step size increases and decreases depending upon the polarity of sequential output levels. In a DPCM quantizer, the set of threshold and reconstruction levels would contract and expand depending upon the sequential utilization of inner or outer levels of the quantizer. For instance, a variable quantizer can be designed where all reconstruction levels expand by a factor p upon two sequential utilizations of the outermost level and they would contract by a factor of 1/p upon two sequential utilizations of the smallest level. This system is advantageous because the receiver would be self-synchronizing.

1.3.3 DELTA MODULATION (DM) [17,19]

A DM system is shown in fig. 1.4. The input signal $V_i(t)$ is low pass filtered and sampled to obtain the $(K+1)^{st}$ sample S(K+1). This sample is then compared to an internally generated $(K+1)^{st}$ estimate, X(K+1), and the sign of the difference is then transmitted. Therefore:

1.3-7 $$E(K+1)=SGN[S(k+1)-X(k+1)]$$

where E(k+1) is the transmitted bit.
The estimate is formed by adding a step size of the proper magnitude and polarity.

1.3-8 $$X(k+1)=X(k)+Y(k+1)$$

where Y(k+1) is the step size.

The step magnitude can be fixed or adaptive. In a linear DM system the magnitude of the step size is fixed. If the magnitude of the step size is adaptive the DM system is said to be adaptive.

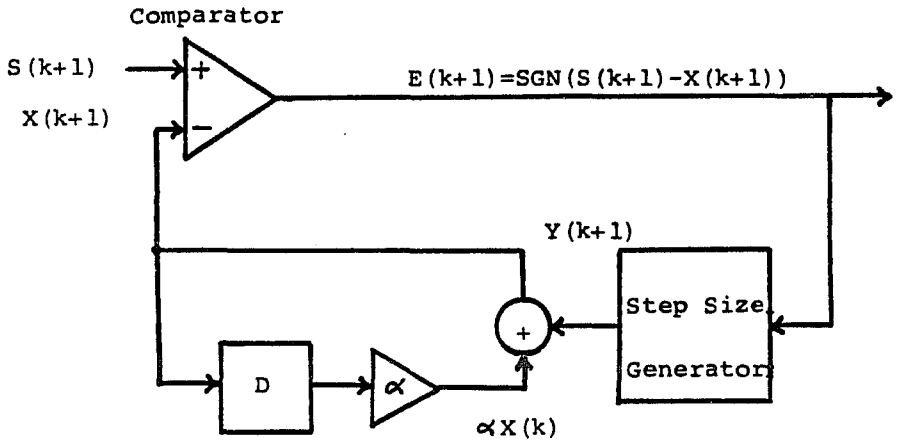


Fig. 1.4 Delta Modulation System (DM)

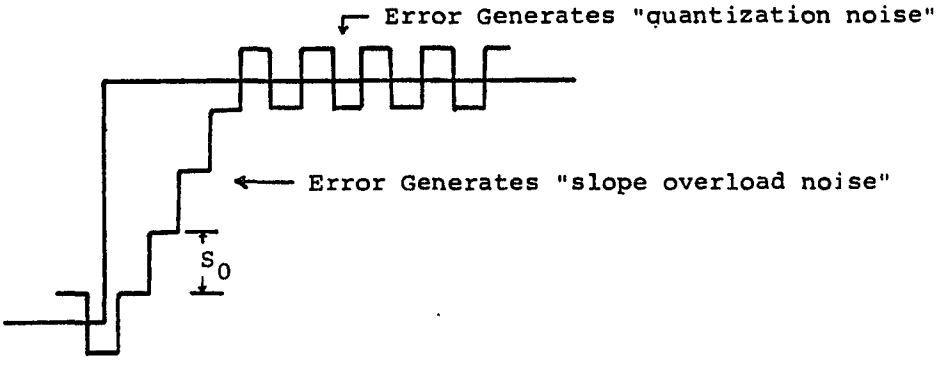


Fig. 1.5 Slope overload in DM system.

A) Linear DM (LDM) [1,8,26]

In a linear DM a new estimate is generated by adding or subtracting a fixed voltage, S_0. The new estimate is given by:

1.3-9 $$X(k+1)=X(k)+S_0*E(k)$$

A channel error would change the E(k) and the minimum error would be given by:

1.3-10 $$X(k+1)-X(k)=2S_0$$

The quantization noise depends on the value of S_0 and in general S_0 is small, so the error is not significant.

Fig. 1.5 shows a signal which changes very rapidly form a low value to a high value. Because the value of S_0 is small it takes the LDM a number of clock pulses to catch up to the signal. This poor slope tracking characteristic of the LDM is called slope overload and makes LDM undesirable for processing video signals. The maximun slope a LDM can track is given by:

1.3-11 MAX SLOPE=$S_0 * F_s$ volts/sec

The slope tracking performance of a LDM could be inproved by increasing the magnitude of S_0. This would result in greater quantizing error and in a greatly increased graininess level. A DM capable of combining a good slope tracking capability together with low graininess when the input signal is approximatly constant is call an Adaptive DM.

B) Adaptive DM (ADM) [5,9,11,16,18,21,25-28,30]

Adaptive DM algorithms are based on the detection of E(k)'s of the same polarity occurring sequentially. This condition indicates that the signal is constantly above or below the estimate. The magnitude of the step Y(k+1) is increased to allow the ADM a higher rate of change to match the rate of change of the input. When the output E(k) pattern is alternating between 1's and 0's this indicates that the estimate is close to the input signal and the magnitude of the step size is decreased. In this manner the granular noise is reduced. One of the best algorithm is Song's algorithms which is described below.

C) The Song ADM Algorithm [25]

The Song algorithm is given by the equations:

1.3-12 $E(k+1)=SGN[S(k+1)-X(k+1)]$

1.3-13 $X(k+1)=X(k)+Y(k+1)$

1.3-14

$$Y(k+1)=\begin{cases} |Y(k)|\,[E(k)+E(k-1)/2] & \begin{cases} |Y(k)| \leq Y_{max}/1.5; E(k)=E(k-1) \\ |Y(k)| < Y_{max}; E(k)\neq E(k-1) \\ 2Y_{min} \leq |Y(k)| \end{cases} \\ 2Y_{min} & |Y(k)| < 2Y_{min} \\ Y_{max} & |Y(k)| > Y_{max}/1.5; E(k)=E(k-1) \end{cases}$$

where $E(k)$ = digital output of the encoder.

$S(k)$ = the encoder estimate of the input signal.

$Y(k)$ = the step size of the delta modulator.

Y_{min} = the minimum step size of the delta modulator.

Y_{max} = the maximum step size of the delta modulator.

A characteristic of the Song mode ADM is its exponential response to an input step signal. This can be

shown by considering the Song mode ADM encoder to have an initial estimate X(0) and an initial step size Y(0) at the time of the input step signal. Thus, future estimates can be written as:

1.3.15 $X(1)=X(0)+Y(0)$

1.3.16 $X(2)=X(1)+Y(1)=X(0)+Y(0)+1.5Y(0)$

1.3.17 $X(3)=X(2)+Y(2)=X(0)+Y(0)+1.5Y(0)+1.5^{2}Y(0)$

.

.

1.3.18 $X(k)=X(0)+Y(0)+1.5Y(0)+1.5^{2}Y(0)+\ldots+1.5^{k-1}Y(0)$

Equation 1.3.18 can be written as

1.3.19 $$X(k)=X(0)+Y(0)\sum_{j=1}^{k-1}(1.5)^{j}$$

1.3.20 $X(k)=X(0)+2Y(0)[1.5^{k}-1]$

If we write

1.3.21 $1.5^{k}=e^{k\ \ln(1.5)}=e^{0.405k}$

then we can write equation 1.3.20 as

1.3.22 $X(k)=X(0)+2Y(0)e^{0.405k}-2Y(0)$

which clearly shows the exponential response of the Song mode ADM to an input step signal.

1.4.0 SLOW SCAN VIDEO ENCODER/DECODER

There are many application in which the video picture does not change for perhaps one minute or more. Such applications are in multimedia presentations, such as map viewing, teleconferencing, computer managed video communication, airline reservations, flight scheduling, etc. When the picture remains stationary for a long period of time there is no need to continually transmit the redundant bits as it adds no information to the present signal. For example, using the ADM, the number of bits necessary to transmit an entire frame at a bit rate of 16 MHz is 540 Kbits. If these bits are transmitted at the normal rate of 30 frames per second we must transmit the data at the encoded bit rate of 16 Mb/s. However, if new information is provided at the rate of one frame each

minute, the average bit rate is reduced to

1.4-1 540 Kbits/frame * 1 frame/60 sec = 9 Kbits/sec.

a significantly reduced bit rate.

1.4.1 FRAME STORAGE

There are two ways to store a frame of video signals; analog or digital. In the analog system the frame of video is stored in a storage tube and, when required, slowly read out into the ADM encoder which can operate at the low rate of, say, 9 Kb/s. Thus, the same ADM could be used for voice and slow scan video.

In the receiver the digital signal is received by the ADM decoder, converted into a analog signal and stored in a second analog storage tube. The output of this tube can then be used to drive a television system.

During the past few years almost all applications using image storage have changed from analog to digital devices. The problem with analog storage is that the system is large, costly and is of inferior quality. The analog storage device stores the image using surface charge concentration techniques. This provides marginal picture quality. System noise is found to increase with time,

degrading the stored picture; also some leakage occurs. Both of these effects act together to produce a somewhat inferior picture.

Digital frame storage techniques are inexpensive, they do not require the periodic maintenance that analog devices require, and we do not observe any degradation of the S/N ratio or of any other aspect of the picture quality with storage time.

To use a digital store, (1) we can PCM encode the analog video, store the samples and convert to ADM operating at say 9 KB/S; or (2) we can directly encode the analog video into ADM at say 16 MB/S and store the ADM bits for later transmission at say 9 KB/S. In the first technique a memory of 8*512*512=2.097 M-bits is needed while using direct ADM conversion only 0.53 M-bits is required to store a frame.

However, since memory is becoming inexpensive the first technique will probably become the more popular in time. Today the latter technique is the more practical.

1.5.0 CODING [20,22,23]

A conventional communication system is shown in fig. 1.6. Here an information source provides a message to the transmitter. The transmitter converts the message to a

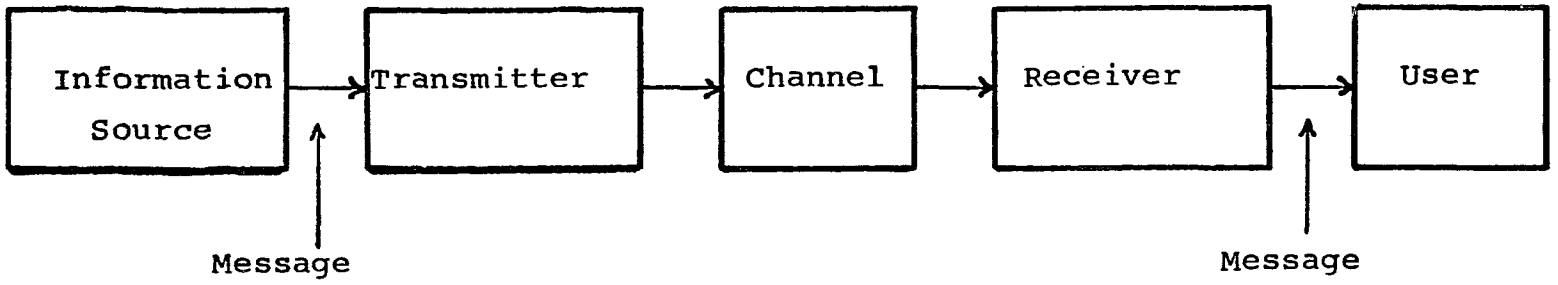


Fig. 1.6 A Commumication System

suitable form for transmission over the channel. At the channel output there is in general a noisy, distorted replica of the transmitted message. The receiver operates on this signal, converts it into a suitable form for the user. It is usually desired that the message supplied to the user be as nearly identical, in some sense, to the source as is possible.

If we consider the ADM to be our binary information source, then we are searching for ways of compressing the amount of digital data that must be transmitted over the channel by properly coding the different bit patterns being produced by the ADM. Almost every picture that has recognizable features contains a high degree of pixel correlation. This is most evident in pictures that contain large background areas where the grey level is almost constant.

A delta modulator converts analog input waveform into a string of binary digits. These binary digits can be transmitted directly or can be processed before transmission. This kind of processing prior to transmission is called entropy encoding. Entropy encoding can be applied to any finite random variable. If the ADM produces an N bit steady state pattern whenever the input signal remains constant, then there is no need to continuously transmit the steady state pattern. The

redundant bit patterns can be removed by proper coding. If a video signal is ADM encoded at a bit rate of 16 MHz, then each video line is represented as 928 binary bits. These 928 binary bits can be partitioned into N bit data blocks. There are 2^N possible N bit data patterns possible. If the probabilities of the occurrance of each of the 2^N combinations of digits are not the same, entropy encoding techniques can be employed to redefine those digits into a more efficient decipherable variable length code and thus reduce the total number of transmitted digits.

Entropy is the average information conveyed by a source symbol. For a binary digital source that consists of K code words, each with a probability of P_i, the entropy, H(S), is defined as:

1.5-1 $$H(S)= -\sum_{i=1}^{K} P_i * \mathrm{Log}_2 P_i$$

The average length of the code word is given by:

1.5-2 $$L=\sum_{i=1}^{K} P_i * L_i$$

where L_i=length of code assigned to the i^{th} source symbol.

The efficency of a code is defined as:

1.5-3 $E(S) = H(S)/L$

A high value of E(S) indicates that the code words chosen for the information source match the statistical property of the source, hence the code used is an efficient one. If the value of E(S) is low, it indicates that either other sets of code words ought to be used for a more efficient transmission or entropy encoding techniques should be employed to save channel bandwidth and reduce the code redundancy. The code redundancy, R(S), is defined as:

1.5-4 $R(S) = 1-E(S) = [L-H(S)]/L$

Different values of E(S) may be obtained under different conditions for the same digital source. If we consider the digital output from a delta modulator, depending on the block length used to calculate the entropy, E(S) can have different value. E(S) is a monotonically decreasing function of the block length. The absolute minimum value of E(S) is obtained when the block length consists of the entire output bit stream.

There are many different entropy encoding techniques. However, for the entropy encoding of the delta modulator output bits, we are interested in only two types of coding; the Huffman code and the run-length code.

1.5.1 HUFFMAN CODE [3]

The Huffman code is not a practical coding scheme to realize using hardware. This code first involves finding the probabilities of the different 2^N patterns being produced by the source. The number of bits assigned to a particular pattern depends on the probabilty of that particular pattern appearing; the least likely patterns are assigned a larger number of bits while more probable patterns are assigned fewer bits. The interest in Huffman codes lies in the fact that they are binary compact codes. A code is called compacted (for a source S) if its average length is less than or equal to the average length of all other uniquely decodable codes for the same source and the same code alphabet.

A systematic proceedure to generate a Huffman code is given below:

1) List the symbols to be encoded in the order of nonincreasing probability.
2) Group the least two probable symbols together and consider these as a new symbol whose probability is the sum of the individual probabilities.

3) Form a new list of symbols containing the remaining original symbols and the new symbol. Repeat step 1 with the new list.

4) Repeat step 2 with the new list.

5) Repeat the regrouping and relisting process until a one element group having a probability of 1 is obtained.

6) Assign binary bits to the original symbols according to the position occupied by the symbol in the various subgroups that were formed.

1.5.2 RUN-LENGTH CODE [13-15]

Run-length coding is not an optimum coding scheme, but is a less complex one. For the same amount of hardware complexity, a good run length code may exceed the performance of the Huffman code. Run-length codes find extensive use in data compression scheme whenever long sequences of a particular pattern are found. Such applications are in facsimile and two tone images.

One way to run-length encode a digital bit stream is to continuously check the digital data stream for the particular pattern that is to be coded. Upon detection of the pattern, the encoder must count the number of times the pattern appears in the sequence. When the sequence ends, the encoder must transmit a header pattern indicating to

the decoder that the following M bits contain the address where the sequence begins and the length of the sequence. There are two points to note with this approach. First, the header pattern must be one that does not occur in the digital data stream. This might necessitate that a long header be used. Second, this approach would result in substantial data compression only if the average number of bits in a sequence is large compared to the total number of bits required to identify a sequence.

Another approach would be to partition the digital bit stream into N bit data blocks. The encoder would then search the digital data bit stream until it detects a data block that contains the desired pattern. Upon detection of a matching data block, the encoder will count the number of successive matching data blocks. When the sequence ends the encoder must transmit a header pattern identifying the next K M bits as the number of consecutive dat blocks in the pattern sequence. We notice that the sequence address does not have to be transmitted because all data blocks begin at predetermined addresses. Here, again, the header pattern must be a pattern that does not appear in the digital data bit stream.

1.6.0 CONCLUSION

The use of transform coding or predictive coding can achieve data compression by reducing the 6-8 bits per pixel required of PCM to 1-3 bits per pixel. Each coding technique has advantages and disadvantages. The transform coding techniqus operates better at low bit rates but requires a large amount of complex hardware. On the other hand, a linear predictor, such as the Song mode ADM, operates well at rates of two bits per pixel and does not require nearly as much or as complex hardware for implementation

With respect to channel error, transform coding techniques are found to tolerate channel errors better than linear predictive coders. The reason for this is that the effects of a channel error in a transformed image will be localize to the pixels within a given block. A typical block might contain four to eight pixels. Linear predictor will suffer more from channel errors because all predictions after the channel error will be affected. Techniques to reduce the effects of channel error in the Song mode ADM include resetting the ADM after each video line to prevent channel error from propagating from line to line, and including a leaky integrator to reduce the effects on a video line.

INVESTIGATION USING PRESENT ADM

2.0.0 INTRODUCTION

Digital systems are preferred over analog systems once one realizes that a digital system usually costs less, is more reliable and provides higher quality performance at low signal to noise ratio (SNR). If near commerical video quality is required, if small size, low power, and low cost are factors than one should consider ADM.

2.1.0 EXPERIMENTAL APPARATUS

In order to investigate the possible data compression achieveable from the video ADM currently being used at the City College of New York the experimental apparatus shown in fig. 2.1 was used. A brief description of the system is given below; a complete description of the FRAME FREEZE unit is given in APPENDIX A.

Camera and Monitor

A SONY black and white camera is used to provide a NTSC video signal. The monitor is a standard black and white monitor.

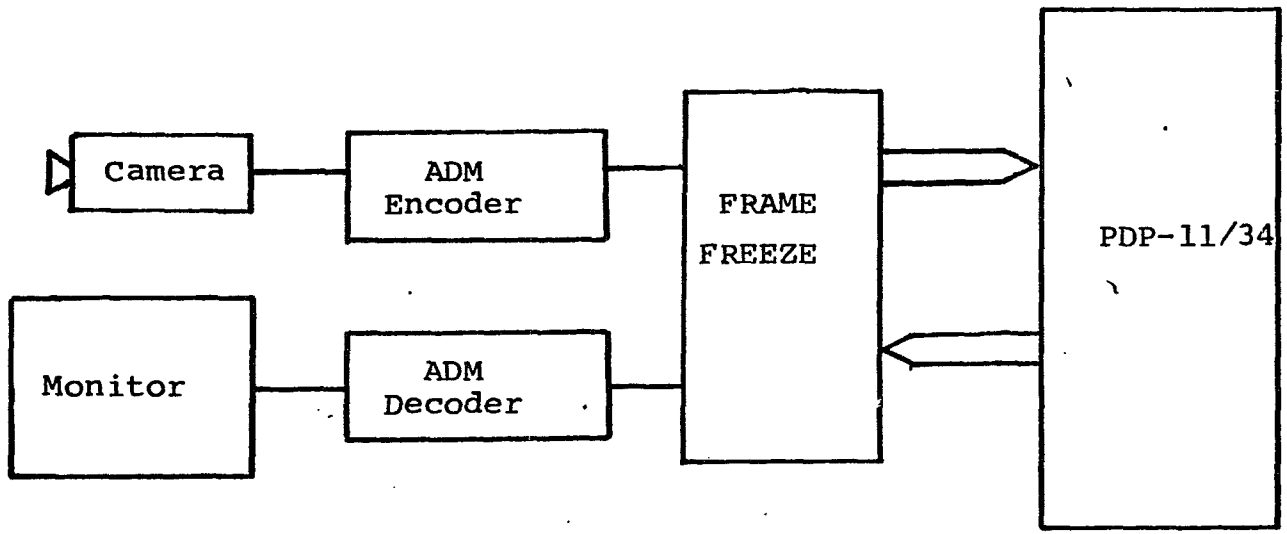


Fig. 2.1 Experimental Apparatus

ADM Encoder

The ADM encoder is used to adaptive delta modulate the input video signal, and output to the frame freeze the E(k) data at a rate of 16 MHz.

ADM Decoder

The ADM decoder is used to convert the E(k) data bits stored in the Frame Freeze into analog samples to be displayed on the monitor.

Frame Freeze (FF)

The FF was designed to store only video data and ignore the horizontal and vertical synch drives. This provides approximatly 10% reduction in data. A video picture is stored as 512 lines per frame and each line is stored as 58 by 16-bit words. The FF can operate in different modes as discribed below:

A) Display Local Image: In this mode any local image can be displayed on the monitor.

B) Frozen Mode: In this mode any local image can be stored in the FF memory and displayed on the monitor. The ADM encoder is disabled and can be removed from the system

C) CPU MODE: In this mode any image that is stored in the FF can be transferred into disk memory of the PDP-11/34 for processing. Also, in this mode any processed image

stored in the PDP-11/34 can be transferred to the FF and viewed on the monitor. During CPU input-output operations both the ADM encoder and decoder can be removed from the system.

D) PDP-11/34: The PDP-11 is used to investigate any data compression that can be achieved by removal of any steady state patterns found in the E(k) data bits.

E) MODEM MODE: In this mode any image stored in the FF can be transmitted over unconditioned telephone lines via a synchronous modem. At present we transmit at 4800 BAUD over local lines and at 1200 BAUD over long distance lines.

2.2.0 ADM HARDWARE

A block diagram of the Song mode ADM encoder build by N. Scheinberg is shown in fig. 2.2. A complete discription of the ADM hardware can be obtain in reference 30 . The equations describing the ADM are given below:

$$E(k+1)=SGN[S(k+1)-X(k+1)] \quad 2.2\text{-}1$$

$$X(k+1)=X^{*}(k)+Y(k+1) \quad 2.2\text{-}2$$

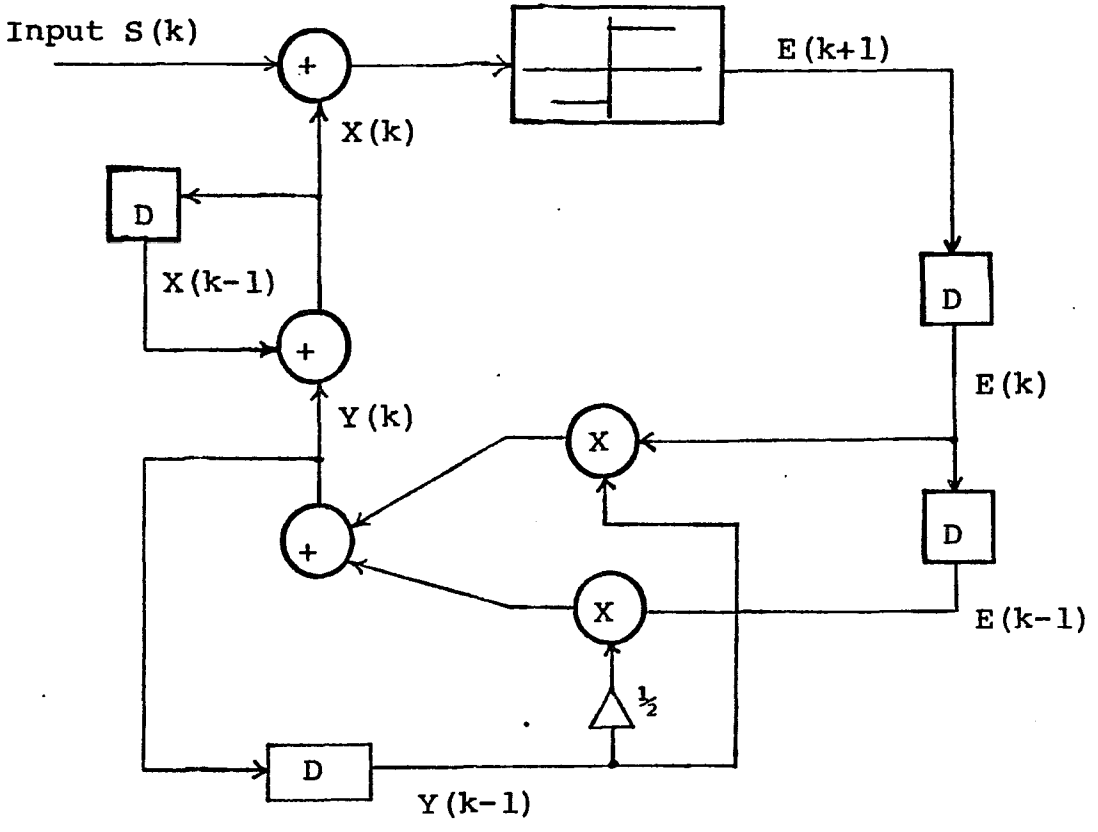


Fig. 2.2 Digital Adaptive Delta Modulator

$$
2.2\text{-}3 \qquad Y(k+1)=\begin{cases} |Y(k)|\,[E(k)+E(k-1)/2] & \begin{cases} |Y(k)| \le Y_{max}/1.5; E(k)=E(k-1) \\ |Y(k)| < Y_{max}; E(k)\neq E(k-1) \\ 2Y_{min} \le |Y(k)| \end{cases} \\ 2Y_{min}E(k) & |Y(k)| < 2Y_{min} \\ Y_{max} & |Y(k)| > Y_{max}/1.5; E(k)=E(k-1) \end{cases}
$$

where $E(k)$ = digital output of the encoder.

$S(k)$ = input analog sample to the encoder.

$X(k)$ = the encoder estimate of the input signal.

$X^{*}(k)$ = the output of the leaky integrator with $X(k)$ as the input.

$Y(k)$ = the step size of the delta modulator.

Ymin = the minimum step size of the delta modulator which is set to one quantization level.

Ymax = the maximum step of the delta modlator which is set to 15*Ymin.

The input video signal, S(k+1), is compared to an internally generated estimate signal, X(k+1), which is quantized into 256 levels. Fig. 2.3 shows a block diagram of how the step size is added to the output of the leaky

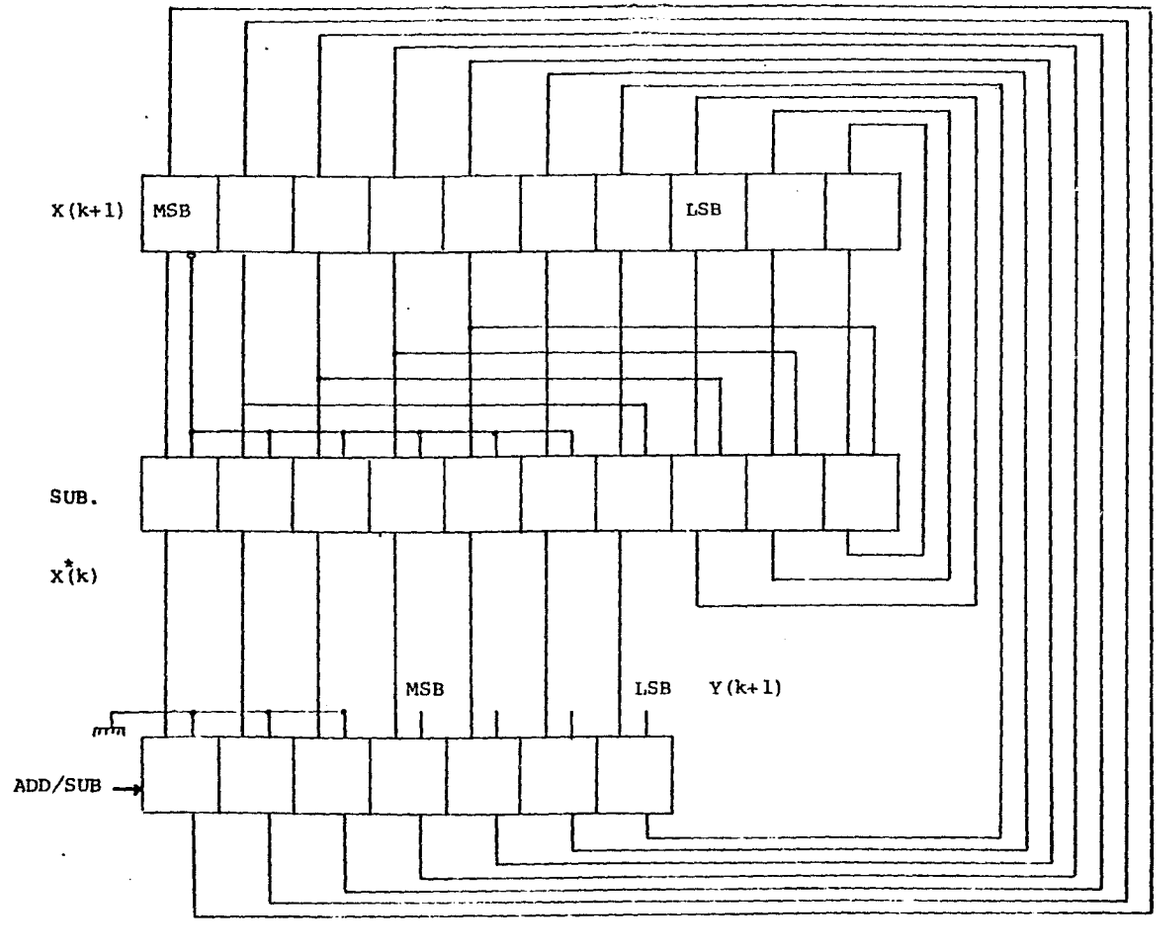


Fig. 2.3 Digital implementation of leaky integrator.

integrator , $X^*(k)$, and how $X^*(k)$ is generated. There are two points to be made with respect to fig. 2.3. First, we notice that the step size is added or subtracted from $X^*(k)$ starting with the second least significant bit. This shift of one bit makes the corresponding Y_{min} equal to two quantization levels. Second, we notice that the leaky integrator will either add or subtract a fraction of X(k) to itself. If the estimate signal is considered to be bipolar then the equation describing the leaky integrator is given by:

$$2.2\text{-}4 \qquad X^*(k) = X(k) - \frac{X(k)}{2^{\alpha}}$$

In future considerations of the leaky integrator all signals will be assume to be unipolar. With unipolar signals the leaky integrator is discribed by

$$2.2\text{-}5 \qquad X^*(k) = \frac{X(k) - |X(k) - 200_8|}{2^{\alpha}}$$

where α determines the amount of leak and 200_8 represents the unipolar estimate mid-value. The subscript eight indicates octal numbers which are used to illustrate the response of the ADM to an input square wave in SECTION 2.3

In the present ADM, α =5 which corresponds to a leaky factor of 0.9688. The use of the leaky integrator is to reduce channel errors and is discussed in chapter four.

When we transmit over telephone lines or in a good channel, P_e 10^{-5}. In this case we can expect 1 error in a line and perhaps 5 lines in error. In this case we should not use leakage as it degrades the picture quality. We should use error correction in the receiver available due to redundancy in the actual picture. As an example, we can take the video signal, A/D using PCM, convert to ADM at a lower rate. The ADM is reset at the beginning of each video line so that errors don't propagate from line to line. At the receiver, ADM decode and convert to PCM. If a channel error occured during transmission the affected PCM samples will differ significantly from its surrounding samples. The samples that differ from its adjacent samples by a predetermined amount can be replaced by the average value of the surrounding samples; i.e.

$$2.2\text{-}6 \qquad P_{i,j} = 0.25[P_{i-1,j}+P_{i+1,j}+P_{i,j-1}+P_{i,j+1}]$$

where $i = i^{th}$ line.

$j = j^{th}$ sample.

$P_{i,j} = j^{th}$ PCM sample on i^{th} line.

2.3.0 RESPONSE TO SQUARE WAVE INPUTS

The two observations in the hardware implementation of the ADM leads us to believe that the expected 0011 steady state pattern will not be realized when a constant input is applied to the ADM encoder. To see the response of the ADM to constant input signals levels an ADM encoder was simulated on the PDP-11/34. The result of the simulation with a square wave input signal can be seen on fig. 2.4 and 2.5 where the input sample, estimate, step size and corresponding output bit pattern are tabulated and plotted for two different input signals. All values in fig. 2.4 and 2.5 except for the parameter K are in octal values. The effect of the leaky integrator can be observed in all the figures. We notice that the expected 0011 pattern is not realized when tracking a constant signal. In addition, we see that every constant input level will produce a different steady state pattern. This is a result of using a leaky integrator since the amount of leak that X(k) undergoes is proportional to its magnitude. We notice that when the input signal level is 200_8, half the maximum amplitude, that the leaky integrator does not leak. Notice that if the input signal is considered to be biploar, as in equation 2.2-4, then the signal zero level would correspond

SIMULATING SONG MODE ADM
LEAKY INTEGRATOR: # OF SHIFTS = 5
ENCODER Y(K) ADDED TO 2ND LSB OF X(K)

K	S(K)	X(K)	E(K)	Y(K)
0 1	3 0 0	0 1 6	1	0 7
0 2	3 0 0	0 4 5	1	1 2
0 3	3 0 0	1 0 6	1	1 7
0 4	3 0 0	1 4 6	1	1 7
0 5	3 0 0	2 0 4	1	1 7
0 6	3 0 0	2 4 2	1	1 7
0 7	3 0 0	2 7 7	1	1 7
0 8	3 0 0	3 3 4	1	1 7
0 9	3 0 0	3 1 3	0	0 7
1 0	3 0 0	2 6 5	0	1 2
1 1	3 0 0	2 7 5	1	0 5
1 2	3 0 0	3 1 1	1	0 7
1 3	3 0 0	3 0 1	0	0 3
1 4	3 0 0	2 6 7	0	0 4
1 5	3 0 0	2 7 2	1	0 2
1 6	3 0 0	2 7 6	1	0 3
1 7	3 0 0	3 0 4	1	0 4
1 8	3 0 0	2 7 6	0	0 2
1 9	3 0 0	2 7 6	1	0 1
2 0	3 0 0	3 0 1	1	0 2
2 1	3 0 0	2 7 5	0	0 1
2 2	3 0 0	2 7 7	1	0 2
2 3	3 0 0	3 0 3	1	0 3
2 4	3 0 0	2 7 7	0	0 1
2 5	3 0 0	3 0 1	1	0 2
2 6	3 0 0	2 7 5	0	0 1
2 7	3 0 0	3 0 0	1	0 2
2 8	3 0 0	2 7 4	0	0 1
2 9	3 0 0	2 7 6	1	0 2
3 0	3 0 0	3 0 2	1	0 3
3 1	3 0 0	2 7 6	0	0 1
3 2	3 0 0	3 0 0	1	0 2
3 3	3 0 0	2 7 4	0	0 1
3 4	3 0 0	2 7 7	1	0 2
3 5	3 0 0	3 0 3	1	0 3
3 6	3 0 0	2 7 7	0	0 1
3 7	3 0 0	3 0 1	1	0 2
3 8	3 0 0	2 7 5	0	0 1
3 9	3 0 0	2 7 7	1	0 2
4 0	3 0 0	3 0 4	1	0 3
4 1	3 0 0	3 0 0	0	0 1
4 2	3 0 0	2 7 2	0	0 2
4 3	3 0 0	2 7 2	1	0 1
4 4	3 0 0	2 7 4	1	0 2
4 5	3 0 0	3 0 0	1	0 3
4 6	3 0 0	2 7 4	0	0 1
4 7	3 0 0	2 7 7	1	0 2
4 8	3 0 0	3 0 3	1	0 3
4 9	3 0 0	2 7 7	0	0 1
5 0	3 0 0	3 0 1	1	0 2
5 1	1 0 0	2 7 5	0	0 1
5 2	1 0 0	2 6 7	0	0 2
5 3	1 0 0	2 6 0	0	0 3
5 4	1 0 0	2 4 6	0	0 4
5 5	1 0 0	2 3 1	0	0 6
5 6	1 0 0	2 0 7	0	1 1
5 7	1 0 0	1 5 5	0	1 5
5 8	1 0 0	1 1 7	0	1 7
5 9	1 0 0	0 6 3	0	1 7
6 0	1 0 0	1 0 3	1	0 7
6 1	1 0 0	0 7 7	0	0 3
6 2	1 0 0	1 0 3	1	0 1
6 3	1 0 0	1 0 0	0	0 2
6 4	1 0 0	0 7 4	0	0 3
6 5	1 0 0	1 0 0	1	0 1
6 6	1 0 0	0 7 6	0	0 2
6 7	1 0 0	1 0 2	1	0 1
6 8	1 0 0	1 0 0	0	0 2
6 9	1 0 0	0 7 3	0	0 3
7 0	1 0 0	0 7 7	1	0 1
7 1	1 0 0	1 0 5	1	0 2
7 2	1 0 0	1 0 5	0	0 1
7 3	1 0 0	1 0 3	0	0 2
7 4	1 0 0	0 7 7	0	0 3
7 5	1 0 0	1 0 3	1	0 1
7 6	1 0 0	1 0 0	0	0 2
7 7	1 0 0	0 7 4	0	0 3
7 8	1 0 0	1 0 0	1	0 1
7 9	1 0 0	0 7 6	0	0 2
8 0	1 0 0	1 0 2	1	0 1
8 1	1 0 0	1 0 0	0	0 2
8 2	1 0 0	0 7 3	0	0 3
8 3	1 0 0	0 7 7	1	0 1
8 4	1 0 0	1 0 5	1	0 2
8 5	1 0 0	1 0 5	0	0 1

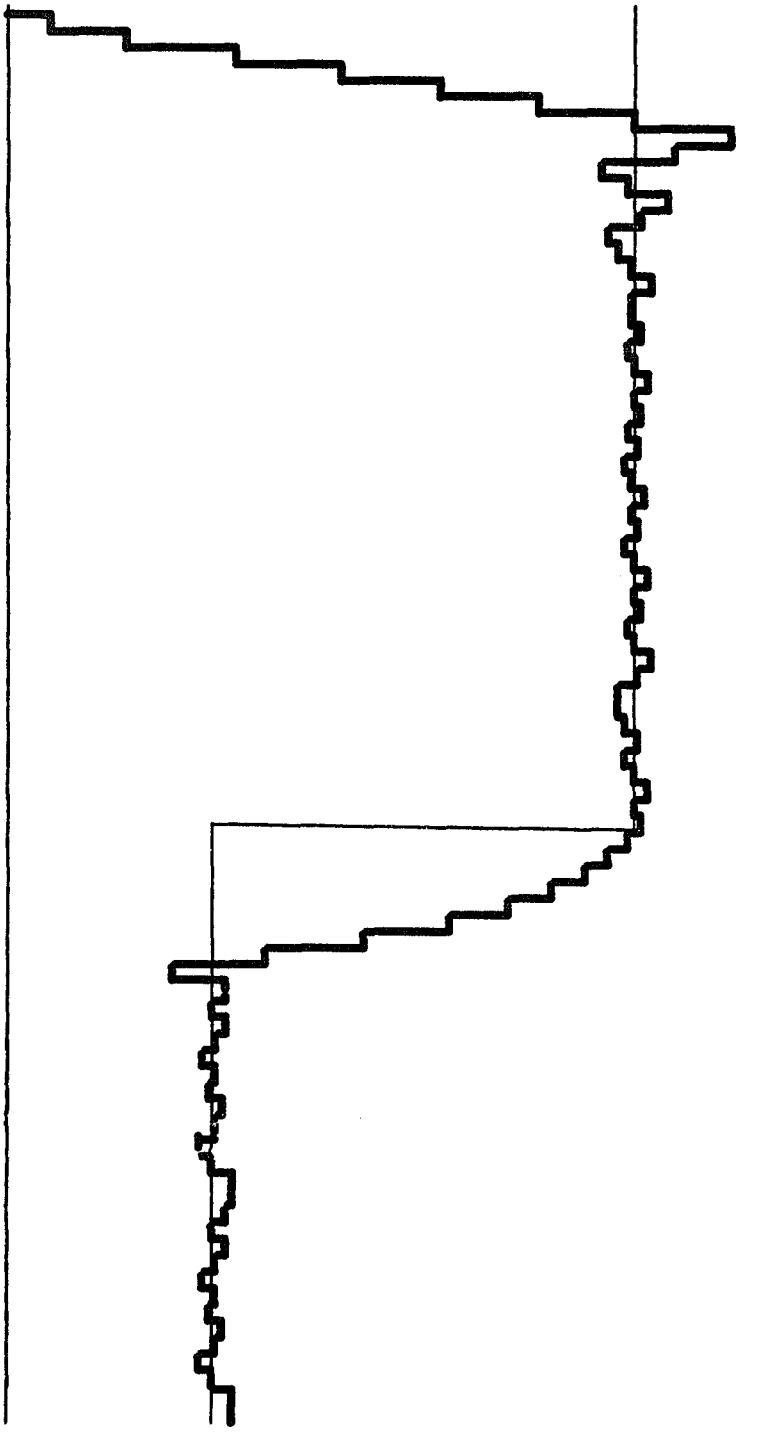


Fig. 2.4 ADM Response-1 (leak factor=0.969, 2nd LSB)

SIMULATING SONG MODE ADM
LEAKY INTEGRATOR: # OF SHIFTS = 5
ENCODER Y(K) ADDED TO 2ND LSB OF X(K)

K	S(K)	X(K)	E(K)	Y(K)
01	200	016	1	07
02	200	045	1	12
03	200	106	1	17
04	200	146	1	17
05	200	204	1	17
06	200	166	0	07
07	200	175	1	03
08	200	205	1	04
09	200	201	0	02
10	200	173	0	03
11	200	175	1	01
12	200	201	1	02
13	200	177	0	01
14	200	203	1	02
15	200	201	0	01
16	200	175	0	02
17	200	177	1	01
18	200	203	1	02
19	200	201	0	01
20	200	175	0	02
21	200	177	1	01
22	200	203	1	02
23	200	201	0	01
24	200	175	0	02
25	200	177	1	01
26	200	203	1	02
27	200	201	0	01
28	200	175	0	02
29	200	177	1	01
30	200	203	1	02
31	200	201	0	01
32	200	175	0	02
33	200	177	1	01
34	200	203	1	02
35	200	201	0	01
36	200	175	0	02
37	200	177	1	01
38	200	203	1	02
39	200	201	0	01
40	200	175	0	02
41	200	177	1	01
42	200	203	1	02
43	200	201	0	01
44	200	175	0	02
45	200	177	1	01
46	200	203	1	02
47	200	201	0	01
48	200	175	0	02
49	200	177	1	01
50	200	203	1	02
51	050	201	0	01
52	050	175	0	02
53	050	167	0	03
54	050	157	0	04
55	050	143	0	06
56	050	122	0	11
57	050	071	0	15
58	050	035	0	17
59	050	056	1	07
60	050	053	0	03
61	050	045	0	04
62	050	054	1	02
63	050	055	0	01
64	050	053	0	02
65	050	050	0	03
66	050	042	0	04
67	050	051	1	02
68	050	051	0	01
69	050	050	0	02
70	050	044	0	03
71	050	051	1	01
72	050	050	0	02
73	050	044	0	03
74	050	051	1	01
75	050	047	0	02
76	050	054	1	01
77	050	053	0	02
78	050	047	0	03
79	050	054	1	01
80	050	052	0	02
81	050	047	0	03
82	050	054	1	01
83	050	052	0	02
84	050	047	0	03
85	050	053	1	01

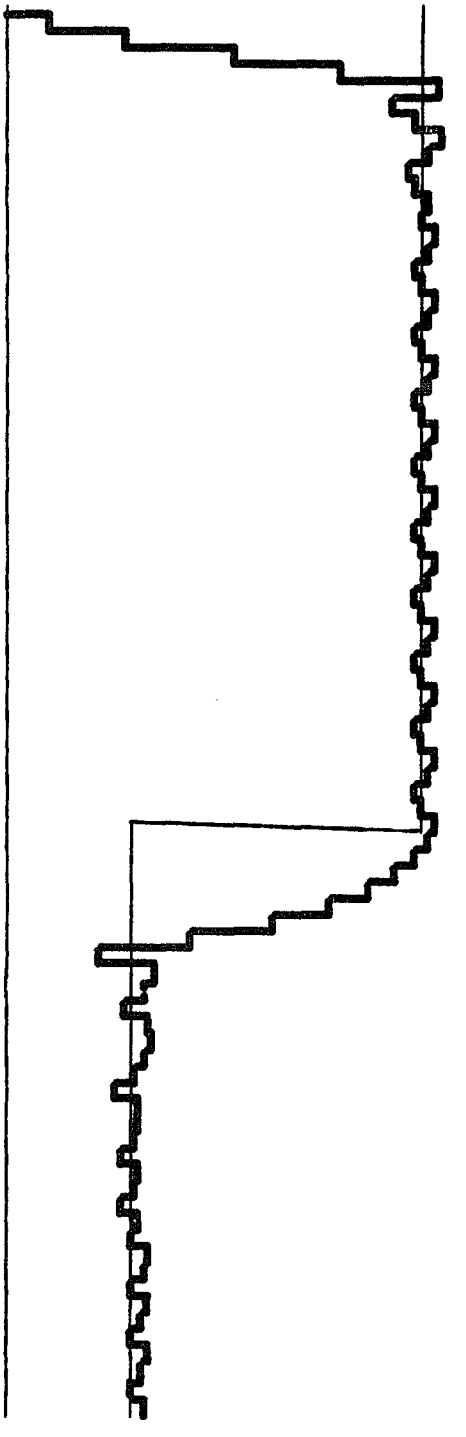


Fig. 2.5 ADM Response -2 (leak factor=0.969, 2nd LSB)

SIMULATING SONG MODE ADM
LEAKY INTEGRATOR: # OF SHIFTS = 5
ENCODER Y(K) ADDED TO LSB OF X(K)

K	S(K)	X(K)	E(K)	Y(K)
1	3 0 0	0 0 7	1	0 7
2	3 0 0	0 2 4	1	1 2
3	3 0 0	0 4 7	1	1 7
4	3 0 0	0 7 0	1	1 7
5	3 0 0	1 1 1	1	1 7
6	3 0 0	1 3 2	1	1 7
7	3 0 0	1 5 2	1	1 7
8	3 0 0	1 7 1	1	1 7
9	3 0 0	2 1 0	1	1 7
10	3 0 0	2 2 7	1	1 7
11	3 0 0	2 4 6	1	1 7
12	3 0 0	2 6 4	1	1 7
13	3 0 0	3 0 1	1	1 7
14	3 0 0	2 7 0	0	0 7
15	3 0 0	2 7 1	1	0 3
16	3 0 0	2 7 4	1	0 4
17	3 0 0	3 0 0	1	0 6
18	3 0 0	2 7 3	0	0 3
19	3 0 0	2 7 2	1	0 1
20	3 0 0	2 7 2	1	0 2
21	3 0 0	2 7 4	1	0 3
22	3 0 0	2 7 6	1	0 4
23	3 0 0	3 0 2	1	0 6
24	3 0 0	2 7 5	0	0 3
25	3 0 0	2 7 4	1	0 1
26	3 0 0	2 7 5	1	0 2
27	3 0 0	2 7 6	1	0 3
28	3 0 0	3 0 0	1	0 4
29	3 0 0	2 7 4	0	0 2
30	3 0 0	2 7 3	1	0 1
31	3 0 0	2 7 4	1	0 2
32	3 0 0	2 7 5	1	0 3
33	3 0 0	2 7 7	1	0 4
34	3 0 0	3 0 3	1	0 6
35	3 0 0	2 7 6	0	0 3
36	3 0 0	2 7 6	1	0 1
37	3 0 0	2 7 6	1	0 2
38	3 0 0	2 7 7	1	0 3
39	3 0 0	3 0 1	1	0 4
40	3 0 0	2 7 5	0	0 2
41	3 0 0	2 7 5	1	0 1
42	3 0 0	2 7 5	1	0 2
43	3 0 0	2 7 6	1	0 3
44	3 0 0	3 0 0	1	0 4
45	3 0 0	2 7 4	0	0 2
46	3 0 0	2 7 4	1	0 1
47	3 0 0	2 7 4	1	0 2
48	3 0 0	2 7 5	1	0 3
49	3 0 0	2 7 7	1	0 4
50	3 0 0	3 0 4	1	0 6
51	1 0 0	2 7 7	0	0 3
52	1 0 0	2 7 1	0	0 4
53	1 0 0	2 6 1	0	0 6
54	1 0 0	2 4 7	0	1 1
55	1 0 0	2 3 1	0	1 5
56	1 0 0	2 1 1	0	1 7
57	1 0 0	1 7 2	0	1 7
58	1 0 0	1 5 3	0	1 7
59	1 0 0	1 3 4	0	1 7
60	1 0 0	1 1 6	0	1 7
61	1 0 0	1 0 1	0	1 7
62	1 0 0	0 6 3	0	1 7
63	1 0 0	0 7 5	1	0 7
64	1 0 0	1 1 1	1	1 2
65	1 0 0	1 0 5	0	0 5
66	1 0 0	1 0 0	0	0 7
67	1 0 0	0 7 0	0	1 2
68	1 0 0	0 7 7	1	0 5
69	1 0 0	1 1 0	1	0 7
70	1 0 0	1 0 6	0	0 3
71	1 0 0	1 0 4	0	0 4
72	1 0 0	1 0 0	0	0 6
73	1 0 0	0 7 0	0	1 1
74	1 0 0	0 7 6	1	0 4
75	1 0 0	1 0 6	1	0 6
76	1 0 0	1 0 5	0	0 3
77	1 0 0	1 0 3	0	0 4
78	1 0 0	0 7 7	0	0 6
79	1 0 0	1 0 4	1	0 3
80	1 0 0	1 0 4	0	0 1
81	1 0 0	1 0 4	0	0 2
82	1 0 0	1 0 3	0	0 3
83	1 0 0	1 0 1	0	0 4
84	1 0 0	0 7 4	0	0 6
85	1 0 0	1 0 1	1	0 3

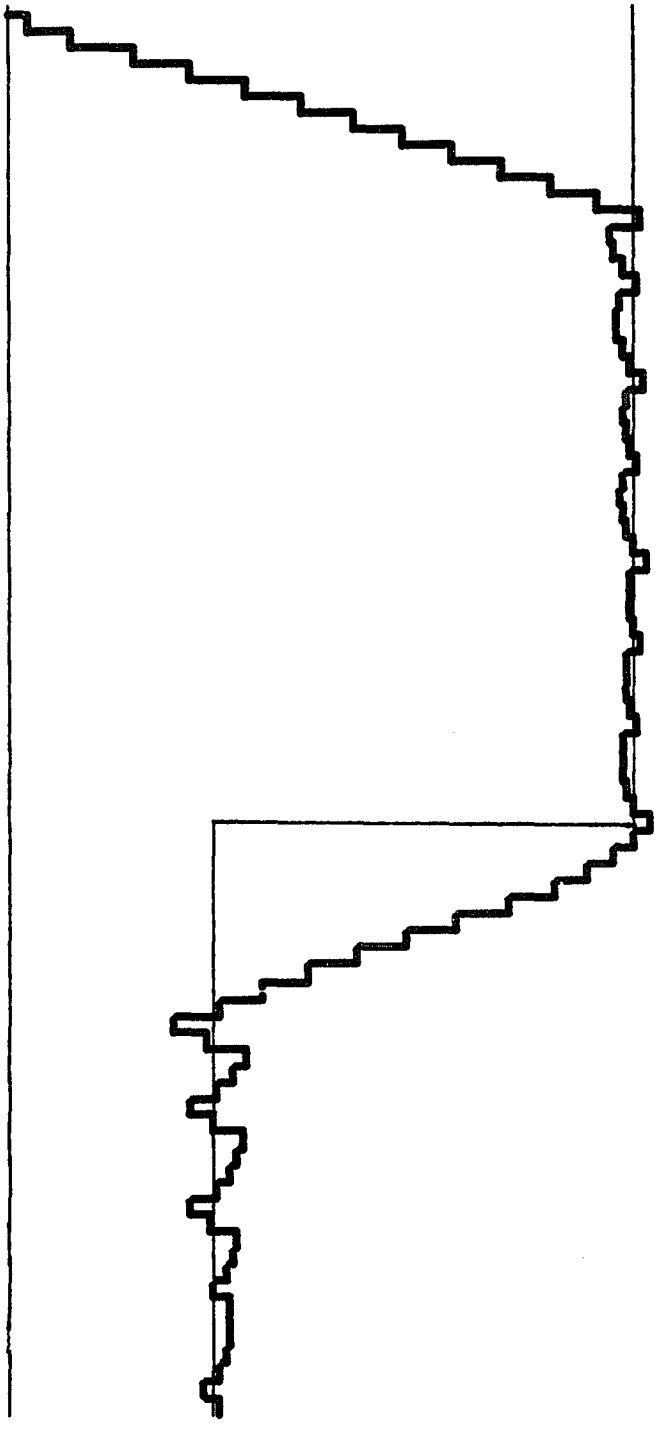


Fig. 2.6 ADM Response-1 (leak factor=0.969, LSB)

to the present value of 200 octal; since at this input level there is no leakage the resulting steady state pattern is 0011. Whenever the input signal level is above 200 octal the leaky integrator will always leak "down". In fig. 2.6 we present the response of the Song mode ADM with the step size added to the LSB of the estimate. The effects of the leaky integrator are more pronounced and can be seen in fig. 2.6 when the input signal level is 300 octal. We notice that sometime the encoder seem to produce a negative step transition even though the estimate is clearly less than the input signal. In fact, the encoder produces a positive step transition but the step size is less than the amount of negative leakage introduced by the leaky integrator, resulting in an estimate that is more negative than the previous estimate. The reverse effect occurs when the input signal level is lower than 200 octal. This effect is countered by adding the step size to the 2^{nd} LSB of the estimate. We should thus expect to find that the data compression possible from the elimination of the steady state patterns is minimal.

An alternate approch would be to redesign the ADM encoder without the leakage. The response of the Song mode ADM without a leaky integrator to both input square waves are shown in figs. 2.7 through 2.9. As can be seen, when the Song mode ADM does not contain a leaky integrator the

SIMULATING SONG MODE ADM
NO LEAKY INTEGRATOR
ENCODER Y(K) ADDED TO 2ND LSB OF X(K)

K	S(K)	X(K)	E(K)	Y(K)
01	300	016	1	07
02	300	042	1	12
03	300	100	1	17
04	300	136	1	17
05	300	174	1	17
06	300	232	1	17
07	300	270	1	17
08	300	326	1	17
09	300	310	0	07
10	300	264	0	12
11	300	276	1	05
12	300	314	1	07
13	300	306	0	03
14	300	276	0	04
15	300	302	1	02
16	300	300	0	01
17	300	274	0	02
18	300	276	1	01
19	300	302	1	02
20	300	300	0	01
21	300	274	0	02
22	300	276	1	01
23	300	302	1	02
24	300	300	0	01
25	300	274	0	02
26	300	276	1	01
27	300	302	1	02
28	300	300	0	01
29	300	274	0	02
30	300	276	1	01
31	300	302	1	02
32	300	300	0	01
33	300	274	0	02
34	300	276	1	01
35	300	302	1	02
36	300	300	0	01
37	300	274	0	02
38	300	276	1	01
39	300	302	1	02
40	300	300	0	01
41	300	274	0	02
42	300	276	1	01
43	300	302	1	02
44	300	300	0	01
45	300	274	0	02
46	300	27C	1	01
47	300	302	1	02
48	300	300	0	01
49	300	274	0	02
50	300	276	1	01
51	100	272	0	02
52	100	264	0	03
53	100	254	0	04
54	100	240	0	06
55	100	216	0	11
56	100	164	0	15
57	100	126	0	17
58	100	070	0	17
59	100	106	1	07
60	100	100	0	03
61	100	070	0	04
62	100	074	1	02
63	100	102	1	03
64	100	100	0	01
65	100	074	0	02
66	100	076	1	01
67	100	102	1	02
68	100	100	0	01
69	100	074	0	02
70	100	076	1	01
71	100	102	1	02
72	100	100	0	01
73	100	074	0	02
74	100	076	1	01
75	100	102	1	02
76	100	100	0	01
77	100	074	0	02
78	100	076	1	01
79	100	102	1	02
80	100	100	0	01
81	100	074	0	02
82	100	076	1	01
83	100	102	1	02
84	100	100	0	01
85	100	074	0	02

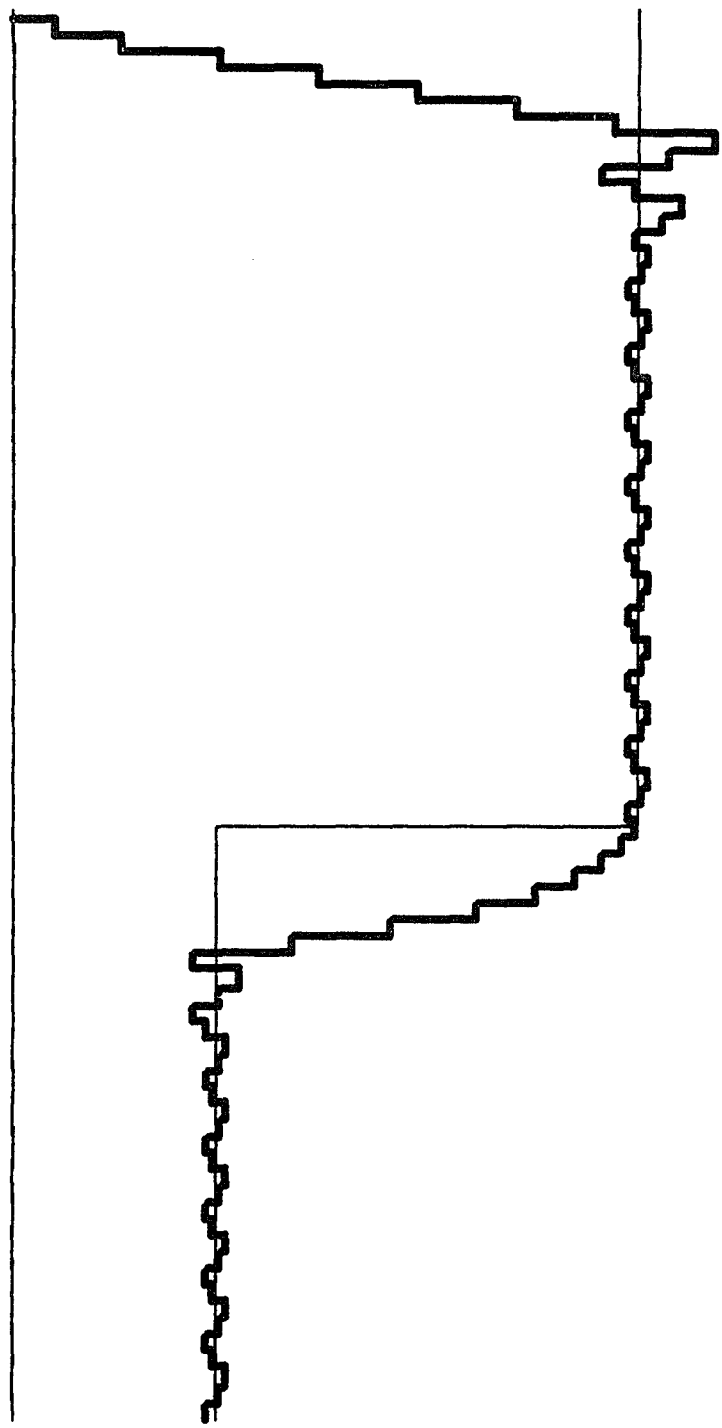


Fig. 2.7 Song Mode ADM Response-1 (2^{nd} LSB)

SIMULATING SONG MODE ADM
NO LEAKY INTEGRATOR
ENCODER Y(K) ADDED TO 2ND LSB OF X(K)

K	S(K)	X(K)	E(K)	Y(K)
01	200	016	1	07
02	200	042	1	12
03	200	100	1	17
04	200	136	1	17
05	200	174	1	17
06	200	232	1	17
07	200	214	0	07
08	200	170	0	12
09	200	202	1	05
10	200	176	0	02
11	200	200	1	01
12	200	174	0	02
13	200	176	1	01
14	200	202	1	02
15	200	200	0	01
16	200	174	0	02
17	200	176	1	01
18	200	202	1	02
19	200	200	0	01
20	200	174	0	02
21	200	176	1	01
22	200	202	1	02
23	200	200	0	01
24	200	174	0	02
25	200	176	1	01
26	200	202	1	02
27	200	200	0	01
28	200	174	0	02
29	200	176	1	01
30	200	202	1	02
31	200	200	0	01
32	200	174	0	02
33	200	176	1	01
34	200	202	1	02
35	200	200	0	01
36	200	174	0	02
37	200	176	1	01
38	200	202	1	02
39	200	200	0	01
40	200	174	0	02
41	200	176	1	01
42	200	202	1	02
43	200	200	0	01
44	200	174	0	02
45	200	176	1	01
46	200	202	1	02
47	200	200	0	01
48	200	174	0	02
49	200	176	1	01
50	200	202	1	02
51	050	200	0	01
52	050	174	0	02
53	050	166	0	03
54	050	156	0	04
55	050	142	0	06
56	050	120	0	11
57	050	066	0	15
58	050	030	0	17
59	050	046	1	07
60	050	072	1	12
61	050	060	0	05
62	050	042	0	07
63	050	050	1	03
64	050	046	0	01
65	050	052	1	02
66	050	050	0	01
67	050	044	0	02
68	050	046	1	01
69	050	052	1	02
70	050	050	0	01
71	050	044	0	02
72	050	046	1	01
73	050	052	1	02
74	050	050	0	01
75	050	044	0	02
76	050	046	1	01
77	050	052	1	02
78	050	050	0	01
79	050	044	0	02
80	050	046	1	01
81	050	052	1	02
82	050	050	0	01
83	050	044	0	02
84	050	046	1	01
85	050	052	1	02

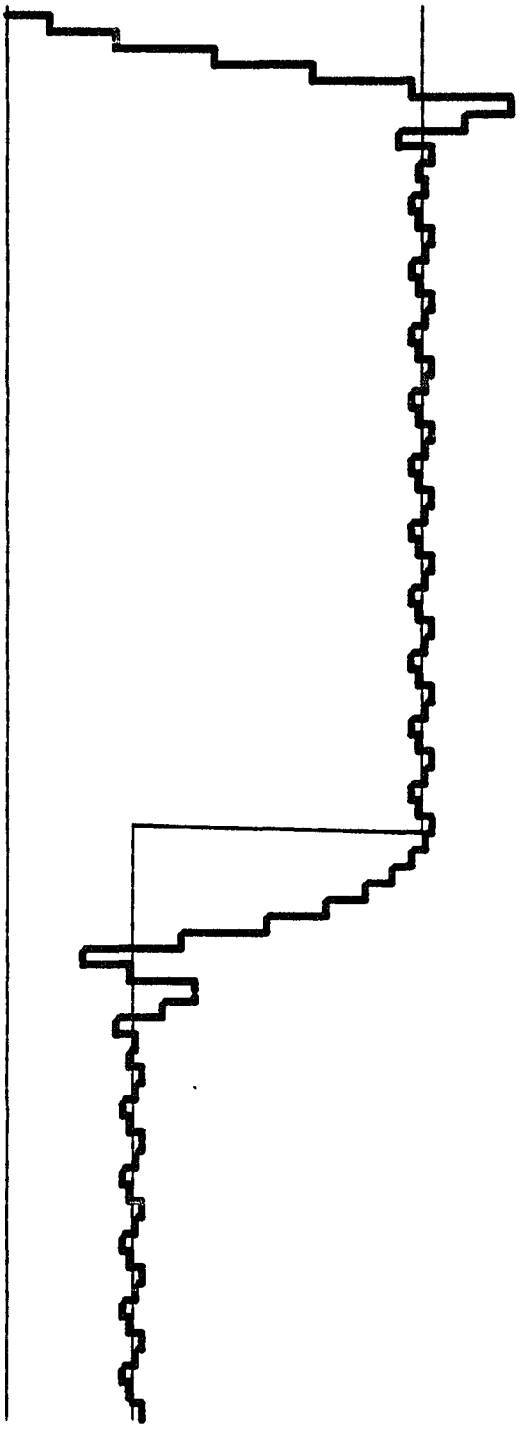


Fig 2.8 Song Mode ADM Response-2 (2^{nd} LSB)

SIMULATING SONG MODE ADM
NO LEAKY INTEGRATOR
ENCODER Y(K) ADDED TO LSB OF X(K)

K	S(K)	X(K)	E(K)	Y(K)
0 1	3 0 0	0 0 7	1	0 7
0 2	3 0 0	0 2 1	1	1 2
0 3	3 0 0	0 4 0	1	1 7
0 4	3 0 0	0 5 7	1	1 7
0 5	3 0 0	0 7 6	1	1 7
0 6	3 0 0	1 1 5	1	1 7
0 7	3 0 0	1 3 4	1	1 7
0 8	3 0 0	1 5 3	1	1 7
0 9	3 0 0	1 7 2	1	1 7
1 0	3 0 0	2 1 1	1	1 7
1 1	3 0 0	2 3 0	1	1 7
1 2	3 0 0	2 4 7	1	1 7
1 3	3 0 0	2 6 6	1	1 7
1 4	3 0 0	3 0 5	1	1 7
1 5	3 0 0	2 7 6	0	0 7
1 6	3 0 0	3 0 1	1	0 3
1 7	3 0 0	3 0 0	0	0 1
1 8	3 0 0	2 7 6	0	0 2
1 9	3 0 0	2 7 7	1	0 1
2 0	3 0 0	3 0 1	1	0 2
2 1	3 0 0	3 0 0	0	0 1
2 2	3 0 0	2 7 6	0	0 2
2 3	3 0 0	2 7 7	1	0 1
2 4	3 0 0	3 0 1	1	0 2
2 5	3 0 0	3 0 0	0	0 1
2 6	3 0 0	2 7 6	0	0 2
2 7	3 0 0	2 7 7	1	0 1
2 8	3 0 0	3 0 1	1	0 2
2 9	3 0 0	3 0 0	0	0 1
3 0	3 0 0	2 7 6	0	0 2
3 1	3 0 0	2 7 7	1	0 1
3 2	3 0 0	3 0 1	1	0 2
3 3	3 0 0	3 0 0	0	0 1
3 4	3 0 0	2 7 6	0	0 2
3 5	3 0 0	2 7 7	1	0 1
3 6	3 0 0	3 0 1	1	0 2
3 7	3 0 0	3 0 0	0	0 1
3 8	3 0 0	2 7 6	0	0 2
3 9	3 0 0	2 7 7	1	0 1
4 0	3 0 0	3 0 1	1	0 2
4 1	3 0 0	3 0 0	0	0 1
4 2	3 0 0	2 7 6	0	0 2
4 3	3 0 0	2 7 7	1	0 1
4 4	3 0 0	3 0 1	1	0 2
4 5	3 0 0	3 0 0	0	0 1
4 6	3 0 0	2 7 6	0	0 2
4 7	3 0 0	2 7 7	1	0 1
4 8	3 0 0	3 0 1	1	0 2
4 9	3 0 0	3 0 0	0	0 1
5 0	3 0 0	2 7 6	0	0 2
5 1	1 0 0	2 7 3	0	0 3
5 2	1 0 0	2 6 7	0	0 4
5 3	1 0 0	2 6 1	0	0 6
5 4	1 0 0	2 5 0	0	1 1
5 5	1 0 0	2 3 3	0	1 5
5 6	1 0 0	2 1 4	0	1 7
5 7	1 0 0	1 7 5	0	1 7
5 8	1 0 0	1 5 6	0	1 7
5 9	1 0 0	1 3 7	0	1 7
6 0	1 0 0	1 2 0	0	1 7
6 1	1 0 0	1 0 1	0	1 7
6 2	1 0 0	0 6 2	0	1 7
6 3	1 0 0	0 7 1	1	0 7
6 4	1 0 0	1 0 3	1	1 2
6 5	1 0 0	0 7 6	0	0 5
6 6	1 0 0	1 0 0	1	0 2
6 7	1 0 0	0 7 7	0	0 1
6 8	1 0 0	1 0 1	1	0 2
6 9	1 0 0	1 0 0	0	0 1
7 0	1 0 0	0 7 6	0	0 2
7 1	1 0 0	0 7 7	1	0 1
7 2	1 0 0	1 0 1	1	0 2
7 3	1 0 0	1 0 0	0	0 1
7 4	1 0 0	0 7 6	0	0 2
7 5	1 0 0	0 7 7	1	0 1
7 6	1 0 0	1 0 1	1	0 2
7 7	1 0 0	1 0 0	0	0 1
7 8	1 0 0	0 7 6	0	0 2
7 9	1 0 0	0 7 7	1	0 1
8 0	1 0 0	1 0 1	1	0 2
8 1	1 0 0	1 0 0	0	0 1
8 2	1 0 0	0 7 6	0	0 2
8 3	1 0 0	0 7 7	1	0 1
8 4	1 0 0	1 0 1	1	0 2
8 5	1 0 0	1 0 0	0	0 1

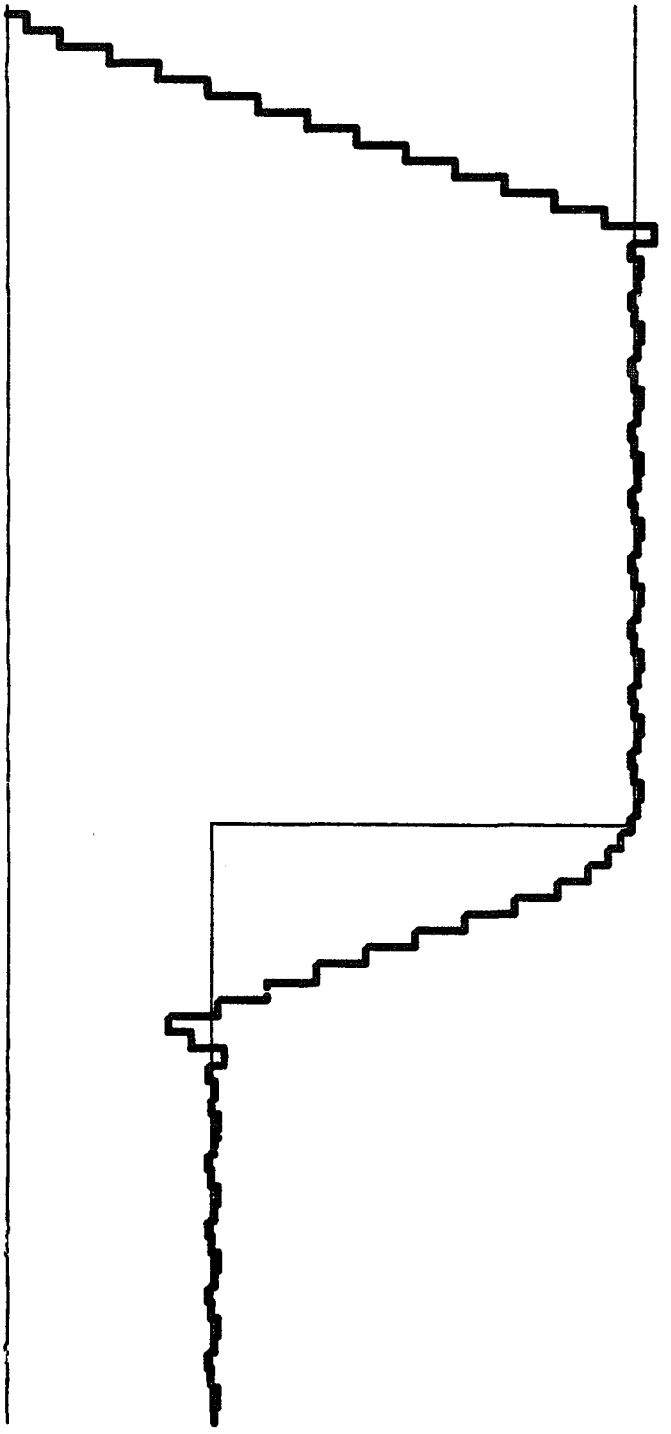


Fig. 2.9 Song Mode ADM Response -1 (LSB)

0011 pattern will result whenever the Song mode ADM is tracking a constant input signal, regardless of the signal amplitude. If we consider the response of the ADM with the step size added to the LSB of the estimate and compare the response of the ADM with and without a leaky integrator we notice that the leaky integrator provides a better slope overload characteritic. This characteristic is desired when encoding video signals because of their rapid variations. This results because the leaky integrator tends to produce, on the average, a larger step size. There are two effects that can be observed when the step size is added to the estimate's LSB. One effect is that the encoder will track any constant the input signal closer. This results because the corresponding step size are smaller by a factor of two. Using a larger step size will result in an increase granular noise which could prove distracting. Simulation with actual video images did not reveal any noticeable granular noise when the step size is added to the 2^{nd} LSB of the estimate. The second effect observed is that by adding the step size to the estimate's LSB the ADM is more vulnerable to slope over-load. Simulation with video images did not reveal any noticeable degragation. Experimental results indicate that a Song mode ADM without a leaky integrator will produce all four possible rotations of the steady state pattern with

approximately equal probability making it unprofitable to encode which particular pattern rotation is being removed. As discussed in SECTION 1.5, if long run-lengths of the particular steady state pattern being removed are expected, then run-length coding might prove profitable. Experimental results are presented in SECTION 2.4 for run-length statistics with actual input video images.

2.4.0 RESPONSE TO INPUT IMAGES

In fig. 2.10 are pictures of the two input images selected for processing. These images have been selected because they contain varying amounts of detail. The picture of the GIRL contains large areas of nearly constant grey level, while on the other hand, the picture of the BOY contains a large amount of detail. Both pictures can be seen to contain edge busyness which is a result of delta modulation. Edge busyness is a distortion of edges perpendicular to the scanning direction of the television camera. In a horizontally scanned image all vertical edges are seen to contain "wiggles". We noticed in figs. 2.4 thru 2.9 that the ADM response to an input step signal takes time to "catch-up" to the input signal due to the slope overload. The amount of time required for the

Fig. 2.10 Input pictures of BOY and GIRL.

estimate to equal the input signal will depend on the initial step size at the time of the step input. A vertical edge corresponds to a step in the input signal. The "wiggles" on the vertical edges of the image is produced because the encoder's step size at the vertical edge will vary with each scanning line. The result is that on some lines the estimate will take a longer time to reach the input signal level, thus displaceing the edge in the encoding direction. If the type of images being delta modulated contain a large amount of vertical edges, then a better image, in the sense that less edge busyness will be noticed, can be obtained if the television camera is made to scan in the vertical direction instead of the horizontal direction.

In TABLE 2-1 and 2-2 are tabulated the probabilities of every 4 bit and 8 bit pattern for the two input images using the present ADM hardware. We noticed in SECTION 1.5 that the easiest way of specifying the address of a particular pattern is by partitioning the E(k) data of each video line into equal blocks of bits. Thus, TABLE 2-1 and 2-2 indicate the probability of a 4 bit or 8 bit data block corresponding to the indicated bit pattern. We notice that no particular bit pattern appears in sufficient numbers to make coding profitable.

To investigate the possible data compression

P_0P_1 \ P_2P_3	00	01	10	11
00	12.15	8.40	9.33	4.94
01	9.53	5.59	5.82	3.34
10	8.70	5.95	5.86	4.24
11	5.04	4.55	3.61	2.96

$P_0P_1P_2P_3P_4$ \ $P_5P_6P_7$	000	001	010	011	100	101	110	111
00000	4.48	0.22	0.28	0.27	1.02	0.21	0.44	0.07
00001	2.46	0.49	0.45	0.09	0.88	0.20	0.12	0.12
00010	2.51	1.12	0.79	0.14	0.73	0.18	0.15	0.11
00011	1.16	0.30	0.34	0.17	0.26	0.14	0.16	0.13
00100	0.70	1.74	1.23	0.33	0.87	0.51	0.24	0.19
00101	0.79	0.57	0.35	0.30	0.32	0.34	0.20	0.23
00110	0.76	0.69	0.35	0.25	0.50	0.29	0.22	0.13
00111	0.26	0.35	0.28	0.23	0.18	0.19	0.14	0.13
01000	0.33	0.89	2.15	0.45	1.22	0.42	0.47	0.16
01001	0.83	0.68	0.71	0.47	0.35	0.29	0.27	0.22
01010	0.52	0.47	0.64	0.37	0.36	0.37	0.44	0.16
01011	0.38	0.38	0.54	0.22	0.25	0.15	0.20	0.18
01100	0.39	0.72	0.50	0.44	0.23	0.42	0.31	0.25
01101	0.41	0.48	0.44	0.37	0.27	0.34	0.13	0.12
01110	0.18	0.21	0.23	0.34	0.21	0.25	0.25	0.19
01111	0.09	0.28	0.17	0.29	0.11	0.24	0.18	0.10
10000	0.71	0.32	1.08	0.27	1.67	0.33	0.63	0.09
10001	1.00	0.51	0.49	0.18	0.64	0.28	0.30	0.19
10010	0.76	0.41	0.62	0.22	0.57	0.44	0.52	0.29
10011	0.36	0.22	0.45	0.18	0.38	0.31	0.21	0.13
10100	0.29	0.55	0.41	0.34	0.36	0.56	0.36	0.29
10101	0.28	0.47	0.42	0.27	0.44	0.50	0.23	0.14
10110	0.22	0.30	0.26	0.32	0.49	0.48	0.35	0.14
10111	0.17	0.22	0.18	0.23	0.19	0.29	0.21	0.14
11000	0.21	0.27	0.52	0.26	0.38	0.23	0.41	0.26
11001	0.18	0.21	0.24	0.41	0.28	0.34	0.46	0.24
11010	0.29	0.25	0.37	0.36	0.30	0.36	0.43	0.18
11011	0.18	0.23	0.43	0.26	0.20	0.26	0.30	0.22
11100	0.09	0.11	0.13	0.18	0.16	0.27	0.32	0.37
11101	0.12	0.19	0.21	0.28	0.19	0.34	0.34	0.34
11110	0.08	0.11	0.18	0.40	0.10	0.21	0.28	0.47
11111	0.06	0.25	0.13	0.32	0.21	0.30	0.14	0.02

TABLE 2-1 Statistics of 4 and 8-bit patterns for input image of the BOY. (%)

P_0P_1 \ P_2P_3	00	01	10	11
00	12.05	9.38	10.70	4.81
01	10.34	5.12	5.21	3.14
10	9.66	6.14	5.30	3.60
11	4.88	3.61	3.43	2.61

$P_0P_1P_2P_3P_4$ \ $P_5P_6P_7$	000	001	010	011	100	101	110	111
00000	4.42	0.16	0.23	0.13	0.67	0.18	0.46	0.08
00001	2.66	0.42	0.46	0.10	1.02	0.20	0.14	0.08
00010	2.60	1.21	1.07	0.18	0.90	0.17	0.14	0.11
00011	1.54	0.25	0.25	0.17	0.25	0.17	0.19	0.08
00100	0.59	2.31	1.51	0.53	1.31	0.77	0.21	0.15
00101	0.96	0.68	0.32	0.22	0.25	0.20	0.20	0.19
00110	1.11	0.69	0.27	0.16	0.31	0.20	0.20	0.17
00111	0.24	0.37	0.25	0.21	0.22	0.22	0.14	0.08
01000	0.22	0.62	2.26	0.59	1.32	0.59	0.72	0.14
01001	0.98	1.09	0.97	0.28	0.36	0.19	0.23	0.17
01010	0.63	0.54	0.79	0.28	0.34	0.25	0.27	0.14
01011	0.31	0.22	0.41	0.21	0.17	0.15	0.20	0.13
01100	0.41	1.00	0.50	0.31	0.22	0.25	0.22	0.22
01101	0.29	0.38	0.36	0.33	0.23	0.31	0.11	0.10
01110	0.17	0.18	0.20	0.30	0.16	0.27	0.21	0.17
01111	0.17	0.25	0.16	0.31	0.16	0.15	0.14	0.14
10000	0.67	0.17	0.95	0.45	1.96	0.44	0.70	0.14
10001	1.15	0.78	0.61	0.14	0.84	0.22	0.22	0.19
10010	0.70	0.72	1.10	0.20	0.89	0.30	0.31	0.24
10011	0.34	0.19	0.33	0.17	0.30	0.24	0.28	0.16
10100	0.24	0.62	0.49	0.30	0.45	0.45	0.27	0.25
10101	0.22	0.32	0.30	0.25	0.30	0.37	0.19	0.17
10110	0.25	0.26	0.21	0.24	0.34	0.39	0.27	0.13
10111	0.16	0.21	0.18	0.23	0.19	0.26	0.17	0.14
11000	0.15	0.39	0.85	0.33	0.47	0.18	0.32	0.29
11001	0.16	0.15	0.19	0.30	0.18	0.27	0.34	0.22
11010	0.27	0.18	0.21	0.24	0.25	0.31	0.33	0.18
11011	0.21	0.17	0.33	0.19	0.15	0.20	0.25	0.19
11100	0.09	0.26	0.14	0.19	0.21	0.23	0.34	0.37
11101	0.14	0.18	0.22	0.23	0.20	0.29	0.28	0.30
11110	0.16	0.18	0.20	0.34	0.14	0.21	0.19	0.35
11111	0.16	0.18	0.08	0.27	0.13	0.12	0.13	0.03

TABLE 2-2 Statistics of 4 and 8-bit patterns for input image of the GIRL. (%)

resulting from using an ADM without a leaky integrator a Song mode ADM without a leaky integrator was simulated on the PDP-11/34. In TABLE 2-3 and 2-4 are tabulated the 4-bit pattern statistics for the picture of the GIRL with the step size added to either the LSB or 2^{nd} LSB of the estimate. The image of the GIRL was selected because it contains large areas of approximate uniform grey level which would tend to produce more steady state pattern. Once again, no particular steady state pattern is produce in sufficient quantity to warrent coding. We notice that by adding the step size to the 2^{nd} LSB of the estimate allows the encoder's estimate to track the input signal at a faster rate as demonstrated by the simulated response of the ADM. This faster tracking capablity allows the ADM to reach steady state quicker and results in the production of more steady state patterns. Comparing the 4 bit pattern statistics for the processed picture of the GIRL without a leaky integrator we see that if the step size is added to the LSB of X(k) all 16 possible patterns are generated with approximately equal probability. However, by adding the step size to the 2^{nd} LSB of the estimate produces a noticeable increase in the generation of the 1100 steady state pattern and all its rotations.

It was stated before that one effect of the leaky integrator is to produce a larger step size in general. To

P_0P_1 \ P_2P_3	00	01	10	111
00	8.92	4.85	6.32	6.07
01	6.29	7.28	7.30	5.14
10	4.81	7.11	7.34	6.18
11	5.92	6.29	5.09	5.11

$P_0P_1P_2P_3P_4$ \ $P_5P_6P_7$	000	001	010	011	100	101	110	1111
00000	4.83	0.34	0.35	0.21	0.52	0.12	0.34	0.14
00001	0.74	0.28	0.12	0.18	0.26	0.31	0.17	0.08
00010	0.70	0.41	0.20	0.31	0.12	0.31	0.23	0.23
00011	0.22	0.48	0.33	0.35	0.23	0.31	0.14	0.10
00100	0.52	0.31	0.33	0.38	0.21	0.43	0.56	0.63
00101	0.10	0.29	0.45	0.71	0.49	0.44	0.39	0.33
00110	0.18	0.50	0.59	0.70	0.59	0.75	0.41	0.26
00111	0.24	0.41	0.38	0.26	0.18	0.21	0.13	0.23
01000	0.42	0.36	0.36	0.23	0.35	0.34	0.46	0.23
01001	0.16	0.28	0.58	0.52	0.58	0.61	0.64	0.53
01010	0.13	0.31	0.53	0.63	0.55	0.72	0.72	0.38
01011	0.25	0.69	0.64	0.29	0.35	0.35	0.24	0.37
01100	0.22	0.24	0.52	0.56	0.25	0.79	0.87	0.45
01101	0.33	0.77	0.85	0.19	0.45	0.29	0.19	0.25
01110	0.22	0.30	0.43	0.39	0.46	0.24	0.30	0.43
01111	0.16	0.26	0.14	0.52	0.22	0.38	0.30	0.40
10000	0.28	0.34	0.32	0.21	0.41	0.13	0.28	0.14
10001	0.38	0.32	0.31	0.36	0.45	0.39	0.28	0.18
10010	0.23	0.26	0.34	0.37	0.25	0.97	0.64	0.35
10011	0.44	0.75	0.86	0.38	0.42	0.39	0.21	0.21
10100	0.21	0.18	0.32	0.37	0.26	0.81	0.79	0.30
10101	0.43	0.97	0.95	0.48	0.43	0.42	0.30	0.17
10110	0.27	0.51	0.49	0.62	0.50	0.39	0.36	0.19
10111	0.29	0.36	0.31	0.36	0.26	0.40	0.42	0.56
11000	0.21	0.23	0.23	0.19	0.27	0.43	0.47	0.21
11001	0.22	0.37	0.67	0.51	0.69	0.68	0.46	0.22
11010	0.23	0.41	0.59	0.44	0.77	0.50	0.22	0.11
11011	0.46	0.39	0.31	0.19	0.25	0.33	0.47	0.73
11100	0.14	0.20	0.23	0.16	0.30	0.38	0.43	0.26
11101	0.33	0.26	0.27	0.12	0.32	0.23	0.46	0.87
11110	0.09	0.13	0.22	0.35	0.11	0.11	0.29	0.81
11111	0.10	0.26	0.12	0.64	0.19	0.46	0.42	0.35

TABLE 2-3 Statistics of 4 and 8-bit patterns for Song mode ADM image of the GIRL with step size added to LSB of the estimate. (%)

P_0P_1 \ P_2P_3	00	01	10	11
00	5.97	2.81	6.29	7.34
01	6.69	9.66	10.18	3.23
10	2.89	10.14	9.72	6.63
11	7.06	6.28	3.40	1.72

$P_0P_1P_2P_3P_4$ \ $P_5P_6P_7$	000	001	010	011	100	101	110	1111
00000	4.12	0.07	0.15	0.09	0.26	0.06	0.10	0.01
00001	0.34	0.18	0.09	0.04	0.17	0.11	0.04	0.01
00010	0.37	0.24	0.17	0.20	0.06	0.20	0.21	0.08
00011	0.10	0.44	0.44	0.13	0.13	0.08	0.02	0.01
00100	0.26	0.20	0.23	0.15	0.16	0.29	0.60	0.50
00101	0.08	0.20	0.63	1.07	0.92	0.87	0.28	0.08
00110	0.08	0.44	1.14	1.64	1.59	0.87	0.23	0.12
00111	0.19	0.41	0.18	0.18	0.04	0.10	0.11	0.03
01000	0.45	0.17	0.21	0.09	0.28	0.19	0.37	0.12
01001	0.15	0.31	0.75	1.00	1.55	0.78	0.76	0.08
01010	0.09	0.25	0.80	0.74	0.47	1.43	1.27	0.28
01011	0.20	1.61	1.26	0.23	0.35	0.23	0.20	0.15
01100	0.10	0.09	0.26	0.39	0.23	1.66	2.28	0.39
01101	0.33	1.83	1.28	0.23	0.26	0.33	0.15	0.13
01110	0.05	0.29	0.42	0.41	0.15	0.25	0.20	0.27
01111	0.03	0.11	0.08	0.36	0.06	0.18	0.14	0.07
10000	0.06	0.16	0.21	0.08	0.26	0.08	0.10	0.02
10001	0.24	0.21	0.23	0.20	0.42	0.44	0.21	0.05
10010	0.11	0.12	0.26	0.24	0.22	1.42	1.66	0.33
10011	0.34	2.33	1.89	0.23	0.47	0.28	0.12	0.12
10100	0.08	0.12	0.23	0.30	0.21	1.31	1.70	0.23
10101	0.29	1.30	1.15	0.95	0.89	0.76	0.26	0.10
10110	0.10	0.26	0.78	1.34	0.93	0.76	0.34	0.20
10111	0.14	0.35	0.20	0.31	0.09	0.38	0.23	0.16
11000	0.06	0.05	0.10	0.03	0.15	0.24	0.49	0.09
11001	0.12	0.24	0.78	1.21	1.54	1.16	0.48	0.10
11010	0.10	0.23	0.96	1.20	0.99	0.52	0.23	0.09
11011	0.16	0.31	0.31	0.22	0.15	0.28	0.28	0.25
11100	0.04	0.04	0.07	0.20	0.12	0.40	0.45	0.19
11101	0.06	0.17	0.25	0.11	0.19	0.20	0.28	0.41
11110	0.01	0.02	0.08	0.22	0.04	0.07	0.17	0.46
11111	0.01	0.12	0.04	0.23	0.06	0.11	0.05	0.01

TABLE 2-4 Statistics of 4 and 8-bit patterns for Song mode ADM image of the GIRL with step size added to 2nd LSB of the estimate. (%)

show that the leaky integrator does in fact produce larger step size the step statistics were calculated for a Song mode ADM with and without a leaky integrator. The step statistics are presented in TABLE 2-5. We notice that in general, we have a greater probability of being in the lower step sizes. In particular, the encoder without a leaky integrator has a combined probability of approximately 60% of being in the two lowest step size. In typical applications a leaky integrator is not needed since typical error rates are 10^{-5} or less and is used only in high noise environments. The introduction of the leaky integrator, which in effect introduces small perterbations on the estimate, is to make the average step size somewhat larger. We notice from TABLE 2-5 that some step size have zero probability of occurring. This a is direct result of how the step size is generated. For example, in order to generate a step size of $|Y(k+1)|=8$ we must find a step size that satisfies

$$8 = ||Y(k)|[E(k+1)+0.5E(k)]| \tag{2.4-1}$$

for any combination of E(k+1) and E(k) which yield an integer value for Y(k). The only solution to this equation is Y(k)=16, E(k+1)=-1, and E(k)=1. However, this solution for Y(k) is not allowed.

IMAGE	BOY				GIRL			
LEAK	NO		YES		NO		YES	
STEP SIZE	LSB	2nd LSB	LSB	2nd LSB	LSB	2nd LSB	LSB	2nd LSB
1	27.32	36.71	16.44	30.4	29.39	38.48	15.2	30.61
2	31.18	39.07	22.75	35.02	33.08	40.56	21.84	35.33
3	15.66	10.62	19.72	17.33	15.25	9.72	21.08	18.87
4	8.33	4.71	15.47	7.65	7.71	3.76	17.08	7.03
5	0.25	0.03	0.21	0.02	0.16	0.03	0.12	0.02
6	5.23	2.64	10.43	3.32	4.57	1.83	11.41	2.47
7	1.21	0.24	1.16	0.22	0.82	0.25	0.79	0.22
8	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
9	2.80	1.20	5.47	1.21	2.28	0.80	5.42	0.79
10	0.32	0.03	0.27	0.02	0.17	0.03	0.13	0.02
11	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
12	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
13	1.59	0.45	2.09	0.41	1.14	0.31	1.54	0.30
14	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
15	6.11	4.31	5.99	4.39	5.44	4.23	5.40	4.33

TABLE 2-5 Step Size Statistics (ADM)

We notice from the pattern statistics that the combined total of the four rotations of the 1100 pattern constitute approximaely 40% of the data. Individually coding each 4 or 8 bit data block does not provide any data compression. If, on the other hand, when the encoder produces any particular rotation of the steady state pattern it does so in long sequences, then run-length coding can be employed to indicate which particular pattern is being generated and the length of the sequence. In TABLE 2-6 are tabulated the run-length statistics for the Song mode ADM encoder with the step size added to the LSB and 2^{nd} LSB. We notice that each pattern usually occurs once and that no particular pattern sustains runs of sufficient length and frequency to make run length coding profitable.

To verify the above observations, that the possible data compression is minimal, the entropy of the ADM signals was calculated. TABLE 2.7 presents the entropy, efficiency and redundancy of the ADM for the two input video pictures. We notice that the ADM is very efficient and hardly any redundancy exists in the code. The small redundancy could be reduced by using a Huffman code, but the possible data compression would be minimal. TABLE 2.8 represents the possible data compression possible if the data of a video line is partitioned into 4 and 8-bit blocks and Huffman

	PATTERN	RUN-LENGTH 1	2	3	4	5	6	7	8	9	10	11
	1100	5565	600	55	8	6	0	0	0	0	0	0
LSB	0110	6793	741	92	17	0	0	0	0	0	0	0
	0011	5515	668	89	8	1	1	1	0	0	0	0
	1001	6569	777	72	6	3	0	0	0	0	0	0
	1100	5195	1008	227	63	17	10	4	1	1	0	0
2nd LSB	0110	7262	1519	291	125	45	12	6	2	0	0	0
	0011	5291	1050	247	80	23	10	1	1	0	1	0
	1001	7289	1461	369	68	36	16	7	2	2	1	1

TABLE 2-6 Run-length statistics for the Song mode ADM image of the GIRL.

Image	Length	Entropy	Efficiency	Redundancy
BOY	4 bits	3.904	97.61%	2.39%
	8 bits	7.621	95.26%	4.74%
GIRL	4 bits	3.847	96.17%	3.83%
	8 bits	7.403	92.54%	7.46%

TABLE 2-7 Present ADM digital source.

Image	Length	Compression	Efficiency	Redundancy
BOY	4 bits	1.35%	99.84%	0.16%
	8 bits	4.40%	99.62%	0.37%
GIRL	4 bits	2.72%	98.89%	1.01%
	8 bits	7.08%	98.63%	1.37%

TABLE 2-8 Huffman coding of present ADM digital source.

coded. We notice that the possible data compression is minimal if any since we would have to transmit the Huffman code used not to mention the difficulty of its implementation.

2.5.0 FIELD INTERPOLATION

A video image is correlated in both the horizontal as well as the vertical direction. The ADM encoder takes advantage of the correlation in the horizontal direction by calculating the present estimate based on the value of the present and previous values of the input signal. Data compression can be achieved by exploiting the correlation in the vertical direction. Since a line of video is more highly correlated to its two adjacent lines, that is the line directly above and below, we can obtain an estimate for the line of video by interpolating its two adjacent lines. The worst case estimate for a line of video would occur whenever the estimated line corresponds to a horizontal edge in the image, but this situation usually does not occur even once in a frame of video.

A television picture is composed of two fields that are interlaced. In fig. 2.11 we show a frame of video

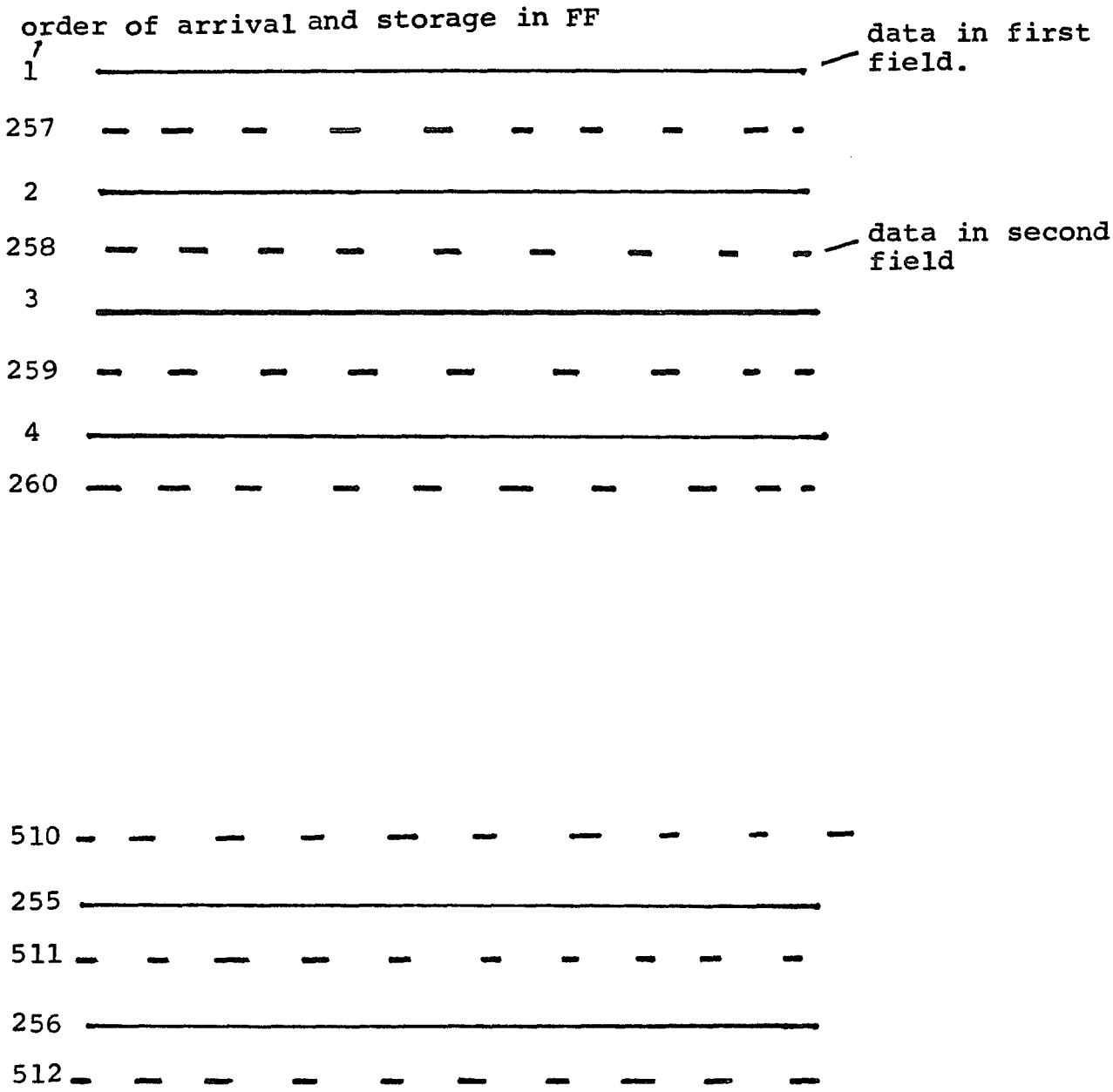


Fig. 2.11 Storage of video frame in FRAME FREEZE.

where each line is numbered in the sequence that they are generated by the television camera. Since the Frame Freeze unit, which is used to store a frame of video for transmission over the computer network, stores a television picture as 512 lines, the first 256 lines in fig. 2.11 corresponds to the first field while the second 256 lines corresponds to the second field of the frame of video. We also notice from fig 2.11 that the two adjacent lines for a line of video in the second field corresponds to lines in the first field. As an example, suppose that we wish to estimate line 258 in fig. 2.11 then we would use lines 2 and 3 to form the estimate.

We can achieve 50% data compression by transmitting only one field and at the receiver estimate the second field by interpolating the received field. The interpolation for the missing lines entails generating the average of the two ajacent lines and can be done digitally using a PCM store. Since this investigation deals with the transmission of video images through a computer network the receiving computer will have to contain a programmed encoder-decoder pair in order to generate the analog estimates used to perform the interpolation and then obtain the resulting E(k) data bits.

The results of transmitting only one field and obtaining the second field by averaging the analog samples

Fig. 2.12 Field interpolated ADM input images.

of the corresponding lines in the first field are shown in fig. 2.12. In order to compare the result of field interpolating an ADM image we process the input images by a program decoder-encoder pair. We notice that the resulting pictures contain less edge busyness and result in a better looking picture. The edge busyness is reduced because the edge busyness of the interpolated field represents the average of the two lines in the first frame. This effect is produced because the interpolated lines, instead of contributing to the edge busyness, tends to smooth it out. We also notice that the processed images contain more graininess. Instead of the smooth skin on the GIRL we now observe a slight texture. The graininess introduce is not very noticeable unless you compare the process pictures with the originals.

Conclusion

It is evident from the results of this section that removal of any particular pattern from the E(k) data bit stream produced by the present Song ADM algorithm will not produce any profitable data compression. In SECTION 2.5 we showed that a 50% data compression can be achieved if only one field of the frame of video is transmitted. At the receiver we obtain the second field by interpolating the

received field. Results with actual video images indicate that the resulting image at the receiver contains less edge busyness and "looks" better. Also, with two fields of storage each even field can be estimated using the two adjacent odd fields. Each line in the even field can be estimated using the two adjacent lines in each odd field, i.e. a total of 4 lines. Each sample on the even fields will thus be estimated using an 8 point interpolation. This technique does not include the possible redundancy removal by using motion prediction. If we wish to achieve data compression by operating on the data it becomes necessary to look at other algorithms that will produce sufficient steady state patterns to make coding profitable. Such an algorithm is the Modified ADM.

MODIFIED ADM

3.0.0 INTRODUCTION

The object of this study is to investigate the possible data compression that can be achieve by encoding a video delta modulated signal. The overall objective is to investigate the feasibility of constructing a system that will capture a frame of video and then have a small computer or built-in microprocessor compress the video data and transmit the information through a computer network.

In the previous chapter we investigated the possible compression that could be achieved using the present ADM hardware. It has been shown that the present ADM algorithm does not produce any steady state pattern in sufficient numbers to merit coding. It was shown that using Song's algorithm a 0011 steady state pattern will be produced for any constant input signal level. However, the steady state patterns are approximatly equally distributed among the four possible pattern rotations. We would thus like to produce a particular bit pattern that begins at predetermined locations and thus reduce the overhead of having to specify what particular rotation is being removed. The Modified ADM (MADM) is a modification of

Song's algorithm that produces a particular steady state pattern whenever the input signal is approximately constant.

3.1.0 Experimental Apparatus

In order to investigate other possible algorithms with the existing hardware at our disposal, the system shown in fig. 3.1 was simulated on the PDP-11. A brief description of the system is given below;

File-1

File-1 is a data file on disk memory. It contains the E(k) data produced by the present ADM and which is transferred to the computer via the FF.

ADM Decoder

This module simulates the hardware ADM decoder and is used to produce an estimate analog signal.

Filters

Both filters are used to smooth the analog samples produced by the ADM decoder and MADM decoder. The analog samples are smoothed by taking a sliding average of the present and past three samples. The equations of the filter are given below:

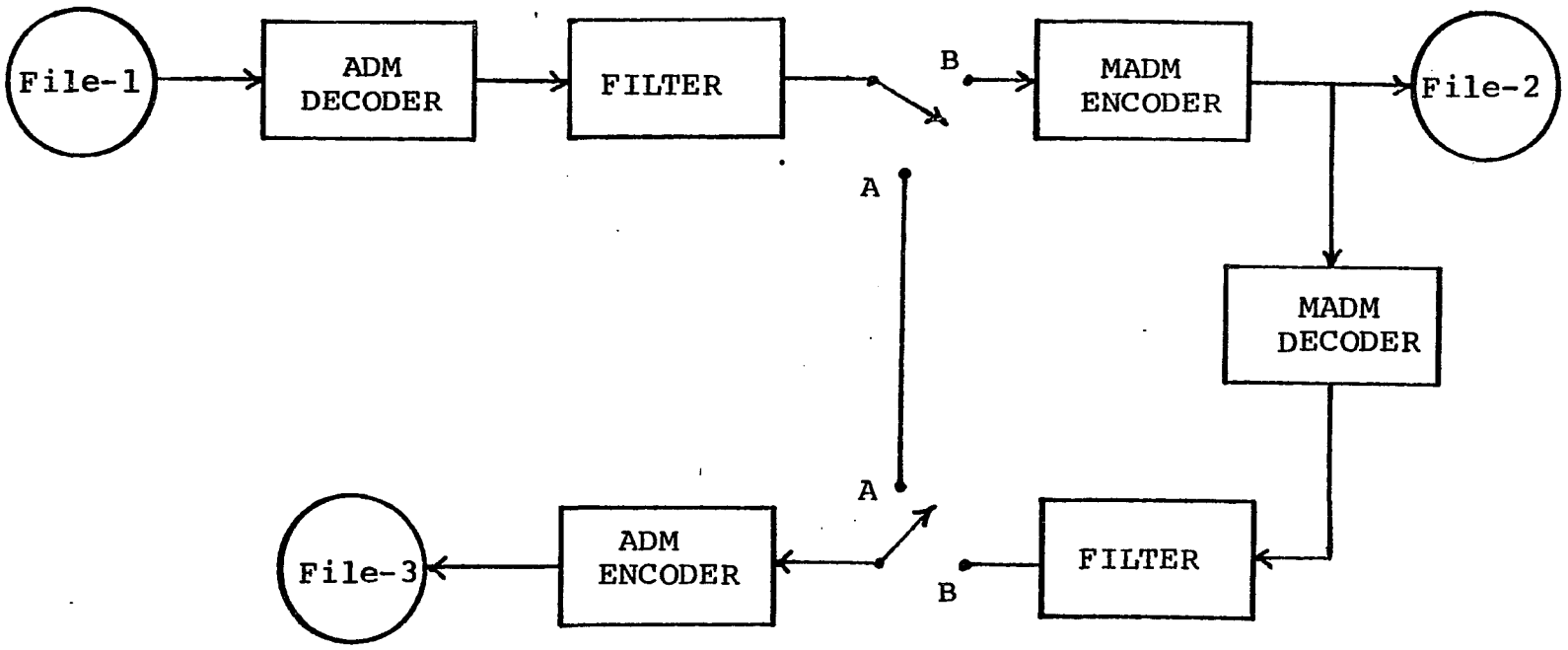


Fig. 3.1 MADM Simulation System.

3.1-1 $S_o(1)=S_i(1)/4$

3.1-2 $S_o(2)=[S_i(1)+S_i(2)]/4$

3.1-3 $S_o(3)=[S_i(1)+S_i(2)+S_i(3)]/4$

3.1-4 $S_o(N)=[S_i(N)+S_i(N-1)+S_i(N-2)+S_i(N-3)]/4$

;N=4,5,...,928

MADM Encoder

The MADM encoder is a modified ADM which produces a particular pattern whenever the input signal is approximately constant.

File-2

File-2 is a data file on disk memory. It contains the E(k) data bits generated by the MADM.

MADM Decoder

This module is used to decode the E(k) data bits generated by the MADM and produces analog samples.

File-3

File-3 is a data file that is stored in disk memory. It contains the processed image. This file can be transferred to the FF for display on the monitor.

Switch

This switch is used to obtain a TEST image, when the switch is close to position A, which can be used to compare the MADM image stored on File-3.

3.2.0 MADM ENCODER

Basically the aim of the MADM algorithm is to "force" a predetermined steady state pattern whenever the input signal is approximately constant. We saw, TABLE 2-5, that the step size has a high probability of being under three quantization levels. In addition, from the response of the ADM to an input square wave we notice that in steady state the encoder's step size will remain in the two lowest step size. The aim of the MADM algorithm is thus to substitute the normal data bits that are being produced by the encoder when it is tracking a constant input by a suitable replacement that can later be removed to achieve data compression. The steady state pattern chosen must maintain the estimate close to the input signal level.

The MADM must be able to provide the following:

1) Increase the number of steady state patterns sufficiently to make coding profitable.

2) Produce the steady state pattern so that it coincides with the data blocks into which a video line is partitioned into. This will not necessitate the transmission of the steady state pattern address.

3) Provide the above two conditions but at the same time maintain a useful image.

The equations for the MADM are given below:

$$E(k+1)=\begin{cases} P_0, \text{ when } F=1, \text{and } |S(k)-X(k)| \leq D_1; \text{ or} \\ \quad F=1, \ |S(k)-X(k)| \leq D_2 \text{ and } E(k)=P_{N-1} \\ P_n, \ n=1,2,..,N-1 \text{ when} \\ \quad F\neq 1, \ |S(k)-X(k)| \leq D_2 \text{ and } E(k)=P_{n-1} \\ SGN[S(k+1)-X(k+1)], \quad \text{otherwise} \end{cases} \tag{3.2-1}$$

3.2-2

$$X(k+1) = X(k) + Y(k+1)$$

3.2-3

$$Y(k+1) = \begin{cases} |Y(k)|[E(k)+0.5E(k-1)] & Y_0 \leq |Y(k+1)| \leq 15Y_0 \\ 2Y_0 & |Y(k)| < 2Y_0 \\ 15Y_0 E(k) & |Y(k)| > 10Y_0 \text{ and } E(k)=E(k-1) \end{cases}$$

where $E(k)$ = the MADM output.

$S(k)$ = the input analog signal.

$X(k)$ = the MADM estimate.

$Y(k)$ = the MADM step size.

Y_0 = the minimum MADM step size.

$P_0P_1..P_{N-1}$ = predetermine forced steady state pattern.

F = the indicator the $E(k)$ coincides with the beginning of a data block.

D_1 = the initial aperture centered on $S(k)$.

D_2 = the aperture on $X(k)$ when MADM reverts to Song mode operation.

A block diagram of the MADM is shown in fig. 3.2. In the above equations the N-bit steady state pattern is represented by Pi, i=0,1,..,N-1. The parameter F_1 indicates when E(k) coincides with the beginning of an N-bit data block. In this manner all steady state patterns that are to be generated by the MADM must originate on one of the 928/N N-bit data blocks that a video line is partitioned into. In order for the MADM to start producing the steady state pattern F_1 must be equal to 1.

Another constraint that must be met before the MADM can start to output the steady state pattern is that the magnitude of the difference between the previous estimate and sample must not be greater than D_1. The previous estimate was chosen to relax timing constraints that will otherwise be met when implementing a real system. There is no noticeable difference in the resulting image if the present values are used in the simulations. Once the MADM is outputting the steady state pattern, it will continue to do so until the magnitude of the difference between the estimate and sample exceeds D_2.

The response of the MADM to an input signal is illustrated in fig. 3.3 where, for the purpose of illustration, the steady state pattern is set to 10110010. The manner in which the MADM operates is as follows: Whenever the MADM encoder is tracking a signal having a

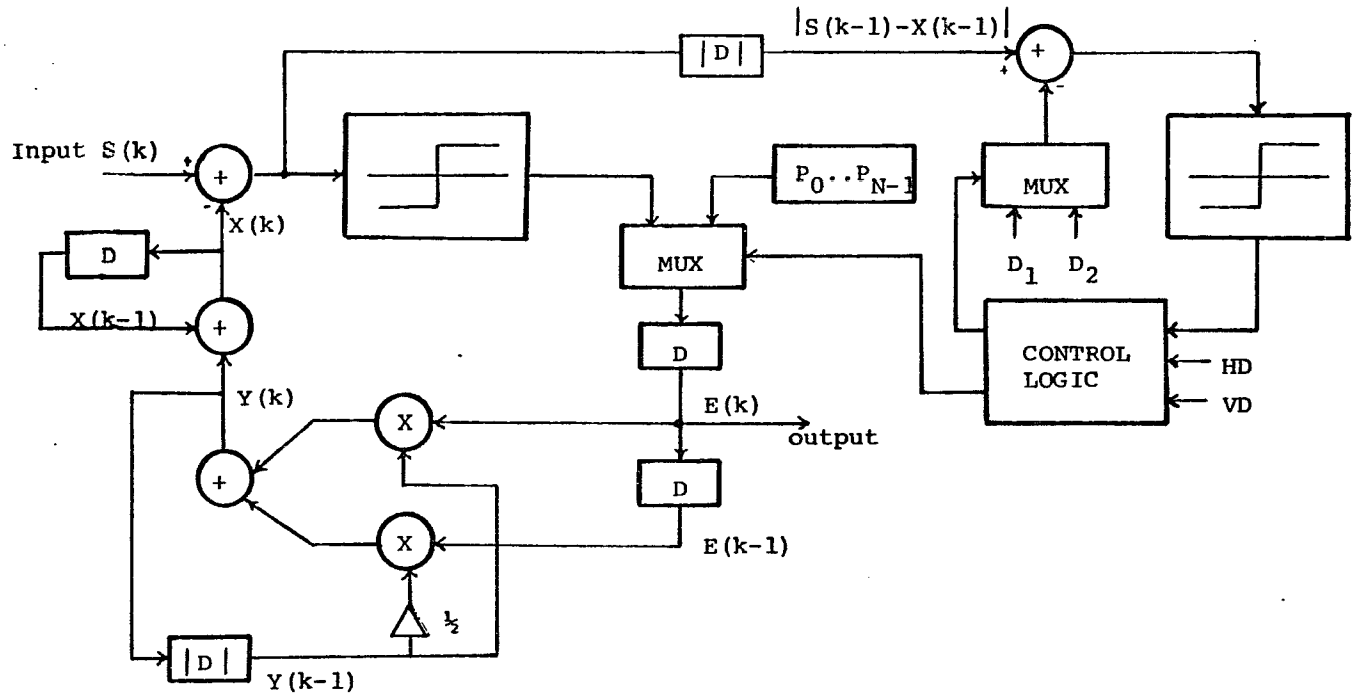


Fig. 3.2 Digital modified adaptive delta modulator.

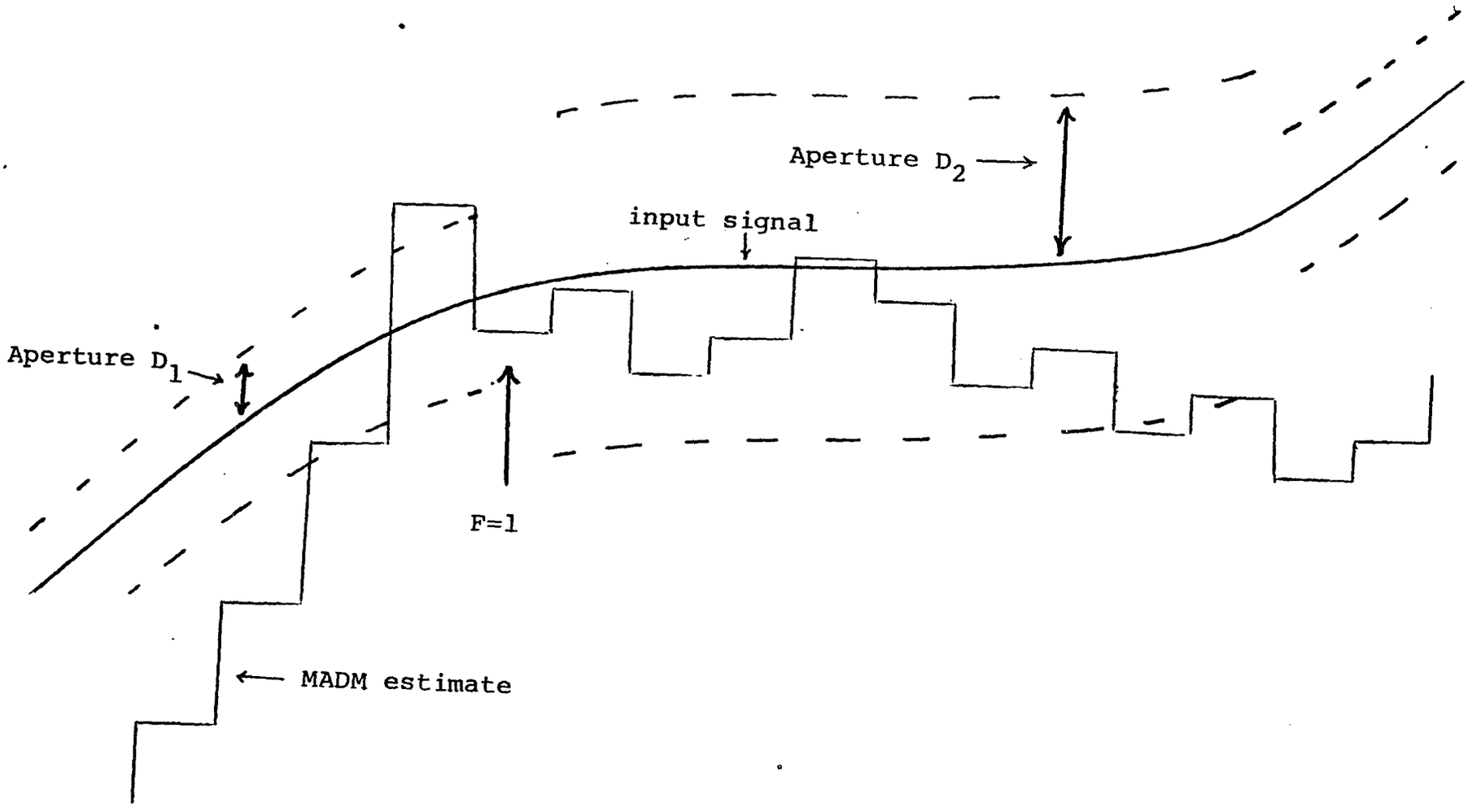


Fig. 3.3 MADM response to input signal.

large slope, the encoder will operate as a Song mode ADM. Whenever the MADM encoder's estimate is close to the input signal, specifically within an aperture of size $2D_1$ centered about the input signal, the MADM encoder will enter into the steady state mode providing the signal F_1 is at logic "1". The initial aperture size D_1 is chosen to insure that the estimate signal is indeed close to the input signal. When the two above conditions are met, a second aperture of size $2D_2$ is opened about the input signal and the encoder begins to produce the predetermind steady state pattern. The value D_2 is chosen to represent the maximum difference between the input signal and the encoder's estimate that will be tolerated before the MADM reverts to operating in the Song mode. Thus, so long as the estimate stays within the aperture $2D_2$ of the input signal, the MADM encoder will continue to produce the predetermined steady state pattern. In order to keep the decoder synchronized to the encoder the E(k) data bits that are being transmitted are also being used to generate the encoder's step size.

3.3.0 MADM RESPONSE TO INPUT SQUARE WAVE

We noticed in SECTION 2.4 that the Song mode ADM will produce the 1100 steady state pattern whenever the input signal is constant. Thus, the most logical "forced"

steady state pattern to choose is one of the four possible rotations of 1100. The results of simulating the MADM with the parameters:

3.3-1 $D_1=5$

3.3-2 $D_2=10$

3.3-3 STEADY STATE PATTERN: $\begin{cases} 11001100 \\ 10011001 \end{cases}$

3.3-4 STEP ADDED TO $\begin{cases} \text{LSB OF } X(k) \\ 2^{nd} \text{ OF } X(k) \end{cases}$

are illustrated in figs. 3.4 thru 3.7 for an input square wave signal. Note that for the purpose of these illustrations the pattern length is set to 8-bits. When the estimate is close to the input signal a predetermine steady state pattern is produced starting at a specified location. Since there are four possible rotation of the steady state pattern about three quarter of the time the "forced" steady state pattern will not correspond to the steady state pattern normally produced by the ADM. Thus, approximately three quarter of the time the D-C level established by the "forced" steady state pattern will either be above or below the level that normally would be established by the Song mode ADM. We notice that when the step size is added to the LSB of the estimate the MADM will track the input signal closer and produce more steady state

SIMULATING MADM: INITIAL APERATURE = 05
MAXIMUM APERATURE = 10

NO LEAKY INTEGRATOR
ENCODER Y(K) ADDED TO LSB OF X(K)
STEADY STATE PATTERN: 11001100

K	S(K)	X(K)	E(K)	Y(K)
01	300	007	1	07
02	300	021	1	12
03	300	040	1	17
04	300	057	1	17
05	300	076	1	17
06	300	115	1	17
07	300	134	1	17
08	300	153	1	17
09	300	172	1	17
10	300	211	1	17
11	300	230	1	17
12	300	247	1	17
13	300	266	1	17
14	300	305	1	17
15	300	276	0	07
16	300	301	1	03
17	300	305	1	04
18	300	313	1	06
19	300	310	0	03
20	300	304	0	04
21	300	276	0	06
22	300	301	1	03
23	300	300	0	01
24	300	276	0	02
25	300	277	1	01
26	300	301	1	02
27	300	300	0	01
28	300	276	0	02
29	300	277	1	01
30	300	301	1	02
31	300	300	0	01
32	300	276	0	02
33	300	277	1	01
34	300	301	1	02
35	300	300	0	01
36	300	276	0	02
37	300	277	1	01
38	300	301	1	02
39	300	300	0	01
40	300	276	0	02
41	300	277	1	01
42	300	301	1	02
43	300	300	0	01
44	300	276	0	02
45	300	277	1	01
46	300	301	1	02
47	300	300	0	01
48	300	276	0	02
49	300	277	1	01
50	300	301	1	02
51	100	300	0	01
52	100	276	0	02
53	100	273	0	03
54	100	267	0	04
55	100	261	0	06
56	100	250	0	11
57	100	233	0	15
58	100	214	0	17
59	100	175	0	17
60	100	156	0	17
61	100	137	0	17
62	100	120	0	17
63	100	101	0	17
64	100	062	0	17
65	100	071	1	07
66	100	103	1	12
67	100	076	0	05
68	100	100	1	02
69	100	077	0	01
70	100	101	1	02
71	100	100	0	01
72	100	076	0	02
73	100	077	1	01
74	100	101	1	02
75	100	100	0	01
76	100	076	0	02
77	100	077	1	01
78	100	101	1	02
79	100	100	0	01
80	100	076	0	02
81	100	077	1	01
82	100	101	1	02
83	100	100	0	01
84	100	076	0	02
85	100	077	1	01

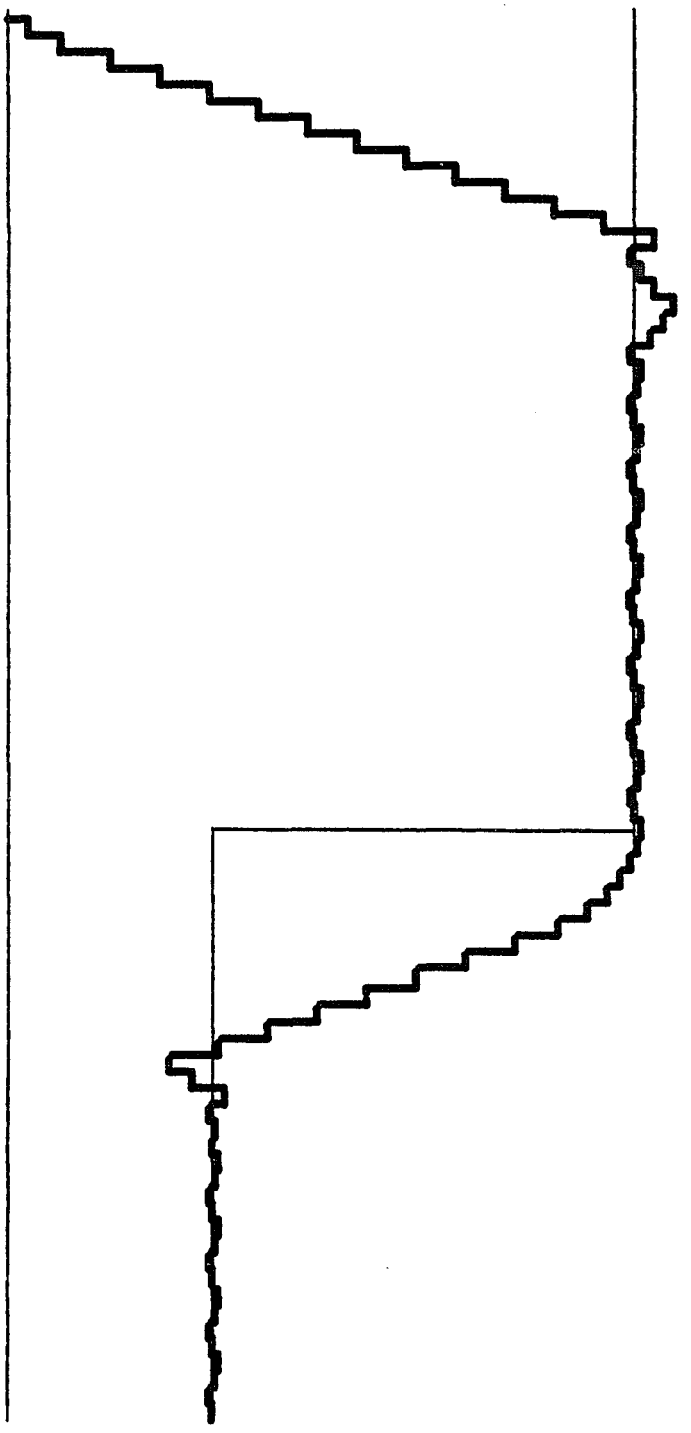


Fig. 3.4 MADM Response-1 (LSB)

```
SIMULATING MADM: INITIAL APERATURE = 05
                 MAXIMUM APERATURE = 10

NO LEAKY INTEGRATOR
ENCODER Y(K) ADDED TO 2ND LSB OF X(K)
STEADY STATE PATTERN: 11001100
```

K	S(K)	X(K)	E(K)	Y(K)
01	300	016	1	07
02	300	042	1	12
03	300	100	1	17
04	300	136	1	17
05	300	174	1	17
06	300	232	1	17
07	300	270	1	17
08	300	326	1	17
09	300	310	0	07
10	300	264	0	12
11	300	276	1	05
12	300	314	1	07
13	300	306	0	03
14	300	276	0	04
15	300	302	1	02
16	300	300	0	01
17	300	304	1	02
18	300	312	1	03
19	300	310	0	01
20	300	304	0	02
21	300	276	0	03
22	300	300	1	01
23	300	274	0	02
24	300	276	1	01
25	300	302	1	02
26	300	310	1	03
27	300	306	0	01
28	300	302	0	02
29	300	304	1	01
30	300	310	1	02
31	300	306	0	01
32	300	302	0	02
33	300	304	1	01
34	300	310	1	02
35	300	306	0	01
36	300	302	0	02
37	300	304	1	01
38	300	310	1	02
39	300	306	0	01
40	300	302	0	02
41	300	304	1	01
42	300	310	1	02
43	300	306	0	01
44	300	302	0	02
45	300	304	1	01
46	300	310	1	02
47	300	306	0	01
48	300	302	0	02
49	300	304	1	01
50	300	310	1	02
51	100	306	0	01
52	100	302	0	02
53	100	274	0	03
54	100	264	0	04
55	100	250	0	06
56	100	226	0	11
57	100	174	0	15
58	100	136	0	17
59	100	100	0	17
60	100	042	0	17
61	100	060	1	07
62	100	104	1	12
63	100	072	0	05
64	100	076	1	02
65	100	104	1	03
66	100	114	1	04
67	100	110	0	02
68	100	102	0	03
69	100	072	0	04
70	100	076	1	02
71	100	104	1	03
72	100	102	0	01
73	100	106	1	02
74	100	114	1	03
75	100	112	0	01
76	100	106	0	02
77	100	100	0	03
78	100	070	0	04
79	100	074	1	02
80	100	102	1	03
81	100	112	1	04
82	100	106	0	02
83	100	100	0	03
84	100	070	0	04
85	100	074	1	02

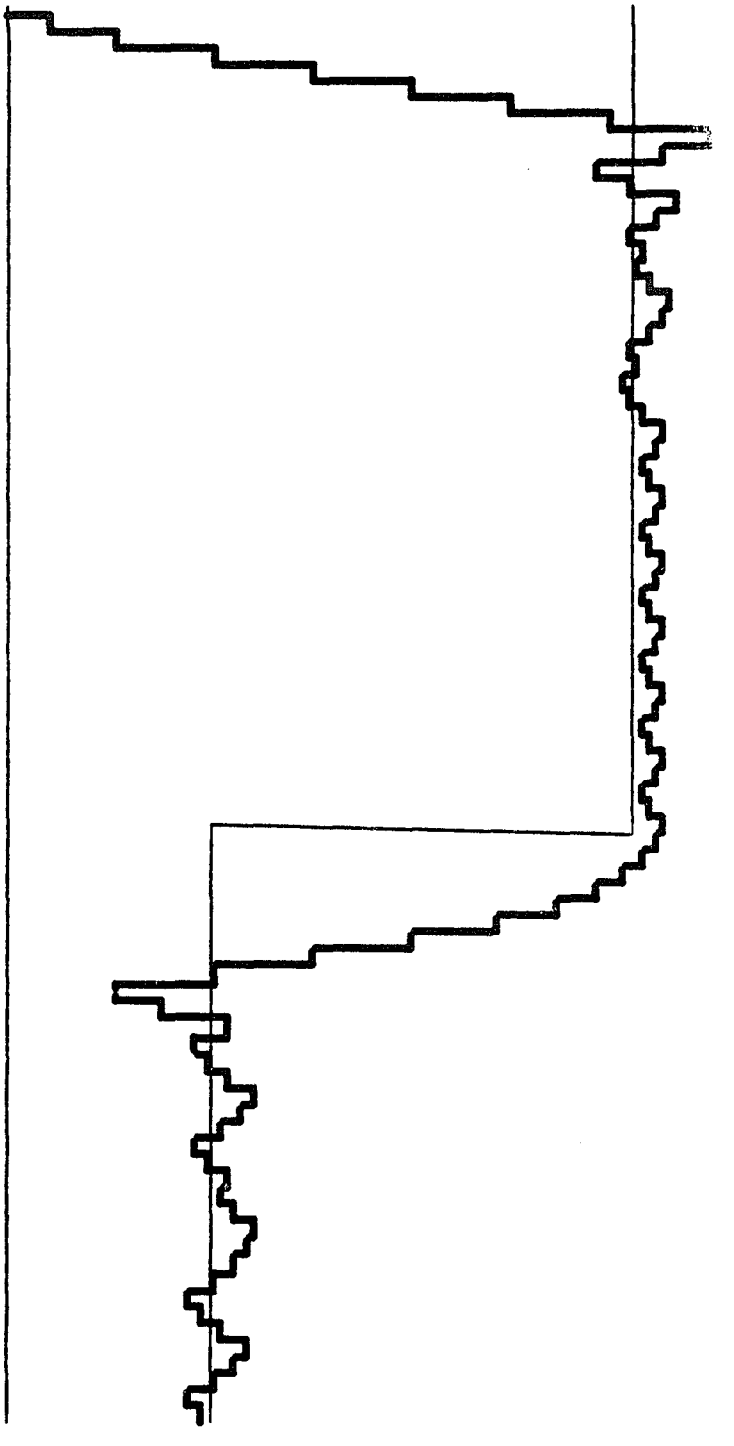


Fig. 3.5 MADM Response-1 (2^{nd} LSB)

patterns. When the steady state pattern is 11001100 and the step size is added to the 2^{nd} LSB of the estimate the encoder will react faster to a step input but the encoder tends to generate a steady state level which is larger than the input signal because the first two bits of the steady state pattern are ones. Since the MADM will tend to generate a higher steady state estimate, occasionally the MADM will revert to operating in the Song mode until the estimate is once again within D_1 of the input signal. An input video signal that on the monitor screen appears to be of a constant grey level will in fact represent a voltage level that contains small perterbations. Thus, the effect of the MADM generating a higher steady state estimate is that the perterbations on the input signal, when the MADM is operationg in the steady state mode, will cause the MADM to revert to operating in the Song mode. This effect can be reduced by making the steady state pattern 10011001 and is illustrated in figs. 3.6 and 3.7. We notice from the response of the MADM that the particular steady state pattern will always begin on an 8-bit boundary for an eight bit pattern. From the above we see that the steady state pattern can be generated in sufficient quantity to warrant coding.

```
SIMULATING MADM: INITIAL APERATURE = 05
                 MAXIMUM APERATURE = 10

NO LEAKY INTEGRATOR
ENCODER Y(K) ADDED TO LSB OF X(K)
STEADY STATE PATTERN: 10011001
```

K	S(K)	X(K)	E(K)	Y(K)
0 1	3 0 0	0 0 7	1	0 7
0 2	3 0 0	0 2 1	1	1 2
0 3	3 0 0	0 4 0	1	1 7
0 4	3 0 0	0 5 7	1	1 7
0 5	3 0 0	0 7 6	1	1 7
0 6	3 0 0	1 1 5	1	1 7
0 7	3 0 0	1 3 4	1	1 7
0 8	3 0 0	1 5 3	1	1 7
0 9	3 0 0	1 7 2	1	1 7
1 0	3 0 0	2 1 1	1	1 7
1 1	3 0 0	2 3 0	1	1 7
1 2	3 0 0	2 4 7	1	1 7
1 3	3 0 0	2 6 6	1	1 7
1 4	3 0 0	3 0 5	1	1 7
1 5	3 0 0	2 7 6	0	0 7
1 6	3 0 0	3 0 1	1	0 3
1 7	3 0 0	3 0 5	1	0 4
1 8	3 0 0	3 0 3	0	0 2
1 9	3 0 0	3 0 0	0	0 3
2 0	3 0 0	3 0 1	1	0 1
2 1	3 0 0	3 0 3	1	0 2
2 2	3 0 0	3 0 2	0	0 1
2 3	3 0 0	3 0 0	0	0 2
2 4	3 0 0	3 0 1	1	0 1
2 5	3 0 0	3 0 3	1	0 2
2 6	3 0 0	3 0 2	0	0 1
2 7	3 0 0	3 0 0	0	0 2
2 8	3 0 0	3 0 1	1	0 1
2 9	3 0 0	3 0 3	1	0 2
3 0	3 0 0	3 0 2	0	0 1
3 1	3 0 0	3 0 0	0	0 2
3 2	3 0 0	3 0 1	1	0 1
3 3	3 0 0	3 0 3	1	0 2
3 4	3 0 0	3 0 2	0	0 1
3 5	3 0 0	3 0 0	0	0 2
3 6	3 0 0	3 0 1	1	0 1
3 7	3 0 0	3 0 3	1	0 2
3 8	3 0 0	3 0 2	0	0 1
3 9	3 0 0	3 0 0	0	0 2
4 0	3 0 0	3 0 1	1	0 1
4 1	3 0 0	3 0 3	1	0 2
4 2	3 0 0	3 0 2	0	0 1
4 3	3 0 0	3 0 0	0	0 2
4 4	3 0 0	3 0 1	1	0 1
4 5	3 0 0	3 0 3	1	0 2
4 6	3 0 0	3 0 2	0	0 1
4 7	3 0 0	3 0 0	0	0 2
4 8	3 0 0	3 0 1	1	0 1
4 9	3 0 0	3 0 3	1	0 2
5 0	3 0 0	3 0 2	0	0 1
5 1	1 0 0	3 0 0	0	0 2
5 2	1 0 0	2 7 5	0	0 3
5 3	1 0 0	2 7 1	0	0 4
5 4	1 0 0	2 6 3	0	0 6
5 5	1 0 0	2 5 2	0	1 1
5 6	1 0 0	2 3 5	0	1 5
5 7	1 0 0	2 1 6	0	1 7
5 8	1 0 0	1 7 7	0	1 7
5 9	1 0 0	1 6 0	0	1 7
6 0	1 0 0	1 4 1	0	1 7
6 1	1 0 0	1 2 2	0	1 7
6 2	1 0 0	1 0 3	0	1 7
6 3	1 0 0	0 6 4	0	1 7
6 4	1 0 0	0 7 3	1	0 7
6 5	1 0 0	1 0 5	1	1 2
6 6	1 0 0	1 0 0	0	0 5
6 7	1 0 0	0 7 1	0	0 7
6 8	1 0 0	0 7 4	1	0 3
6 9	1 0 0	1 0 0	1	0 4
7 0	1 0 0	0 7 6	0	0 2
7 1	1 0 0	0 7 7	1	0 1
7 2	1 0 0	1 0 1	1	0 2
7 3	1 0 0	1 0 4	1	0 3
7 4	1 0 0	1 0 3	0	0 1
7 5	1 0 0	1 0 1	0	0 2
7 6	1 0 0	1 0 2	1	0 1
7 7	1 0 0	1 0 4	1	0 2
7 8	1 0 0	1 0 3	0	0 1
7 9	1 0 0	1 0 1	0	0 2
8 0	1 0 0	1 0 2	1	0 1
8 1	1 0 0	1 0 4	1	0 2
8 2	1 0 0	1 0 3	0	0 1
8 3	1 0 0	1 0 1	0	0 2
8 4	1 0 0	1 0 2	1	0 1
8 5	1 0 0	1 0 4	1	0 2

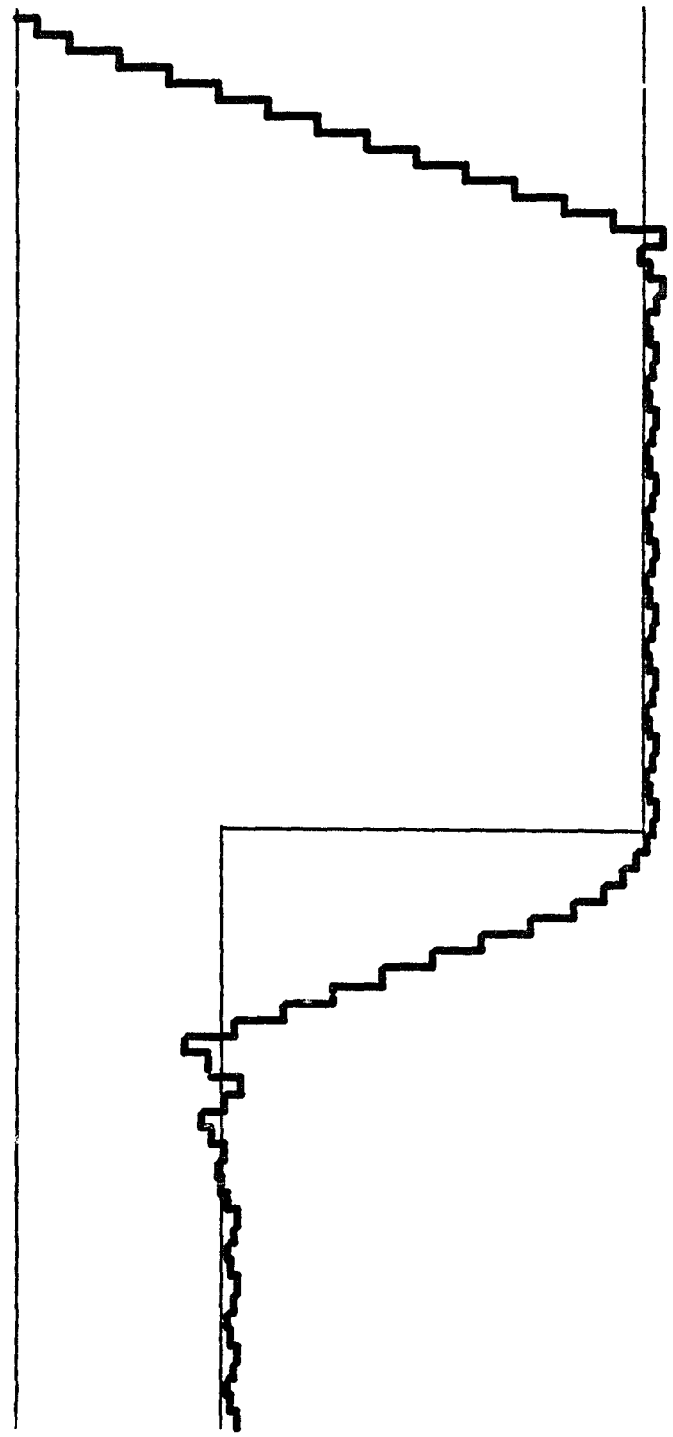


Fig. 3.6 MADM Response

```
SIMULATING MADM: INITIAL APERATURE = 05
                 MAXIMUM APERATURE = 10

NO LEAKY INTEGRATOR
ENCODER Y(K) ADDED TO 2ND LSB OF Y(K)
STEADY STATE PATTERN: 10011001
```

K	S(K)	X(K)	E(K)	Y(K)
0 1	3 0 0	0 1 6	1	0 7
0 2	3 0 0	0 4 2	1	1 2
0 3	3 0 0	1 0 0	1	1 7
0 4	3 0 0	1 3 6	1	1 7
0 5	3 0 0	1 7 4	1	1 7
0 6	3 0 0	2 3 2	1	1 7
0 7	3 0 0	2 7 0	1	1 7
0 8	3 0 0	3 2 6	1	1 7
0 9	3 0 0	3 1 0	0	0 7
1 0	3 0 0	2 6 4	0	1 2
1 1	3 0 0	2 7 6	1	0 5
1 2	3 0 0	3 1 4	1	0 7
1 3	3 0 0	3 0 6	0	0 3
1 4	3 0 0	2 7 6	0	0 4
1 5	3 0 0	3 0 2	1	0 2
1 6	3 0 0	3 0 0	0	0 1
1 7	3 0 0	3 0 4	1	0 2
1 8	3 0 0	3 0 2	0	0 1
1 9	3 0 0	2 7 6	0	0 2
2 0	3 0 0	3 0 0	1	0 1
2 1	3 0 0	3 0 4	1	0 2
2 2	3 0 0	3 0 2	0	0 1
2 3	3 0 0	2 7 6	0	0 2
2 4	3 0 0	3 0 0	1	0 1
2 5	3 0 0	3 0 4	1	0 2
2 6	3 0 0	3 0 2	0	0 1
2 7	3 0 0	2 7 6	0	0 2
2 8	3 0 0	3 0 0	1	0 1
2 9	3 0 0	3 0 4	1	0 2
3 0	3 0 0	3 0 2	0	0 1
3 1	3 0 0	2 7 6	0	0 2
3 2	3 0 0	3 0 0	1	0 1
3 3	3 0 0	3 0 4	1	0 2
3 4	3 0 0	3 0 2	0	0 1
3 5	3 0 0	2 7 6	0	0 2
3 6	3 0 0	3 0 0	1	0 1
3 7	3 0 0	3 0 4	1	0 2
3 8	3 0 0	3 0 2	0	0 1
3 9	3 0 0	2 7 6	0	0 2
4 0	3 0 0	3 0 0	1	0 1
4 1	3 0 0	3 0 4	1	0 2
4 2	3 0 0	3 0 2	0	0 1
4 3	3 0 0	2 7 6	0	0 2
4 4	3 0 0	3 0 0	1	0 1
4 5	3 0 0	3 0 4	1	0 2
4 6	3 0 0	3 0 2	0	0 1
4 7	3 0 0	2 7 6	0	0 2
4 8	3 0 0	3 0 0	1	0 1
4 9	3 0 0	3 0 4	1	0 2
5 0	3 0 0	3 0 2	0	0 1
5 1	1 0 0	2 7 6	0	0 2
5 2	1 0 0	2 7 0	0	0 3
5 3	1 0 0	2 6 0	0	0 4
5 4	1 0 0	2 4 4	0	0 6
5 5	1 0 0	2 2 2	0	1 1
5 6	1 0 0	1 7 0	0	1 5
5 7	1 0 0	1 3 2	0	1 7
5 8	1 0 0	0 7 4	0	1 7
5 9	1 0 0	1 1 2	1	0 7
6 0	1 0 0	1 0 4	0	0 3
6 1	1 0 0	0 7 4	0	0 4
6 2	1 0 0	1 0 0	1	0 2
6 3	1 0 0	0 7 6	0	0 1
6 4	1 0 0	1 0 2	1	0 2
6 5	1 0 0	1 1 0	1	0 3
6 6	1 0 0	1 0 6	0	0 1
6 7	1 0 0	1 0 2	0	0 2
6 8	1 0 0	1 0 4	1	0 1
6 9	1 0 0	1 1 0	1	0 2
7 0	1 0 0	1 0 6	0	0 1
7 1	1 0 0	1 0 2	0	0 2
7 2	1 0 0	1 0 4	1	0 1
7 3	1 0 0	1 1 0	1	0 2
7 4	1 0 0	1 0 6	0	0 1
7 5	1 0 0	1 0 2	0	0 2
7 6	1 0 0	1 0 4	1	0 1
7 7	1 0 0	1 1 0	1	0 2
7 8	1 0 0	1 0 6	0	0 1
7 9	1 0 0	1 0 2	0	0 2
8 0	1 0 0	1 0 4	1	0 1
8 1	1 0 0	1 1 0	1	0 2
8 2	1 0 0	1 0 6	0	0 1
8 3	1 0 0	1 0 2	0	0 2
8 4	1 0 0	1 0 4	1	0 1
8 5	1 0 0	1 1 0	1	0 2

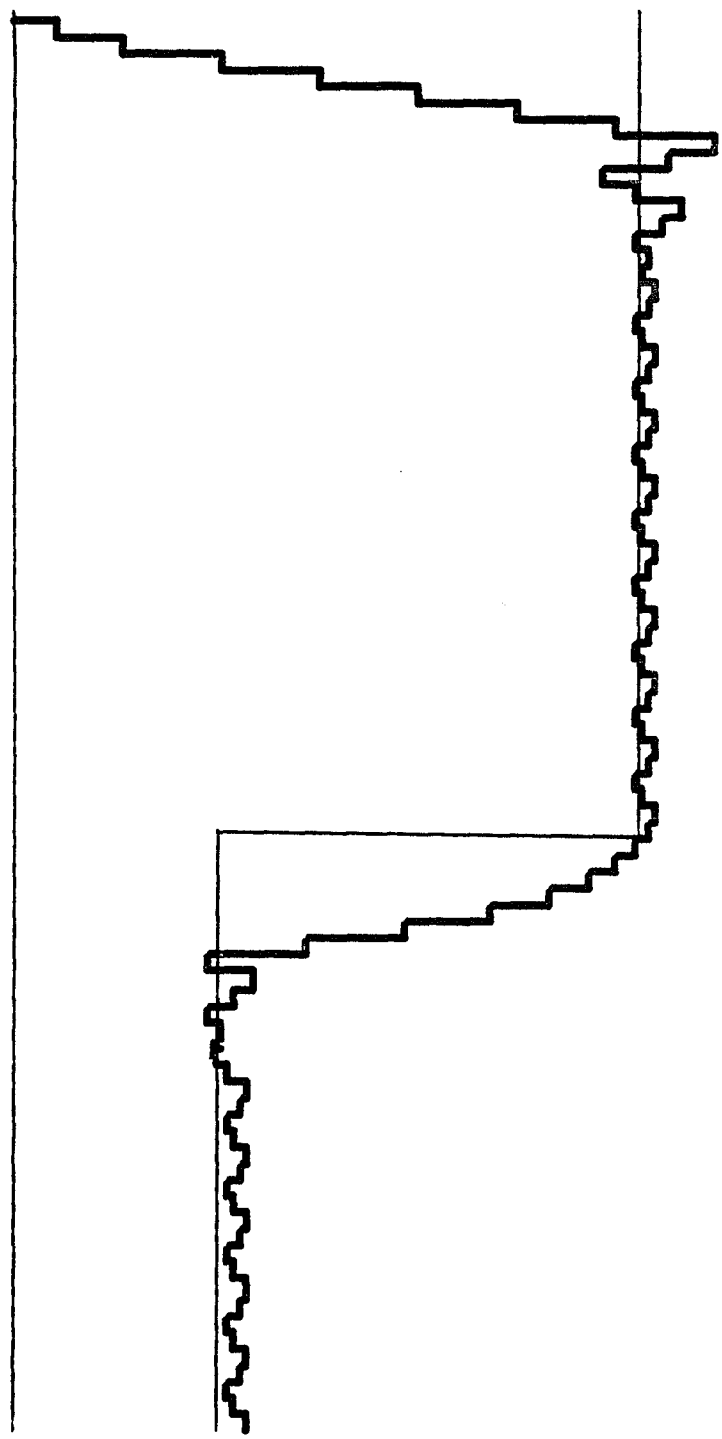


Fig. 3.7 MADM Response

3.4.0 MADM RESPONSE TO INPUT IMAGES

In order to investigate the effects of applying the MADM algorithm to a video picture we must first obtain test images against which to compare the resulting MADM images. The two pictures shown in fig. 2.10 are passed through the system shown in fig. 3.1 with the switch closed to position B. The resulting images stored in File-3 are shown in fig. 3.8. These images will be used to compare the processed images when the switch is closed to position B. Comparing the TEST images with the originals we see that the test images contain slightly greater edge busyness. This should be expected since the the analog output from the first ADM decoder will contain the distortion from the original ADM; the program ADM encoder will thus try to follow this distorted analog signal and thus introduce more distortion which takes the form of greater edge busyness.

The two origial images were processed by the MADM simulation system with the parameters:

3.4-1 $D_1=7$

3.4-2 $D_2=12$

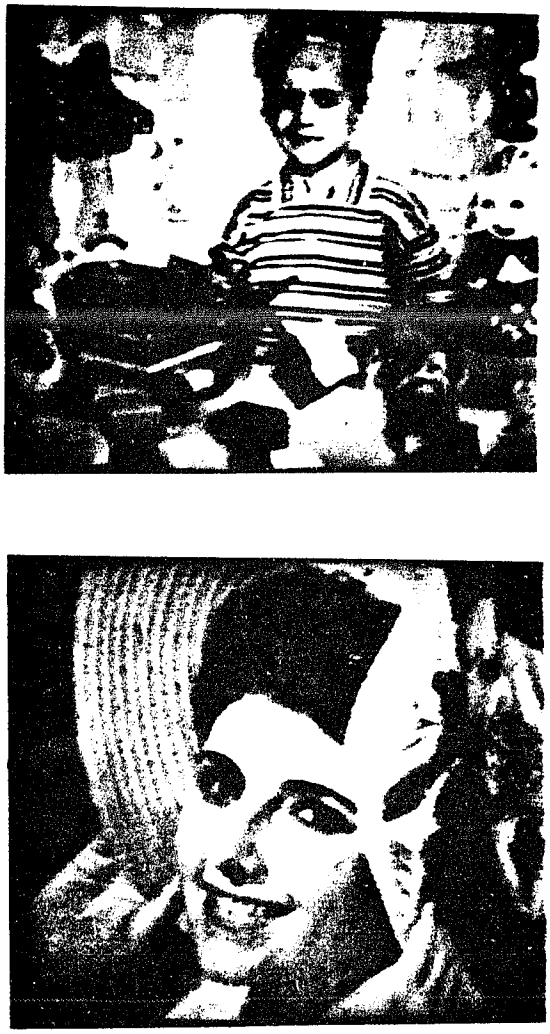


Fig. 3.8 Test images of BOY and GIRL.

3.4-3 Y(k) ADDED TO { LSB OF X(k) / 2^{nd} LSB OF X(k)

3.4-4 STEADY STATE PATTERN: { 1100 / 1001

In figs 3.9 we present the resulting processed images of the GIRL. The most noticeable degradation is the contour noise which results whenever a band of grey levels are replaced by a single grey level. This does not allow a smooth transition from one grey level to another in areas of the image where the grey level is approximately constant. Since the MADM will generate a particular pattern as long as the encoder estimate is within D_2 of the input signal contour noise will be more pronounce in pictures that contain large areas of approximate equal grey levels. Subsequently, the picture of the GIRL will be more vulnerable to the effects of contour noise. The contour noise in the picture of the GIRL is most evident on the black background and is exhibited by the breaking up of the original solid black background into segments with varying grey levels which appear as streaks in the processed picture. However, filtering should improve this. Horizontal filtering is done in real time. Vertical filtering can be done using a averaging filter and using digital PCM store.

LSB, PATTERN=1100 2^{nd} LSB, PATTERN=1100

LSB, PATTERN=1001 2^{nd} LSB, PATTERN=1001

Fig. 3.9 MADM pictures of the GIRL with $D_1=7$, $D_2=12$ and two different steady state patterns, Y(k) added either to LSB or 2^{nd} LSB of X(k).

Comparing the processed pictures we notice that the picture processed by the MADM with the steady state pattern 1100 and the step size added to the 2^{nd} LSB of the estimate has less contour noise. In the picure of the GIRL we can notice the difference in contour noise in the area of the chin and especially in the dark background. The reason that this pattern will produce less contour noise is that the estimate will contain larger perterbations when the step size is added to the 2^{nd} LSB of the estimate. This tends to break up some of the contour noise but also results in a loss in data compression. We perfer the 1001 steady state pattern and step size added to the 2^{nd} LSB because it will track the input signal closer and, judging from the computer simulations, does not produce much more contour noise.

The lighter streaks in the black background result from the fact that the present "forced" steady state pattern will, in general, yield a steady state estimate which is larger than the input signal and results in a lighter grey level than the input signal. By using the complement of the "forced" steady state pattern the opposite effect will be observed, that is, the MADM encoder will tend to produce an estimate signal which is lower than the input signal. The result of simulating the MADM with the parameter:

3.4-5 $D_1 = 7$

3.4-6 $D_2 = 12$

3.4-7 Y(k) added to 2^{nd} LSB of X(k)

3.4-8 PATTERN = 0110

are shown in fig. 3.10A. We notice that the effects of contour noise are less pronounced in the resulting image of the GIRL. This occurs because the eye is less sensitive to a dark streak on a dark background than to a ligher streak on a black background which results when the 1001 steady state pattern is used. In the very light areas of the picture the 0110 steady state pattern produces the effect of increasing the image contrast. This results because those area that are very light will continue to appear light whereas the dark areas will now tend to become darker. The net result is that a better "looking" picture is produced, and it does not result in any loss in data compression.

We notice that the picture of the BOY contains a large amount of detail and, unlike the picture of the GIRL, does not contains large areas of approximately constant grey level, resulting in the contour noise being less noticeable. The contour noise in the picture of the BOY is observed in the background wall where the light differences

Fig. 3.10A MADM pictures of BOY and GIRL with $D_1=7$, $D_2=12$, Y(k) added to 2nd LSB of X(k), and PATTERN=0110.

in shading in the original picture are less evident in the processed picture. We see, from the picture of the boy, that detail is not lost and all the small objects can clearly be identified. A net increase in edge busyness is observed on vertical edges. This increase in edge busyness results partly from the slope overloading introduced by the smaller step size which results from the madm, but is mainly do to processing the video image through three delta modulators. In fig. 3.10B we present the effects of varying the aperture size D_2. The main result is the increase in contour noise. From the experimental results of fig. 3.10 we select the aperture size D_2=12 as providing the best trade off between picture quality and data compression. Notice that even the resulting pictures with D_2=14 maintain their detail and the distortion introduced by the contour noise is not severe

3.4.1 CODING

The object of this study is to examine other techniques of transmitting video data over a computer network using the simplest and least expensive means. We assume that this study will lead to the development of the CUNY SLOW SCAN SYSTEM (CSSS) where images are frozen into the FRAME FREEZE and transmitted over the computer network.

D_2 = 10 D_2 = 14

D_2 = 10 D_2 = 14

Fig. 3.10B MADM pictures of BOY and GIRL with D_1=7, D2= 10 or 14, Y(k) added to 2nd LSB of X(k), and PATTERN=0110.

The host computer connected to the CSSS serves only as an input port and does not do any processing on the video data other than to regulate the packet flow and detect when there is a channel error that occurs in the computer network. In addition we will consider coding techniques that can be performed by a computing element as small as a microprocessor. In this way the CSSS can be implemented with a built-in processor that performs the coding and communicates with the host computer through a standard modem.

In the previous section we saw that using the 0110 steady state pattern and adding the step size to the 2^{nd} LSB of the estimate produces the better "looking" picture. Since the steady state pattern is four bits we considered coding multiples of the four bit data blocks. In TABLE 3-1 are tabulated the run-lengths for data blocks of length 4 and 8-bits corresponding to the MADM image of the BOY and GIRL using the parameters: D_1=7, D_2=12, Y(k) added to 2^{nd} LSB of X(k), and PATTERN = 0110. The picture of the GIRL will produce less number of run-lengths but the run-lengths will generally be longer. It is the longer sequence of steady state patterns in the picture of the GIRL that produces the contour noise. The picture of the BOY will produce shorter sequence of steady state patterns with the result that the contour noise is less noticeable. We

4-BIT BLOCK (average run-length=4.31 blocks)

	1	2	3	4	5	6	7	8	9	10
0	6785.	3269.	2022.	1387.	1053.	782.	728.	589.	396.	344.
10	337.	256.	299.	95.	72.	61.	44.	39.	57.	21.
20	30.	80.	11.	7.	4.	4.	6.	9.	8.	9.
30	5.	1.	3.	0.	2.	1.	0.	0.	1.	1.
40	1.	0.	0.	0.	1.	1.	0.	2.	0.	0.
50	1.	0.	1.	0.	1.	0.	0.	0.	0.	0.
60	0.	0.	0.	1.	1.	1.	1.	1.	1.	0.
70	1.	0.	1.	1.	0.	0.	0.	0.	1.	0.
80	0.	0.	0.	0.	0.	0.	1.	0.	0.	0.
90	0.	0.	1.	0.	0.	0.	0.	0.	1.	1.
100	0.	0.	0.	0.	0.	0.	1.	0.	1.	0.
110	0.	0.	0.	0.	0.	1.	2.	1.	1.	0.
120	0.	0.	0.	0.	1.	0.	0.	0.	0.	0.
130	2.	1.	0.	0.	0.	0.	0.	1.	0.	0.
140	0.	0.	1.	0.	0.	0.	0.	1.	1.	1.
150	0.	1.	0.	0.	0.	1.	0.	0.	0.	0.
160	1.	0.	0.	0.	0.	0.	0.	0.	0.	0.
170	0.	0.	0.	0.	0.	0.	1.	0.	0.	0.
180	1.	0.	0.	1.	0.	1.	0.	0.	0.	0.
190	0.	0.	0.	0.	1.	0.	0.	0.	0.	1.
200	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
210	0.	1.	0.	0.	5.	0.	0.	0.	0.	0.
220	4.	0.	0.	0.	0.	0.	3.	0.	0.	2.
230	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

8-BIT BLOCK (average run-lenth=2.99 blocks)

	1	2	3	4	5	6	7	8	9	10
0	4334.	2083.	1428.	893.	643.	426.	148.	95.	86.	112.
10	25.	11.	14.	15.	10.	3.	3.	1.	1.	2.
20	0.	2.	2.	0.	1.	1.	1.	0.	0.	0.
30	0.	3.	2.	1.	1.	2.	0.	0.	1.	0.
40	0.	0.	1.	0.	0.	1.	0.	0.	1.	1.
50	0.	0.	1.	1.	0.	0.	1.	2.	2.	0.
60	0.	1.	0.	0.	2.	1.	0.	0.	1.	0.
70	1.	0.	0.	3.	1.	0.	1.	0.	0.	1.
80	0.	0.	0.	0.	0.	0.	0.	1.	0.	1.
90	1.	1.	0.	0.	0.	0.	1.	0.	1.	0.
100	0.	0.	0.	0.	1.	0.	5.	0.	0.	4.
110	0.	0.	3.	2.	0.	0.	0.	0.	0.	0.

TABLE 3-1A Run-length statistics for 4 and 8-bit data block of the MADM picture of the BOY.

4-BIT BLOCK (average run-length=5.62 blocks)

	1	2	3	4	5	6	7	8	9	10
0	6212.	2784.	1756.	1114.	834.	594.	522.	454.	276.	275.
10	231.	191.	329.	69.	84.	59.	49.	35.	106.	39.
20	48.	116.	42.	43.	25.	32.	17.	64.	49.	33.
30	25.	22.	22.	24.	15.	15.	13.	8.	6.	2.
40	6.	6.	4.	5.	9.	6.	5.	8.	2.	2.
50	6.	5.	5.	6.	3.	2.	0.	0.	1.	0.
60	0.	0.	1.	1.	0.	0.	1.	0.	0.	1.
70	2.	0.	0.	0.	0.	1.	0.	1.	0.	0.
80	0.	1.	0.	1.	2.	1.	0.	0.	0.	0.
90	0.	1.	0.	0.	0.	1.	0.	0.	1.	0.
100	1.	0.	0.	0.	0.	0.	0.	0.	1.	0.
110	0.	1.	0.	0.	0.	1.	0.	2.	3.	1.
120	0.	0.	0.	0.	1.	0.	0.	0.	0.	0.
130	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
140	0.	0.	1.	0.	0.	0.	0.	0.	0.	0.
150	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
160	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
170	0.	2.	1.	1.	0.	0.	0.	0.	0.	0.
180	0.	0.	0.	2.	1.	0.	1.	0.	0.	1.
190	0.	0.	0.	0.	1.	0.	0.	0.	2.	0.
200	0.	0.	0.	0.	0.	2.	0.	0.	0.	0.
210	0.	0.	0.	0.	0.	1.	0.	0.	0.	0.
220	9.	0.	0.	0.	0.	0.	2.	0.	0.	3.
230	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

8-BIT BLOCK (average run-length=3.86 blocks)

	1	2	3	4	5	6	7	8	9	10
0	3724.	1698.	1033.	670.	453.	437.	150.	86.	147.	156.
10	75.	62.	93.	72.	50.	52.	29.	25.	8.	11.
20	7.	16.	14.	5.	11.	10.	7.	0.	1.	0.
30	2.	0.	1.	1.	2.	0.	1.	1.	0.	0.
40	2.	2.	1.	0.	1.	0.	0.	1.	1.	1.
50	0.	0.	0.	1.	0.	1.	1.	2.	4.	0.
60	0.	1.	0.	0.	0.	0.	0.	0.	0.	0.
70	1.	0.	0.	0.	0.	0.	0.	0.	0.	0.
80	0.	0.	0.	0.	2.	2.	0.	0.	0.	0.
90	2.	1.	1.	1.	0.	0.	1.	0.	2.	0.
100	0.	3.	0.	0.	0.	0.	1.	0.	0.	9.
110	0.	0.	2.	3.	0.	0.	0.	0.	0.	0.

TABLE 3-1B Run-length statistics for 4 and 8-bit data block of the MADM picture of the GIRL.

notice that the the average run-length for the 4-bit steady state pattern is 4.3 for the BOY and 5.6 4-bit blocks for the GIRL. If different rotations of the steady state pattern or if the step size is added to the LSB of the estimate and the above apertures are used the MADM will produce similar results. In general, whenever the data block length is four the MADM encoder will produce runs with an average run-length greater than four. Data blocks larger than four bits will produce average run-lengths between two and three. Notice that there are several video lines that are almost completely encoded into steady state patterns. These lines are probably located at the top and bottom of the picture.

We calculate the data compression obtained by using the MADM algorithm and run-length coding the steady state pattern as follows: If a data block does not correspond to the steady state pattern then a binary "0" is transmitted followed by the N data bits where N equals the number of bits in a data block which each video line is partitioned into. If the data block corresponds to the steady state pattern, we proceed to count the number of consecutive steady state patterns that are produced by the MADM. We then transmit a binary "1" followed by a M bit word which specifies the number of consecutive steady state patterns counted. The value of M is given by:

$$M < \log_2[\text{average runlength}]$$

where M is the largest integer.

Fig. 3.11 shows the run-length encoding of the different 4-bit data blocks. From TABLE 3-1 we see that the value of M is 1 or 2. In TABLE 3-2 the percent data compression that can be achieved for the input image of the GIRL with the step size added to the LSB or 2^{nd} LSB of the estimate and using either the 0011 or 0110 steady state pattern. We notice that the processed images where the step size is added to the LSB of the estimate produces the greater data compression but this also produces resulting images which contain greater contour noise. Experimentally we find that the reduction in data compression when adding the step size to the 2^{nd} LSB bit of the estimate is justified by the better quality resulting image. Between the two steady state patterns the 0110 produces a greater amount of data compression due to the fact that this pattern will track the input signal closer. In fig. 3.12 and 3.13 we present a plot of the percent data compression achieveable for the MADM picture of the GIRL and BOY with Y(k) added to the 2^{nd} LSB of the estimate and steady state pattern 0110 as a function of D_1 and D_2 for M = 1 and 2. We notice from the figures that the data compression will depend more on the

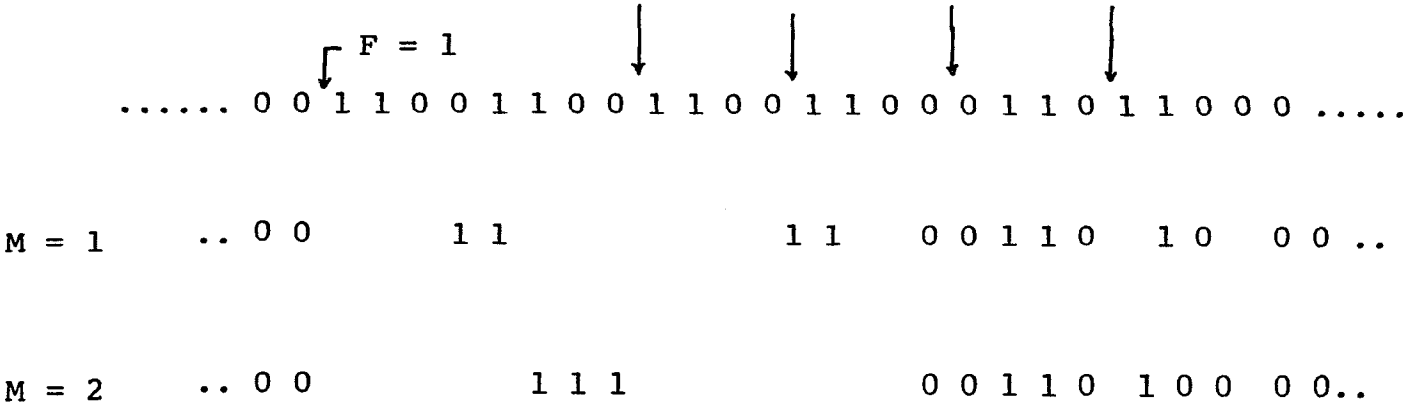


Fig. 3.11 Run-length encoding of 4-bit blocks (1100) for M = 1, 2.

PATTERN	STEP ADDED TO	M	% COMPRESSION
0011	LSB	1	49.50
		2	51.53
	2^{nd} LSB	1	47.66
		2	48.94
0110	LSB	1	49.75
		2	51.65
	2^{nd} LSB	1	49.85
		2	50.67

TABLE 3-2 Run-length data compression for MADM picture of the GIRL with 0011 and 0110 steady state pattern and D_1=7, D_2=12.

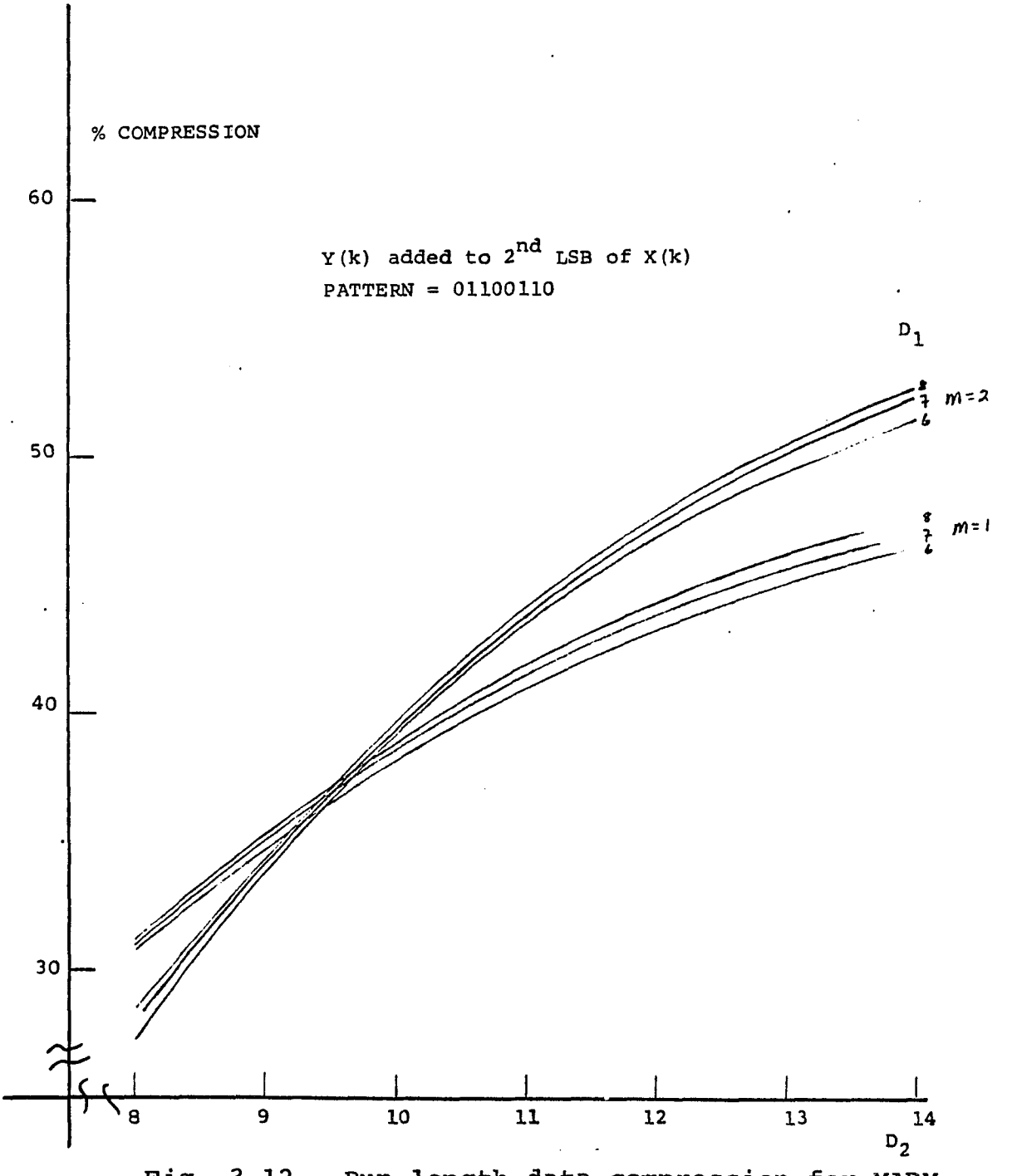


Fig. 3.12 Run-length data compression for MADM pictures of the GIRL as a function of M, D_1, and D_2.

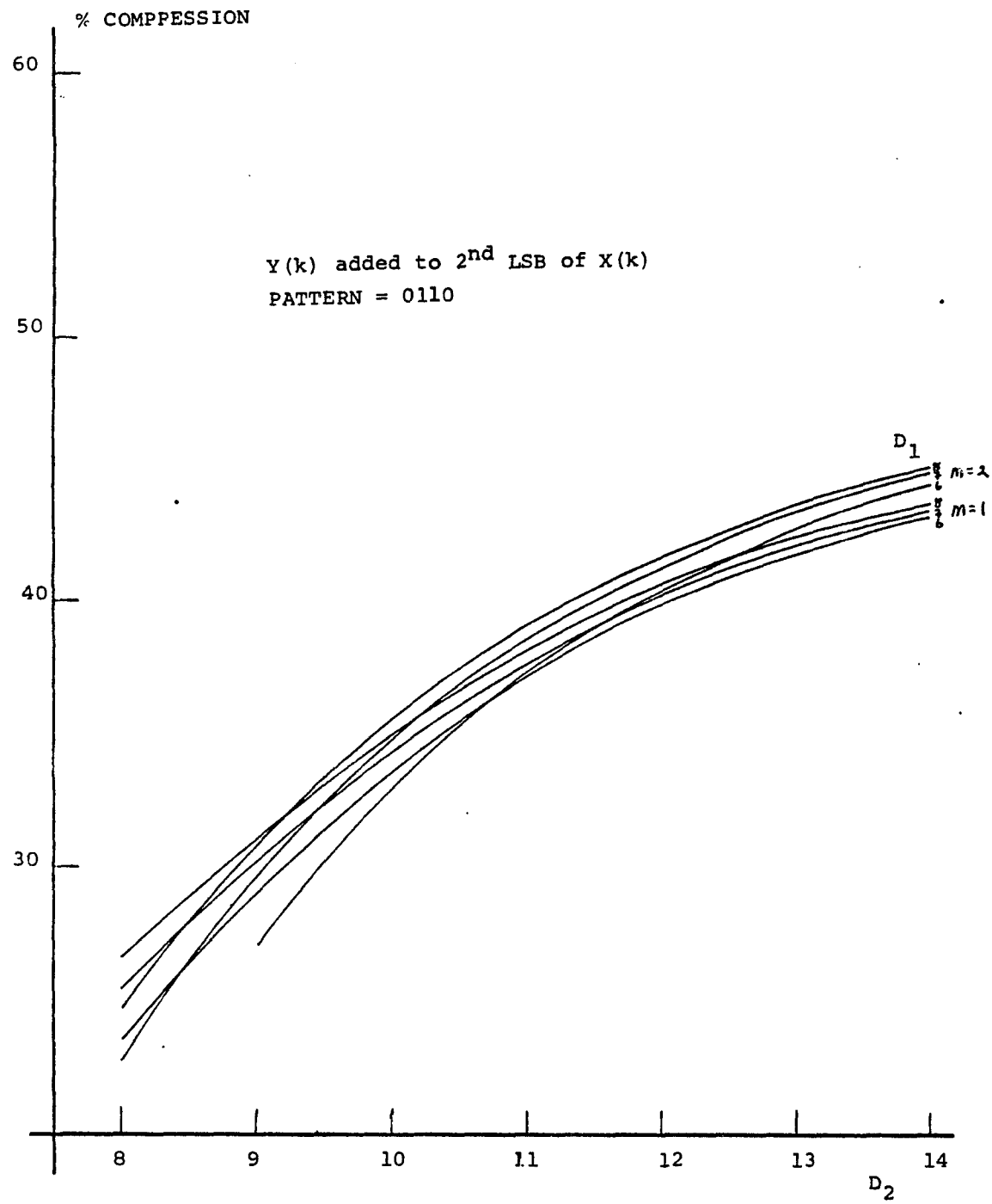


Fig. 3.13 Run-length data compression for MADM picture of the BOY as a function of M, D_1, and D_2.

size of aperture D_2 then on the aperture D_1. When the MADM is tracking a video signal that is approximately constant the number of steady state E(k) data bits that are produced increases with size D_2 because the MADM will tolerate larger perterbations on the input signal. Thus, as shown in the figures, the most data compression can be achieved is by setting M = 2 for the larger values of D_2 and for the lower values we should set M = 1. The process picture of the BOY will produce results similar to those of the GIRL except that the amount of data compression will be lower. From the resulting images in fig.3.10 we notice that in order to maintain a good quality image we don't want to operate with an aperture D_2 greater than 12. From fig. 3.13 we see that for the parameters D_2=12 we should use M=2 which provides, at this value of D_2, about 2% greater data compression when encoding both pictures. By setting M=1 we can maintain a data compression close to the possible maximum even if the picture contains a large amount of detail resulting in lower data compression. From the above we conclude that if run-length coding is employed, then M=1. With M=1 we can indicate that up to two consecitive N bit steady state patterns have been detected. If a run-length greater than two is detected then it is encoded in groups of twos. The percent compression can be calculated as follows:

$$3.4\text{-}9 \qquad C = \left(1 - \frac{\left[\frac{512*928}{N} - \sum_{j=1}^{928/N} jY_j \right] (N+1) + \sum_{\substack{j=2 \\ j \text{ even}}}^{928/N} j[Y_j + Y_{j-1}]}{512*928} \right) * 100$$

where C = percent data compression.

512 = lines per frame.

928 = bits per line.

N = bits in data block.

j = # of data blocks in run-length.

Y_j = # of runs of length j.

The above equation can be reduced to :

$$3.4\text{-}10 \qquad C = \left[\frac{1}{512*928} \left([N \sum_{j=1}^{928/N} jY_j + \sum_{j \text{ odd}}^{928/N} Y_j] \right) - \frac{1}{N} \right] * 100$$

where the first term in brackets represents the total number of bits in all the run-lengths. The second term represent the total number of odd run-lengths. If we divide and multiply by the total number of run-lengths the result is:

$$3.4\text{-}11 \quad \% \text{ compression} = \left(\frac{T[NR+P_0]}{928*512} - \frac{1}{N} \right) * 100$$

where T = total number of run length sequence.

N = number of bits in the pattern.

R = average run length.

P_0 = % of odd run-length.

A simpler way of coding the data blocks, and one which is easier to realize in hardware, is to set M=0. In this way a binary "1" is transmitted whenever a steady state pattern is detected. If the data block does not correspond to the steady state pattern a binary "0" is transmitted followed by the N data bits. In TABLE 3-3 and 3-4 the probability of every 4 and 8-bit data block is tabulated for the MADM pictures of the BOY and GIRL with the parameters stated in equations 3.4-5 thru 3.4-8. The percent compression achievable by setting M=0 is:

$$3.4\text{-}12 \quad \% \text{ compression} = \frac{NX-1}{N} * 100$$

where N = number of bits in data block.

X = probobility of data block occurring.

P_0P_1 \ P_2P_3	00	01	10	11
00	4.79	3.74	1.53	1.78
01	3.32	0.22	68.97	7.37
10	0.16	0.24	0.27	0.51
11	1.53	1.45	1.91	2.23

$P_0P_1P_2P_3P_4$ \ $P_5P_6P_7$	000	001	010	011	100	101	110	111
00000	0.01	0.22	0.30	0.24	1.59	0.10	1.11	0.32
00001	0.11	0.11	0.08	0.04	0.51	0.27	0.11	0.05
00010	0.12	0.06	0.03	0.01	0.13	0.00	2.98	0.07
00011	0.00	0.04	0.01	0.05	0.05	0.07	0.06	0.03
00100	0.11	0.08	0.04	0.05	0.04	0.00	1.12	0.09
00101	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01
00110	0.04	0.08	0.09	0.10	0.01	0.01	1.17	0.13
00111	0.00	0.01	0.00	0.02	0.02	0.03	0.04	0.06
01000	0.42	0.39	0.26	0.22	0.02	0.01	2.00	0.06
01001	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01
01010	0.00	0.00	0.00	0.00	0.00	0.00	0.16	0.03
01011	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.02
01100	3.60	2.76	0.41	0.56	1.19	0.01	53.17	5.10
01101	0.00	0.00	0.00	0.00	0.02	0.00	0.37	1.38
01110	0.06	0.08	0.21	0.24	0.08	0.02	3.54	0.17
01111	0.01	0.01	0.10	0.23	0.59	0.73	0.76	0.53
10000	0.01	0.01	0.02	0.01	0.01	0.00	0.08	0.02
10001	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.00
10010	0.01	0.00	0.00	0.00	0.02	0.00	0.17	0.01
10011	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.00
10100	0.02	0.00	0.00	0.00	0.01	0.00	0.24	0.02
10101	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
10110	0.00	0.00	0.00	0.00	0.01	0.00	0.39	0.02
10111	0.00	0.00	0.01	0.01	0.01	0.03	0.01	0.01
11000	0.02	0.02	0.02	0.02	0.04	0.00	0.64	0.38
11001	0.02	0.01	0.00	0.00	0.13	0.12	0.10	0.04
11010	0.01	0.00	0.00	0.00	0.02	0.00	0.91	0.10
11011	0.00	0.00	0.00	0.01	0.07	0.07	0.12	0.11
11100	0.01	0.03	0.04	0.01	0.05	0.02	1.06	0.51
11101	0.00	0.00	0.00	0.00	0.04	0.04	0.11	0.12
11110	0.01	0.04	0.11	0.31	0.04	0.03	0.62	0.39
11111	0.01	0.07	0.05	0.15	0.08	0.12	0.08	0.00

TABLE 3-3 4 and 8-bit pattern statistics for the MADM picture of the BOY with D_1=7, D_2=12 and PATTERN=0110.

P_0P_1 \ P_2P_3	00	01	10	11
00	3.71	3.64	1.12	1.57
01	2.81	0.15	74.29	6.07
10	0.14	0.19	0.21	0.37
11	1.33	1.14	1.58	1.68

$P_0P_1P_2P_3P_4$ \ $P_5P_6P_7$	000	001	010	011	100	101	110	111
00000	0.03	0.20	0.19	0.16	1.44	0.07	0.76	0.24
00001	0.08	0.08	0.04	0.01	0.45	0.22	0.13	0.03
00010	0.07	0.05	0.02	0.01	0.10	0.00	3.02	0.05
00011	0.00	0.03	0.02	0.04	0.06	0.08	0.05	0.02
00100	0.05	0.07	0.03	0.02	0.02	0.00	0.88	0.09
00101	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01
00110	0.01	0.05	0.07	0.04	0.02	0.00	1.17	0.08
00111	0.00	0.00	0.00	0.01	0.02	0.03	0.03	0.03
01000	0.27	0.25	0.16	0.22	0.01	0.03	1.89	0.03
01001	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
01010	0.00	0.00	0.00	0.00	0.00	0.00	0.12	0.00
01011	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
01100	2.74	2.90	0.37	0.71	0.89	0.01	60.23	4.52
01101	0.00	0.00	0.00	0.00	0.05	0.00	0.39	1.07
01110	0.08	0.07	0.14	0.18	0.13	0.01	3.11	0.09
01111	0.04	0.03	0.09	0.17	0.51	0.50	0.47	0.36
10000	0.00	0.01	0.01	0.01	0.01	0.00	0.09	0.01
10001	0.00	0.01	0.00	0.00	0.00	0.01	0.00	0.00
10010	0.00	0.00	0.00	0.00	0.01	0.00	0.16	0.01
10011	0.00	0.00	0.00	0.01	0.01	0.00	0.00	0.00
10100	0.00	0.00	0.00	0.00	0.01	0.00	0.22	0.00
10101	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
10110	0.00	0.00	0.00	0.00	0.02	0.00	0.26	0.00
10111	0.00	0.00	0.01	0.01	0.01	0.01	0.02	0.00
11000	0.02	0.02	0.02	0.01	0.02	0.01	0.61	0.24
11001	0.00	0.01	0.00	0.00	0.12	0.10	0.12	0.03
11010	0.00	0.00	0.00	0.00	0.01	0.00	0.81	0.09
11011	0.00	0.00	0.00	0.01	0.04	0.05	0.11	0.07
11100	0.01	0.03	0.02	0.01	0.04	0.03	0.98	0.40
11101	0.00	0.00	0.00	0.00	0.01	0.01	0.06	0.12
11110	0.02	0.02	0.07	0.22	0.03	0.02	0.42	0.33
11111	0.01	0.04	0.03	0.14	0.07	0.10	0.08	0.01

TABLE 3-4 4 and 8-bit pattern statistics for the MADM picture of the GIRL with D_1=7, D2=12 and PATTERN=0110.

In fig. 3.14 we present the precent data compression as a function of the two aperture sizes, D_1 and D_2, with N=0 or simple block coding. In fig. 3.14 the encoding block size is 4-bits and the steady state pattern is 0110. The two sets of curves represent the compression if the block length used by the computer for coding is either 4 or 8 bits. We notice that for low values of D_2 that using a 4-bit coding block length provides about 10 percent greater data compression over an 8-bit coding block length. The reason for this is that whenever the MADM enters into the steady state mode it will not remain in this state very long if the aperture size D_2 is small. Thus, the MADM will tend to generate more 4-bit steady state sequence than 8-bit sequences.

As the aperture size D_2 increases, the MADM will begin to produce longer sequence of steady state data bits. This result is an increase in data compression regardless of which coding block size is used. However, if we use an 8-bit coding block one bit can represent 8 data bits of data while if a 4-bit coding block is used we need 2 bits to specify the same eight data bits. This is evident from the curves in fig. 3.13 where we notice that the initial slope corresponding to the 8-bit coding block is greater than the 4-bit coding block.

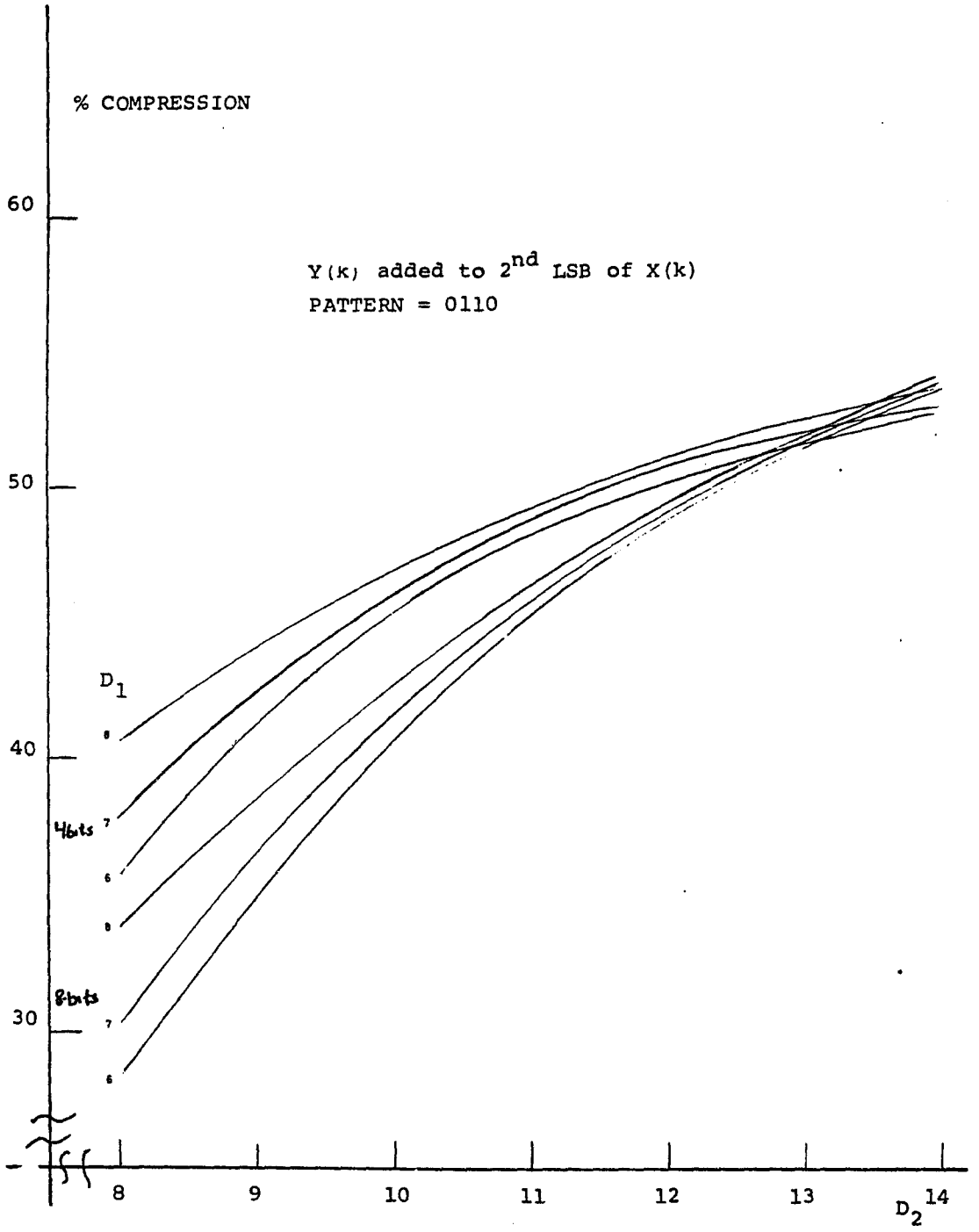


Fig. 3.14 Data compression using block coding (M=0) for the MADM picture of the GIRL as a function of D_1, D_2, and coding block length.

For values of D_2 = 13 or 14 both coding block length yield about equal data compression. We notice, however, that the curves corresponding to the 8-bit coding block continues to have a greater slope than for the 4-bit coding block which has a very small slope. This results because the MADM will, for large values of D_2, produce longer sequences of steady state data bits which are more economically coded using a longer coding block length.

If the MADM encoding data block length were eight bit rather than the four bits used above, then the resulting data compression curves are shown if fig. 3.15. For low values of D_2 the MADM will produce less steady state E(k) data bits. This result because the MADM encoder can initiate steady state operation at only half the number of locations that a 4-bit pattern length can. This effect is more evident when using the 4-bit coding block. We notice that at higher values of D_2 the 8-bit coding blocks in both the figures are identical as should be expected.

Our simulation results indicate that using the MADM algorithm which divides each video line into 4-bit data blocks produce better images, for a given compression, than if the video line is partitioned into 8-bit data blocks. This results because for a given initial aperture D_1, using a 4-bit data block requires a lower aperture size D_2 to

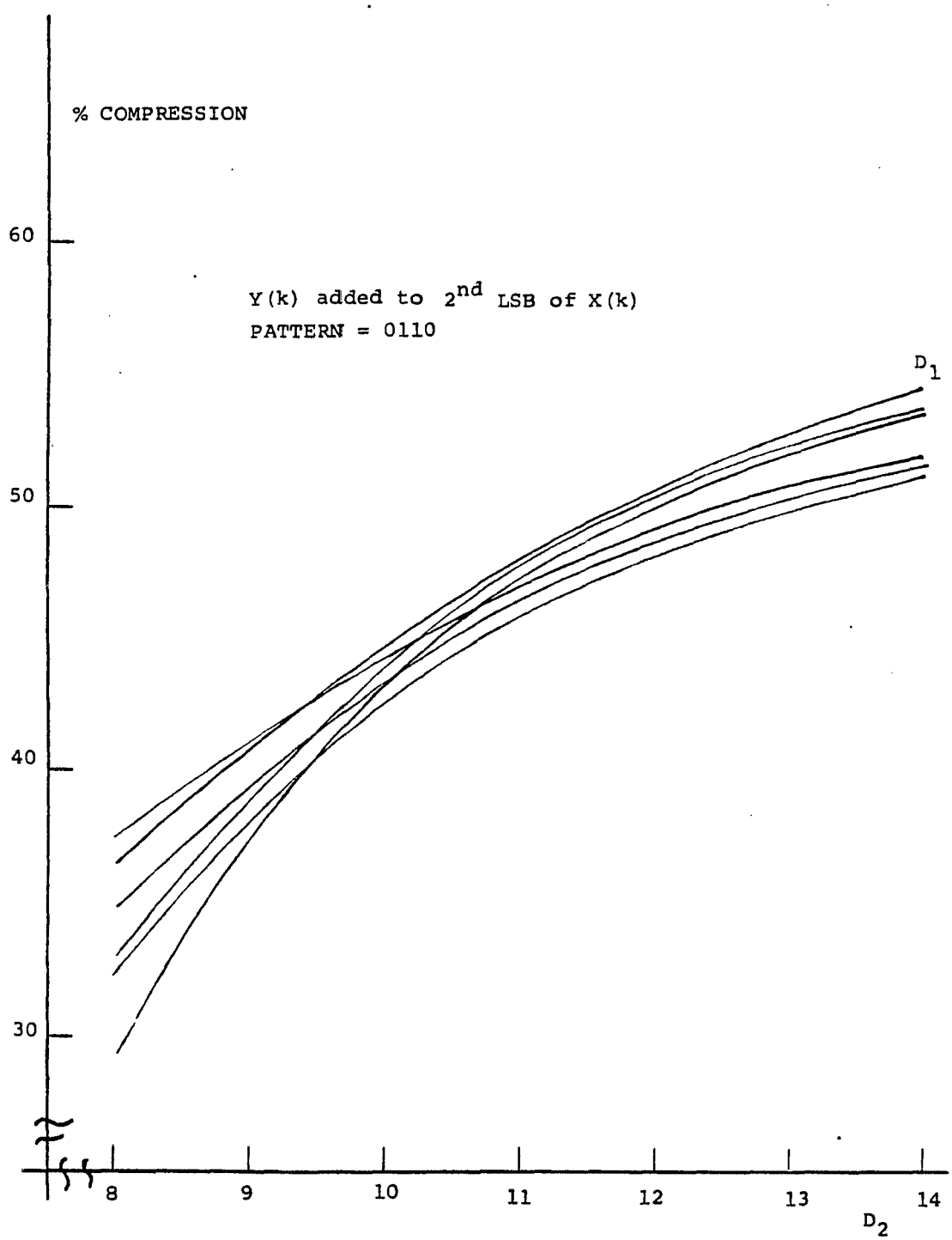


Fig. 3.15 Data compression using block coding(M=0) for the MADM picture of the GIRL as a function of D_1, D2, and coding block length.

achieve a given compression. The smaller aperture size D_2 requires that the encoder establish a steady state estimate which is closer to the input signal. Comparing the compression curves for M=0 and M=2 we see that if the amount of data compression expected is under 50% that using simple block coding(M=0) will provide the greater data compression. Since both schemes produce aprroximately equal compression due to the fact that the average run-length is not large the easiest scheme to implement in hardware, when M=0, should be used.

For comparison we present the result of using a 4 and 8-bit coding block and using a Huffman code in fig. 3.16. The compression curves shown in fig. 3.16 do not take into account the coding overhead which has to be incurred. This overhead is about 2%. Comparing the Huffman code to the block code we see that in the region D_1=7 and D_2=12 that the Huffman code provides about 10% greater compression. This increase in compression is off-set by the increase complexity introduced in implementing the Huffman code.

When using block coding the receiver will have to keep track of the number of data blocks it has recieved. At the beginning of each data block a decision will be made by the reciever depending on the value of the header bit. If the header bit is "0" then the following N bits

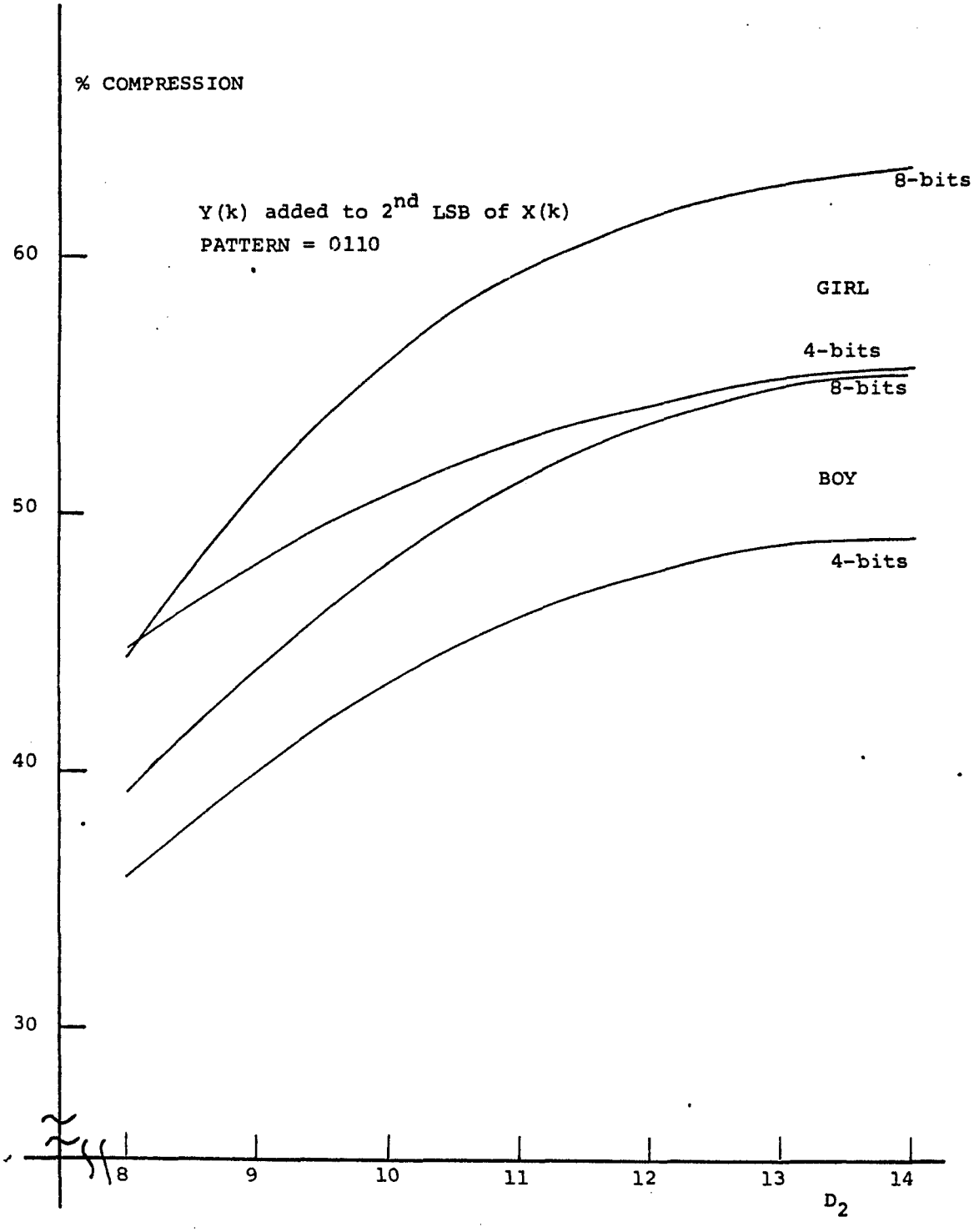


Fig. 3.16 Data compression for MADM pictures of BOY and GIRL using Huffman code of 4 and 8-bit blocks.

represent data bits. If the header bit is "1" then it represents one N-bit steady state patterns. The effects of channel errors could severely degrade the performance of the system and will have to be investigated.

3.4.2 FIELD INTERPOLATION

In SECTION 2.5 we saw that data compression could be achieved by transmitting only one field and at the receiver generate the missing field by interpolating the received field. This technique can also be performed on the resulting MADM images to achieve approximately 50% compression on the already compressed data. In fig. 3.17 we present the result of field interpolating the MADM input images with the parameters specified in equations 3.4-5 thru 3.4-8. Again we notice that the effects of interpolating the second field results in a decrease in edge busyness and the effects are more apparent now. Another result of field interpolating the received picture is a reduction, due to averaging, in the streaks that were produced by the MADM. A more positive result produced by field interpolating the received picture is an increase in the picture graininess which when imposed on the contour noise produces a better looking picture. The graininess

70.4% COMPRESSION

76.3% COMPRESSION

Fig. 3.17 Field interpolation of MADM with D_1=7, D_2=12, Y(k) added to 2nd LSB of X(k).

adds texture to the previously "flat" regions affected by contour noise. The results of field interpolating the MADM picture presents a very favorable means of achieving data compression, 70-75% for the two input pictures. As stated above, in order to implement field interpolation the receiving computer must contain a programmed MADM encoder and decoder.

3.5.0 HARDWARE CONSIDERATIONS

In order to implement the MADM algorithm modification of the existing ADM hardware have to be made. There are three conditions that must be determined in order to select which of the two modes the MADM is to operate in; as a Song mode ADM or in the steady state mode in which the MADM produces the predetermined steady state pattern. The conditions that must be determined are summerize below:

1) When F=1, indicating the begining of an 8 bit data block.

2) When $|ERROR| = |S(k)-X(k)| \leq D_1$, indicating that encoder's estimate is close enough to the input signal so that the MADM encoder can start to produce the predetermined steady state pattern if the E(k) data bit

corresponds to the beginning of a N-bit data block.

3) When |ERROR| > D_2, indicating that MADM should revert back to operating as a Song mode ADM.

We would like to investigate methods of determining the three above conditions without having to implement the block diagram shown in fig. 3.2.

3.5.1 F=1 SIGNAL

A requirement of the MADM is being able to partition the E(k) data bits of each video line into N-bit blocks. This is necessary so that every steady state pattern produced by the MADM encoder begins at a predetermine address. The Frame Freeze unit, see APPENDIX A, was designed to store only the E(k) data bits corresponding to the video data and ignores the E(k) data bits produced during the horizontal and vertical synch. pulses. Fig. 3.18 shows a possible circuit to partition the video lines into N-bit data blocks.

The operation of the circuit shown in fig. 3.18 is as follows: Whenever a horizontal synch. pulse is detected the down counter is loaded with the N-bit count. While the horizontal drive remains at logic '0' the down counter is disabled and the borrow signal is at logic '1'. As soon as

the horizontal drive goes to logic '1' the down counter is enable and starts to count the clock pulses which is equivalent to counting the E(k) data bits being produced by the MADM. As shown in fig. 3.18, after every N clock pulses the down counter will produce a pulse which is used to generate the signal F.

3.5.2 $\underline{|S(k)-X(k)| \leq D_1}$

In order for the MADM to produce the predetermined steady state pattern we must be able to detect when the encoder estimate is close to the input signal. Instead of having to obtain the absolute value of the ERROR signal and comparing it to D_1 the encoder's step size can be used to indicate when the estimate is close to the input signal. We saw from the simulations of the delta modulators that one characteristic they all shared is that whenever the encoder is tracking a constant input signal the step size remains in the two smallest values. We can thus expect that using the step size to initiate the MADM steady state mode should produce satisfactory results.

A simple circuit that can be used to detect when the encoder's step size is less than some specific value is shown in fig. 3.19. The circuit shown detect whenever the step size is less than four quantization levels. The

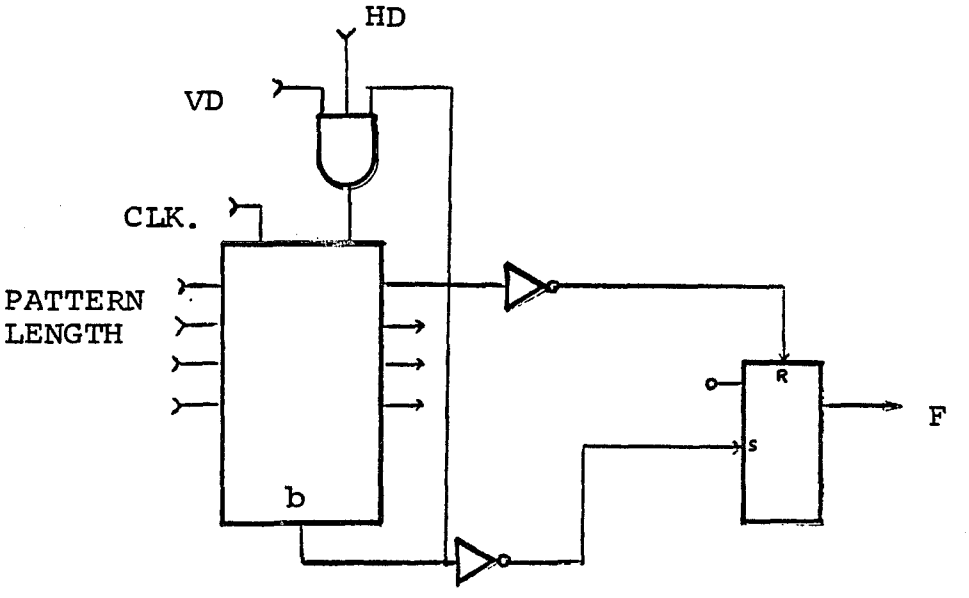


Fig. 3.18 Circuit to partition a video line.

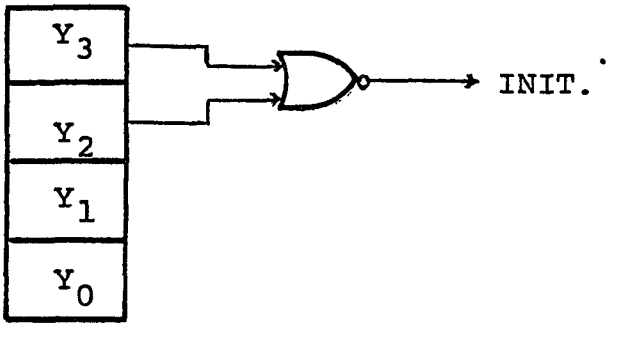


Fig. 3.19 Circuit to initiate MADM steady state using step size.

register $[Y_0Y_1Y_2Y_3]$ corresponds to the four bit step size. The output of the NOR gate will equal logic '1' only if both the two most significant bits of the step size are logic '0'.

In order to observe the effects of using the step size to initiate the MADM steady state mode simulations were performed on the two input images. The result of the computer simulations for the picture of the BOY are shown in fig. 3.20 for different initiating step sizes. In these simulations the MADM encoder will revert to operating in the Song mode whenever the magnitude of the error signal is greater than D_2. We notice from the resulting images that using the step size to initiate the MADM steady state mode produces images comparable to the MADM algorithm using the aperture D_1=7 if the initiating step size is three quantization levels or less. Using a smaller initiating step size results in a better picture but at a loss of data compression. If a larger initiating step size is used then the MADM will attempt to initiate steady state operations the majority of the time, since the step size has a high probability of being under four, resulting in the MADM "jumping" in and out of the steady state and in a slight loss in data compresion. The "jumping" in and out of the steady state pattern also creates greater edge busyness since when an edge is encountered the MADM will sometimes

ISTEP = 2 C = 37.47% ISTEP = 3 C = 43.68%

ISTEP = 4 C = 43.21%

Fig. 3.20 MADM steady state initiated by step size D_2=12, Y(k) added to 2nd LSB of X(k), and PATTERN=0110. (C=compression)

be in the steady state mode and take a longer amount of time to react to the edge; at other times the MADM is out of steady state and can react quicker to the step input.

3.5.3 $|S(k)-X(k)| > D_2$

Several techniques have been attempted to detect the condition to force the MADM to operate in the Song mode. The technique that produced the best results is decribed below. The method used to indicate when $|S(k)-X(k)| > D_2$ makes use of the fact that when the ADM is tracking a constant input signal the resulting E(k) data bits will contain values of both +1 and -1 since the estimate will be oscillating about the input signal. Thus, we can attempt to detect the $|S(k)-X(k)|$ D_2 condition by observing the output of the comparitor during steady state operation. If we detect a long sequence of consecutive outputs of the same polarity, which indicates that the steady state estimate is either constantly above or below the input signal, then the MADM should be forced to revert operating in the Song mode. The number of consecutive outputs of the same polarity should be greater than the pattern length, otherwise the MADM will not produce sufficient steady state patterns to warrent coding. In fig. 3.21 we present the results of simulating the MADM with the initial aperture of

N = 6 C = 34.07% N = 7 C = 37.37%

N = 8 C = 41.42%

Fig. 3.21 MADM halts steady state mode if N consecutive comparitor outputs of same polarity, D_1=7, Y(k) added to 2nd LSB, PATTERN=0110.

D_1=7 and the MADM forced to return to the Song mode if the specified number of consecutive outputs of the comparitor are of the same polarity. We notice from the resulting images that the effect of increasing the number of consectutive bits with the same polarity is to increase the edge busyness since, in the worst case, it might take that number of clock pulses for the encoder to react to an edge. In fig 3.22 we present the result of using the step size to initiate the steady state mode and eight consecutive bits of the same polarity to force the MADM into the Song mode. The resulting images produce approximately the same data compression as using D_1=7 and D_2=12. However, there is a noticeable increase in picture distortion. The distortion is further increased if we use field interpolation. The resulting pictures, even though they contain distortion, are good when the resulting amount of data compression is considered.

3.6.0 Conclusion

We have shown that employing a MADM to process video signals, we can produce a particular steady state pattern in sufficient quantity (greater than 50% for the two input pictures) to make compression possible. The processed image represents a net data compression of 40-50% which

includes the coding overhead. We have also shown that good quality pictures can result from transmitting only one field and interpolating the second field yielding a net compression of 70-75%. In addition we showed that the MADM algorithm can be implemented by modifying the present ADM hardware. This data compression refers only to the video data and does not include the additional savings that can be obtained by not sending the horizontal and vertical synch drives. The horizontal and vertical synch drives represent approximatly 10% of the video signal.

CHANNEL ERRORS

4.0.0 INTRODUCTION

In the previous chapter it was shown that a possible data compression can be achieved by using an MADM encoder and then compressing the data by block coding the resulting E(k) data stream. The main objective of the research is the transmission of video data over computer networks so the subject of channel errors is an important aspect of this study.

Fig. 4.1 illustrates a portion of a possible computer network. There are three important quantities that make-up a computer network; the HOST COMPUTER, the SWITCHING COMPUTER, and the REMOTE TERMINAL. Usually, a computer customer has access to the computer network via a remote terminal connected to a host computer. The remote terminal is connected to the host computer via a common carrier. In our particular case the PDP-11/34, which is interfaced to the CUNY SLOW SCAN SYSTEM, is connected to a host computer through a 9.6Kb/s channel. The purpose of the PDP-11/34 is to assemble in the proper format all messages that are to be transmitted over the computer

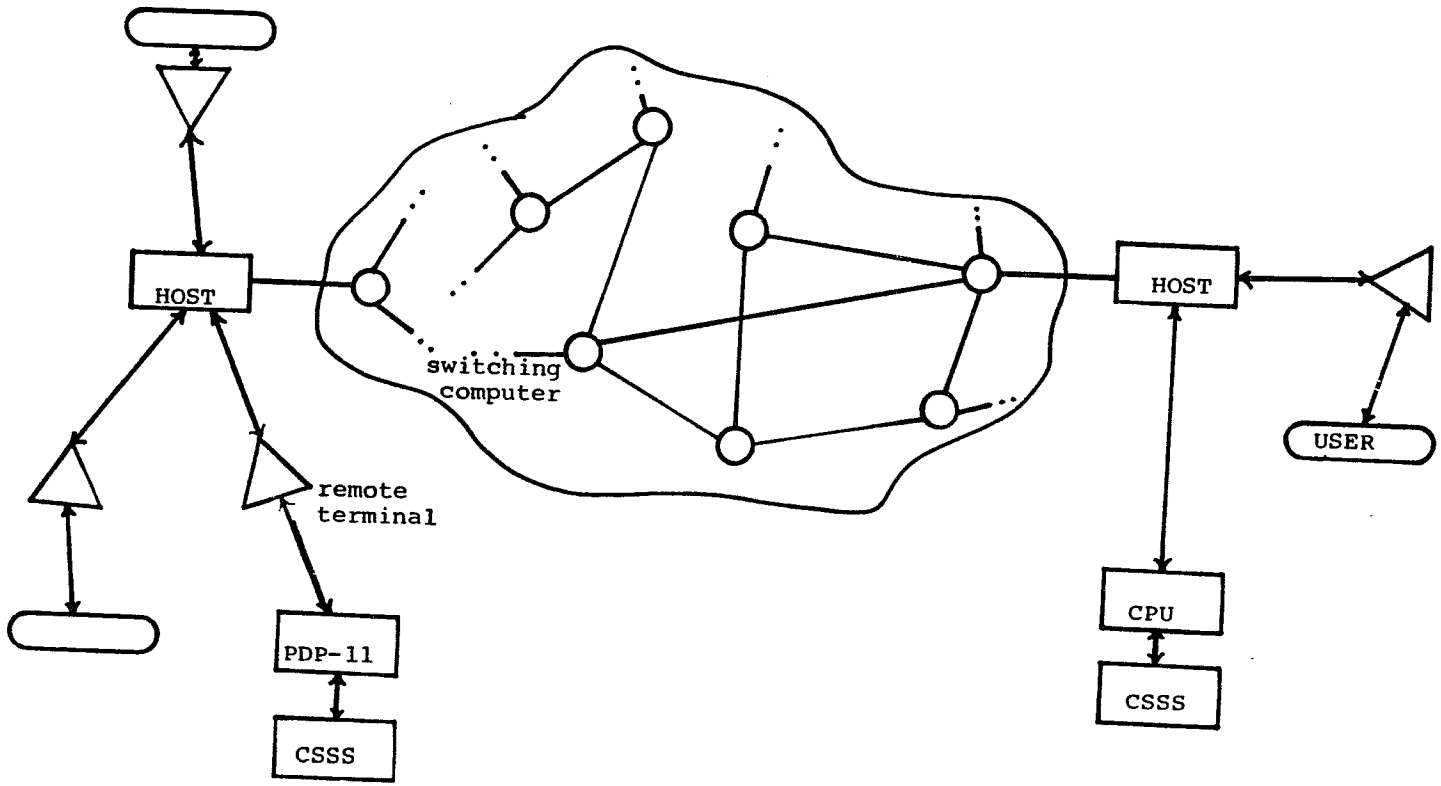


Fig. 4.1 CUNY SLOW SCAN SYSTEM connected to a computer network.

network. In our case, the message will be the compressed E(k) data representing a video image.

The host computer serves as the actual port into the computer network. The host computer will have several network customers attached to it via remote terminals. The function of the host computer is to regulate the messages between the network and the various different remote terminals.

The switching computers represent the nodes within the computer network. Messages are transmitted from one host computer to another host computer via the switching computers. Messages that are to be transmitted through the computer network are divided into smaller messages called packets. Each packet will contain a header indicating its destination, its priority and its place in the sequence of packets that constitute the original message. The message is partitioned into packets before entering the network and each packet is treated independently by the switching computers. The function of the switching computers is to direct the packets to the next switching computer which represent the shortest path to the packets destination at any instant of time. In addition, the host computer is able to detect when a received packet contains an error. Thus, packets that represent portions of the same message can be transmitted over different paths in the network. If

one path is very busy the switching computer will direct packets through another path which may be geometrically longer but because it is not congested the packets may arrive to their destination in a shorter amount of time.

In this investigation, we will assume that each compressed video line is transmitted as a packet over the network. Statistics gathered on the MADM processed pictures reveal that the average packet length is 507 bits/packet with a variance of 105 bits. There are three types of errors that will be discussed:

1- Errors in the uncompressed MADM E(k) data.

2- Packet losses over the computer network.

3- Random bit errors on the compressed E(k) data.

4.1.0 RANDOM BIT ERRORS IN E(K) DATA

Random bit errors in the Song mode ADM E(k) data has been studied by N. Scheinberg and resulted in the use of a leaky integrator in the present ADM encoder-decoder pair. The effects of channel errors and how the leaky integrator reduces their effects is discussed below.

A random channel error on the E(k) data bit stream produces two effects:

1- The step size is changed.

2- A D-C shift is produced in the decoder's estimate which remains after the step size corrects itself.

The two above effects are illustrated in fig. 4.2. The error in the decoder step size will correct itself after a few samples, whenever the step size reaches its minimum value. After the step size corrects itself the decoder's estimate will remain with a permanent D-C shift. In order to calculate the expected D-C shift that a channel error will produce we must find the expected number or received bits that are in error after the occurrence of a channel error and before the decoder's step size corrects itself.

4.1.1 EXPECTED NUMBER OF STEP SIZES IN ERROR

The number of transmitted bits that will reach the decoder after the occurence of a channel error and before the step size corrects itself can be found from the step size state transition diagram shown in fig 4.3. The state diagram in fig 4.3 shows all the possible step sizes that a Song mode ADM can have along with all the possible state transitions that are allowed. In fig 4.3 P_{i-j} represents the probability of going to step size Y(k+1)= j if the present step size is Y(k)= i. If a channel error occurs

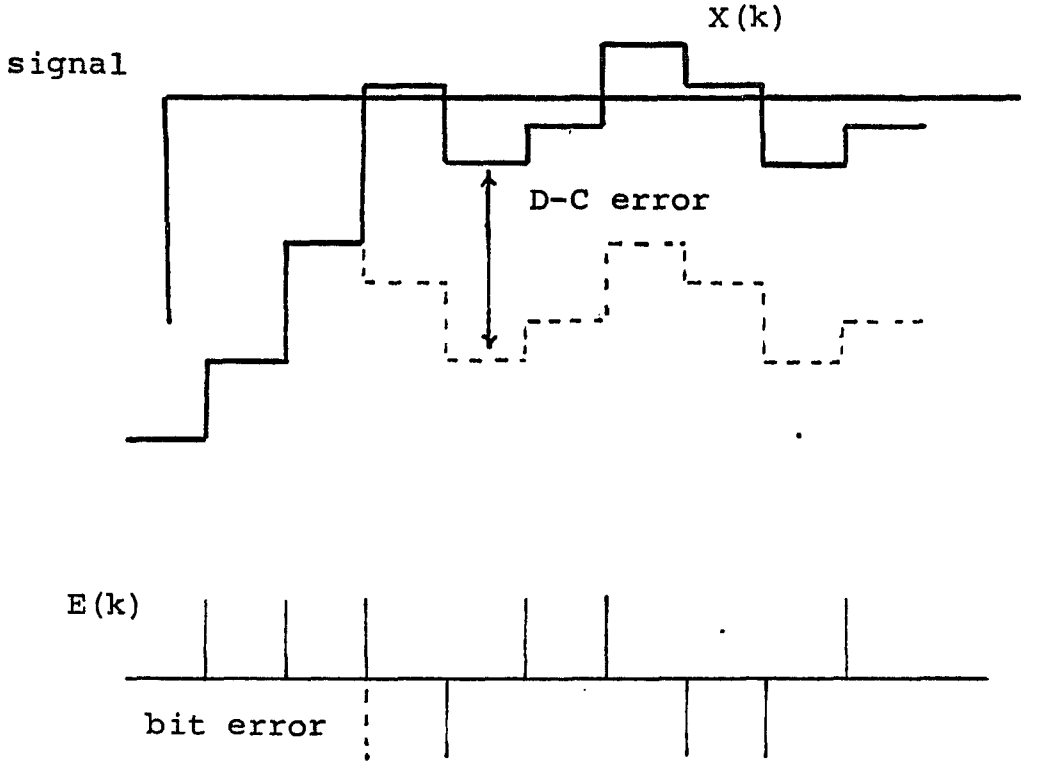


Fig. 4.2 Effect of a single channel error on step size and estimate.

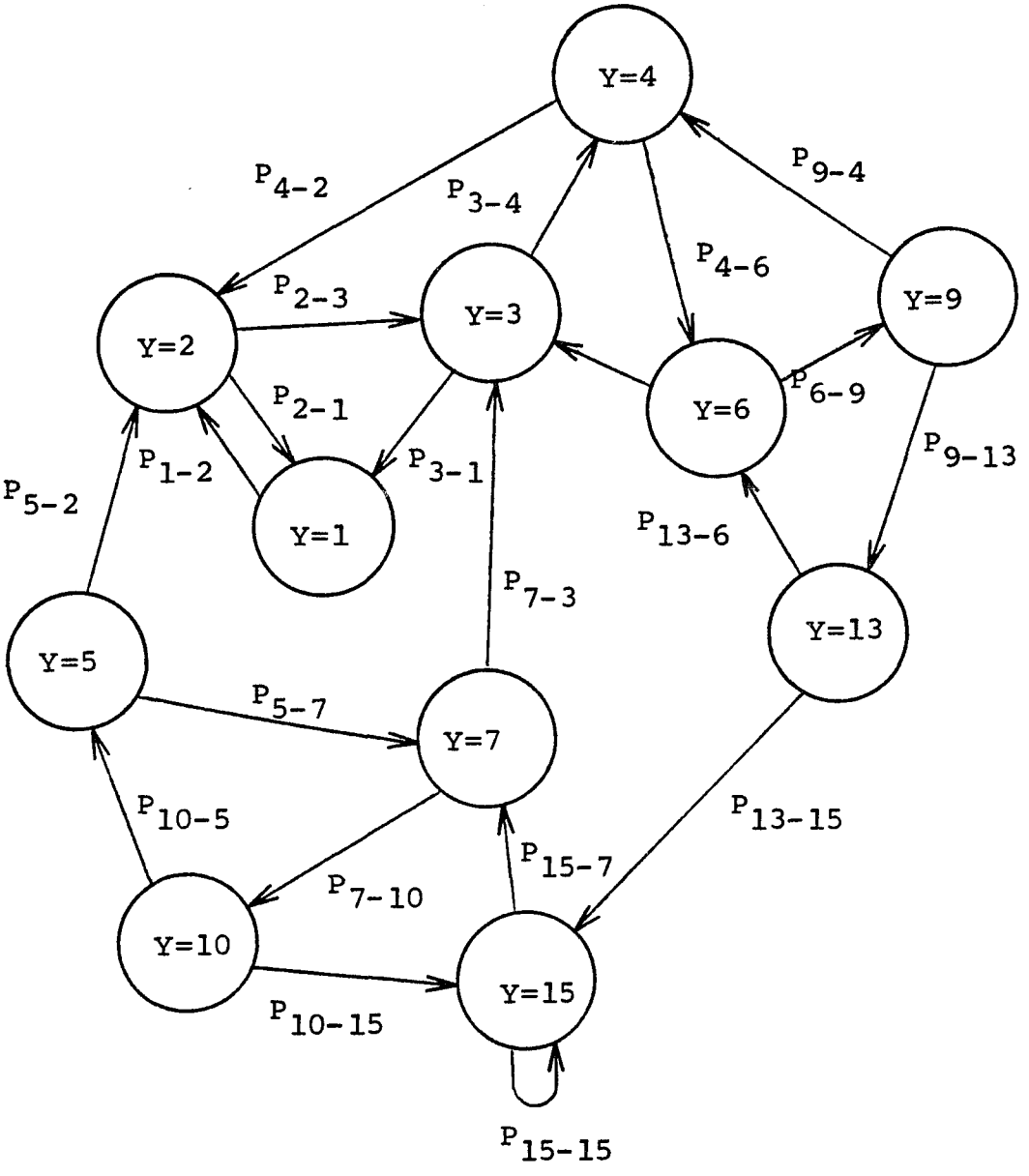


Fig. 4.3A Step size state transition diagram.

causing data bit E(k+1) to be in error then Y(k+1) and subsequent step sizes will be in error until the encoder's step size reaches the minimun step size. If a channel error occurs on a data bit when the encoder's step size is minimum, Y(k)=1, then no error in the decoder's step size will occur since the next step size, Y(k+1)=2, will be reached regardless of the value of the data bit. The average number of bits received after the occurence of a channel error and before the step size corrects itself is a function of the step size just prior to the channel error, Y(k). Thus:

$$4.1.1 \qquad E[N(s)] = \sum_{i=2}^{15} N(s|s=i)P(s=i)$$

where $E[N(s)]$ = expected number of received bits after a channel error and before the step size corrects itself.

$N(s|s=i)$ = expected number of received bits in error given that the step size just prior to the channel error is $Y(k)=i$

$P(s=i)$ = the probability that $Y(k)=i$.

In order to evaluate equation 4.1-1 we must obtain an expression for N(s|s=i). The step size state diagram in fig. 4.3 can be considered as a multiple input system, where the inputs states correspond to the possible step size just prior to the channel error, and the system output state corresponds to the minimum step size. The expected number of step size transitions from a particular input state to the minimum step size state is equal to the average path length from the particular input state in fig. 4.3 to the output state. In order to find the average path length from a particular input state to the output state we multiply each path segment by the variable X where the variable is used to keep track of the number of path segments that are crossed in going from the particular step size to the minimum step size. Then:

4.1-2 $$N(s|s=i)=\frac{d}{dx}\left[T_i(x)\right]\Big|_{X=1}$$

where $T_i(x)$ = the transfer function from input state $Y(k)=i$ to output state $Y(k)=1$.

The expression for $T_i(x)$ can be obtain from the state diagram in fig. 4.3 by using Mason's Rule and is given by:

4.1-3 $$T_i(x) = \frac{1}{\Delta(x)} \sum_{\text{all } n} T_{in}(x)\,\Delta_n(x)$$

4.1-4 $\Delta(x)$ = 1-(sum of transmission of all loops)
+(sum of products of transmissions of all pairs of disjoint loops)
-(sum of products of transmissions of all triples of disjoints loops)
+.....

4.4-5 $\Delta_n(x)$ = 1-(sum of transmissions of all loops that are disjoint from direct path n)
+(sum of products of transmissions of all pairs of disjoint loops that are disjoint from direct path n)
-.....

where T_{in} = n^{th} direct path from input state Y(k)=i to output state Y(k)=1.

$\Delta(x)$ = the determinant of the state diagram.

$\Delta_n(x)$ = the coefactor of the state diagram with respect to the n^{th} direct path.

To evaluate E[N(s)] a particular video image must be

considered. The state transition probabilities were calculated for the MADM images of the BOY and GIRL with the parameters specified in equation 3.4-5 thru 3.4-8. The state transition probabilities are presented in TABLES 4-1 and 4-2. The probability of being in each state is presented in TABLE 4-3. Comparing the step size with those of the Song mode ADM we notice that the difference is that the MADM will generate the maximum step size only a fraction of the amount of time the Song mode ADM does. This indicates that the number of step size in error after the occurrence of a channel error will be less in the MADM decoder than in an ADM decoder. This would lead us to expect that the D-C shift produced by a channel error will be greater for the Song mode ADM than for the MADM. The results of applying the above statistics are presented in TABLE 4-4 and yield the expected number of step sizes that will be in error after the occurrence of a channel error. We notice that on the average the MADM decoder will correct its step size within two clock pulses or one pixel whereas the Song mode decoder will take twice as long. Since the picture of the GIRL contains larger areas of approximate equal grey level we see that the MADM will generate a larger number of steady state E(k) data bits which will keep the step size in the lower size. This results in the MADM decoder receiving a fewer number of E(k) data bit

i \ j	1	2	3	4	5	6	7	9	10	13	15
1		1.00									
2	.802		.198								
3	.441			.559							
4		.419				.581					
5		.215					.785				
6			.574					.426			
7			.699						.301		
9				.581						.419	
10					.939						.061
13						.811					.189
15							.703				.297

P_{i-j}

TABLE 4-1 State transition probabilities for MADM picture of the BOY.

i \ j	1	2	3	4	5	6	7	9	10	13	15
1		1.00									
2	.831		.169								
3	.460			.540							
4		.459				.541					
5		.173					.827				
6			.616					.384			
7			.765						.235		
9				.545						.455	
10					.961						.039
13						.788					.212
15							.675				.325

P_{i-j}

TABLE 4-2 State transition probabilities for MADM picture of the GIRL.

STEP	BOY	GIRL
1	37.98	35.36
2	40.55	38.27
3	9.32	10.61
4	5.79	7.13
5	0.07	0.10
6	3.63	4.84
7	0.30	0.36
8	0.00	0.00
9	1.40	2.06
10	0.07	0.11
11	0.00	0.00
12	0.00	0.00
13	0.64	0.86
14	0.00	0.00
15	0.26	0.29

TABLE 4-3 MADM step size probabilities.

before the step size corrects itself. By assuming that the step size corrects itself in two clock pulses we can calculate the expected D-C shift that the MADM decoder estimate will undergo from a channel error.

GIRL	SONG	4.65
	MADM	1.97
BOY	SONG	5.18
	MADM	2.21

TABLE 4-4 Expected number of step sizes in error due to a single channel error.

4.1.2 EXPECTED D-C SHIFT

To calculate the expected permanent D-C shift that the decoder's estimate will undergo due to a channel error we will assume that the step size corrects itself within two bits after a channel error. If we consider a four bit pattern where the step size corresponding to the first data bit is known and the second data bit is "flipped" due to a channel error then the decoder's D-C shift can be calculated due to the channel error and this particular four bit pattern. In order to calculate the expected D-C shift for a particular image we must calculate the D-C shift produced by every combination of four bit patterns

and all possible step sizes corresponding to the first bit. Then:

$$4.1\text{-}6 \qquad E[S(i,j)] = \sum_{j=2}^{15} \sum_{i=0}^{15} S(P_t=i,Y=j)\,P(P_t=i,Y=j)$$

where $E[S(i,j)]$ = expected D-C shift due to a channel error.

$S(P_t,Y)$ = D-C shift due to pattern P_t where the step size corresponding to the first pattern bit is $Y(k)=j$ and an error "flips" the second pattern bit.

$P(P_t=i,Y=j)$ = the probability of pattern $P_t=i$ occurring with the step size corresponding to the first pattern bit is $Y(k)=j$.

In TABLE 4-5 we present the D-C shift produced by the sixteen possible four bit patterns for every possible step size corresponding to the first pattern bit and the step size added to the LSB of the estimate. If the step

STEP	1	2	3	4	5	6	7	9	10	13	15
PATTERN											
0000	8	9	15	27	31	37	40	42	45	44	45
0001	7	5	7	14	15	20	22	24	27	27	28
0010	7	6	5	8	9	12	12	17	20	22	23
0011	8	6	5	8	9	12	11	17	20	26	30
0100	8	9	15	27	31	37	40	42	45	44	45
0101	7	5	7	14	15	20	22	24	27	27	28
0110	7	6	5	8	9	12	12	17	20	22	23
0111	8	6	5	8	9	12	11	17	20	26	30
1000	8	6	5	8	9	12	11	17	20	26	30
1001	7	6	5	8	9	12	12	17	20	22	23
1010	7	5	7	14	15	20	22	24	27	27	28
1011	8	9	15	27	31	37	40	42	45	44	45
1100	8	6	5	8	9	12	11	17	20	26	30
1101	7	6	5	8	9	12	12	17	20	22	23
1110	7	5	7	14	15	20	22	24	27	27	28
1111	8	9	15	27	31	37	40	42	45	44	45

TABLE 4-5 D-C error produce for 4-bit patterns with initial step size specified.

step size added to the LSB of the estimate. If the step size is added to the 2^{nd} LSB of the estimate then the values in TABLE 4-5 have to multiply by two. The probability of a particular 4-bit pattern with a particular step size corresponding to the first bit were calculated for the MADM images with the parameter specified in equations 3.4-5 thru 3.4-8. The resulting probabilities are presented in TABLE 4-6 . In TABLE 4-7 we present the expected D-C shift that the decoder's estimate will undergo due to a channel error.

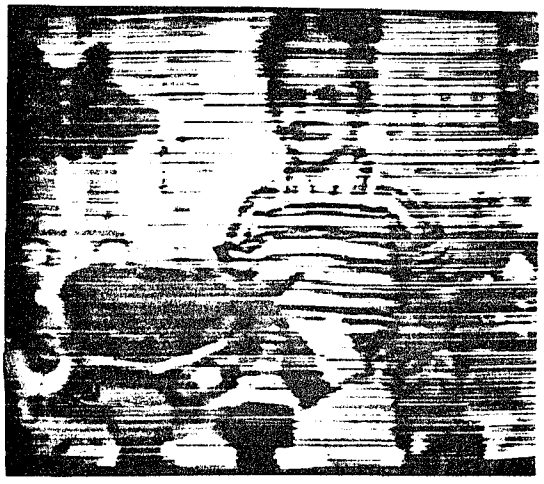
The effect of a channel error on the decoder's estimate results in a permanant D-C shift which manifests itself in horizontal streaks accross the television screen. In fig. 4.3 the picture of the BOY with different channel error rates is shown. Whenever a channel error occurs on a horizontal line the D-C shift in the decoder's estimate will persist until the end of the video line. The error is prevented from propagating to the next horizontal line by resetting the encoder and decoder to a predetermined step size and estimate at the beginning of each horizontal line. Resetting the encoder and decoder to its initial state at the beginning of each horizontal line is accomplished by the horizontal drive. Since the horizontal drive is more negative than the actual video signal, the encoder will transmit negative E(k) bits causing the step size to

STEP	1	2	3	4	5	6	7	9	10	13	15
PATTERN											
0000			0.01	0.01							0.04
0001			0.02	0.01							
0010	0.05										
0011	0.07							0.09			
0100	0.04	0.02									
0101	0.03	0.07									
0110		0.09	0.01								
0111		0.02	0.01								
1000		0.02	0.01								
1001		0.09	0.01								
1010	0.03	0.07									
1011	0.05	0.02									
1100	0.07										
1101	0.05	0.01									
1110			0.02	0.01							
1111			0.01	0.01							

% probability

TABLE 4-6 Probability of 4-bit patterns given that the first bit corresponds to a given step size.

10^{-4} rate (44 errors) 10^{-3} rate (449 errors)



10^{-2} rate (4705 errors)

Fig. 4.3 MADM picture of BOY with channel errors.

increase to its maximum value and the estimate to decreases to its most negative value in both the encoder and decoder.

4.1.3 LEAKY INTEGRATOR

The effects of channel errors on the MADM pictures can be reduced by introducing a leaky integrator in both the encoder and decoder as shown in fig.2-3. If we consider the estimate to be bipolar than the equation describing the leaky integrator is given by:

4.2-1 $X(k+1) = X(k)-X(k)/2 +Y(k+1)$

4.2-2 $X(k+1) = L_f X(k)+Y(k)$

where L_f = the leak factor.

The response of the Song mode ADM with a leaky integrator to a step input waveform can be obtained by considering an initial estimate X_0 and initial step size Y_0. Then

4.2-3 $Y(1) = 1.5Y(0) = 1.5Y_0$

4.2-4 $X(1) = L_f X_0+Y(1) = L_f X_0+1.5Y_0$

4.2-5 $$Y(2) = 1.5Y(1) = 1.5^2Y_0$$

4.2-6 $$X(2) = L_f{}^2X_0+L_f1.5Y_0+Y(2)$$
$$= L_f{}^2X_0+L_f1.5Y_0+1.5^2Y_0$$

4.2-7 $$Y(3) = 1.5Y(2)=1.5^3Y_0$$

4.2-8 $$X(3) = L_f{}^3X_0+L_f{}^21.5Y_0+L_f1.5^2Y_0+Y(3)$$
$$= L_f{}^3X_0+L_f{}^21.5Y_0+L_f1.5^2Y_0+1.5^3Y_0$$

.

.

4.2-9 $$Y(k) = 1.5Y(k-1) = 1.5^kY_0$$

4.2-10 $$X(k) = L_f{}^kX_0+Y_0\sum_{j=0}^{k-1} L_f{}^j1.5^{k-j}$$

$$= L_f{}^kX_0+1.5^kY_0[(L_f/1.5)^k-1]/(L_f-1.5)$$

Since L_f is approximately equal to 1 then

4.2-11 $$X(k) = L_fX_0+2Y_0[1-(L_f/1.5)^k]1.5^{k+1}$$

$$= X_0e^{k\ln(L_f)}+2Y_0[e^{(k+1)\ln1.5}-1.5e^{k\ln(L_f)}]$$

```
SIMULATING MADM: INITIAL APERATURE = 05
                 MAXIMUM APERATURE = 10

LEAKY INTEGRATOR: # OF SHIFTS = 7
ENCODER Y(K) ADDED TO 2ND LSB OF X(K)
STEADY STATE PATTERN: 01100110
```

K	S(K)	X(K)	E(K)	Y(K)
01	300	016	1	07
02	300	042	1	12
03	300	101	1	17
04	300	137	1	17
05	300	175	1	17
06	300	233	1	17
07	300	271	1	17
08	300	327	1	17
09	300	311	0	07
10	300	264	0	12
11	300	276	1	05
12	300	314	1	07
13	300	305	0	03
14	300	275	0	04
15	300	300	1	02
16	300	276	0	01
17	300	272	0	02
18	300	273	1	01
19	300	277	1	02
20	300	275	0	01
21	300	271	0	02
22	300	272	1	01
23	300	276	1	02
24	300	274	0	01
25	300	270	0	02
26	300	271	1	01
27	300	275	1	02
28	300	273	0	01
29	300	267	0	02
30	300	270	1	01
31	300	274	1	02
32	300	272	0	01
33	300	266	0	02
34	300	267	1	01
35	300	273	1	02
36	300	301	1	03
37	300	276	0	01
38	300	302	1	02
39	300	300	0	01
40	300	273	0	02
41	300	275	1	01
42	300	301	1	02
43	300	276	0	01
44	300	302	1	02
45	300	277	0	01
46	300	303	1	02
47	300	301	0	01
48	300	274	0	02
49	300	266	0	03
50	300	270	1	01
51	100	263	0	02
52	100	255	0	03
53	100	245	0	04
54	100	231	0	06
55	100	207	0	11
56	100	155	0	15
57	100	117	0	17
58	100	061	0	17
59	100	077	1	07
60	100	124	1	12
61	100	112	0	05
62	100	074	0	07
63	100	103	1	03
64	100	101	0	01
65	100	075	0	02
66	100	100	1	01
67	100	104	1	02
68	100	102	0	01
69	100	077	0	02
70	100	101	1	01
71	100	105	1	02
72	100	104	0	01
73	100	100	0	02
74	100	102	1	01
75	100	106	1	02
76	100	105	0	01
77	100	101	0	02
78	100	103	1	01
79	100	107	1	02
80	100	106	0	01

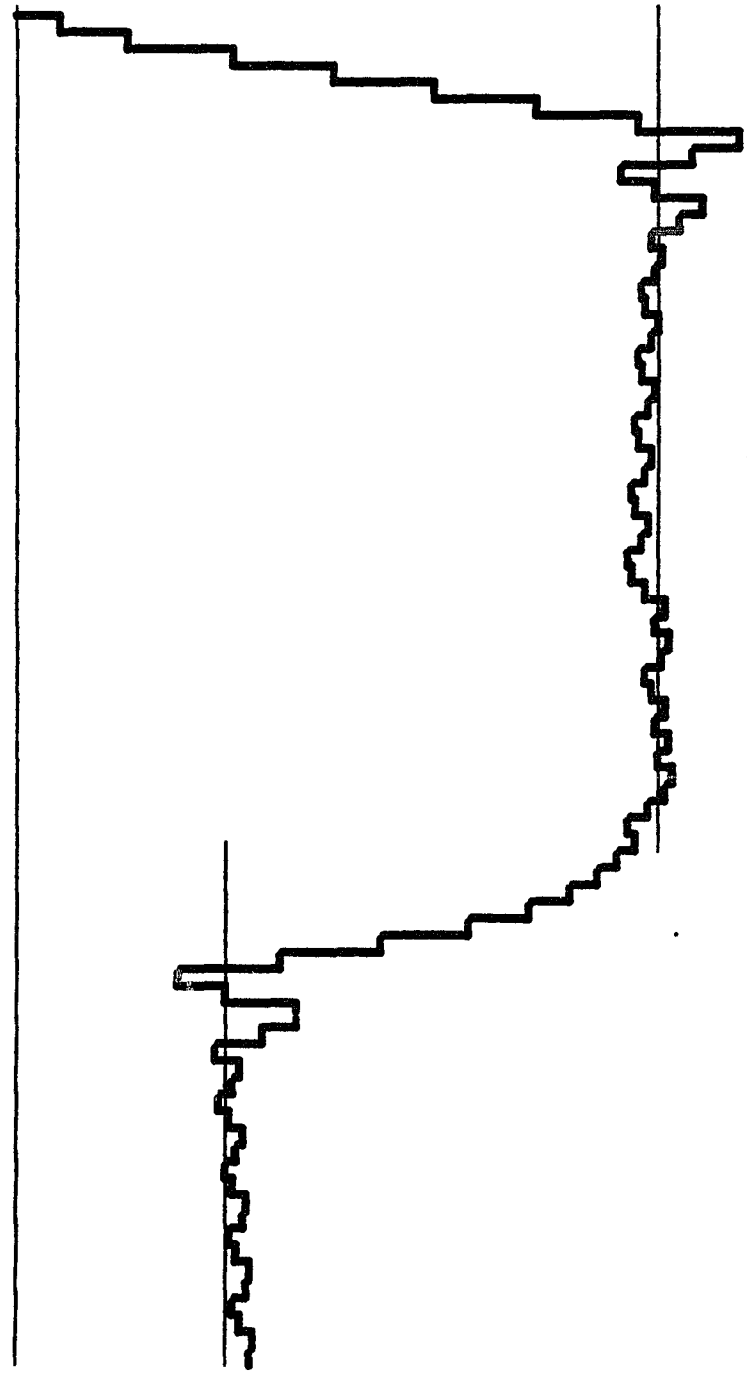


Fig. 4.4 MADM response-1 (leak=.992, 2nd LSB)

```
SIMULATING MADM: INITIAL APERATURE = 05
                 MAXIMUM APERATURE = 10

LEAKY INTEGRATOR: # OF SHIFTS = 5
ENCODER Y(K) ADDED TO 2ND LSB OF X(K)
STEADY STATE PATTERN: 01100110
```

K	S(K)	X(K)	E(K)	Y(K)
01	000	016	1	07
02	000	045	1	12
03	000	106	1	17
04	000	146	1	17
05	000	204	1	17
06	000	242	1	17
07	000	277	1	17
08	000	334	1	17
09	000	313	0	07
10	000	265	0	12
11	000	275	1	05
12	000	311	1	07
13	000	301	0	03
14	000	267	0	04
15	000	272	1	02
16	000	276	1	03
17	000	272	0	01
18	000	274	1	02
19	000	301	1	03
20	000	275	0	01
21	000	267	0	02
22	000	267	1	01
23	000	272	1	02
24	000	266	0	01
25	000	271	1	02
26	000	275	1	03
27	000	303	1	04
28	000	275	0	02
29	000	275	1	01
30	000	300	1	02
31	000	274	0	01
32	000	276	1	02
33	000	272	0	01
34	000	274	1	02
35	000	301	1	03
36	000	275	0	01
37	000	267	0	02
38	000	267	1	01
39	000	272	1	02
40	000	266	0	01
41	000	271	1	02
42	000	275	1	03
43	000	303	1	04
44	000	275	0	02
45	000	275	1	01
46	000	300	1	02
47	000	274	0	01
48	000	276	1	02
49	000	272	0	01
50	000	274	1	02
51	100	271	0	01
52	100	263	0	02
53	100	253	0	03
54	100	242	0	04
55	100	225	0	06
56	100	203	0	11
57	100	151	0	15
58	100	113	0	17
59	100	057	0	17
60	100	077	1	07
61	100	125	1	12
62	100	114	0	05
63	100	100	0	07
64	100	056	0	12
65	100	072	1	05
66	100	112	1	07
67	100	106	0	03
68	100	077	0	04
69	100	105	1	02
70	100	105	0	01
71	100	103	0	02
72	100	077	0	03
73	100	071	0	04
74	100	077	1	02
75	100	107	1	03
76	100	106	0	01
77	100	104	0	02
78	100	110	1	01
79	100	115	1	02
80	100	115	0	01

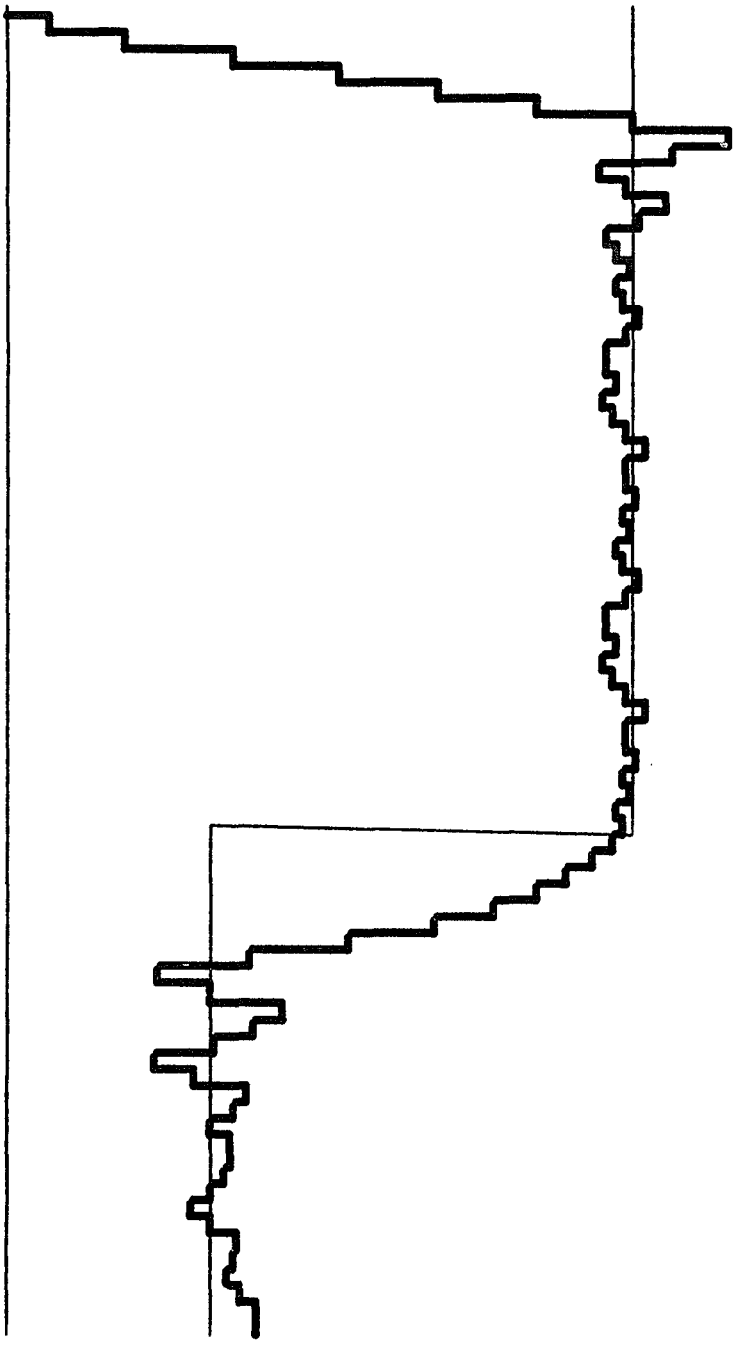


Fig. 4.5 MADM response-1 (leak=0.969, 2nd LSB)

We noticed before that the MADM decoder will suffer a permanent D-C shift in its estimate whenever a channel error occurs. To see how the inclusion of a leaky integrator reduces the effect of a channel we consider that a channel error occurs at time k. Then

$$X(k+1) = L_f[X(k)+e]+Y(k+1) \tag{4.2-12}$$

$$X(k+2) = L_f^2[X(k)+e]+L_fY(k+1)+Y(k+2) \tag{4.2-13}$$

.

$$X(k+j) = L_f^j e+L_f^j X(k)+ \sum_{i=1}^{j} L_f^{j-i} Y(k+i) \tag{4.2-14}$$

We notice from the above equation that the error produced by a channel error will leak away at a rate of L_f^j. From the above equation we notice that the smaller the value of the leak factor the sooner the error will disappear. Computer simulations performed showed that the smallest leak factor that can be used without introducing annoying granularity into the MADM video image is L_f=0.992. The granularity results from the larger step size that result from using a leaky integrator. In figs. 4.4 and 4.5 we show the response of the MADM with a leaky integrator to an input square wave. The leak factors in the figures correspond to leak factors of 0.992 and 0.969. From these

responses we notice that if the leak factor is reduced the amount of data compression will also be reduced because the leaky integrator will depart from the input signal at a faster rate. Another effect that the leaky integrator will produce is that long run-lengths of the steady state pattern will become unlikely even if the input signal is constant. From the manner in which the leaky integrator is implemented we notice that the above is only true for input signals that have a magnitude greater than 37-octal(if bipolar signals are considered and a leak factor of 0.992 is used) because for smaller values the result of shifting seven places to the right, see fig. 2.3, will produce a zero value. In fig. 4.6 we show the effects of the leaky integrator on the MADM picture of the boy and using a leak factor of 0.992. We notice that the effects of the channel errors are reduced and the D-C error is seen to leak away. Because the leaky integrator is implemented using a 10 bit register the error will not always leak off completely. We also observe that the streaks due to contour noise are reduced because the leaky integrator will introduce perturbations about the estimate which tends to break up some of the contour noise. The contour noise is also reduced because the leaky integrator will tend to reduce the generation of long sequences of the steady state pattern. The use of a leaky integrator reduces the amount

NO INTERPOLATION

FIELD INTERPOLATION

Fig. 4.6 MADM pictures of the BOY with leaky integrator (leak factor = 0.992) and 10^{-3} error rate.

of data compression by about 2-3% but the loss is out-weighted by the benefits of the leaky integrator. In the following discussions, unless otherwise stated, the MADM will be simulated with the parameters specified in equations 3.4-5 thru 3.4-8 and a leaky integrator is used with a leak factor of 0.992.

4.2.0 MADM PACKET LOSS

The MADM E(k) data bits that are stored in the FRAME FREEZE unit is transferred to the PDP-11/34 for compression, using the block code discussed in SECTION 3.4.1, and transmission over the computer network. Transmission is accomplished by transmitting each compressed video line as a packet. Even though transmission over the computer network is usually reliable there are instances where packets are lost due to a defective node. One means of guarding against packet loss is to have the receiving host computer to transmit a RECEIVE ACK to the transmitting host computer. If the transmitting computer does not receive an ACK signal from the receiving host computer it will retransmit the lost packet again. In the case of video information this technique may not be practical because the video image may no longer be present for retransmission. Thus, other

30% LOSS (147 packets lost)

40% LOSS (201 packets lost)

Fig. 4.7 Direct substitution on MADM picture of the BOY with packet loss.

means of handling packet loss should be considered.

Video packet loss over a computer network is equivalent to losing a line of video. In the previous chapters it was shown that a good quality picture can be obtained by transmitting only one field and interpolating the second field at the receiver. Thus, if 512 lines of video are transmitted and the lost packets are replaced by interpolating the analog samples of the two adjacent video lines the results would be similar to that obtained when field interpolation is performed. In order to perform the above interpolation the receiving host computer must contain a programed MADM encoder and decoder. If the host computer does not contain such a program, then another means of generating the lost packet information is by direct substitution. In this method the lost packets are replaced by an adjacent packet. Direct substitution results in a video line being displayed twice. The result of applying direct substitution is shown in fig. 4.7 for the MADM picture of the BOY for different percent packet loss. The resulting pictures show the effect of displaying a number of lines twice. We see that for a 40% packet loss rate that those lines that are displayed twice create the appearance of flattening the image and extending any horizontal edge in the vertical direction. This effect becomes more pronounced if more than two consecutive

packets are lost because then that area of the picture will appear as a strip.

If a transmitted video picture is to be field interpolated then a packet loss will not only effect the lost line but also those lines in the second field that require to the lost line in their interpolations. In fig. 4.8 we illustrate the effects of interpolating the missing packet or applying direct substitution on a missing line prior to interpolating the missing field. The two MADM pictures using both these methods were simulated on the computer. We noticed that field interpolating the missing packet produces the better results and does not contain the "flat" effect that is evident when using the direct substitution method. This should be expected from fig. 4.8 where we see that the interpolated lines in the second field are the weighted sum of the two closest received lines. In the direct subsitution method there will be three identical lines, two in the first field and one in the second field, which are adjacent to each other. If field interpolation is to be used to transmit a MADM picture, then there is no additional overhead introduced by interpolating the lost packets since the receiving host computer will already contain the program needed to perform the interpolation. From the simulation results we conclude that direct substitution can be used effectively if the

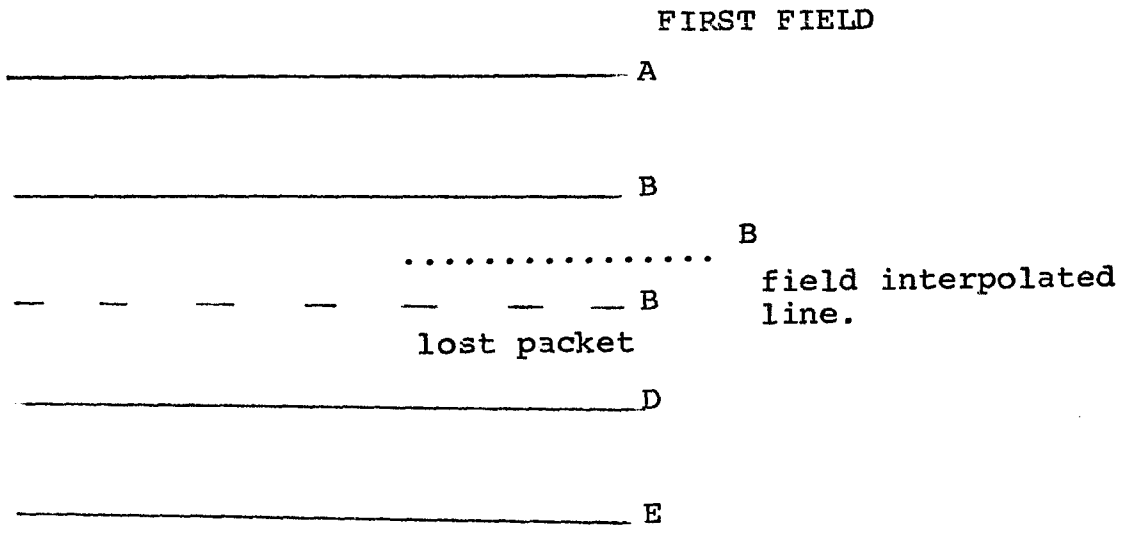


Fig. 4.8 Effect of field interpolation on direct substitution.

packet loss rate is under 20%; at higher packet loss the image degradation becomes severe. Interpolation, on the other hand, is seen to provide the best means of dealing with packet loss.

4.3.0 RANDOM BIT ERRORS IN COMPRESSED DATA

There are two types of compressed bits that can be complemented due to a channel error: a data bit and a header bit. If a data bit in the compressed data is inverted due to a random channel error the results are identical to those discussed in SECTION 4.2.0. The situation becomes more complicated if the channel error occurs on a header bit. The initial results of a header bit error is that either a 4 bit steady state pattern is inserted into the uncompressed data, when an error occurs on a logic "0" header bit, or a 4 bit steady state pattern is deleated from the uncompressed data, when an error occurs on a logic "1" header bit. The initial effect is the displacement of the resulting uncompressed data by four bits, or two pixels. This in itself does not present a major problem. The more severe problem results from the fact that now the CPU has lost synchronization and subsequent data bits will be mistakenly interpreted as header bits and header bits will be used as data bits.

This results in the uncompressed data containing errors and the generation of either more or less data bits than were transmitted. If the compressed were data were transmitted as one record then the effects of an error on a header bit would ripple through the entire frame of video. In order to minimize this effect and keep the effects of a channel error from propagating from line to line each compressed video data is transmitted as a packet. Thus, we need only consider how to deal with channel errors on header bits within a packet.

Each compressed video packet contains both data and geader bits. errors on the data bits can effectively be reduced by incorporating a leaky integrator in the MADM. The problem now becomes a question of how to deal with errors on any of the header bits. One approch is to use an error correcting code on the 232 header bits that are contained in each packet. If we consider our channel to be a binary symmetric channel then in order to identify and correct a single channel error in a N-bit code the check bits must be able to specify (N+1) things: whether or not a channel error has occured and, if it did, in which of the N bits did it occur. Hence, the following ineqsuality must be satisfied:

4.3-1 $$2^{(N-K)} \geq (N+1)$$

where N = # of bits in the code.

K = # of data bits.

N-K = # of check bits required.

This technique is inpractical not only from the loss of data compression which results from including the check bits but more important in its inplementation. In addition to the check bits transmitted to protect the 232 header bits information has to be included to specify which 232 bits in each packet is being coded since their location will vary from packet to packet.

It has been shown in previous section that a line of video can be estimate by interpolating its two adjacent lines. Thus, a mean of handling channel error on header bits in the compressed data is to disregard those packets that are determined to contain channel errors in any of its header bits. If channel errors are considered to be independent, then the probability that a packet will contain an error in one of is header bits is given by:

4.3-2 $$P\{error\} = \sum_{x=1}^{232} \binom{232}{x} p^x \, (1-p)^{232-x}$$

where x = number of header bits in error.

p = probability of header bit being in error.

The above equation can be written as:

$$4.3\text{-}3 \qquad P\{error\} = 1 - \binom{232}{0} p^{0} \, (1-p)^{232-0}$$

For a 10^{-3} bit error rate this yields that a packet will have a header bit in error with a probability of 21%. We have shown in previous sections that simulation results with hegher probability of packet loss have yield satisfactory picture quality by using field interpolation. Thus, this leads us to belive that interpolating those packets that have been determined to contain errors in their header bits will prove to work satisfactory.

We now consider the means of detecting when an error has been cause on a header bit. We notice that when a header bit is inverted due to a channel error that there will be generated more or less data bits than were transmitted. By having the receiving computer count the number of bits that are produced when decompressing a packet we will be able to detect when a header bit is in error. When a packet is found to contain a header bit in

RECEIVED IMAGE

DIRECT SUBSTITUTION
(when header bit error detected)

Fig. 4.9 10^{-4} error rate on compressed data (26 bit errors, 8 header bit errors detected).

error, it is treated as a lost packet and is estimated by either interpolating the two closest lines or by direct substitution. To see how effective this technique is we simulated on the computer random bit error on compressed video data. The results of these simulation are presented in fig. 4.9 where we present the effects of channel error on the compressed data without replacing any packets and by interplating those packets that have been determined to contain channel errors on header bit. We notice that there is a marked inprovement on the resulting picture. In one case the received image is highly distorted due to errors on header bits while the processed picture does not contain the horizontal distortion associated with errors on header bits. Both pictures are seen to contain the effects of errors on the data bits.

CONCLUSION

We considered the effects of errors on the MADM pictures and found that by including a leaky integrator the MADM could operate at much higher error rates. Errors were also shown to be reduced by interpolating. Errors in the computer network were seen to be reduced by interpolating those packets that were either lost or were determined to contain error on header bits.

5.0.0 CONCLUSIONS

This study has shown that the present Song mode ADM does not produce sufficient redundancy to achieve data compression. We have presented the Modified Adaptive Delta Modulator algorithm which was shown to produce enough redundancy to achieve 40-50 % data compression and acceptable amount of degradation in picture quality which took the form of contour noise. We were able to increase the amount of data compression to about 70-75% by employing field interpolation on the resulting MADM data and produced tolerable degradation in picture quality. The MADM forces the production of the redundant steady state pattern at predetermined locations so that data stored in the frame freeze can be accessed by a CPU for compression and transmission over a computer network. By using a simple 4-bit block code the aquisiton and compression of the MADM data stored in the frame freeze can be performed by a simple processor and does not require that statistics be generated on the entire frame of video as is the case for the Huffman code. Using 4-bit blocks to perform the line partitions and the coding were shown to produce better results than using a larger block size. We also showed that the hardware implementation of the MADM algorithm is

possible by modifing the existing ADM hareware.

The subject of channel errors was also considered. The used of a leaky integrator was demonstrated to reduce the effects of channel errors on data bits. Packet loss over the computer network were also considered and their effects were shown to be reduced by interpolating the missing packets or by using direct substitution. Direct substitution performs well when the probability of a packet loss is under 20% while line interpolation works well up to 40-50% packet loss. We showed that the effects of channel errors in header bits of the compressed data can be reduced if those packets which can be determined to contain header bits in error are treated as lost packets. We showed that an effective means of determining if a header bit is in error is to assume an error if the total number of decompressed data bits don't equal the number of transmitted data bits per line.

Future research is needed in the area of real time reduced frame video transmission over computer networks. If we assume a 45% data compression then by employing field interpolation we can transmit at the reduce rate of 4.5frames/min. over a 9.6 Kb/sec line. This rate can be increased if two and three dimensional coding is used.

THE CUNY SLOW SCAN SYSTEM

A.0.0 INTRODUCTION

The CUNY SLOW SCAN SYSTEM (CSSS) is a television system that will interface with the ARPANET to provide a slow scan image transmission capability. The CSSS uses Adaptive Delta Modulation (ADM) as the source encoder and operator at 16 MHz. The CSSS includes all the hardware necessary to be used both under manual (local or modem) control or under CPU (PDP-11/34) control. An overview of the CSSS is shown in fig. A.1

A.1.0 SYSTEM DESCRIPTION

The CSSS block diagram is shown in fig. A.2

1- Camera: standard NTSC B/W camera.

2- Monitor: standard NTSC compatible B/W monitor.

3- ADM Encoder: Delta Modulates the input video signal at 16MHz rate.

4- ADM Decoder: Converts the data bit stream from the Frame Freeze (FF) into an analog signal that

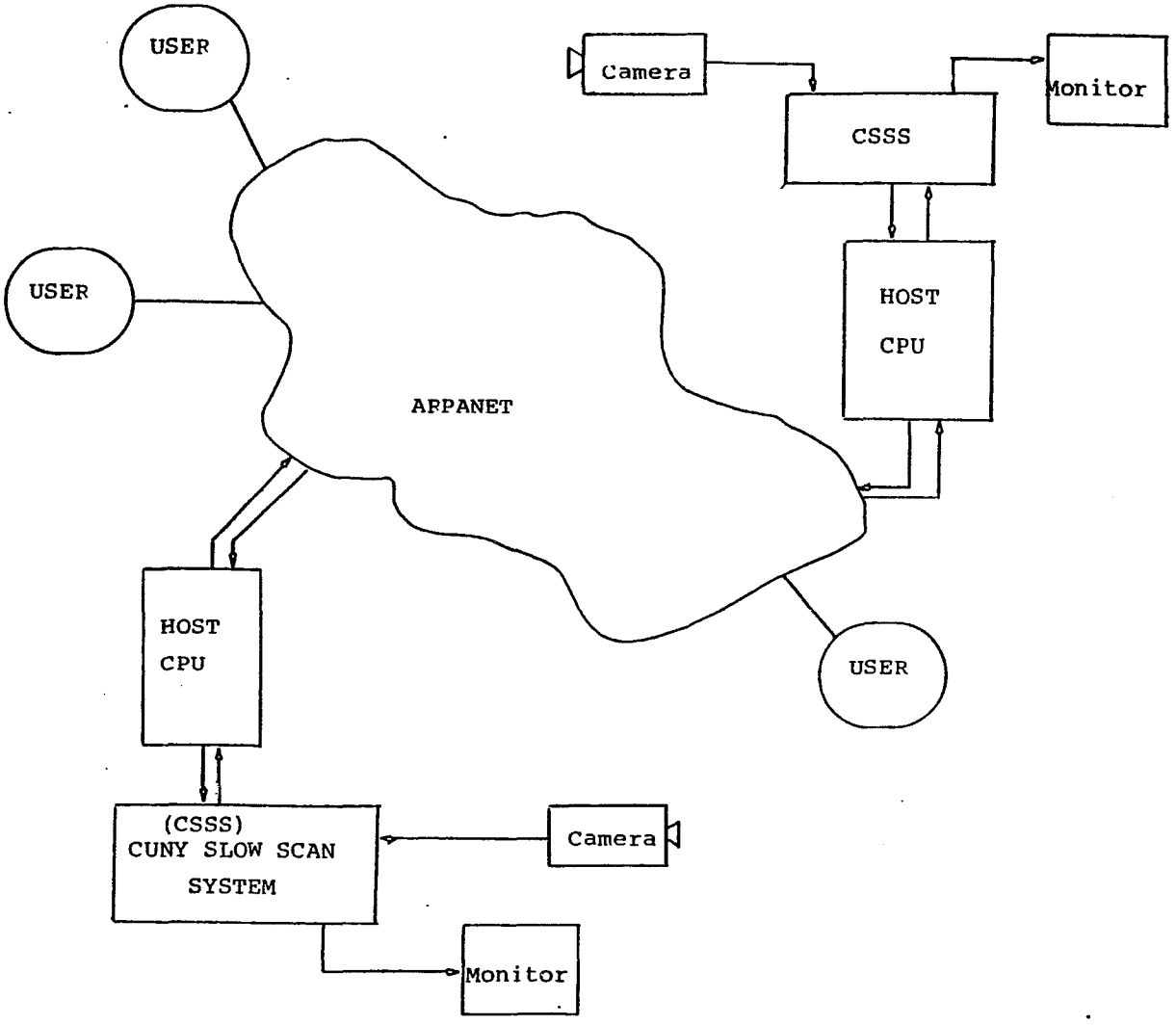


Fig. A.1 CUNY SLOW SCAN SYSTEM Overview.

is to be displayed on the monitor.

5- Frame Freeze: The frame freeze stores one frame (2 Fields) of Delta Modulated video.

The CSSS can operate in three distint modes which are selected by a four position rotor switch. There is one position open which can be used to define a future operation. The three modes of operations are described below.

A.1.1 MANUEL MODE (rotor switch R-1)

In the manuel mode the frame freeze (FF) unit is under the control of a local operator. Any computer command transmitted to the FF will be ignored when the FF is under manuel control. There are two operations that may be performed in the manuel mode which are described below.

1) VIEWING OF LOCAL IMAGES (toggle switch S-1): When in the manuel mode one position of S-1 will allow the FF to continously store the ADM encoded video data into the FF memory. By employing a WRITE-READ memory cycle in the FF memory we continuously output the E(k) data bits that are being stored in the memory to the ADM decoder for viewing on the monitor.

2) "FREEZING" OF LOCAL IMAGES (toggle switch S-1): When in the manuel mode the other position of S-1, see above section, will discontinue the WRITE signal into the FF memory. Thus, any image stored in the FF memory will be continuously read from the memory to the ADM decoder for viewing on the monitor. In this mode the ADM encoder can be disconnected from the system.

A.1.2 CPU MODE (rotor switch R-2)

The CPU mode is selected by the second position of the rotor switch. When an image is stored, it can be read into the PDP-11/34 disk memory for processing and transmission over the computer network. Images received from the computer network can be stored in the PDP-11/34 disk memory and then transferred to the FF for viewing. Communications between the PDP-11/34 and the FF is through a pair of DR11-K interface units and are described below.

OUTPUT-1: The lower 15 bits of this output port provide the 15 bits of address needed for the FF memory. The 16th (MSB) output bit is used to indicate whenever the data on OUTPUT-2 represents CPU status information.

INPUT-1: This input port is used to transfer the 16 bit data word addressed by OUTPUT-1 into the computer whenever the CSSS is in the CPU mode and the CPU read status bit is enable.

OUTPUT-2: This output port is used to write a 16 bit data word into the FF memory at the location specified by OUTPUT-1 when in the CPU mode and the CPU write status bit is enable. This output port will also provide the status inforation when in the CPU mode. The status is indicated when the MSB of OUTPUT-1 is enable. The status word is latched and stored in the LOCAL CARD of the FF unit.

INPUT-2: At present there is no need to use this input port. Future use could be to indicate FF status information.

A.1.3 MODEM MODE (rotor switch R-3)

The third setting of the four position rotor switch selects the modem mode. In the modem mode any image "frozen" in the FF memory can be transmitted over unconditioned telephone line through a synchronous modem to

a remote FF. The transmission rate is determined by the clock rate of the synchrouous modem. The transmission rate is limited by the telephone channel. At present we transmit at a rate of 4800 BAUDS over local telephone lines and 2400 BAUDS over long distance telephone lines. If dedicated telephone line are to be used then the BAUD rate is limited to sixteen times the horizontal drive frequency or approximately 2400 BAUDS. If a higher transmission rate is required then the BAUD rate can be increased to any rate under 16 MHz. by making a simple modification. To operate in the modem mode the followng manuel switches must be set:

1) MODEM RECEIVE/TRANSMIT (toggle switch S-2): This control only works when the FF is in the MODEM MODE. This switch indicates whether the modem is transmitting or receiving data from the telephone line.

2) MANUEL FREEZE ON/OFF (toggle switch S-1): This switch must be set to the appropriate state depending on the setting of the modem receive/transmit switch. If transmitting data the manual freeze switch should be in the ON position so that the video image remains "frozen" in the FF memory while it is being transmitted over the telephone line. If receiving data from the modem the manual freeze

switch should be set to OFF so that that the receive data can be written into the FF memory.

3) MODEM INIT/RESET (toggle switch S-3): Initially this switch will be set to the position RESET regardless of the modem function. If the modem is to receive the video information S-3 can be toggled to the INIT position to await the reception of the transmitted video data. If the modem is to transmit the video information then S-3 will be toggled to the INIT position whenever the operator wishes to initalize data transmisson.

A.2.0 FRAME FREEZE HARDWARE

The FF unit is a stand alone unit which contains two 32K by 8-bits memory cards, a modem card, a local card, and a synch. card. The FF unit provides all the neccessary signal need to synchronize the television camera and monitor and also provides the clock signal required for the ADM encoder and decoder. The operations and hardware inplementation of the various FF cards are discussed below.

A.2.1 SYNCH. CARD

The synch card provides all the system clocks and video synch signals needed to synchronize the CSSS. The synch card is shown in fig A.2. The 16 MHz. system clock is provided by the 16MHz crystal. The 2.04545 MHz clock is required by the MM5321 television camera synch generator. The outputs of the MM5321 that are used are the composite synch, the horizontal drive, and vertical drive. The voltage range for these signal is +5 to -5 volts. The LM311 are used as buffers and also to produce the proper voltage levels required for various synch signal; a -5 volt synch level is required for the television monitor and camera while a TTL level is required for the FF circuits. The televison monitor can be synchronized by the composite synch signal. The televison camera used does not have a direct external synch input. However, the television camera can be externally synchronized by applying an external horizontal and vertical drive which have a -5 volt pulse.

A.2.2 LOCAL CARD

The local card is partitioned into several sections; control section, addressing section, input section, and output section. The various section are described below.

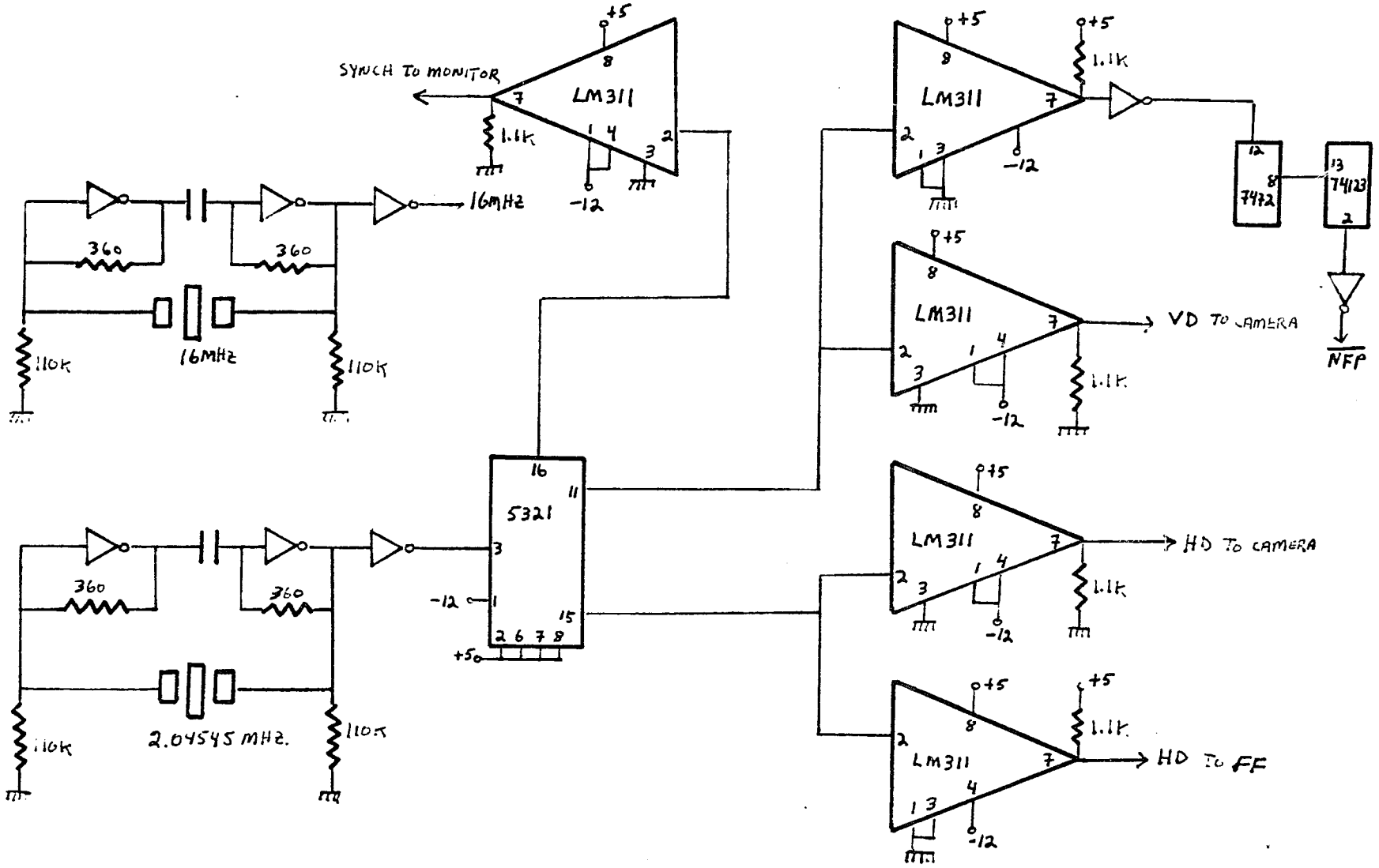


Fig. A.2 FRAME FREEZE Synch. Card

1) CONTROL SECTION: The control section of the local card is shown in fig A.3. The function of the control section is to provide the READ and WRITE to the memory cards and regulate the transfer of data from the FF and the external world. Since we employ two 32K by 8-bits memory cards, we store the ADM video data 16 bits at a time. The ADM encoder operates at 16 MHz thus the E(k) data will be stored at a 1 MHz rate. In order to partition the one micro second cycle time allowed for the processing of each 16 E(k) data word the 16-bit down-counter, IC A1, and the two 8-bit shift registers, IC's A2 and A3, are used.

The 16 bit down-counter will divide the clock rate by sixteen. Thus, the borrow output of the down-counter will produce a locic "0" singal every sixteen clock pulses. Since the down-counter's borrow signal is clocked into the 16-bit shift registers, two 8-bit shift registers, at the 16 MHz clock rate only one bit in the shift register will be a logic "0" and the rest at logic "1". Thus, a "walking zero" is used to initiate the various control signals. The down-counter is reset by the horizontal drive so that no E(k) data will be stored while the horizontal drive is low.

The computer status word is stored in register chip A6. The computer status word is used by the CPU to control the operations of the FF when in the CPU mode. If the FF is not in the CPU mode the CPU status word is ignored. The

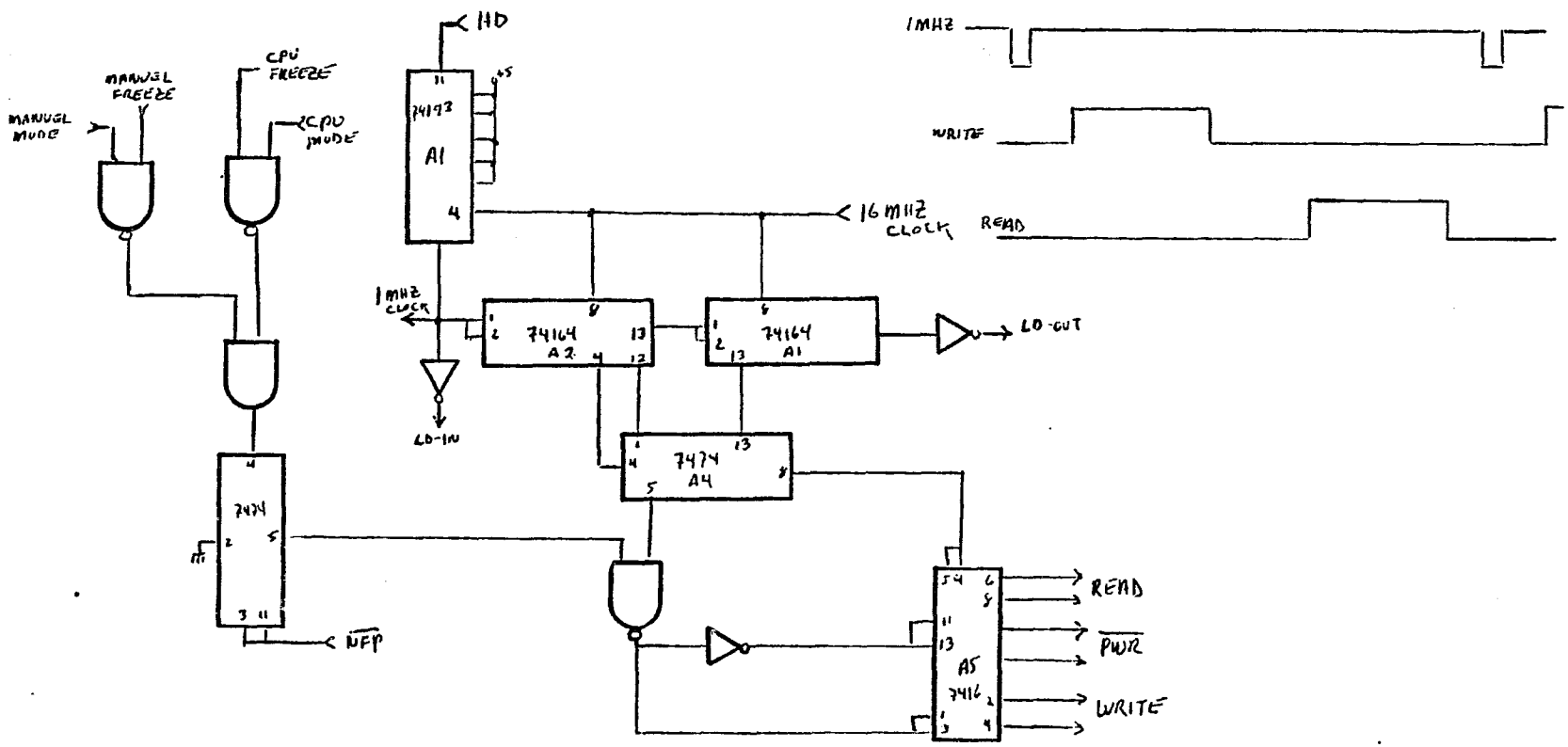


Fig. A.3 Control section of local card.

CPU status word is provided by the CPU via OUTPUT-2 and is latched into the register by the MSB of OUTPUT-1. The status bits are described below:

A) CPU FREEZE: This status bit is used by the CPU to freeze any local image being viewed. This allows the computer operator to view an image on the monitor and freeze it in memory whenever he wants to.

B) CPU I/O: This bit is used by the CPU to indicate that it is accessing the FF memory. This bit is used to blank the monitor screen during the I/O operation.

C) CPU READ This status bit indicates that the CPU is accessing the FF memory and is reading a frame of video for processing.

D) CPU WRITE This status bit indicates that the CPU is accessing the FF memory an is writing a frame of video into the FF memory.

A timing diagram is included with fig. A.3. The one micro second cycle time is divided by the system clock into sixteen intervals which correspond to the 16 outputs of the shift registers. The FF address is changed at the begining of the cycle and coincides with the loading of the input data word. The input data is first written into the FF memory and then read out to the output latches to be serial shifted to the ADM decoder for viewing on the monitor. We

notice that the manner in which the cycle time is partitioned that the various control signals will be initiated at their proper time regardless of the system clock rate.

2) ADDRESS SECTION: The FF memory address is generated by the down counters as shown in fig. A.4. The memory address is divided into two parts; 1) A horizontal address which can select one of the fifty-eight possible 16-bit data words which composes a line of video. 2) A line address which selects one of the 512 lines that composes a frame of video. The address counters are driven by the 1 MHz clock, which is derived by IC A1 in the control section, whenever a local image or a "frozen image" is being viewed. Address lines A_0-A_5 are used to specfied the horizontal address and are reset when the horizontal synch is low. The horizontal address will proceed to count when the horizontal synch signal goes high and in so doing only data pertaining to the actual video picture will be stored. The horizontal drive will also drive the down counters, IC's B6-B8, which contain the line address, A6-A
14. Thus, the line address will change with every new video line. In order to store the frame of video at a specific location in memory the line address is reset every two field which is specified by the NFP.

In order to allow the computer or the modem card to

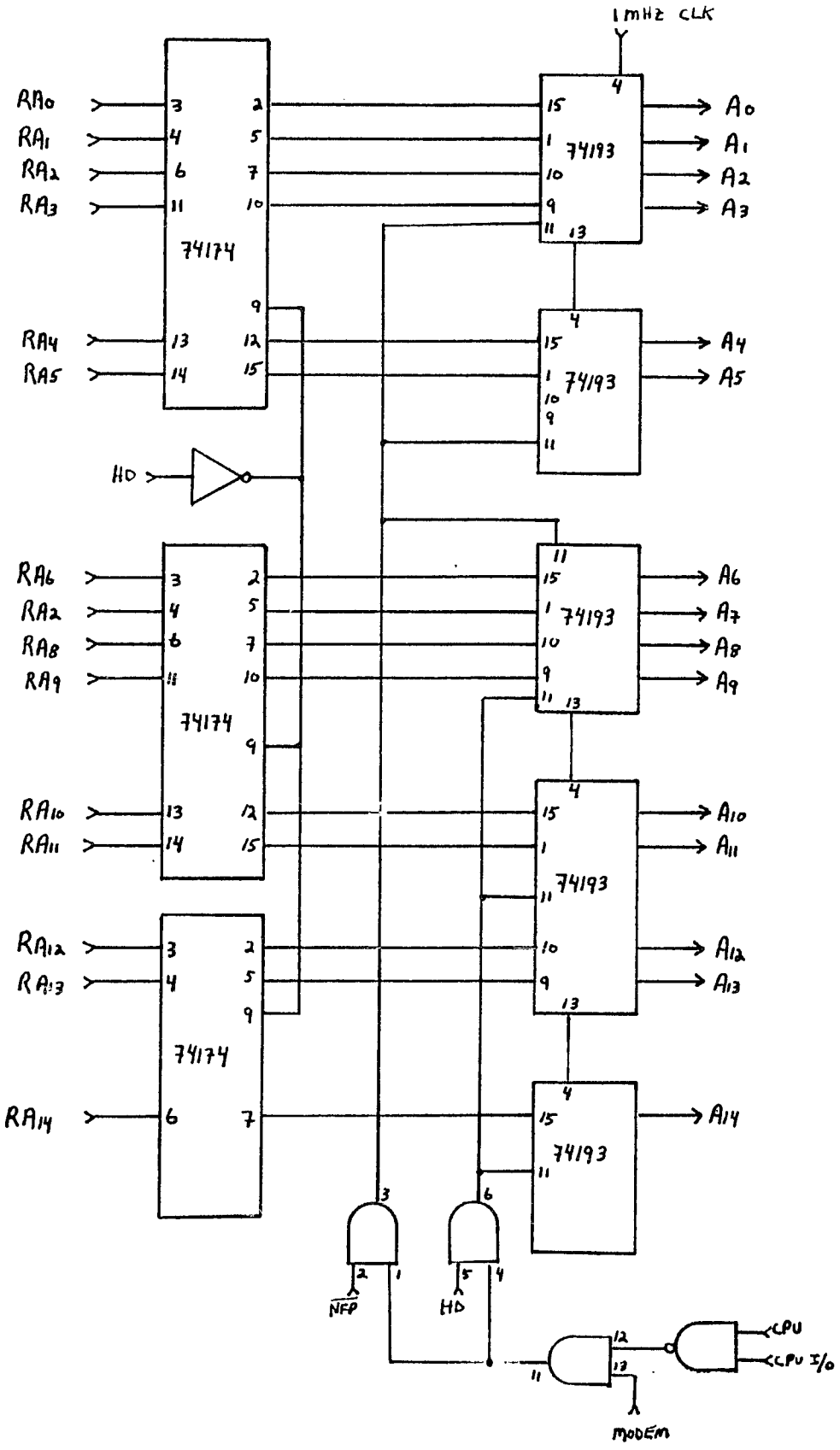


Fig. A.4 Address Section of LOCAL CARD

have access to the FF memory the address counter can be made to look transparent to any address stored in latches B1-B3 by enabling the counter's reset pin which is done every horizontal drive. Latches B1-B3 are loaded via a 16-bit ribbon cable from the modem card. The address presented to the latches is multiplexed from either the computer generated address or the modem card generated address.

3) INPUT SECTION: The input section of the local card is shown in fig A.5. This section allows the E(k) data bits generated by the ADM encoder to be stored in the FF memory at the location specified by the Addressing section. The ADM hardware is build using ECL devices, thus, the input data bits are converted to TTL level by IC C2. Since the data is to be stored in 16-bit words the data is shifted into two 8-bit serial shift registers and latched at the proper time.

The IC's C8-C11 are 2-to-1 multiplexers. The multiplexers provide the capability of storing the data produced by the ADM encoder or data provided by an external source. The external source can be either the computer or the modem.

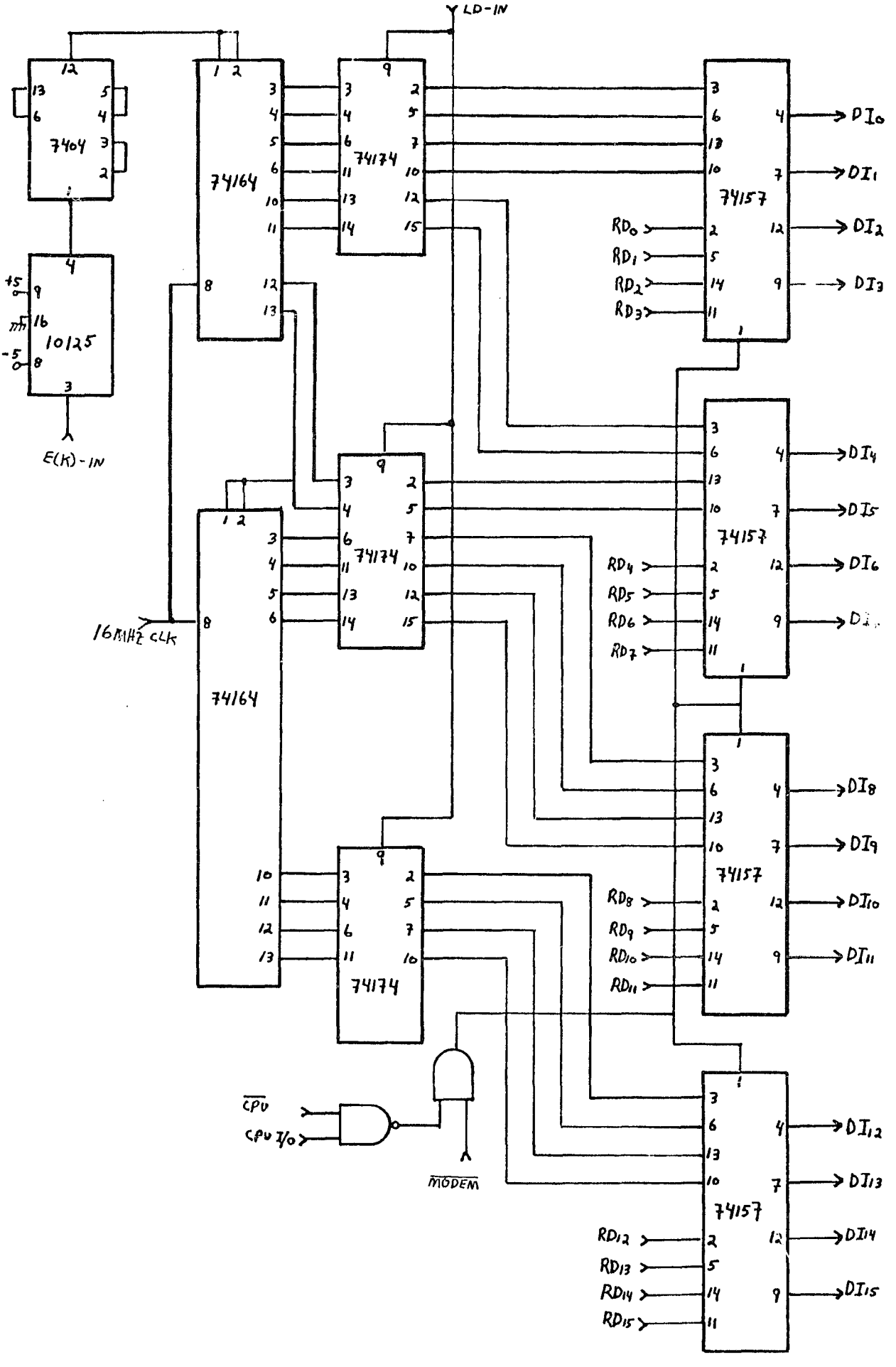


Fig. A.5 Input Section of LOCAL CARD

4) OUTPUT SECTION: The output section of the local card is shown in fig A.6. The output data from the memory location specified by the addressing section is latched into IC's D1-D4. The latched data can be transfered to a parallel-in seial-out shift registers, IC's D9 and D10, and clocked out to the ADM decoder for viewing on the television monitor. The multiplexer on the output is used to output to the ADM decoder either the E(k) data or else a logic "0" signal. The logic "0" signal serves two purposes: 1) When displaying images on the television monitor the ADM decoder must receive data during the horizontal drive to reset the state of the decoder. Since the FF does not store the E(k) data produced by the ADM encoder during the horizontal drive we must generate the required E(k) data. Since the horizontal drive signal is more negative than the video signal, during the horizontal drive the ADM encoder will produce -1, logic "0", E(k) data bits which results in the encoder having a maximum step size and a minimum estimate at the beginning of each video line. By multiplexing a logic "0" to the ADM decoder during horizontal drive we reset the decoder step size to its maximum value and the estimate to its minimum size at the beginning of each video line. 2) When the FF is either in the CPU or MODEM MODE and data is being transferred in

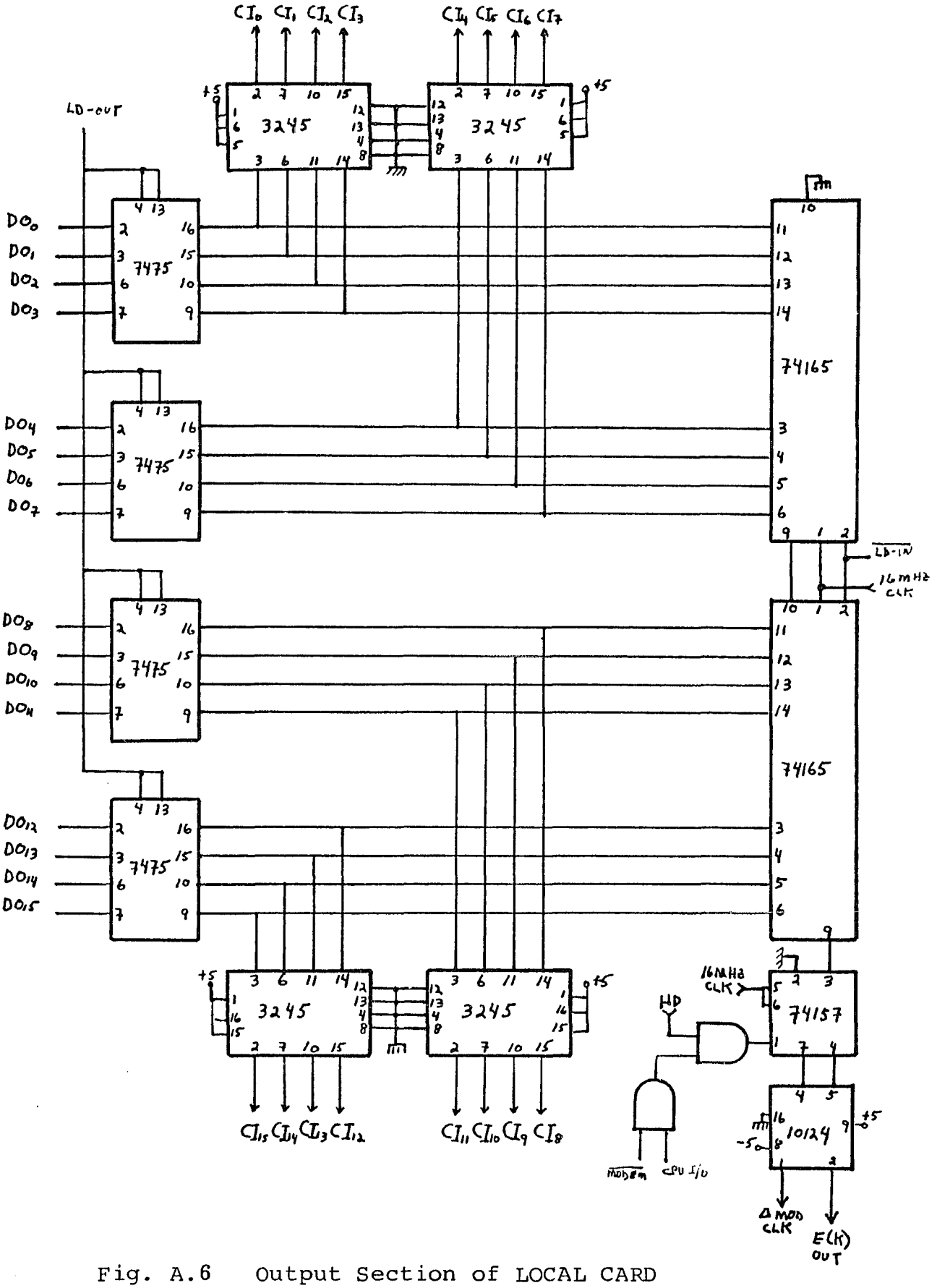


Fig. A.6 Output Section of LOCAL CARD

or out of the FF memory the ADM decoder will receive only a logic "0" signal so that the monitor's screen is blank. If the data is transmitted to the ADM decoder during FF memory read operations in the CPU or modem mode the television monitor screen will be flickering.

The data latched form the FF memory can also be transmitted to the CPU or to the telephne modem. Transmission to the CPU is through a 40-pin ribbion cable. The 3245, IC's D5-D8, are line drivers necessary to drive the long cables to the computer.

A.2.3 MODEM CARD

The function of the modem card is to : 1) transmit the "frozen" video image in the FF memory over unconditoned telephone line via a synchronous modem 2) store in the FF memory a video image being received over the telephone line via a synchronous modem. The modem card is partitioned into three sections; modem address, transmitter, and receiver.

In order to set the FF into the modem mode the rotor switch has to be set to position R-3. If a video image is to be transmitted, it is assume that the image was "frozen" when the FF was in either the manuel mode or the CPU mode and then the rotor switch was placed into the modem mode.

Since transmission over telephone lines is through a synchronous modem the following signal are provided from the synchronous modem and are RS232 voltage level.

TRANSMIT CLOCK: This modem clock output is used to synchronize the FF unit to the transmitter portion of the synchronous modem when transmitting a frozen image from the FF to the telephone line.

TRANSMIT DATA: This serial modem input is used to transmit the E(k) data from the FF to the telephone line at the TRANSMIT CLOCK rate.

RECEIVE CLOCK: This modem output provides the clock signal used to synchronize the FF unit whenever it is receiving a video image from the telephone line.

RECEIVE DATA: This modem output represents the serial E(k) data that is being transmitted over the teilephone line at the RECEIVE CLOCK rate.

The different sections of the modem card are described below:

1) MODEM ADDRESS: The modem address and modem transmit section of the modem card are shown in fig. A.7. The modem address is generated the same way that is is generated in

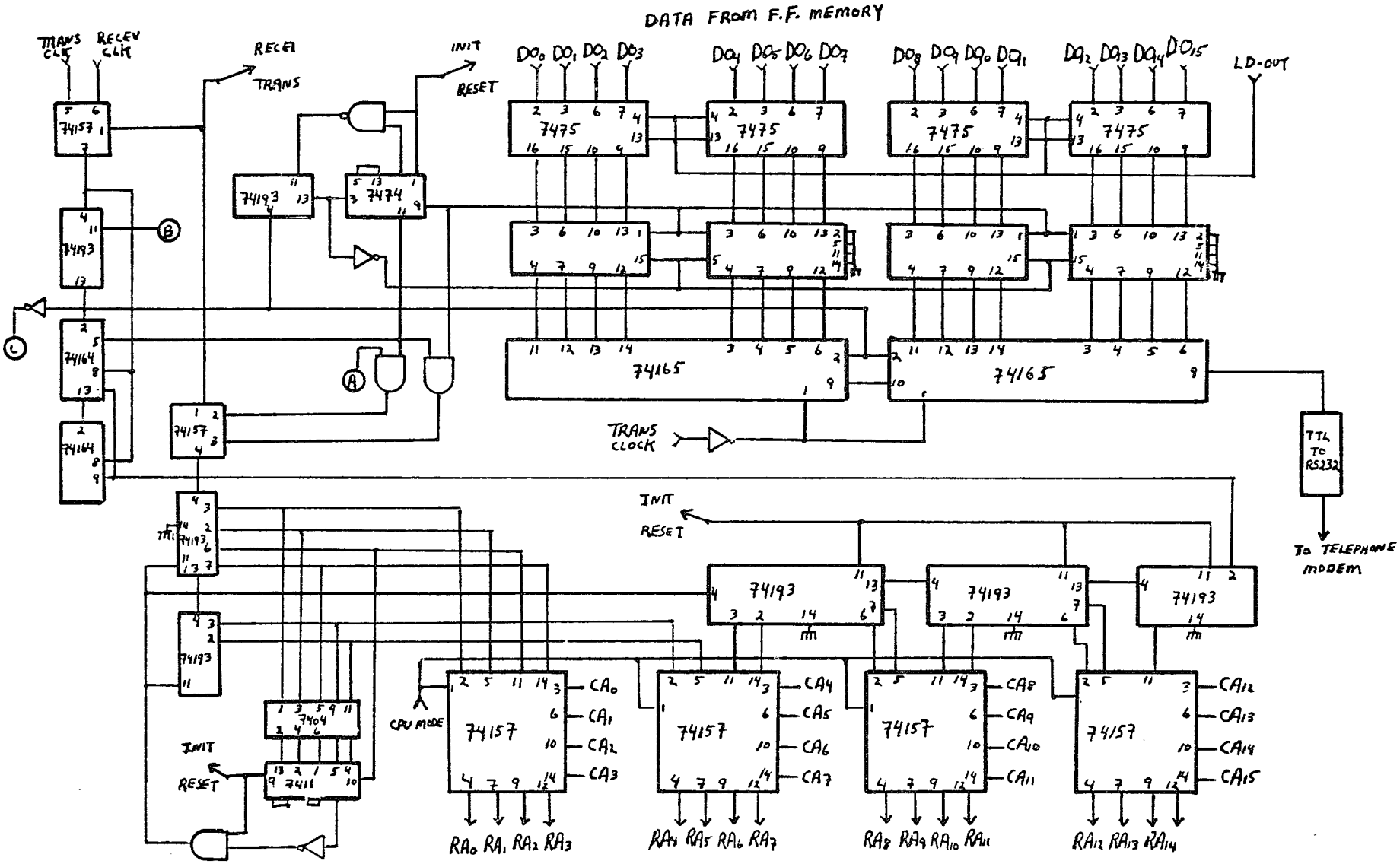


Fig. A. 7 Address and Transmit Section of MODEM CARD

the LOCAL CARD except that a few control signal are added. In fig. A.7 the modem transmit section is blocked off from the addressing section.

Since the FF has to be synchronized to one of two clocks, TRANSMIT CLOCK or RECEIVE CLOCK, one of the two clocks is multiplexed to drive the system by IC E1. Switch S-3 which indicates whether the modem is receiving or transmitting a video image is used to select the clock signal used. IC E2 is a down counter that is used to divide the clock by sixteen since it takes sixteen clock pulses to shift in or out the serial data of the synchonous modem. Shift registers E3 and E4 are used to shift the "walking zero" that is used to generate the various control signals. One of the control signals required is to decrement the modem address by one.

The modem address is generated by the down counters, E6-E10. The horizontal address is contain in E6 and E7 while the line address is contain in E8-E10. When operating at 16 MHz only 58 16-bit words are stored per video line. Thus, when accessing the FF memory from an external source care must be taken to insure that not more than 58 words per line are accessed. In the modem mode chips E15 and E16 are used to reset the horizontal address after every 58 words and also to up-date the line address.

The outputs of the multiplexer, IC's E11-E14, are

use to provide an address to the FF memory via the LOCAL CARD. Notice that the address generated by the modem card or the address provided by the CPU, address CA_0-CA_{15}, provided by OUTPUT-1 of the DR11-K parallel interface, can be multiplexed to RA_0-RA_{14}, the ribbon address to the local card. CA_{15} is not used for addressing since only 15 bits are required to address 32-K of memory. CA_{15}, however, is used to indicate when the data on OUTPUT-2 of the DR11-K is a CPU status word.

2) MODEM TRANSMITTER: In fig. A.7 the transmitter portion of the modem card is blocked off. When transmitting a video image over the telephone line the two modem at each end must somehow be synchronized so that the recevier will know when it is receiving a video image. To achieve data synchronization the transmitter will transmit 16 words of 170360_8, or 32 8-bit synch patterns, followed by an all zero word which is then immediately followed by th video data.

Shift registers E26 and E27 contains the data that is transmitted to the synchronous modem. Since the modem operates at RS232 voltage level IC E28 is used to make the translation from TTL. The shift registers are loaded every 16^{th} clock pulse and is control by the borrow pin of chip

E2. The multiplexers, IC's E22-E25, selects whether to transmit the video data in latches E29-E32 or to transmit the synch pattern that is hard wired to the multiplexer's second inputs.

When transmission is initiated IC's E22-E25 will multiplex the synch pattern to the output shift registers. The number of synch words transmitted is determined by the down counter, IC E17. When the borrow signal of E17 goes low, indicating that the prescribed number of synch words have been transmitted, pin 15 on the multiplexers will go high for half a clock period. When pin 15 is high the multiplexer output become zero and the output shift registers will latch the all zero word. The borrow signal from E17 also sets a flip-flop, IC E18, which allows the video data latched in E29-E32, to be multiplexed to the output shift registers. At the same time, the flip-flop enables the address counters.

3 MODEM RECEIVER: The modem receiver section of the modem card is shown in fig. A8. The received data is converted from RS232 to TTL voltage level and serially shifted into IC's F2 and F3 at the RECEIVER CLOCK rate. Chips F6 and F7 are used to detect the synch patterns. Every time a synch pattern is detected the down counter F9 is decremented. The down counter is used to keep track of the number of

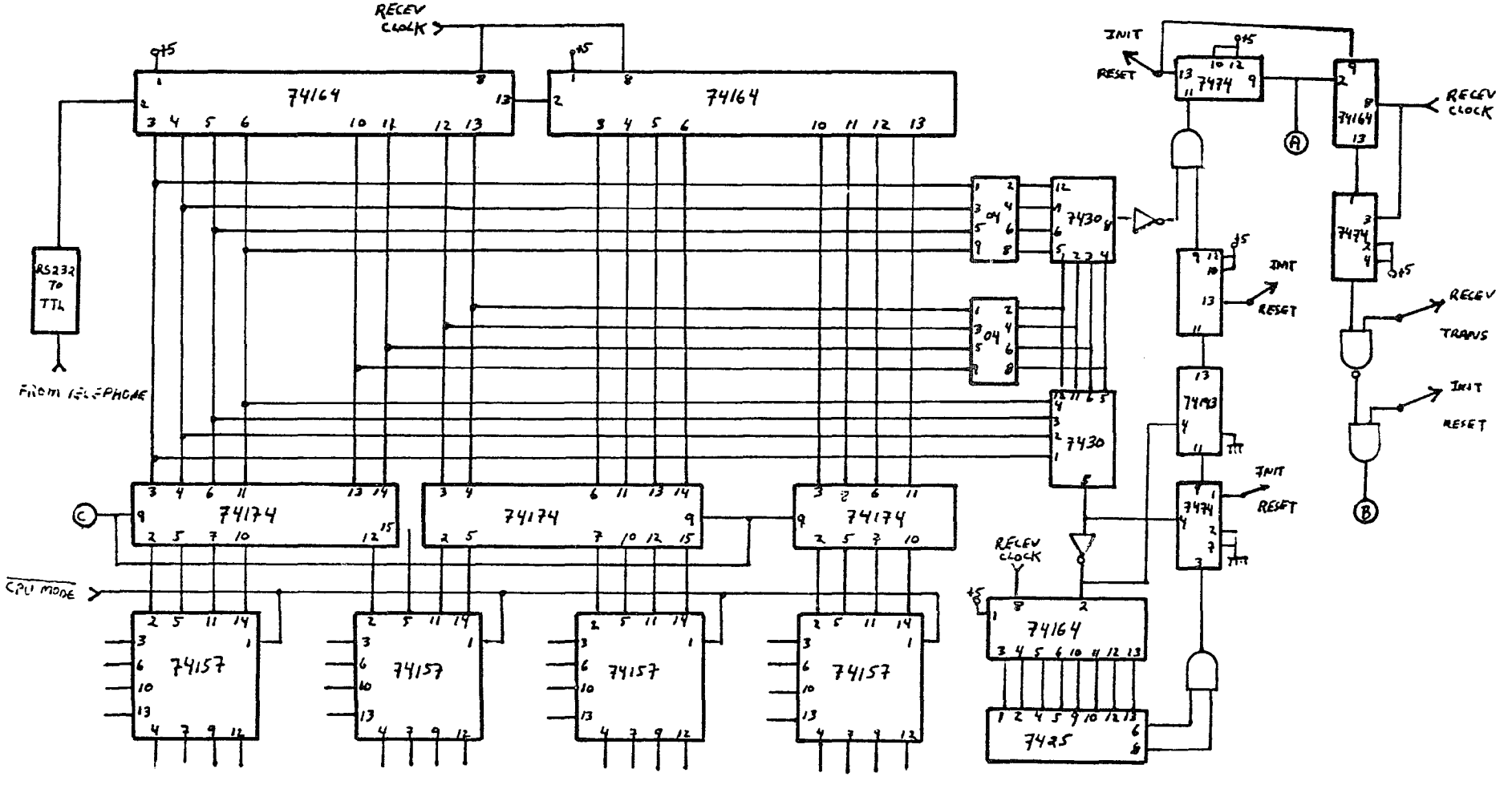


Fig. A.8 Receiver Section of MODEM CARD

consecutive synch pattern needed to indicate that a video image is in fact being received. Chips F11 and F12 are use to insure that the synch patterns are in consecutive order. If the sequence of synch patterns is broken before 16 consecutive patterns are detected the down counter, F9, is reset.

When the receiver has detected 16 consecutive synch patterns, flip-flop F8 is set, the receiver will then wait for the all zero word. When the first all zero pattern is detected the video information will begin eight bits later, since two all zero synch patterns are transmitted. When the all zero pattern is detected, provided F8 is set, flip-flop F13 is set. This enables the address counters and eight clock pulses later flip-flop F15 is set, which enables the down counter E2 which provides the clock pulses to the address counters.

The received data is latched into chips F16-F18 every sixteen clock pulse and is presented to the multiplexers, IC's F19-F22. The multiplexers are used to multiplex either the received data from the modem or the data from the CPU via a 16 bit ribbon cable to the multiplexers, IC's C8-C11, in the LOCAL CARD for storage into the memory.

A.3.0 INDICATORS

The following indicator are on the frame freeze unit.

1 - Power ON/OFF [LED] - Indicates power is being supplied to the system.

2 - Manual/CPU [LED] - Indicates the mode of operation of the FF.

3 - Modem transmit/receive [LED] - Indicates whether data is being transmitted or received from telephone modem.

4 - Mode data transfer over [LED] - Indicates when data transfer over telephone modem is complete.

5 - CPU Read/Write [LED] - Indicates whether the computer is transmitting or receiving data from the FF.

REFERENCES

1) R. Lippmann, "Influence of Channel Errors on DPCM Picture Coding," ACTA Electronica, 19, 1976, pp 289-294

2) W. Zschunke, "DPCM Picture Coding With Adaptive Prediction," IEEE Trans. on Comm., Vol. COM-25, No. 11, Nov. 1977, pp 1295-1301

3) D. A. Huffman, "A Method For The Construction of Minimum Redundancy Codes," Proc. IRE, Vol. 40, Sept. 1952, pp 1098-1101

4) J. B. O'Neal, Jr., "Predictive Quantizing Systems [Differential Pulse Code Modulation] For The Transmission of Television Signals," Bell Syst. Tech. J., May-June 1966, pp 689-721

5) Tsu-Luen R. Lei, N. Scheinberg, D. L. Schilling, "Adaptive Delta Modulation System For Video Encoding," IEEE Trans. on Comm., Vol. COM-25, No. 11, Nov. 1977

6) H. Jeong, C. Kwan Un, "A PCM/AND and ADM/PCM Code Converter," IEEE Trans. on Acoustics, Speech, and Signal Processing, Vol. ASSP-27, No. 6, Dec. 1979, pp 762-768

7) J. L. LoCicero, D. L. Schilling, "An All Digital Technique For ADM to PCM Conversion," Conf. Rec., 1976 National Telecommunication Conf., pp 29:3-1-29:3-6

8) C. W. Harrison, "Experiments With Linear Prediction In Television," Bell Syst. Tech. J., July 1952, pp 764-784

9) D. L. Schilling, N. Scheinberg, J. Garodnick, "Video Encoding Using Adaptive Delta Modulation," IEEE Trans. on Comm., Vol. COM-26, No. 11, Nov. 1978, pp 1682-1689

10) E. R. Kretzmer, "Statistics of Television Signals," Bell Syst. Tech. J., July 1952, pp 751-763

11) A. Habibi, G. S. Robinson, "A Survey of Digital Picture Coding," Computer, Vol. 7, Num. 5, May 1974, pp 22-34

12) P. J. Ready, D. J. Spenscer, "Block Adaptive DPCM Transmission of Images," Conf. Rec., 1975 Nat. Telecomm. Conf., pp 22:10-22:17

13) D. R. Weber, "An Adaptive Run Length Encoding Algorithm," Conf. Rec., 1975 Internat. Conf. Comm., pp 7:4-7:7

14) T. S. Huang, "Easily Implementable Suboptimum Runlength Codes," Conf. Rec., 1975 Internat. Conf. Comm., pp 7:8-7:11

15) D. Preub, "Comparison of Two Dimensional Facsimile Coding Schemes," Conf. Rec., 1975 Internat. Conf. Comm., pp 7:12-7:16

16) N. R. Scheinberg, D. L. Schilling, M. Z. Ali, I. Paz, "A Technique For Correcting Errors in Video Delta Modulation Channels," Conf Rec., 1975 Internat. Conf. on Comm., pp 27:21-27:25

17) J. W. Mark, "A Dithered Adaptive Predictive Coding Scheme For Compression of Analog Sources," Conf. Rec., 1975 Internat. Conf. on Comm., pp 27:26-27:30

18) N. S. Jayant, "Adaptive Delta Modulation With a One Bit Memory," Bell Syst. Tech. J., Vol. 49, Mar. 1970, pp 321-342

19) H. Taub, D. L. Schilling, "Principles of Communication System," McGraw-Hill, 1971, Chap. 6

20) R. Ash, "Information Theory," Interscience Division of John Wiley Sons, 1965, Chap. 1-3

21) Tsu-Leun R. Lei, "Delta Modulation Encoding Of Video Signals," Ph.D. Dissertation, The City University of New York, New York, 1978

22) L. S. Schwartz, "Principles of Coding, Filtering and Information Theory," Spartan Books Inc., 1963

23) L. Brillouin, "Science and Information Theory," Academic Press Inc., 1957

24) J. L. LoCicero, "Arithmetic Processing And Digital Conversion of Adaptive Delta Modulation Encoded Signals," Ph.D. Dissertation, The City University of New York, New York, 1976

25) C. L. Song, J. Garodnick, D. L. Schilling, "A Variable Step Size Roburst Delta Modulator," IEEE Trans. on Comm. Tech., Vol. COM-19, Dec. 1971, pp 1033-1099

26) "Special Issue On Redundancy Reduction," Proceedings of the IEEE, Vol. 55, Num. 3, March 1967

27) "Special Issue On Image Bandwidth Compression," IEEE Trans. on Comm., Vol. COM-25, Num. 11, Nov. 1977

28) "Special Issue On Digital Picture Processing," Proceedings of the IEEE, Vol. 60, Num. 7, July 1972

29) Schweiber, "Picture Coding," Proceedings of the IEEE, Vol. 60, No. 7, July 1972

30) N. R. Scheinberg, "The Delta Modulation Of Video Signals," Ph.D. Dissertation, The City University of New York, New York, 1979

31) C. M. Kortman, "Redundancy Reduction - A Practial Method of Data Compression," Proceedings of the IEEE, Vol. 55, No. 3, March 1967, pp 253-263

32) C. A. Andrews, J. M. Davies, G. R. Schwarz, "Adaptive Data Compression," Proceedings of the IEEE, Vol. 55, No. 3, March 1967, pp 267-277

33) D. Hochman, H. Katzman, D. R. Weber, "Application of Redundancy Reduction to Television Bandwidth Compression," Proceedings of the IEEE, Vol. 55, No. 3, March 1967, pp 263-266

34) L. Ehrman, "Analysis of Some Redundancy Removal Bandwidth Compression Techniques," Proeedings of the IEEE, Vol. 55, No. 3, March 1967, pp 278-287