# DESIGN AND CONSTRUCTION OF A (75,50) FORWARD ERROR CORRECTING CODEC FOR HIGH SPEED DIGITAL COMMUNICATIONS

© Jeremy Clark

A Major Technical Report

in

The Faculty

of

Engineering

December 1981

Concordia University

Presented in Partial Fulfillment of the Requirements
for the degree of Master of Engineering at
Concordia University
Montreal, Quebec, Canada

ABSTRACT

Design and Construction of a (75,50) Forward Error Correcting Codec for High Speed Digital Communications

Jeremy Clark

Code properties necessary for forward error correction in high speed digital communication systems are reviewed. A (75,50) double error correcting quasi-cyclic code is presented as a reasonable embodiment of these properties. A hypothetical codec (encoder-decoder) assembly incorporating this (75,50) code is described. Practical considerations for the implementation of the codec are discussed. An ECL-based encoder prototype, a realization of the hypothetical model, is presented. The encoder circuitry and performance at a data rate of 15MHZ are discussed in detail. Design considerations for a matching decoder are presented. Cost figures are developed for a complete codec package, together with possible future applications.

## ACKNOWLEDGEMENTS

The author wishes to thank Dr. V.K. Bhargava for his support and enthusiasm during the course of this project.

Appreciation is also extended to all those in the Graduate Communications Laboratory H847/849 for their welcome comments and suggestions.

Thanks also to Miss Jennifer Clark for the typing and report layout.

# TABLE OF CONTENTS

TABLE OF CONTENTS

# LIST OF SYMBOLS AND ABBREVIATIONS

| | |
|---|---|
| Codec | Encoder Decoder |
| ECL | Emitter Coupled Logic |
| FEC | Forward Error Correction |
| BCH | Bose-Chaudhuri-Hocquenghem |
| erfc | Error Function Complement |
| $E_b$ | Energy per bit |
| $N_o$ | Energy Spectral Density of Additive White Gaussian Noise |
| $d_{min}$ | Code Mihimum Distance |
| t | Error Correcting Capability of Code |
| p | Binary Symmetric Channel Error Probability |
| $P_B$ | Probability of Post Decoding Block Error |
| Pb | Probability of Post Decoding Bit Error |
| DISR | Data Input Shift Register |
| PBG | Parity Bit Generator |
| CWSR | Code Word Shift Register |
| DOSR | Data Output Shift Register |
| MECL | Motorola Emitter Coupled Logic |
| GaAs | Gallium Arsenide |
| VLSI | Very Large Scale Integration |
| IC | Integrated Circuit |
| SIP | Single in Line Package |
| VCO | Voltage Controlled Oscillator |

# LIST OF FIGURES

# LIST OF FIGURES

# LIST OF TABLES

# CHAPTER 1

# INTRODUCTION

## 1.1 Historical Perspective

Data communications has undergone an explosive growth since World War II, especially in the last two decades. From essentially simple machine-to-machine communication over a single channel such as teletype on an open wire line, vast networks of integrated switched digital networks have developed. International standards [E.8] have evolved allowing users to join a network with the appropriate data structure and rate. Simple structures at low rates permit communications such as telex, digitized voice etc. These may in turn be multiplexed together to higher rates. More involved structures at these higher rates allow transmission of complex signals such as digitized voice and radar, or the multiplexed combination of thousands of digitized voice and data channels. For economic reasons, the trend in recent years has been to design digital systems with the highest capacity (rate) possible.

Parallel with the development of digital networks, has been the study of securing data signals from transmission errors due to the imperfect nature of channels within the network. Several methods have evolved. One such method is

to repeat the digital message several times (Automatic Request Scheme) to insure integrity. This proves very effective and economical at low data rates but becomes less desirable at higher rates. High data rates require the uninterrupted and smooth flow of data.

This requirement has prompted the use of Forward Error Correction (FEC) schemes. In these methods, additional information or parity bits are added to the message at the send end, and decoded with the message at the receive end, allowing error detection and correction. The process occurs synchronously with the message, without interruption.

Table 1.1 lists common data formats and rates in use today. Also given are the rates that would be required if these formats were rate 2/3 encoded. i.e. For every 2 bits of actual data, 1 bit of redundancy or parity is added.

TABLE 1.1

Common Data Formats

| | | | ← Rate 2/3 Encoded → | |
|---|---|---|---|---|
| System | Rate MHZ | Period nsec | Code Rate MHZ | Period nsec |
| Bell T1 | 1.544 | 648 | 2.316 | 432 |
| CCIR | 1.544 | 648 | 2.316 | 432 |
| CCIR | 2.048 | 488 | 3.072 | 326 |
| Bell T1C | 3.152 | 317 | 4.728 | 212 |
| Bell T2 | 6.312 | 158 | 9.468 | 106 |
| CCIR | 6.312 | 158 | 9.468 | 106 |
| CCIR | 8.448 | 118 | 12.672 | 79 |
| CCIR | 32.064 | 31 | 48.096 | 21 |
| CCIR | 34.368 | 29 | 51.552 | 19 |
| CCIR | 44.736 | 22 | 67.104 | 15 |
| Bell T3 | 46.304 | 22 | 69.456 | 14 |
| CCIR | 139.264 | 7 | 208.896 | 5 |
| Bell T4 | 274 | 3.7 | 411 | 2.4 |

## 1.2 Code Properties Necessary for High Speed Forward Error Correction

Nowadays one would define a high speed digital system as one that operates over 40 Mbit/sec. Examples might be the 50 Mbit/sec coding system used in the American Space Shuttle [C.12], the proposed 120 Mbit/sec Intelsat V TDMA system [C.5], or the Norwegian Telecommunication Authority 140 Mbit/sec coaxial line system [C.11].

Digital system designers seek to minimize the overall system probability of error in keeping with budgeted performance objectives and economic constraints. System structures are chosen on the basis of maximum 'system gain'. For example in a radio or satellite system, this would mean minimum antenna size and required transmitter power. Forward error correction, or simply coding, may be used to increase this system gain factor.

In (n,k) linear block coding, incoming data is segmented into blocks of k bits, whereupon n-k redundant or parity bits are calculated and added to the block. A continuous stream of data arrives at the encoder and thus must also leave the decoder in a like fashion. This means that the coded data must be transmitted at a faster rate than the incoming data due to the overhead parity bits. Unfortunately this increase in coded data speed means more channel band-

width is required. Hence, depending on the modem format used (QPSK, 8ϕPSK, 16QAM etc.), and the channel characteristics (additive white Gaussian noise, fading etc.) more errors will be introduced. However, the code has error correcting properties, and if more errors can be corrected than those introduced due to the increase in speed, then a net coding gain is achieved. This net coding gain concept is discussed by Avni and Bhargava in [C.4] and Avni in [C.19].

Several characteristics of a coding scheme are required if it is to fit in a high speed system. First and foremost, it must provide the required net coding gain in a cost effective way, i.e. making it a more desirable alternative than another scheme (e.g. increase transmitting power etc.). Existing literature points to the decoder as the major cost element in a coding scheme. Hsiao [C.2] cites the requirements of parallel processing and uniform structure for ultra high speed decoding. As he points out, the aspects of low redundancy and high speed operation are conflicting requirements. Low redundancy usually implies a decoder requiring a long execution time. An example might be a typical BCH type decoder utilizing the iterative algorithm of Berlekamp. High speed operation requires increased redundancy, i.e. longer code lengths, with increased hardware requirements. However, using one step majority logic codes, the decoder circuitry can be simplified to a very

uniform structure, with the independent and parallel processing of all decoded bits.

Although there may be many circuits in a one step majority logic decoder, as this report will show, they are uniform and of simple connection. Recent large scale integration efforts with high speed circuits will make it feasible to implement long shift register combinations etc. on one LSI chip. This will make the hardware requirement of such a majority logic decoder trivial.

## 1.3 (75,50) Double Error Correcting Quasi-Cyclic Code

The encoder described in this report, and the matching decoder utilize a (75,50) double error correcting code of the Rahman and Blake type, described by Bhargava in [C.1]. The code is systematic quasi-cyclic and the explicit form for the parity check matrix is as follows:

$$H = \begin{bmatrix} a & b & c & d & e & v & w & x & y & z \\ e & a & b & c & d & z & v & w & x & y \\ d & e & a & b & c & y & z & v & w & x \\ c & d & e & a & b & x & y & z & v & w \\ b & c & d & e & a & w & x & y & z & v \end{bmatrix} \quad I_{25}$$

where

$$a = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix} \qquad b = e = x = y = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$c = z = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \qquad d = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$v = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \end{bmatrix} \qquad w = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

In general, for a (75,50) linear block code, the generator matrix G is given by:

$$G = [I_{50}, P] \qquad [C.13]$$

and the parity check matrix H by:

$$H = [P^T, I_{25}]$$

The first 50 (0,49) bits of the transmit code vector $\bar{u}$ is the message vector $\bar{m}$ itself.

$$\bar{u} = \bar{m}\, G$$

The last 25 (50,74) bits are the parity bits. The calculation of a parity bit may be done by inspection of H. ($P^T$ in particular).

EX:

$u_i = m_i, \quad i = 0,49$

$u_{49} = m_{49}$, last message bit

$u_{49+j} = p_{1j}m_i + p_{2j}m_2 + \ldots p_{49j}m_{49} \qquad j = 1,25$

$u_{50} =$ 1st parity bit $= m_0 + m_1 + m_{13} + m_{16} + m_{25} + m_{28} + m_{34} + m_{48}$

The previous example may be easily shown with reference to Appendix I, which gives the full matrix $P^T$, and the complete set of encoder equations.

Since each row of $P^T$ (column of P) contains 8 1's, it follows that each parity bit is simply calculated by taking the mod 2 sum of 8 bits. Hence encoding is uniform for each parity bit, and fast. Also, the encoding of any parity bit is independent of the encoding of any other parity bit.

To decode the 50 information bits of the 75 bit received code vector $\bar{u}$, one step majority logic decoding with parallel processing is used. This avoids the use of a linear feedback shift register. The majority logic equations used to estimate each bit can be obtained by inspection of H, and are included in detail in Appendix II.

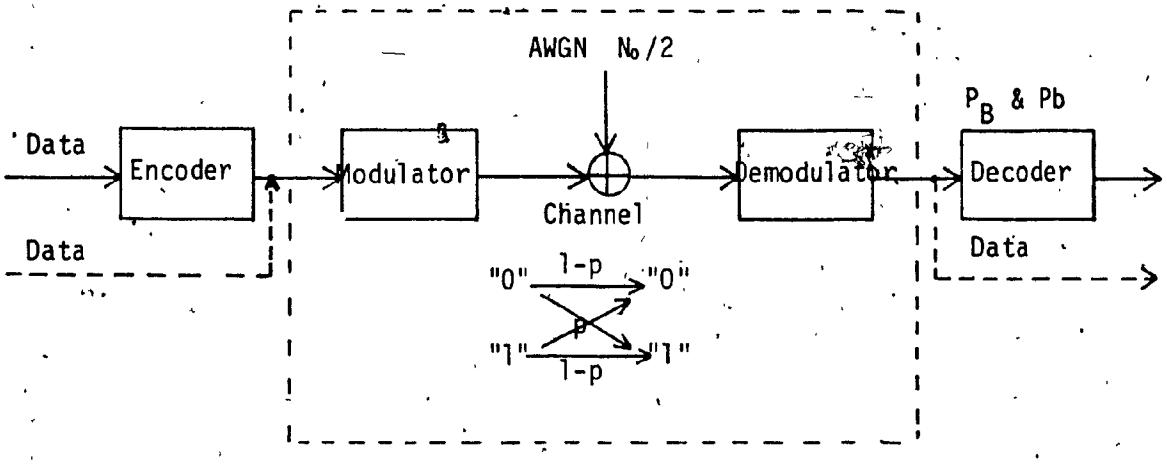
EX: $\hat{m}_0 \triangleq$ estimate of 1st information bit (decoded value)

$$\hat{m}_0 = \text{maj}\begin{bmatrix} u_0 \text{ (itself)} \\ u_1+u_{13}+u_{16}+u_{25}+u_{28}+u_{34}+u_{48}+c_0 \\ u_4+u_{12}+u_{15}+u_{27}+u_{29}+u_{33}+u_{47}+c_4 \\ u_{10}+u_{14}+u_{22}+u_{32}+u_{37}+u_{39}+u_{43}+c_{14} \\ u_8+u_{17}+u_{18}+u_{35}+u_{40}+u_{42}+u_{46}+c_{17} \end{bmatrix}$$

$c_j \triangleq u_{50+j} = j^{th}$ received parity bit, $j = 0,24$

Again, due to the uniformity of 1's in each row of H, the decoding of each bit is done uniformly and independently with a 5 input majority logic gate and four 8 input mod 2 adders per bit.

## 1.4 Theoretical Performance of the (75,50) Code




Coded vs Uncoded Communication Channel

Fig.1.4.1

To evaluate the (75,50) quasi-cyclic code, the communications model as shown in Fig.1.4.1 is postulated. Data arrives at the encoder, parity bits are calculated and added to the data word, and the composite code word is fed to the modulator. The modulator translates the coded bits into various waveforms, QPSK, 8ϕPSK, 16QAM etc. as dictated by the channel requirements. The signal is now fed to the communications channel, where additive white Gaussian noise of spectral density $N_o/2$ is added. The demodulator performs the inverse process of the modulator and produces the received code word. The received code word is then passed to

the decoder where it is decoded to produce the received data word. The channel is assumed to be memoryless and binary symmetric. That is, errors occur randomly with probability p. If coding is not used, then data simply goes directly to the modulator and emerges from the demodulator with the same error probability p.

At the input to the decoder, each code word or block will contain errors, which will be assumed to be binomially distributed. [C.19]. Due to the error correcting capability of the code, the decoder will correct any pattern of t or fewer errors where

$$t = \left[\frac{d_{min} - 1}{2}\right]_{GI}$$

GI = Greatest integer part of
$d_{min}$ = Code minimum distance

The probability of the block containing j errors will be equal to

$$\text{Probability (j errors)} = p^j (1-p)^{n-j}$$

Hence the probability of a decoding block error will equal the sum of probabilities of j error patterns for t(j=

$$\text{Probability (block error)} = \sum_{j=t+1}^{n} \binom{n}{j} p^j (1-p)^{n-j}$$

To obtain the post decoding probability of bit error, the assumption is made that on the average, every n bit code word contains $d_{min}$ errors [C.19]. Thus,

$$d_{min} \sum_{j=t+1}^{n} \binom{n}{j} p^j (1-p)^{n-j}$$

errors are produced per block. Since there are n bits per block, the probability of bit error Pb reduces to:

$$Pb = \frac{d_{min}}{n} \sum_{j=t+1}^{n} \binom{n}{j} p^j (1-p)^{n-j}$$

For the (75,50) double error correcting quasi-cyclic code,

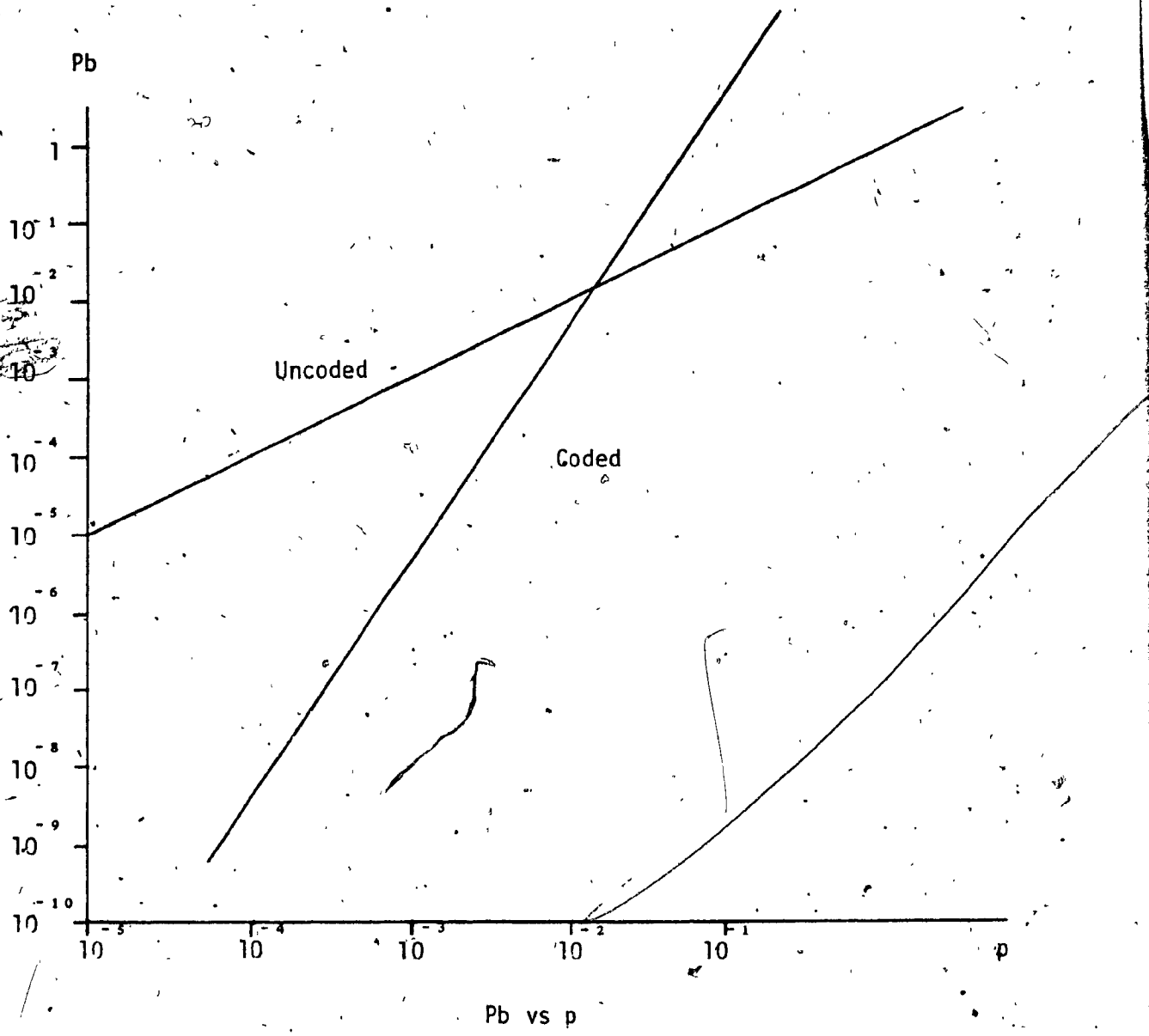
$$d_{min} = 5$$

$$Pb = \frac{5}{75} \sum_{j=3}^{75} \binom{75}{j} p^j (1-p)^{75-j}$$

$$\simeq \frac{5}{75} \times \frac{75 \times 74 \times 73}{3 \times 2 \times 1} p^3 \qquad \text{for } p \ll 1$$

$$\simeq 4.5 \times 10^3 \, p^3$$

The relationship between the post decoding bit error rate Pb and the channel error rate probability p is shown in Fig.1.4.2. Note when $p < 10^{-2}$, Pb is much smaller than p for the coded case.




Pb vs p

Fig.1.4.2

To establish the net coding gain, a particular modem characteristic must be considered (in keeping with the communication model as postulated in Fig.1.4.1). Fig.1.4.3 is a plot of the probability of bit error p vs $\frac{E_b}{N_o}$ for a simple QPSK system with coherent detection, without coding.[C.20] To plot the curve with coding, the following must be considered [C.19]. At any particular uncoded working point $\left[\frac{E_b}{N_o}\right]_u$ vs $p_u$, the corresponding coded bit energy will be $10\log (n/k)\text{db} = 10\log 75/50 \simeq 1.8\text{db}$ lower, due to the extra parity bits required. This lower value of $\left[\frac{E_b}{N_o}\right]_i$ will have a corresponding probability of bit error $p_i$. However, due to the error correcting capability of the code, this value of $p_i$ will yield a lower probability of error $p_o$, as given in Fig. 1.4.2.

EX: For QPSK,

$$p = \tfrac{1}{2}\operatorname{erfc}\sqrt{\frac{E_b}{N_o}} \qquad [C.20]$$

$$\operatorname{erfc}(x) = \frac{2}{\sqrt{\pi}}\int_x^\infty e^{-y^2}\,dy \simeq \frac{1}{x\sqrt{\pi}}\exp(-x^2) \quad \text{for } x \text{ large}$$

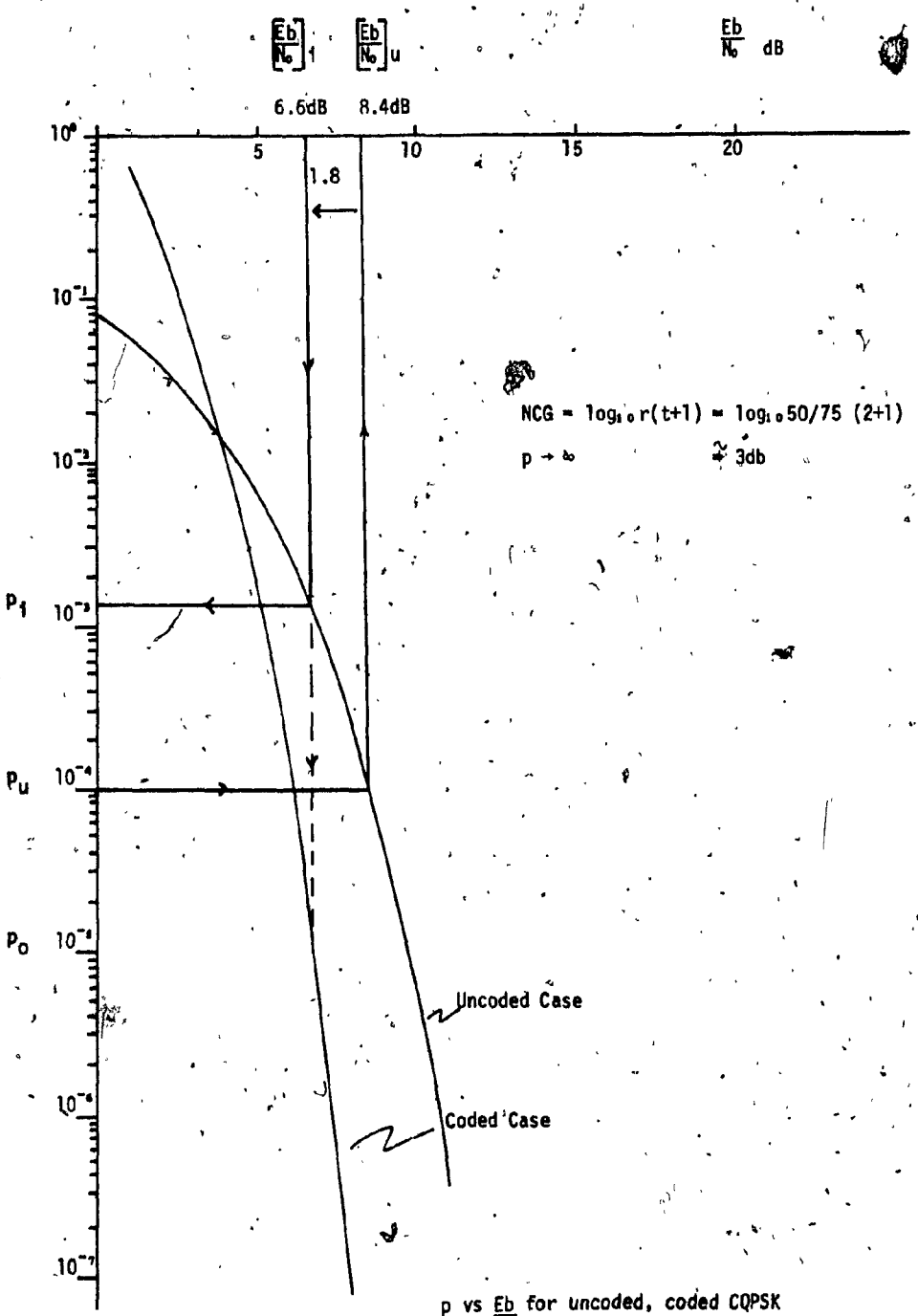
$$p_u = 10^{-4}, \left[\frac{E_b}{N_o}\right]_u \simeq 8.4\text{db} \quad \text{by inspection from graph}$$

$$\left[\frac{E_b}{N_o}\right]_i = 8.4 - 1.8 = 6.6\text{db}$$

$$p_i \simeq 1.3\times10^{-3} \quad \text{by inspection from graph}$$

$$p_o \simeq 4.5\times10^{3} \times (1.3\times10^{-3})^3 = 9.9\times10^{-6}$$

Hence the point (6.6, $9.9 \times 10^{-6}$) can be plotted. Other points can be plotted in a similar fashion, and give rise to the coded curve. The difference in db between the uncoded and coded case, represents the net coding gain.




p vs $\frac{Eb}{No}$ for uncoded, coded CQPSK

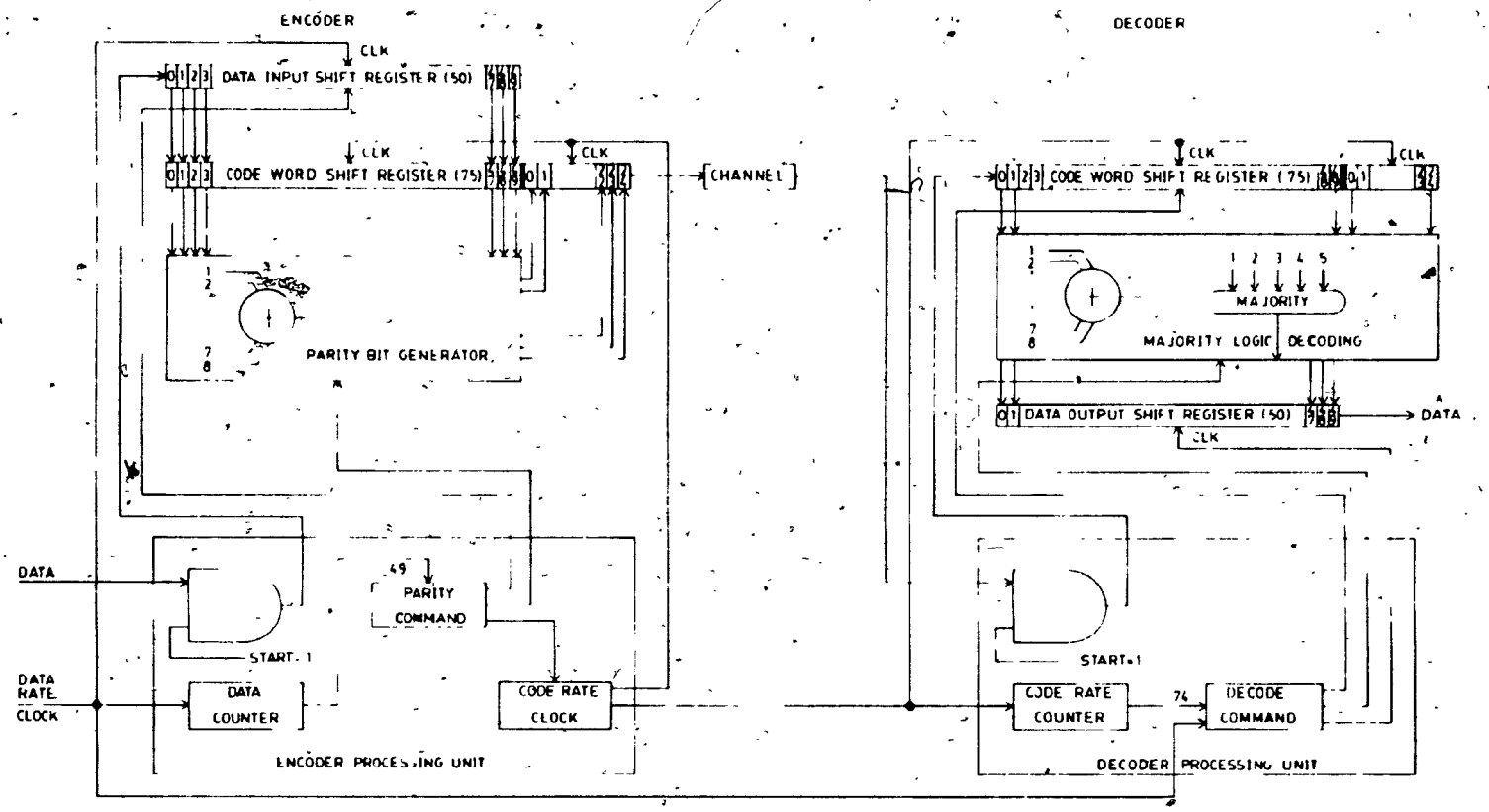
Fig.1.4.3

## 1.5 Hypothetical Encoder Decoder Assembly

The (75,50) double error correcting quasi-cyclic code, may be implemented using the structure as illustrated in Fig. 1.5. For purposes of the codec evaluation only, it is assumed that the various VCO, data and clock rates are available to the decoder. This removes the added complexity of having to build timing extraction circuits, but still permits evaluation of performance in various channel configurations. (Additive white Gaussian noise, fading sinusoidal interference etc).

Data enters the encoder processing unit where it is fed in to the data input shift register (DISR) upon a 'start' command. After the 50th bit has entered ($m_{49}$), the data counter senses the occurrence and signals the parity command unit. This unit causes the data input shift register to immediately dump its full 50 bit word in parallel to the code word shift register (CWSR). This dumping occurs within 1 bit interval so that data flows again sequentially into the data input shift register. Meanwhile the parity bits are calculated in accordance with sec.1.3 and loaded in the last 25 positions of the code word shift register. The contents are then shifted out at the code rate (1.5 x data rate).

The transmitted code word is received by the decoder in the code word shift register. After an appropriate 'start' command, the code rate counter signals the decode command module when 75 bits have been received ($..u_{73}$, $u_{74}$). The full 75 bit content of the code word shift register is then immediately dumped in the majority logic decoding circuitry as discussed in sec.1.3. The 50 estimated message bits are then loaded in the data output shift register. These are then shifted out in synchronism with the data entering the encoder, but delayed by several bit intervals.




CODEC BLOCK DIAGRAM

fig 1.5

# CHAPTER 2

# IMPLEMENTATION CONSIDERATIONS

## 2.1 High Speed Logic Families

When it comes to building a high speed logic circuit, there are very few comercially available logic families that can be used. Motorola MECL III or MECL 10K are two of these families. The older series MECL I and MECL II are now out of production. Nippon Electric use in-house a family known as CML, but not much information is available yet. The Motorola 10K family is second sourced by several other suppliers (Fairchild, Texas Instruments), which is very important for any production considerations. Table 2.1 summarizes the various properties of these logic families.

F. Blum of the Rockwell Corporation [E.10] announced some very startling possibilities for high speed logic families using gallum arsenide (GaAs) substrates instead of silicon. High electron mobility and the semi-insulating properties of the substrate promise gate propagation delays of less than 100 picoseconds (.1nsec). Very large scale integration (VLSI) of these gates also appears imminent. However, these devices are not yet available off the shelf.

| Family | MECL I | MECL II | MECL 10K | MECL III | NEC CML | Rockwell GaAs |
|---|---|---|---|---|---|---|
| Status | Obsolete | Obsolete | Available 2nd Source | Available | Available | Announced |
| Gate Prop.Dly. | | | 2 nsec | 1 nsec | .7 nsec | .1 nsec |
| Freq. Toggle | 30 MHZ | 120 MHZ | 160,250 MHZ | 300,500 MHZ | ? | > 1 GHZ |
| Gate Power | | | 25 mW | 60 mW | 10 mW | ~ 1 mW |
| Logic Levels | | | High -.9V Low -1.75V | High -.9V Low -1.75V | High 0V Low -.5V | ? |
| Logic Margin | | | 850 mV | 850 mV | 500 mV | ? |

Fast Logic Families

Table 2.1

## 2.2 Choice of Integrated Circuits

Due to the shortage of high speed logic families to choose from, selection of the integrated circuits for this project was fairly simple. Essentially the choice boiled down to Motorola MECL III (family used by Medlin and Bryg C.6), MECL 10K or NEC CML. At the time the circuits were purchased, it was virtually impossible to obtain NEC CML (Feb.'79). Delivery for MECL III was prohibitive, and the cost of the various circuits was approximately 5 to 10 times more expensive than the MECL 10K. So MECL 10K logic was ordered as a reasonable compromise.

The encoder and decoder circuitry is basically a repetition of various building blocks. The encoder consists of 2 shift registers, one 50 bit DISR and one 75 bit CWSR. It also has a parity bit generator which consists of 25 8 input exclusive OR circuits. The processing circuits consist of a small number of counters, and gates, drivers etc. The decoder consists of 2 shift registers, one 75 bit CWSR and one 50 bit DISR. The majority logic decoding consists of 4 8 input exclusive OR circuits per decoded bit, or a total of 200 8 input exclusive OR's. It also has 50 majority logic gates, which may be constructed in several ways. Various gates, drivers etc. are also required. Thus in summary, the major circuit requirements for the complete

encoder decoder are:

A - 2 x 50 bit shift registers and 2 x 75 bit shift registers

B - 225 8 input exclusive - OR gates

C - 50 majority logic gates

D - Misc. gates, counters, drivers etc.

Table 2.2.1 lists the various possibilities for implementing the required shift register chains. All possible MECL 10K and MECL III storage elements are considered ( D flip flop, SR flip flop etc.). The MC10141 was chosen due to its immediate availability, relative low cost compared to the other choices and flexibility. Each chip contains a universal 4 bit shift register, capable of shifting left, right or parallel input and output. All decoding circuitry is included on each chip.

Table 2.2.2 lists the various possibilities for implementing the 8 input exclusive-OR circuitry. The MC10107/ 10113 are optimum from the low cost point of view, but require more IC's and a lot more wiring. The MC10170 is the next logical choice, as it is the cheapest IC with the minimum #'s required. However, it was unavailable at the time of purchase. The MC10160 was ordered as a sensible compromise.

| IC NO MC | 10130 | 10131 | 10231 | 10133 | 10135 | 10141 | 10153 | 10168 | 10175 | 10176 | 1666 | 1668 | 1670 | 1694 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Type | MECL 10K | 10K | 10K | 10K | 10K | 10K | 10K | 10K | 10K | 10K | MECL III | III | III | III |
| Description | 2xDFF S/R/CLK | 2xMSDFF S/R/CLK | 2xMSDFF S/R/CLK | 4xDFF GTDOPT /CLK | 2xJKMS FF J/K/S/R CLK | 4xUSR → ← 11 | 4 xDFF GTDOPT /CLK | 4xDFF GTDOPT /CLK | 5xDFF R/CLK | 6xMSDFF CLK | 2xRSFF R/S/CLK | 2xDFF R/S/CLK | 1xMSDFF R/S/CLK | 4xSR R/CLK |
| Freq. Max. MHZ | | 160 | 225 | | 140 | 200 | | | | 150 | | | 350 | 325 |
| Data Prop.Dly. Min/Max ns | 1/3.5 | | | 1/5.4 | | 1.8/3.8 | 1/5.4 | 1/5.4 | 1/3.5 | 1.6/4.5 | | | | |
| Clk Prop.Dly Min/Max ns | 1/4 | 1.8/4.5 | 1.5/3.3 | 1/5.4 | 1.8/4.5 | | 1/5.6 | 1/5.6 | 1/4.3 | | 1/2.5 | 1/2.5 | | 1/3 |
| Setup Time Min/Max ns | 2.5/ | 2.5/ | 1/ | 2.5/ | 2.5/ | 2.5Data 5Select | 2.5/ | 2.5/ | 2.5/ | 2.5/ | | | .4/ | |
| Hold Time Min/Max ns | 1.5/ | 1.5/ | .75/ | 1.5/ | 1.5/ | 1.5/ | 1.5/ | 1/ | 1.5/ | 1.5/ | | | .3/ | |
| $CDN IC (1979) | 2.75 | 3.73 | 5.68 | | | 8 | | | | | | 27.27 | 24.79 | 58.75 |
| #IC's for 2x50 bit & 2x75 bit | 126 | 126 | 126 | | | 64 | | | | | | 126 | 250 | 64 |
| $CDN Total (1979) | 347 | 470 | 716 | | | 512 | | | | | | 3,436 | 6,198 | 3,760 |

Shift Register Selection

Table 2.2.1

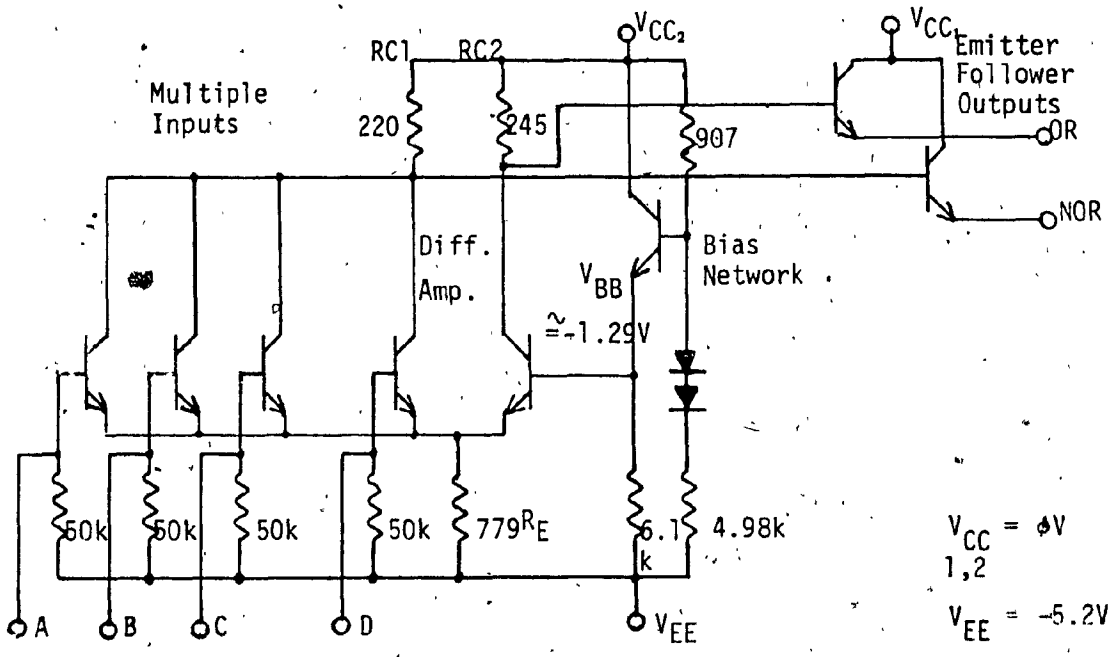
| IC NO. | MC 10107 | MC 10113 | MC 10160 | MC 10193 | MC 10170 | MC 1672 |
|---|---|---|---|---|---|---|
| Type | MECL 10K | 10K | 10K | 10K | 10K | MECL III |
| Descrip. | 3x2 Input EX-OR/NOR | 4x2 Input EX-OR+Enable | 1x12 Input EX-OR | Error-Detect Circuit | 1x9 Input +1x2 Input EX-OR | 3x2 Input EX-OR |
| Prop.Dly. Min/Max ns | 1.1/3.7 | 1.3/4.5 | 2/7.5 | 2/8.5 | 2/6 | 1.3/2.3 |
| $Cdn. IC (1979) | .59 | .59 | 4.00 | 6.24 | 3.88 | $$$ |
| #IC's Reqd. 225 EX-OR | 526 | 394 | 225 | 225 | 225 | 525 |
| $Cdn. Total (1979) | 310 | 232 | 900 | 1,404 | 873 | 5,250 |

8 Input Exclusive - OR Modulo 2 Adder [E.1]

Table 2.2.2

## 2.3 ECL Basics

As mentioned in Sec.2.2, due to economic and delivery considerations, MECL 10K logic was purchased for the codec. The 10K logic series is a toned-down version of the MECL III series. Speed is traded off for much lower power dissipation, and decreased crosstalk. Fig.2.3.1 shows the essential gate structure used in the 10K family. [E.2a]




ECL Gate Structure
Fig.2.3.1

Each gate consists of a differential-amplifier input circuit of high impedance (approx. 50KΩ pulldown resistor, 3 pF cap.), a temperature and voltage compensated bias

network, and low impedance emitter follower output (approx. 7Ω). The high input impedance allows high fan out operation. Up to 4 or 5 gates can be fed from one output, but beyond this, capacitive loading corrupts the rise and fall times of the waveforms.

The input pulldown resistor allows unused inputs in most cases to remain unconnected ("0"). The low output impedance is very useful for high speed operation, and allows easy connection to 50Ω coaxial lines, or wire wrap above a suitable ground plane. Logic levels are as follows:

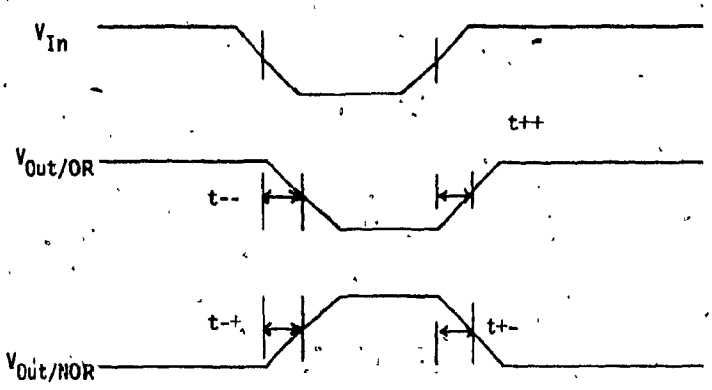
"1" = (-0.960V,-0.810V), -0.9V typical. Noise Margin 125mV

"0" = (-1.850V,-1.650V), -1.7V typical. Noise Margin 155mV

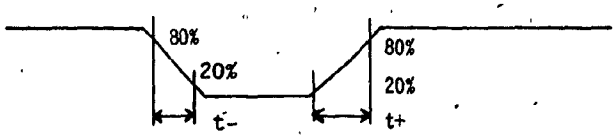
Note that the emitter follower outputs have complementary OR and NOR outputs, so that in most circuit configurations additional inverters are not required.

Several parameters must be carefully considered when using 10K logic, due to the high speed of operation. These are: gate propagation delay, rise and fall time, set-up and hold time. These are illustrated and defined in Fig.2.3.2. The gate propagation delay, is a measure of the time needed for a waveform to propagate through a given logic device. It is measured between the 50% points. The rise and fall times are measured between the 20% and 80% points as opposed to other families where they are measured between the 10% and 90% points. This is due to the built-in

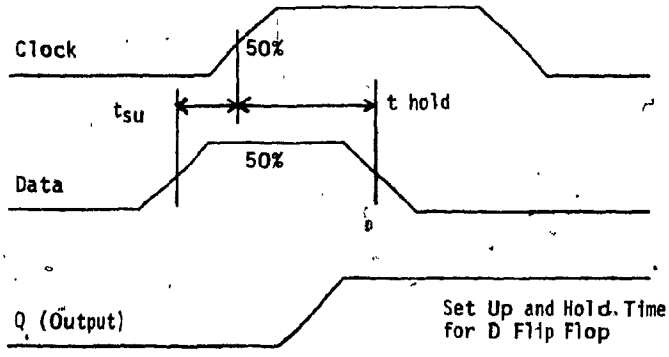
roundness of the waveforms. The set-up time is the minimum time that a data signal must be present at a flip flop (hence shift register) input before the transition of the clock pulse, for proper operation. The hold time is the minimum time after the clock transition that the data signal must remain unchanged at the input. For 10K logic, the gate propagation delay, rise time, fall time and hold time, are typically 2 nsec. The set-up time is approximately 3 nsec. (See Tables 2.2.1, 2.2.2)




Gate Propagation Delay $t_{x+y+}$




Rise and Fall Time $t_{x+x-}$




Important MECL 10K Operating Parameters
Fig.2.3.2

## 2.4 Circuit Layout

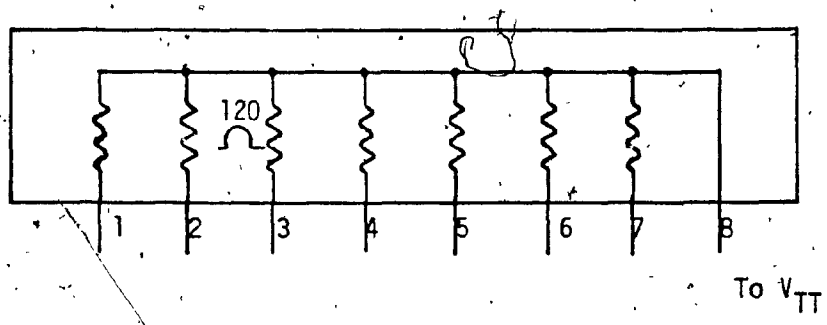
Due to the high speed (radio frequency) nature of 10K logic, careful circuit layout is imperative. To implement the codec, specially designed ECL 10K logic panels are employed (Augat type ECL 21-180). Fig.2.4.1 shows 3 such panels.

Augat ECL 21-180 Logic Panels

Fig.2.4.1

These panels are of the wire wrap type, with 3 ground planes. The first ground plane is on the IC side, and forms the $V_{CC}$ plane (0 volts, pins 1&16). The second ground plane is on the pin side and forms the $V_{EE}$ plane (-5.2 volts, pin 8). The $V_{EE}$ ground plane is also used for the signal (ac) ground, as it is the quietest. The third ground plane is sandwiched between the $V_{CC}$ and $V_{EE}$ planes, and forms the $V_{TT}$ plane (-2 volts).

Any interconnection between ECL 10K integrated circuits poses a transmission line problem, due to the edge speeds involved (≈2 nsec). Fortunately the use of several techniques greatly simplifies the problem [E.3,4,5]. Using #30 wire (typical for wire wrapping, 1 nsec ≈ 6 inches) over the $V_{EE}$ ground plane results in a line impedance within the range of 100 to 120Ω. A single wire run from one IC to another and terminated in 120Ω to the $V_{TT}$ plane, results in a good match, with <10% ringing. This termination also serves the dual purpose, from the DC point of view, of a pulldown resistor (all emitter coupled outputs are floating!) These 120Ω terminating-pulldown resistors are contained in single-in-line resistor packages (SIP), one SIP between two adjacent IC's. Each SIP contains 7 resistors, the common point being brought out on the 8th pin to the $V_{TT}$ plane as in Fig.2.4.2.

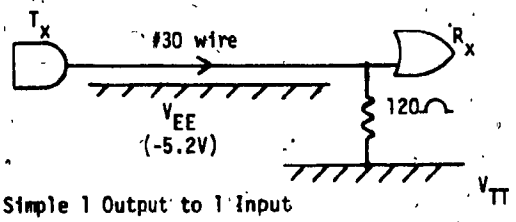



120Ω SIP Connection

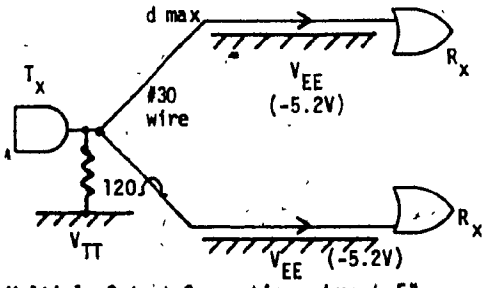
Fig.2.4.2

Several IC to IC modes of connection are used, and are illustrated in Fig.2.4.3. For a simple connection of 1 output to 1 input, the terminating-pulldown resistor is placed at the receiving end as in a. If a multiple connection is to be made, and the length of the largest connection is $\leq$5 inches, then the terminating-pulldown resistor is placed at the send end as in b. For any run over 16 inches, i.e. one board length, miniature 50$\Omega$coaxial line terminated in 120$\Omega$must be used as in c. All delay lines are implemented with a series damping resistor as shown in d.

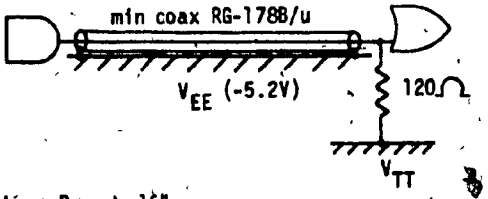
For circuit monitoring purposes, a separate gate output must be used to directly drive the monitoring instrument. The gate should be connected by miniature 50$\Omega$coax directly to the instrument, and terminated in 50$\Omega$. This is shown in e.(Note: just bridging an instrument arbitrarily, say a 1M$\Omega$/20pf scope on a line, produces excessive ringing and makes monitoring impossible).



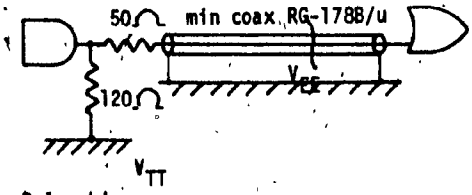

a - Simple 1 Output to 1 Input



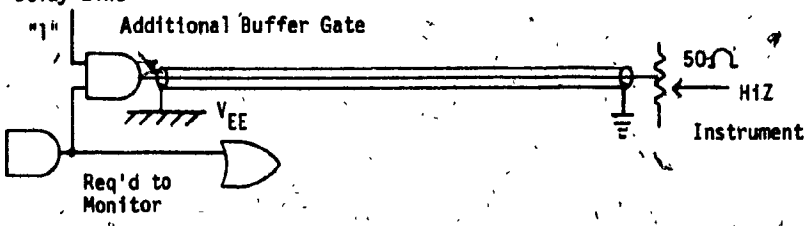

b - Multiple Output Connection, dmax ≤ 5"




c - Long Run ≥ 16"




d - Delay Line



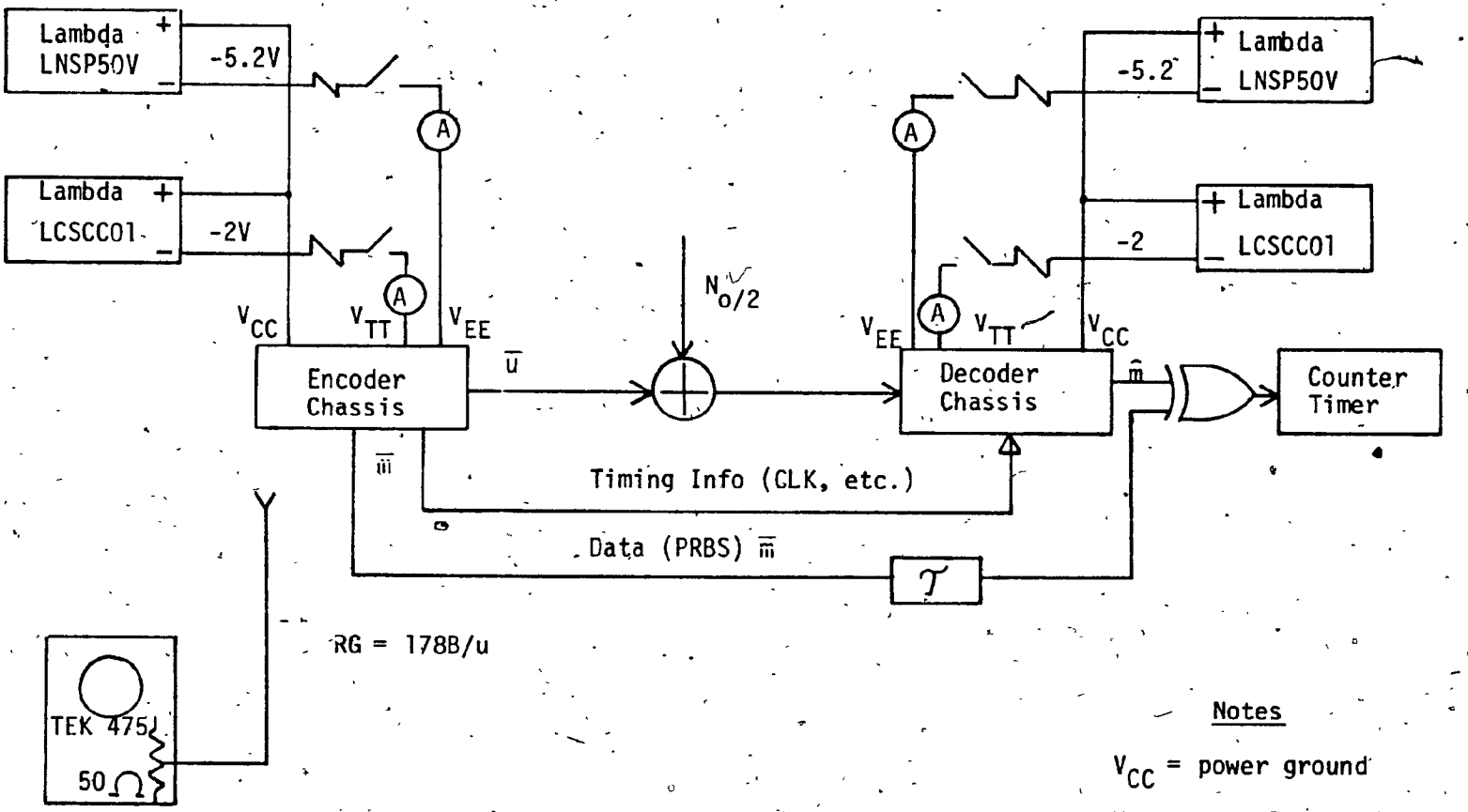

e - Monitor Point

ECL 10K IC to IC Interconnection
Fig.2.4.3

## 2.5 Experimental Set-up

Fig.2.5.1 illustrates the experimental set-up to be used to evaluate the codec performance. ECL requires 2 voltages: -5.2V for $V_{EE}$ and -2V for $V_{TT}$, with regulation better than 10%. A Lambda LNS-P-50V power supply provides the -5.2V. It has a 22A capability, with line and load regulation of .1% and 1.5mV rms ripple. A Lambda LCS-CC-01 provides the -2V. It has a 9A capability. Separate sets of fused and metered power supplies are provided for both the encoder and decoder. Each set has a capability of powering approximately 300 ECL 10K ICs (100mA/IC ex: MC10141 etc.).

As mentioned in Sec.1.5, the encoder provides all timing information directly to the decoder. The channel consists of a simple addition circuit (resistive adder), with an adjustable white Gaussian noise source (ex: 2xHP462A wideband amplifiers in cascade, with 1st amplifier input unterminated). Due to financial constraints, it was not possible to purchase equipment for the channel, or for a high rate pseudo-random data generator. For that reason, a compact pseudo-random generator is implemented as part of the encoder circuit. To evaluate the codec performance, it is simply a matter of comparing the transmitted message vector $\bar{m}$, available from the encoder (and suitably delayed), with the decoded message vector $\hat{m}$. Differences are recorded on a counter, and hence the error-rate performance can be calculated as a function of




Notes

$V_{CC}$ = power ground

$V_{EE}$ = signal ground

Experimental Setup

Fig.2.5.1

the level of additive white Gaussian noise.

All circuit monitoring is done with a Tektronix 475 osciloscope, with terminated coaxial fittings as outlined in the previous section. Instrumentation considerations [E.2b] dictate the measurement system response time should be approximately 10 times better than what is being tested. This means for faithful waveform reproduction, a scope needs a rise time of approx. .2 ns for ECL 10K. The 475 has a 200 MHZ bandwidth and a 1.8 nsec rise time. This means for a real rise time of 2 ns, the observed rise time is:

$$t_{robs} = \sqrt{t_{rs}^2 + t_{r_{DUT}}^2} = \sqrt{(1.8)^2 + (2)^2} = 2.7 \text{ ns}$$

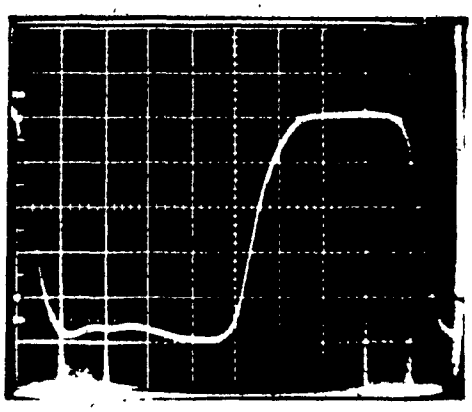
where

$t_{robs}$ = observed rise time on scope

$t_{rs}$ = measurement system rise time

$t_{r_{DUT}}$ = rise time of device under test

Note that this is 35% longer than the actual value. This factor must be considered when measuring actual ECL 10K rise times. It is not necessary, however, when measuring circuit delays between equivalent amplitude points on a waveform. Fig.2.5.2 illustrates the rise time of 10K pulse displayed on the TEK 475.



5 ns/div

ECL 10K Rise Time

Fig.2.5.2

Note: ECL 10K waveforms are measured from 20%-80% points,

$\therefore t_{rt_{20/80}} \cong 1.4 \times 2.7 = 3.8$ ns, 1.4 approx. 10/90 to 20/80 correction factor, due to waveform rounding.

# CHAPTER 3

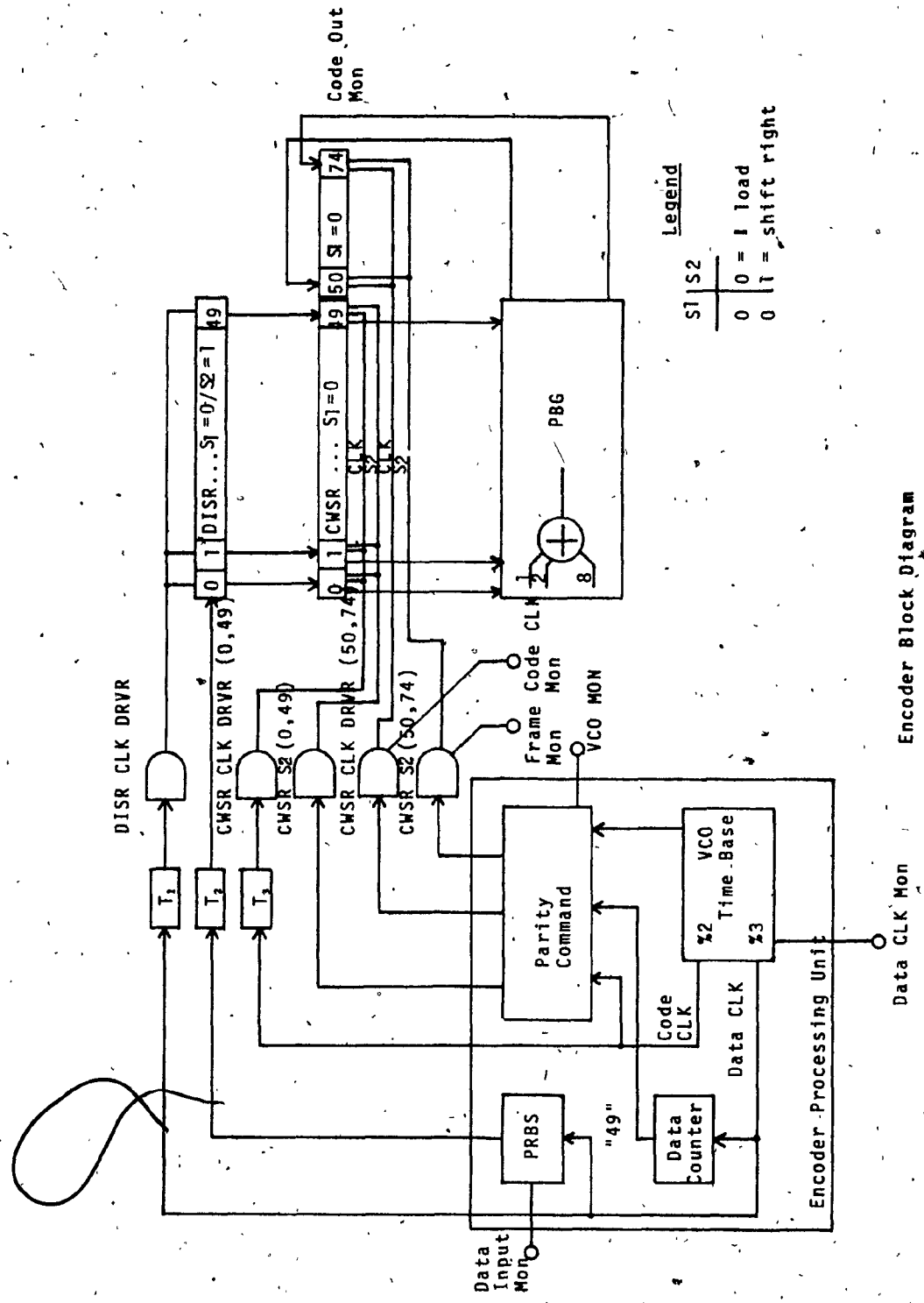
# ENCODER DESIGN

## 3.1 Detailed Functional

Fig.3.1.1 is a detailed block diagram of the complete encoder. It consists of 5 independent parts: the data input shift register (DISR), the code word shift register (CWSR), associated shift register drivers, the parity bit generator (PBG), and the encoder processing unit. The encoder processing unit consists of a time base, data counter, pseudo random binary sequence generator (PRBS) and parity command unit.

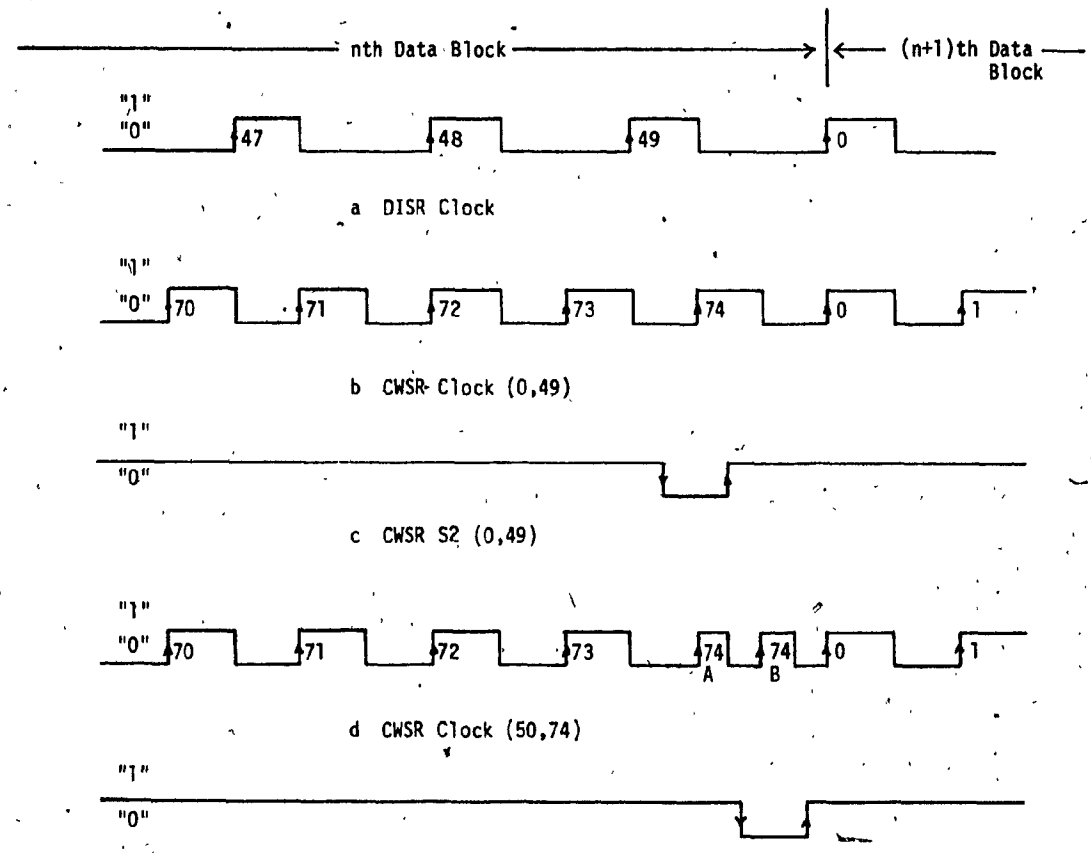
To eliminate the need for costly external generating equipment, and lengthy setup time, the encoder features a built-in time base and pseudo random generator. This means all clocking and data waveforms are available directly and retain a constant phase relationship to one another, as they are derived from a master oscillator. The pseudo random generator also provides a direct means of real time testing any shift register sequence, using a regular analog oscilloscope. All necessary waveforms are available to the decoder via external monitor points.

Fig.3.1.2 shows the essential operating encoder waveforms. Their exact generation and properties are discussed in more detail in Sec.3.5. The data clock applied to the DISR is shown in a. Upon the positive going edge of the clock, data is transferred one bit to the right. Note that after the




Encoder Block Diagram
Fig.3.1.1

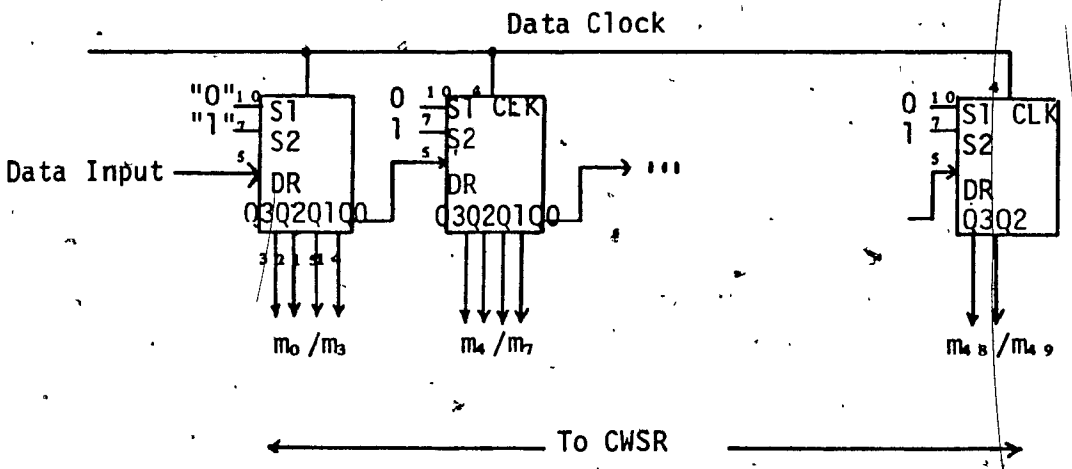
50th pulse (49), the next block of 50 data bits enters the DISR. The DISR shifts continuously to the right. The CWSR clock for the first 50 positions (0,49) is shown in b. Note that these positions shift right continuously at 1.5 times the data rate (code rate). After the 50th bit has been loaded into the DISR, and upon receipt of the 75th (74) CWSR clock pulse, the CWSR S2 (0,49) signal goes to "0" as in c. This tells the first 50 bit positions of the CWSR to parallel load data from the DISR. The CWSR clock waveform for the last 25 positions of the register is shown in d. These positions shift uniformly to the right at 1.5 x data rata (code rate) as the first 50 positions. At the 75th clock position, there is a double clock pulse, however. The first pulse (74A) tells the register to shift the last or 75th code bit into the channel. The second pulse (74B) tells the 25 bit positions (50,74) to parallel load the 25 parity bits of the next data word. Note that this parallel entry is triggered by the CWSR S2 position going to "0" as in e.




Encoder Clocking and Loading Waveforms

Fig.3.1.2

## 3.2. Data Input Shift Register (DISR) and Driver

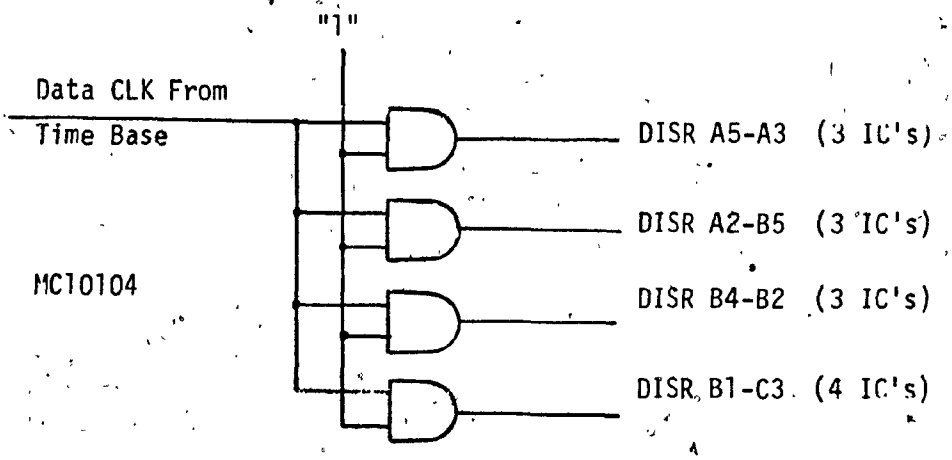



Data Input Shift Register
Fig.3.2.1

Fig.3.2.1 illustrates the construction of the data input shift register (for actual wiring diagram, see Appendix III Fig.AIII-3). The 50 bit ($m_0$ ,$m_{49}$) register is formed by using 13 MC10141 IC's (Fig.AIII-1). Each MC10141 IC contains a universal 4 bit shift register. Only the first two bits are used on the 13th IC. This register shifts right continuously as per sec.1.5, and is programmed by hard-wiring the S1 input (pin 10) to "0" and S2(pin 7) to "1". Data from the pseudo

random generator is connected to the data input on the 1st IC at pin 5. Each subsequent IC is wired in series with the previous IC, the data input pin 5 connected to the last output Q0, pin 14. Each MC10141 IC has separate buffered outputs from the 4 flip flop storage elements, Q3,Q2,Q1 and Q0. These outputs, which represent $m_0, m_1, m_2$ etc. are wired directly to the corresponding inputs of the code word shift register (forming $u_0, u_1, u_2$ etc.) as shown in Fig.3.1.1.

The clock signal for the 50 bit shift register is provided by 4 separate gates, as in Fig.3.2.2. Each gate drives 3 separate IC's, with the fourth driving 4 IC's.



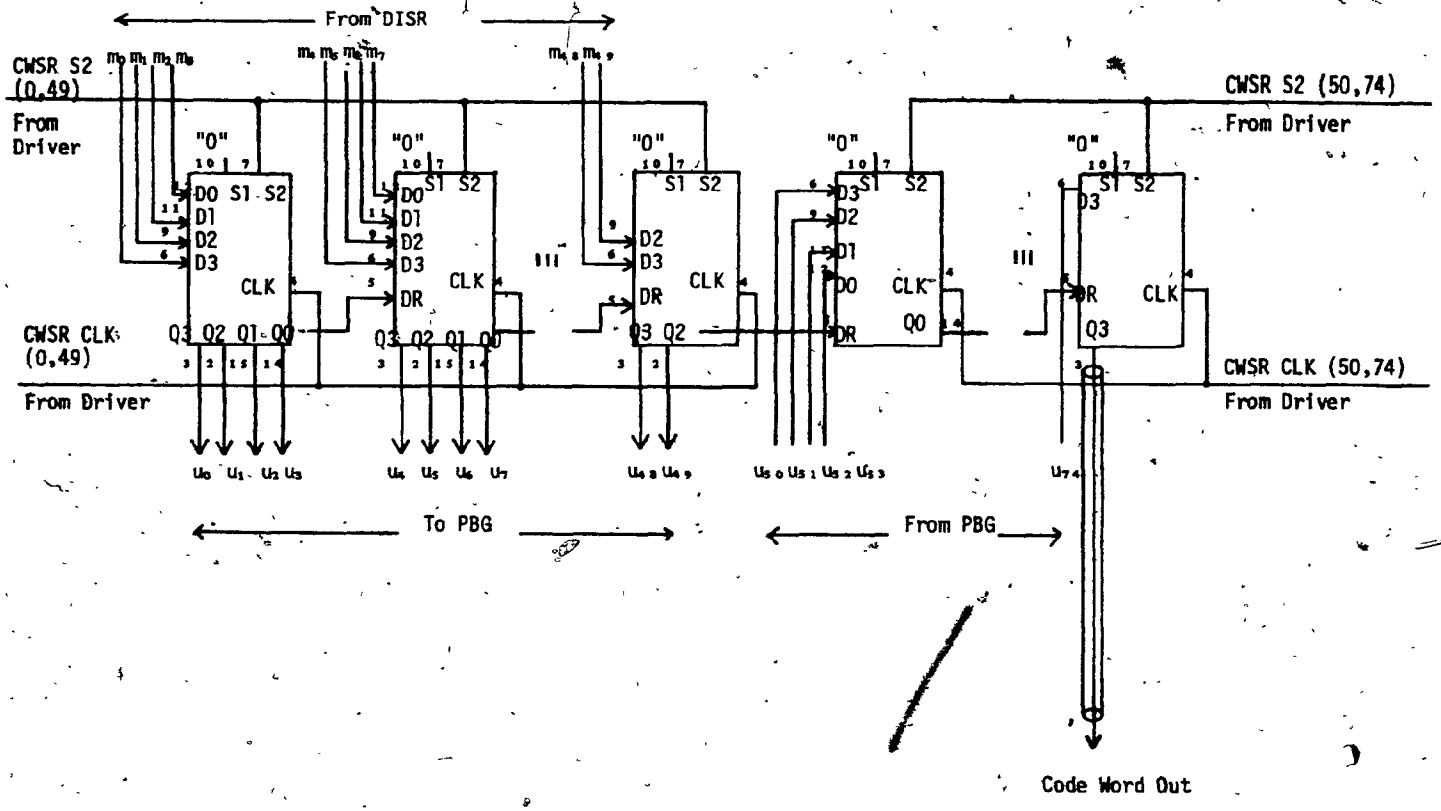

DISR Clock Driver
Fig.3.2.2

For actual wiring see Fig.AIII-3. The individual IC locations on the logic panel are shown in Fig.3.6.2.

## 3.3 Code Word Shift Register (CWSR) and Drivers

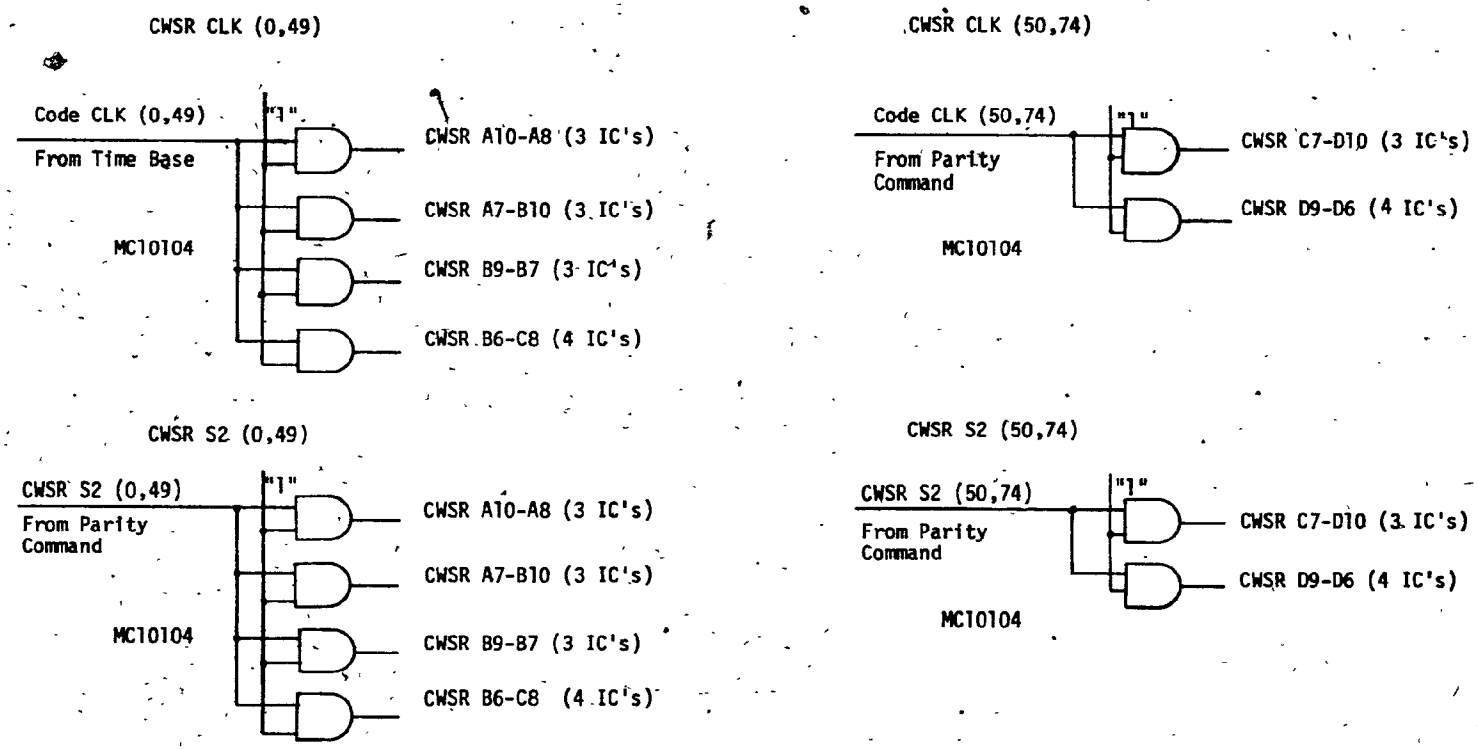
Fig.3.3.1 shows the code word shift register. It consists of 20 MC10141 IC's wired in cascade. The register is divided into two functionally different sections: the first 13 IC's receive parallel data from the DISR ($m_0, m_1, ..m_{49}$) and the last 7 IC's are loaded with the calculated parity bits ($u_{50}, u_{51}, ..u_{74}$). For actual wiring diagram, see Fig.AIII-4. Note there are two clocking signals, the CWSR CLK (0,49) for the first 13 IC's, and the CWSR CLK (50,74) for the last 7 IC's. There are similarly 2 shift right/parallel load controls CWSR S2(0,49) and CWSR S2(50,74). The buffered outputs of the first 13 IC's ($u_0, u_1, ..u_{49}$) provide the input data to the parity bit generator, to enable calculation of the required 25 parity bits.

The 2 clock signals and shift right/parallel load commands are provided by the drivers shown in Fig.3.3.2. Each gate drives 3 separate IC's, with the last driving 4 IC's. For actual wiring, see Fig.AIII-5. The individual IC locations on the logic panel are shown in Fig.3.6.2.




Code Word Shift Register

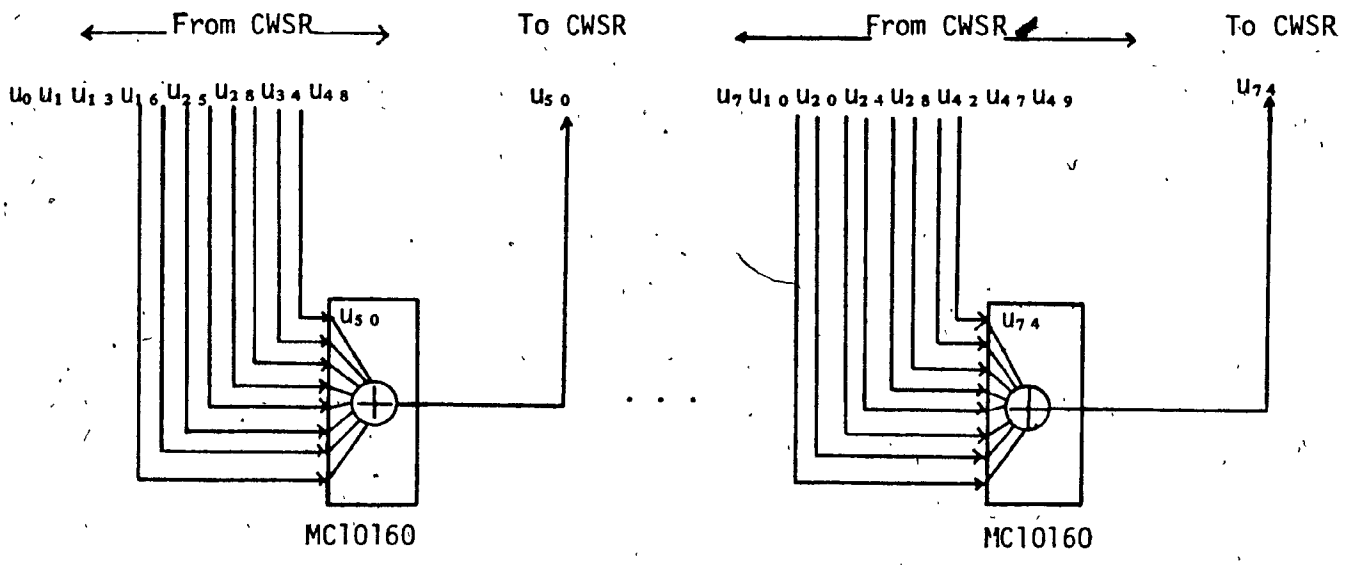
Fig.3.3.1




Code Word Shift Register Drivers

Fig.3.3.2

## 3.4 Parity Bit Generator (PBG)




Parity Bit Generator

Fig.3.4

Fig.3.4 shows the parity bit generator. It consists of 25 MC10160 exclusive -OR IC's (Fig.AIII-2), one per parity bit. The inputs are taken from the buffered outputs of the CWSR, 8 inputs per parity bit. These inputs are given in Appendix I. The outputs are loaded into the last 25 positions of the CWSR as discussed in Sec.3.1. Although there is a continuous waveform at the output of each IC, only the value during the 75th code clock pulse is operative. For actual wiring diagram see Fig.AIII-6.

## 3.5 Encoder Processing Unit

As discussed in Sec.3.1, the encoder processing unit generates all the necessary data and control functions to supply and operate the DISR and the CWSR. Its principal components are: a time base, data counter, pseudo random binary sequence generator and parity command unit. They are described in the following subsections.

### 3.5.1 Time Base

The time base is shown in Fig.3.5.1. All waveforms are derived from the master oscillator. The master oscillator is an MC 1648 IC, free running at 45MHZ. The free running frequency $f_o$ is given by:

$$f_o = \frac{1}{2\pi\sqrt{L(C+C\ shunt)}}\ ,\ C\ shunt\ depends\ on\ setup$$

$L$ = 7 turns #24 wire, on a SF(6) T50 core

$\therefore\ L \tilde{=}\ 40\mu H/100T = .196\mu H/7\ turns,\ \propto 7^2/100^2$ [E.9]

$C$ = 27pF

$$f_o = \frac{1}{2\pi\sqrt{.196\times10^{-6}\times27\times10^{-12}}} = 69MHZ$$

This indicates that the shunt capacitance is approximately of the same order as the tank capacitor, to yield an output of 45MHZ.

The MC1648 is designed to operate as a VCO (hence phase locked) by addition of suitable varactors. Thus, if necessary, the oscillator can be synchronized to an external clock. And gate buffers provide the VCO to the rest of the circuit and to the parity command unit.

The data clock is obtained by dividing the VCO by 3 (15MHZ). This is accomplished by using a modulo 3 ring counter [E.6,7], implemented by an MC10135 dual JK flip flop. The code clock (0,49) is obtained by dividing the VCO by 2 (22.5MHZ). This is performed by a simple flip flop, implemented by ½ of an MC10135. Note that the MC10135 uses complementary inputs $\bar{J}$ & $\bar{K}$ vs J & K. And gates provide buffering for feeding other circuits and monitoring.

Depending on the initial states of the flip flops, the phase relationship of the %2 and %3 outputs may not be as given in Fig.3.1.2. If this happens, the circuit must be turned off and powered up again, until the correct relationship appears (Sec.3.7 explains further).

The complete wiring schematic is given in Fig.AIII-7.

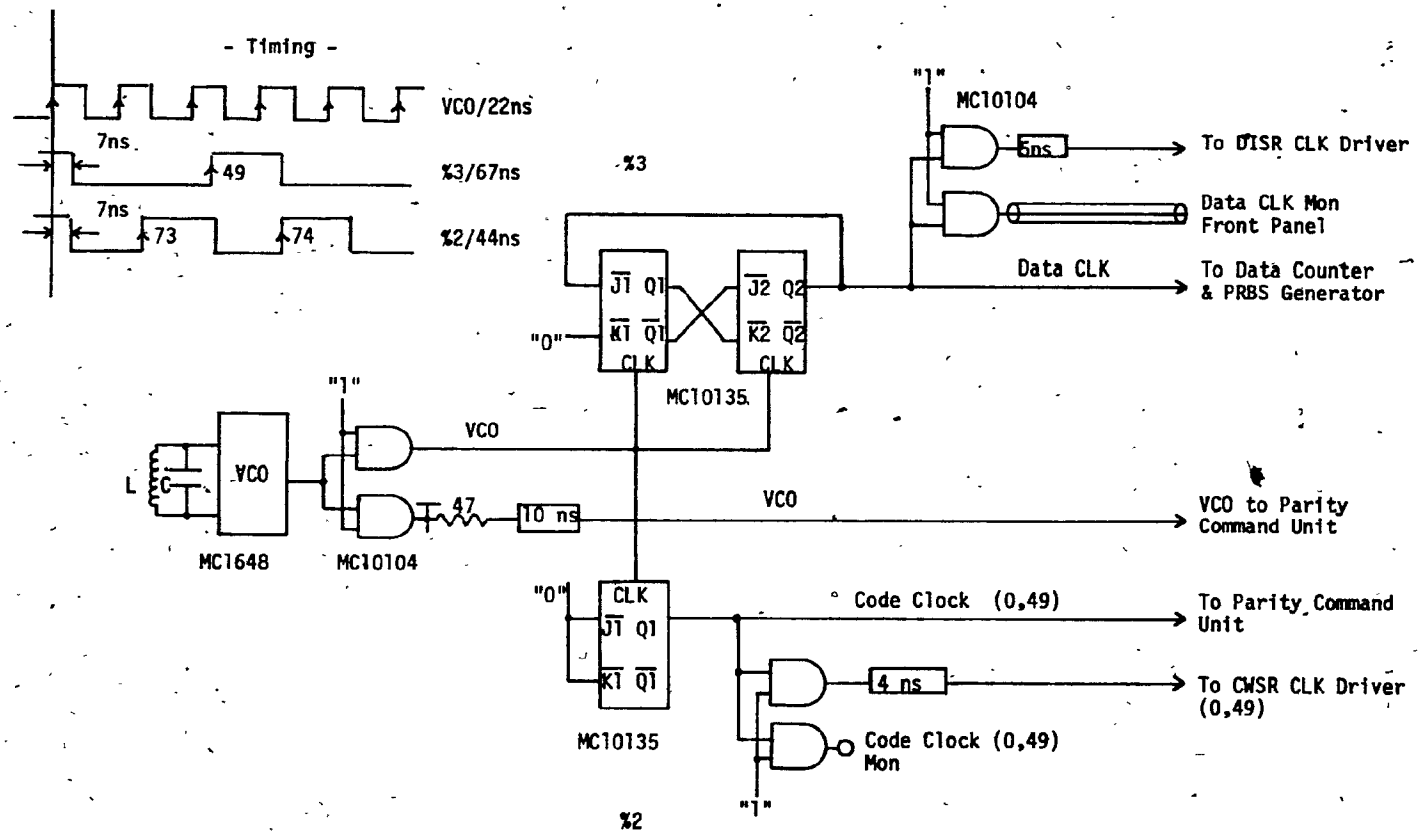
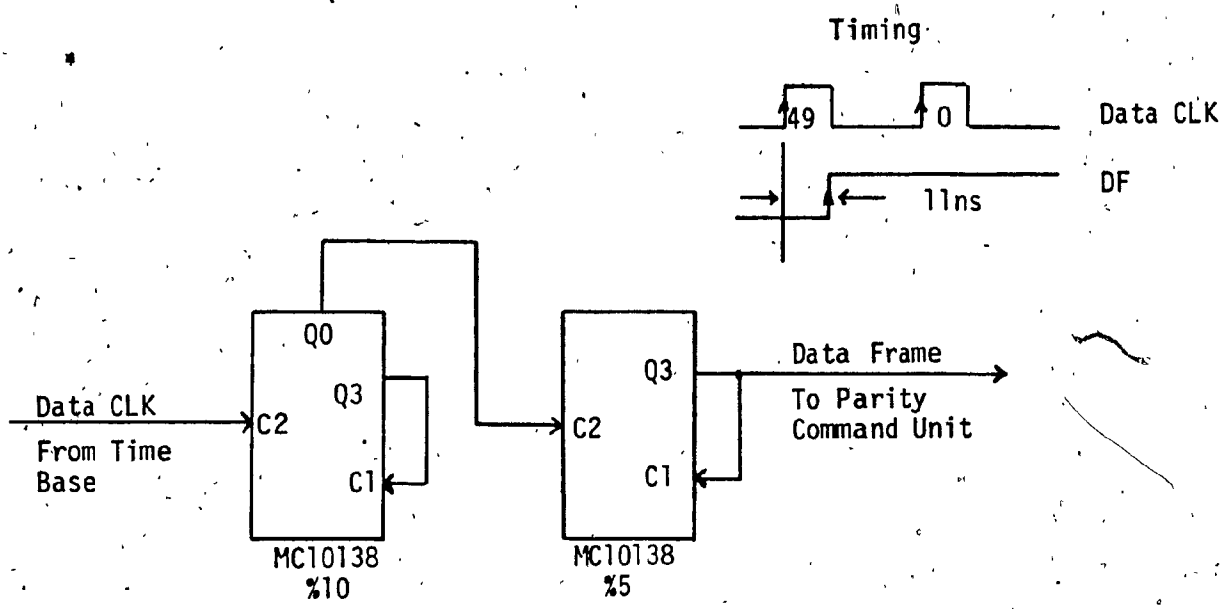

- Timing -
VCO/22ns
7ns
49
%3/67ns
7ns
73
74
%2/44ns
%3
"1"
MC10104
5ns
To DISR CLK Driver
Data CLK Mon
Front Panel
Data CLK
To Data Counter
& PRBS Generator
J1 Q1
K1 Q1
CLK
J2 Q2
K2 Q2
CLK
"0"
MC10135
"1"
VCO
L
C
VCO
47
10 ns
VCO
VCO to Parity
Command Unit
MC1648
MC10104
"0"
CLK
J1 Q1
K1 Q1
Code Clock (0,49)
To Parity Command
Unit
4 ns
To CWSR CLK Driver
(0,49)
Code Clock (0,49)
Mon
MC10135
"1"
%2
Time Base

Fig.3.5.1

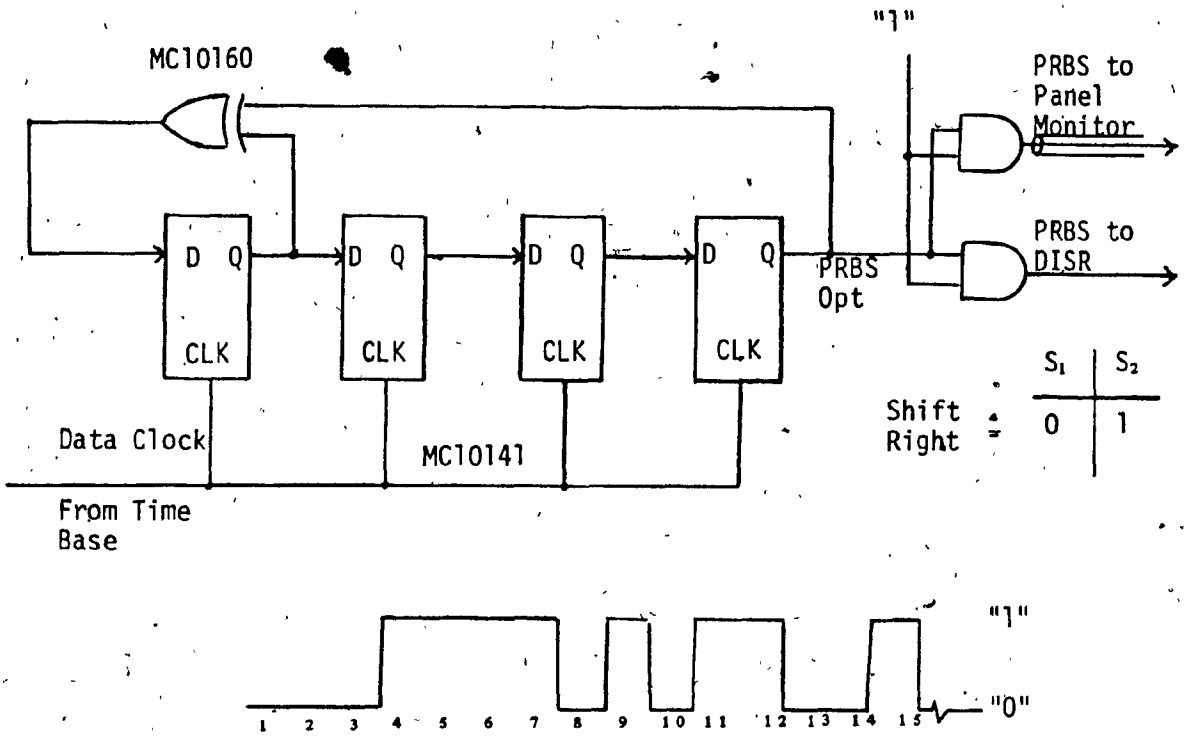
### 3.5.2 Data Counter




Data Counter
Fig.3.5.2

The data counter is shown in Fig.3.5.2. It senses the occurrence of the 50th (49) data clock pulse. It is implemented by 2 MC10138 bi-quinary counters wired in cascade, the first %10, the second %5. The output provides the data frame pulse for the parity command unit. See Fig.AIII-8 for the wiring schematic.

### 3.5.3 Pseudo Random Binary Sequence Generator




Pseudo Random Binary Sequence Generator

Fig.3.5.3.1

The pseudo random binary sequence generator is formed from a 4 stage linear feedback shift register, using an MC10141 IC. To generate a maximal length, or m sequence the feedback taps are taken from the 1st and 4th positions as shown in Fig.3.5.3.1 [C.14]. The output sequence is = $2^n-1 = 2^4-1 = 15$ bits long, and is as shown.

The primary purpose of the generator is to provide a simulation for data, and yet have a set pattern to enable

testing of the various modules, DISR, CWSR etc. For example, after 3 blocks of data, or 150 bits, the 15 bit m sequence starts again at exactly the same position (150/15=10). This makes it easily verifiable on a suitable oscilloscope. Longer or shorter patterns do not give this effect and make testing difficult. (When all circuitry of the codec is tested 100%, the generator can be easily modified to produce longer sequences to measure bit error rates in a channel etc., although this is not the primary purpose as stated here.)

The power spectrum of an m sequence is given by:

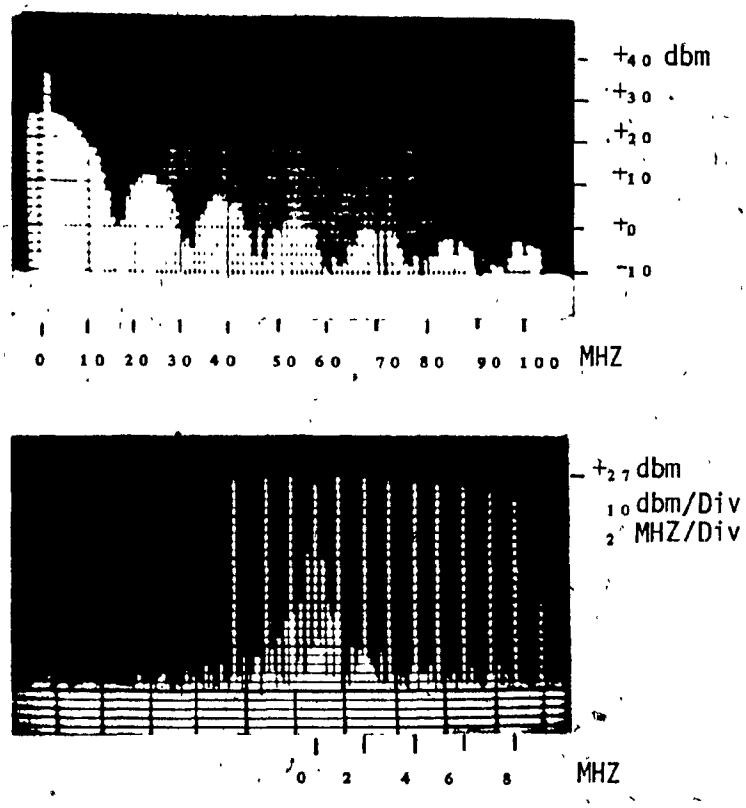
$$S(f) = \left[\frac{p+1}{p}\right]^2 \left[\frac{\sin \pi f t_0}{\pi f t_0}\right]^2 \sum_{\substack{n=-\infty \\ n \neq 0}}^{\infty} S\left[f - \frac{n}{p t_0}\right] + \frac{1}{p^2}\delta(f)$$

$$p = 2^n - 1 = 15$$

$$t_0 = \frac{1}{\text{data clock}} = \frac{1}{15\text{MHZ}}$$

$$S(f) = 1.14 \left[\frac{\sin \pi f/15\text{MHZ}}{\pi f/15\text{MHZ}}\right]^2 \sum_{\substack{n=-\infty \\ n \neq 0}}^{\infty} S\left[f - n\text{x}1\text{MHZ}\right] + \frac{1}{225}\delta(f)$$

The spectrum is shown in Fig. 3.5.3.2. Note that the individual lines spaced at 1MHZ, with the $(\sin x/x)^2$ roll off, with the first null at 15MHZ.




Power Spectrum of 15 bit/15MHZ m Sequence

Fig.3.5.3.2

### 3.5.4 Parity Command Unit

The parity command unit generates the load and clock commands for the CWSR. The complete set of waveforms is shown in Fig.3.5.4.1, together with their derivation. The logic diagram is shown in Fig.3.5.4.2 with the corresponding waveforms A,B,C etc. marked. The wiring schematic is shown in Fig.AIII-10.

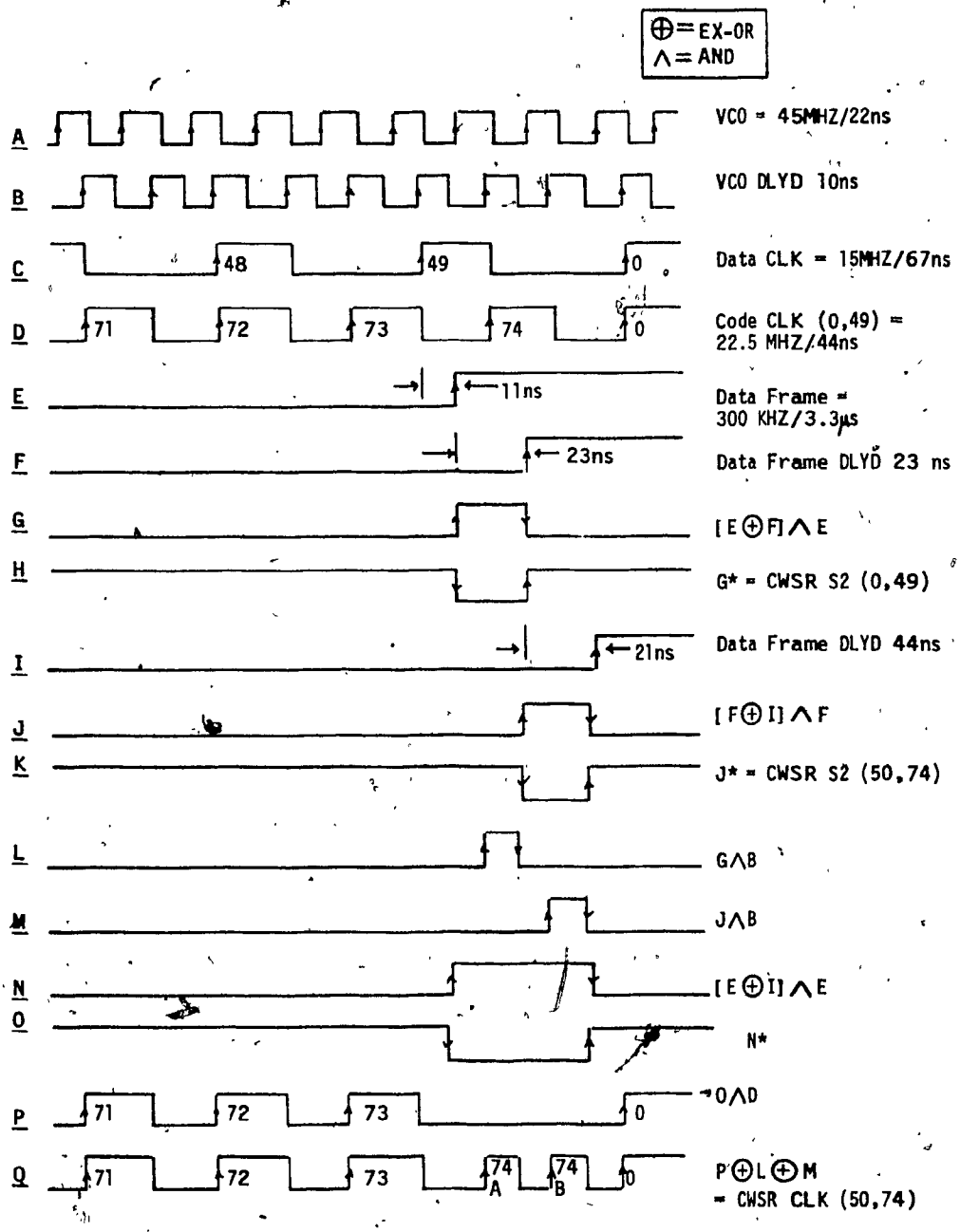
Essentially 3 waveforms have to be constructed. The parallel load command for the first 50 bits of the CWSR-CWSR S2 (0,49), the parallel load command for the last 25 bits of the CWSR-CWSR S2 (50,74), and the composite clock for the last 25 bits - CWSR CLK (50,74).

CWSR S2 (0,49) The data frame pulse E is delayed by 23ns and ex-OR'd with itself and then gated by itself, to form the pulse shown in G. G is then inverted to form H. H is the required result, as it is in the exact position relative to the data and code clock pulses to parallel load the 50 data bits into the CWSR.

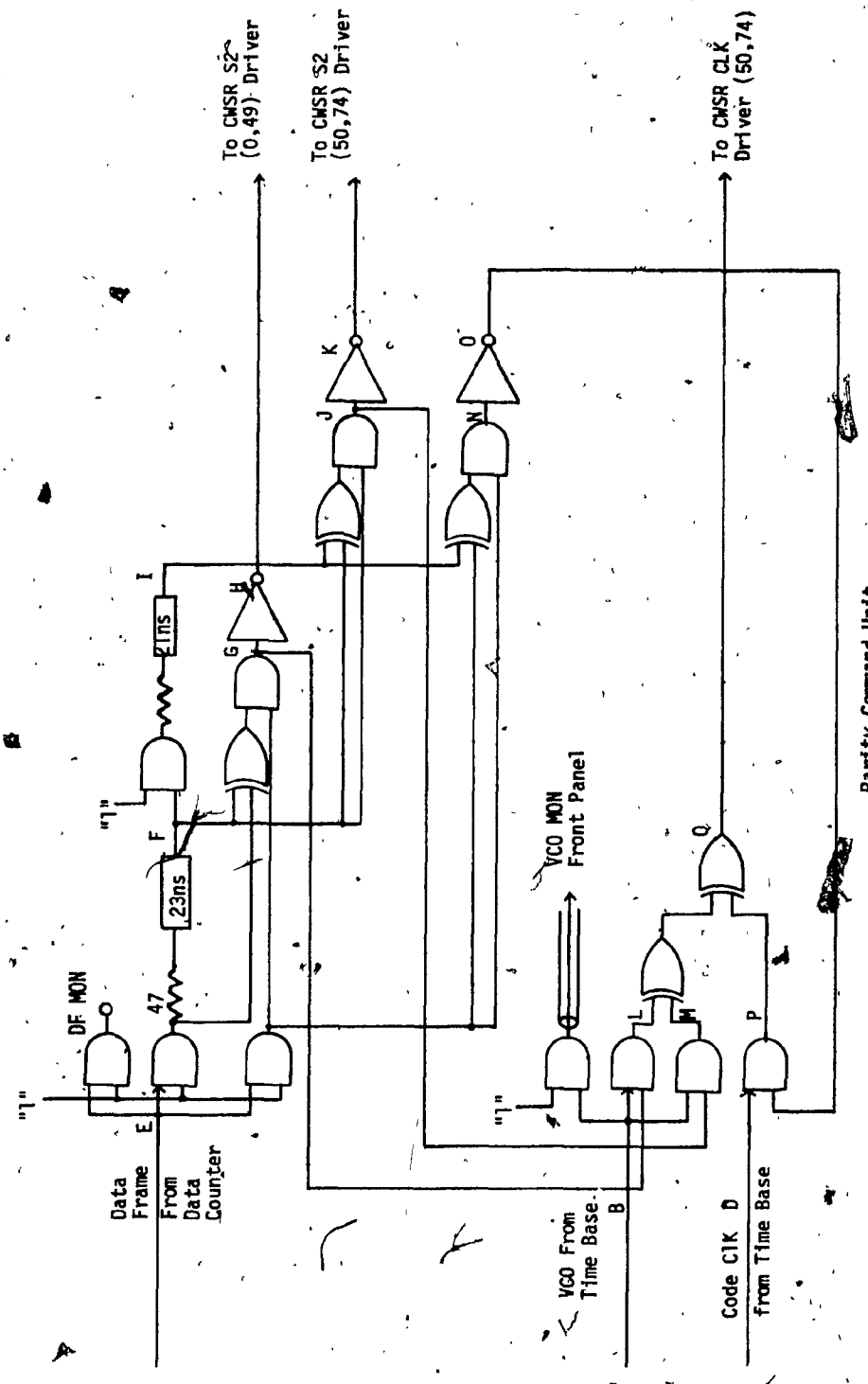
CWSR S2 (50,74) The F waveform (DF delayed 23ns) is further delayed by 21 ns. This is ex-OR'd with itself and then gated by itself to form the pulse shown in J. J is then inverted to form K, the required result. Note that the 25 parity bits are loaded on the 74B clock pulse.

CWSR CLK (50,74) The G and J waveforms are used as gates to select two clock pulses L and M in the 74th clock period.

A blanking waveform (data frame ex-OR'd data frame delayed 44 ns, and gated data frame) sets the 74th clock period blank as in P, where upon the 2 clock pulses are added to form the required composite form of Q.



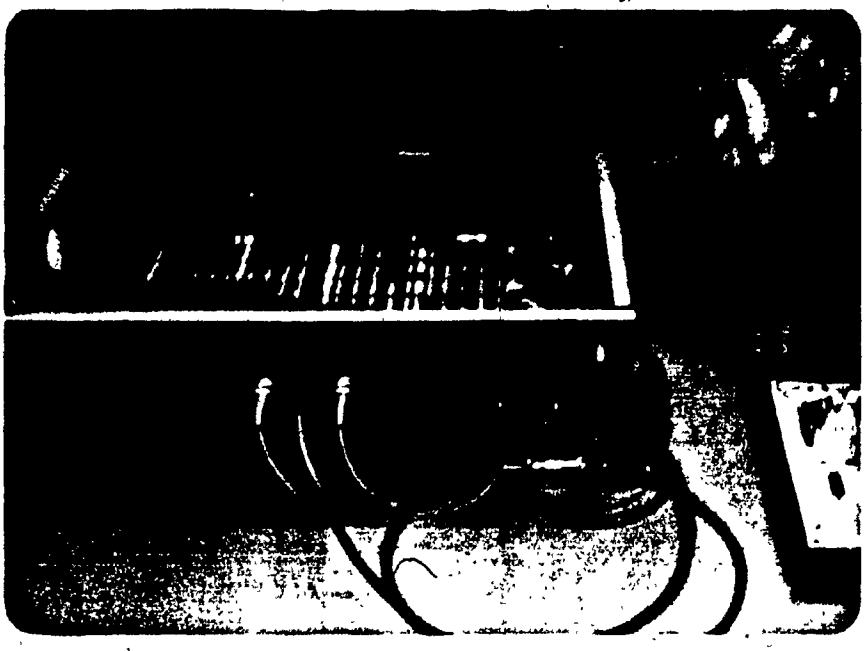

Parity Command Unit Waveforms
Fig.3.5.4.1




Parity Command Unit

Fig.3.5.4.2

## 3.6 Encoder Chassis



Encoder Chassis
Fig.3.6.1

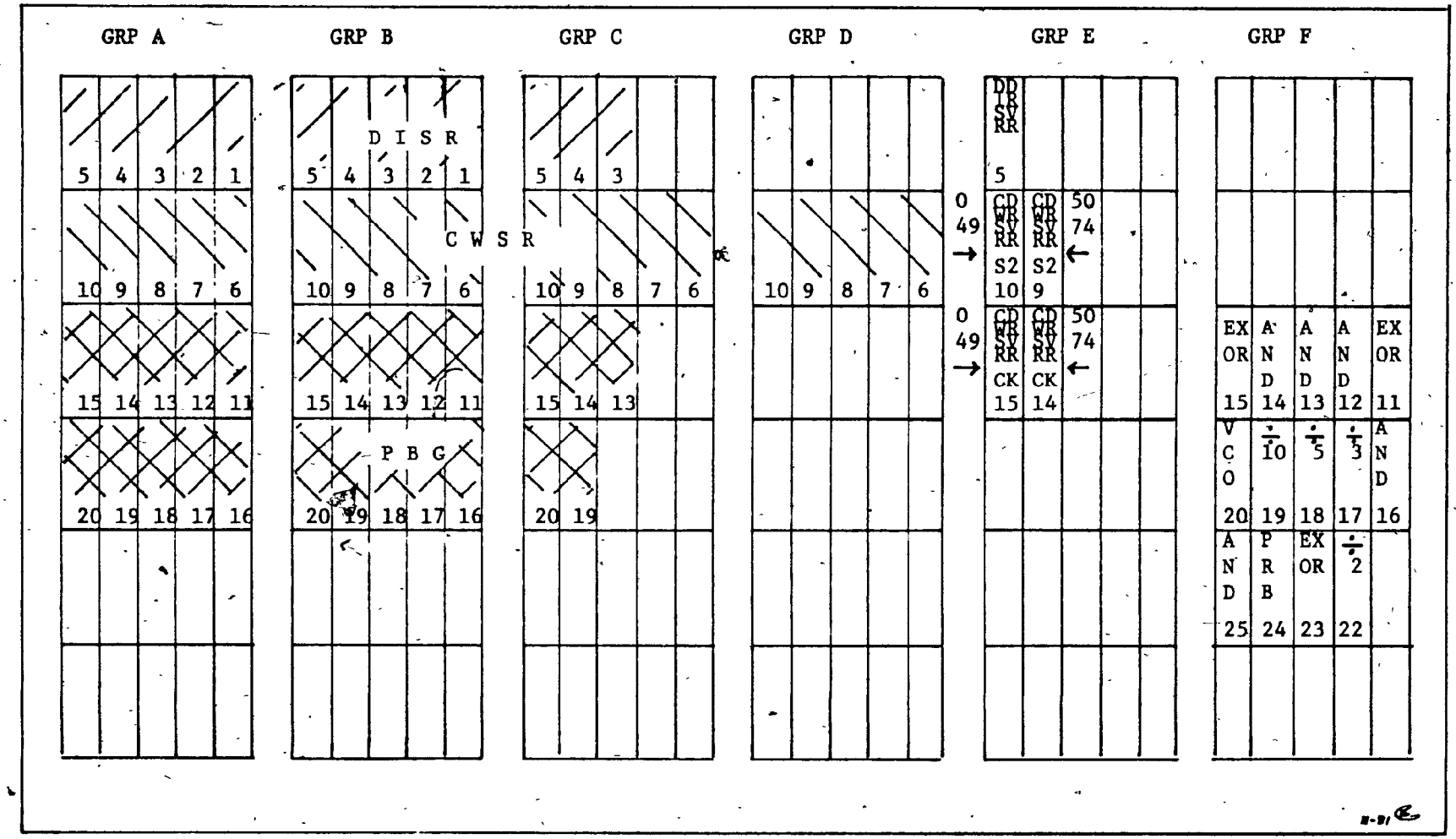
Fig.3.6.1 shows the completed encoder chassis. The Augat logic panel is mounted in the chassis with pins facing upwards to facilitate changes and testing. A small muffin fan is mounted on the rear of the unit and must be operated whenever the panel is powered (IC's will burn up otherwise).

The separate $V_{EE}$ and $V_{TT}$ power connections to the logic

panel are first brought out to the front of the unit and fused. The connection to the actual power supplies is made at the rear of the unit on terminal strips provided. To ease operation, the $V_{EE}$ and $V_{TT}$ buses are strapped together and the unit powered from -5.2V alone. This is a simplification and does away with using the $V_{TT}$ supply. The drawback is the increased power consumption and circuit heat dissipation.

All the various circuit monitoring points are brought out to unterminated BNC terminals on the front face. To monitor a particular waveform, a 50Ω cable should be run and terminated at the instrument in 50Ω. For oscilloscope use, the 'frame' point is used as a trigger source (then use delayed sweep). As mentioned in Sec.3.5.1, the proper phase relationship must be present between the %2 and %3 circuitry in the time base. Thus the unit may have to be powered on /off several times before an output appears at the 'code out' position (positive check will show correct position of the code clock pulses 74A & 74B. In the incorrect position they overlap with 73).

The exact location of any IC is determined from Fig.3.6.2. The DISR, CWSR and PBG are located in the top left hand corner of the panel, with the other elements in the bottom right hand corner.




ENCODER CIRCUIT BOARD LAYOUT/ IC SIDE

Fig.3.6.2

## 3.7 Encoder Performance

Some of the most important characteristics of the encoder may be seen in Fig.3.7.1-3.7.6.

Fig.3.7.1 shows the data clock (15MHZ) vs the code clock (22.5MHZ) at the output of the time base. Note the code clock is 1.5 times faster.

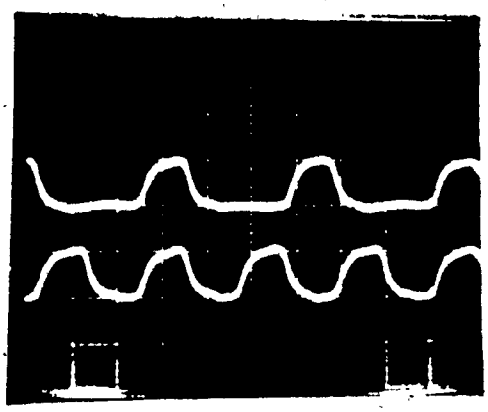
Fig.3.7.2 shows the CWSR S2 (0,49) command, which tells the CWSR when to parallel load the 50 bit data word into the first 50 register positions. It occurs as shown in the 75th code clock period, when all code bits except the last one in position $u_{74}$ have been shifted into the channel. Code clock pulse 74A shifts this last position into the channel.

Fig.3.7.3 shows the CWSR S2 (50,74) command, which tells the CWSR when to parallel load the 25 parity bits. Note that it occurs after code clock pulse 74A, which shifts the last code bit into the channel. Code clock pulse 74B loads the 25 parity bits.

Fig.3.7.4 shows a comparision between the data in and the coded data out. Since the code is in systematic form, the coded data output is separate from the parity bits. Note the compression due to the higher (x1.5) code rate. (Only data is shown).

Fig.3.7.5 shows the last 19 positions of the code word, showing only the various parity bits.

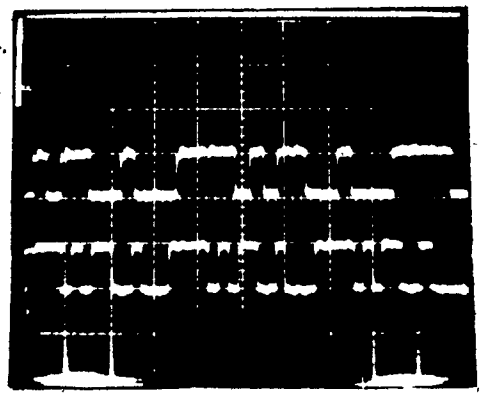
Fig.3.7.6 shows a complete code word with 50 data bits (PRBS pattern) and 25 parity bits. The pattern repeats itself exactly after 3 data frames, as the 150 bits is exactly divisible by the m sequence length 15.



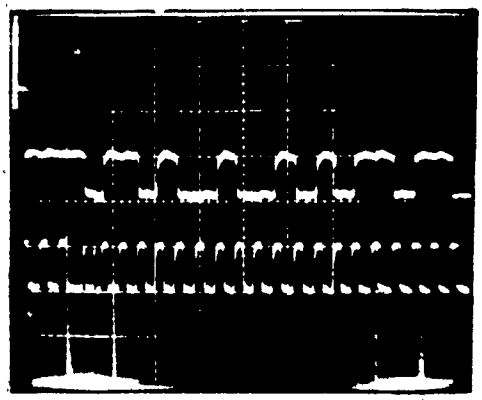
Data Clock & Code Clock, 20 ns/cm
Fig.3.7.1

CWSR S2 (0,49) & CWSR Clock (50,74), 20 ns/cm
Fig.3.7.2

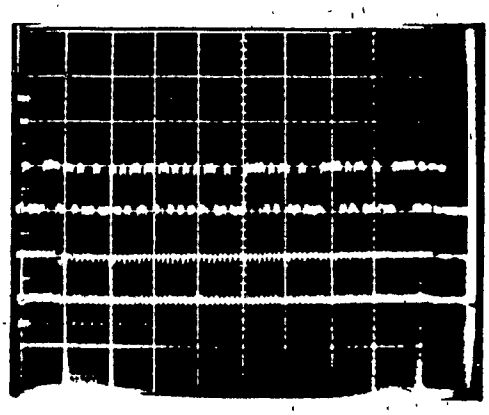
CWSR S2 (50,74) & CWSR Clock (50,74), 20 ns/cm
Fig.3.7.3



Data Word In & Code Word Out, 100 ns/cm
Fig.3.7.4



Code Word Parity Bits & CWSR Clock (50,74), 100 ns/cm
Fig.3.7.5



Complete Code Word & CWSR Clock (50,74)
Fig.3.7.6

# CHAPTER 4

# DECODER DESIGN

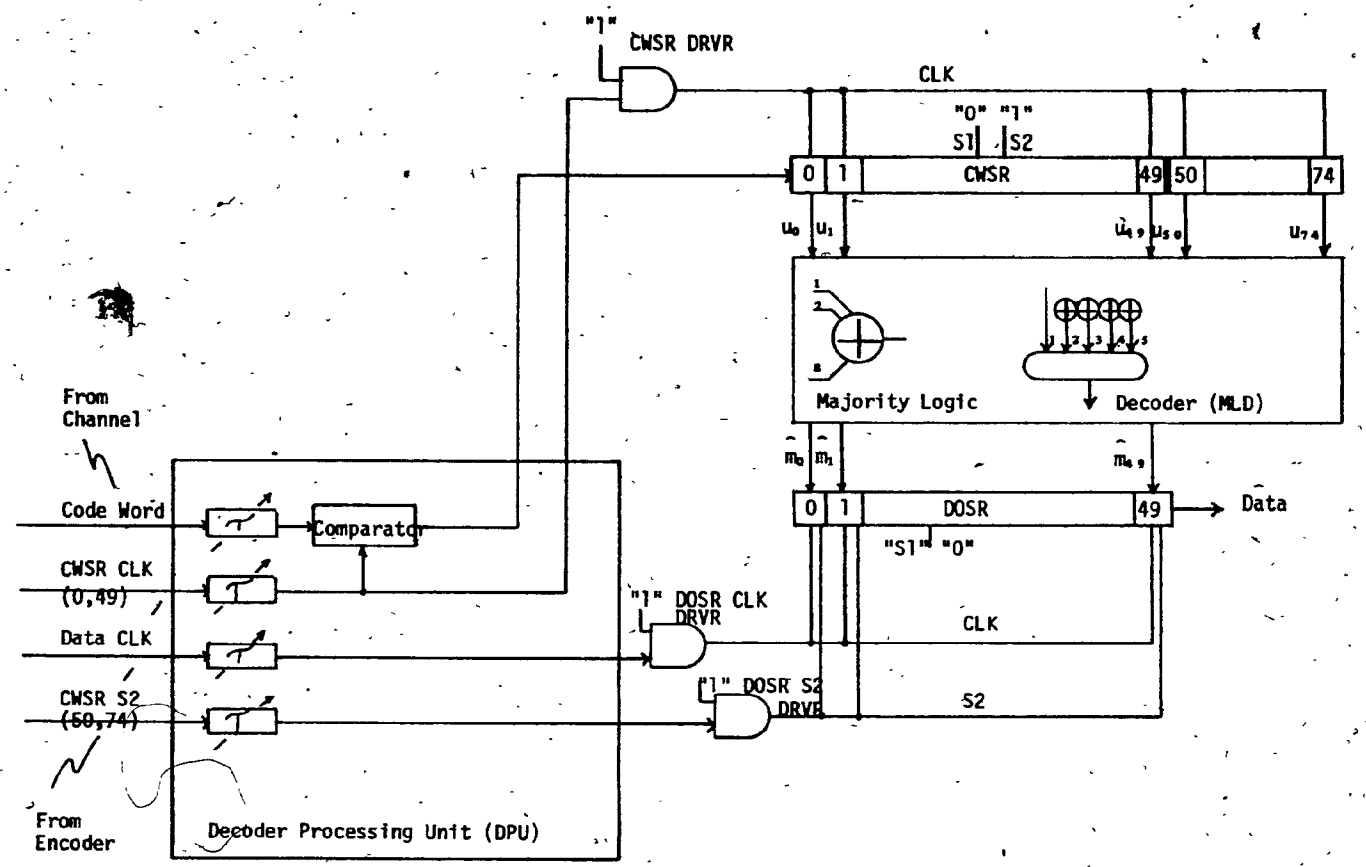
## 4.1 Detailed Functional

Fig.4.1.1 shows the complete decoder design. As mentioned previously, extensive use is made of the various encoder waveforms, specifically the code clock (0,49), data clock, and CWSR S2 (50,74). This removes the need for local timing regeneration which is a separate problem apart from the evaluation of the codec performance. The major components are the code word shift register (CWSR), the majority logic decoder (MLD), the data output register (DOSR), associated drivers and the decoder processing unit.

The important decoder operational waveforms are shown in Fig.4.1.2. Note that all encoder waveforms received by the decoder first pass through adjustable delay networks so that any arbitrary 'channel' can be constructed, and the waveforms retain their proper phase relationship.

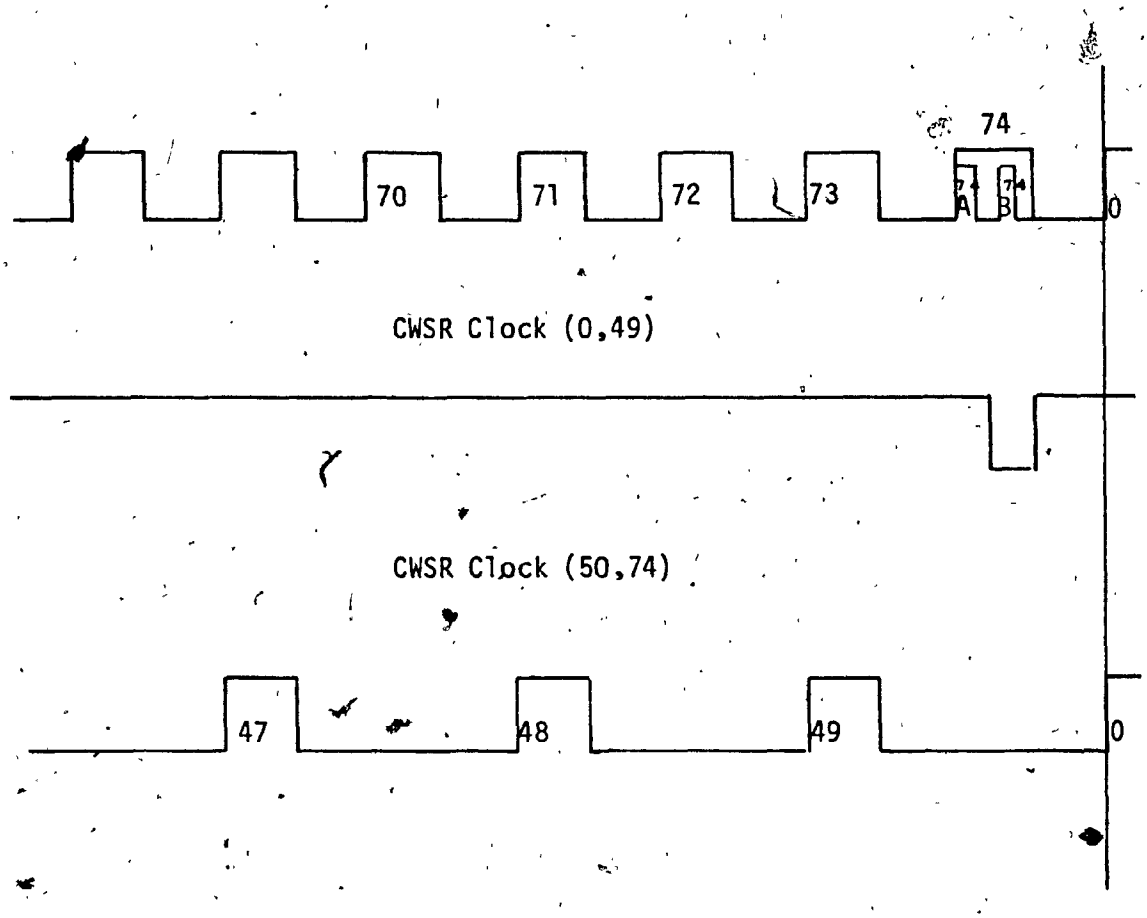
The code word received from the channel is corrupted by noise, interference etc. Thus it must first pass through a voltage comparator which samples this signal at the code rate, to produce a 'clean' waveform. This then passes directly to the CWSR. The CWSR is continually shifting right at the code rate. It is hard wired to the majority logic decoding circuitry through buffered parallel outputs. The output of the majority logic gates are hard wired to the parallel loading inputs of the DOSR. When the 75th code

bit has been loaded into the CWSR, the CWSR S2 (50,74) command tells the DOSR to parallel load the 50 estimated message bits. This occurs a sufficient time after the 75th code clock pulse (74A) to allow proper majority logic decoding. Note that the outputs of the MLD are all spurious until the 75th clock pulse period, when they form the estimated message bits. The DOSR then shifts the decoded message bit out at the data rate.




Detailed Decoder Design

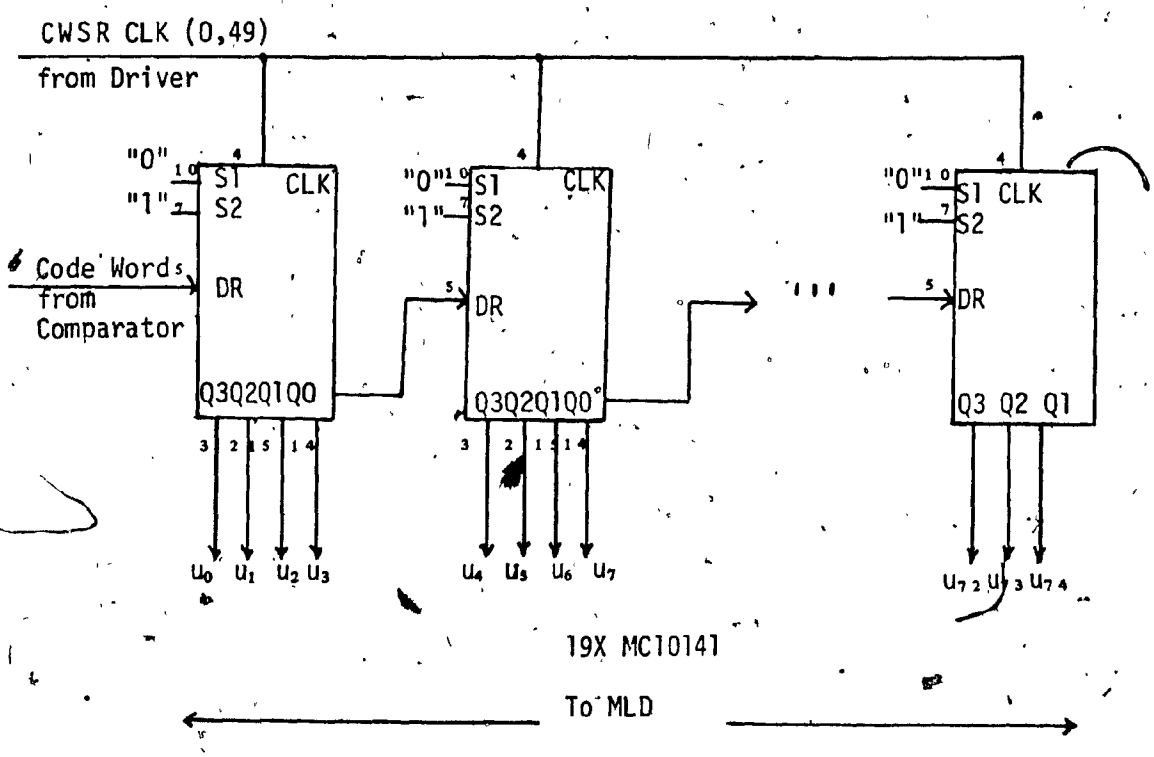
Fig.4.1.1




Decoder Clocking and Loading Waveforms

Fig.4.1.8

## 4.2 Code Word Shift Register & Driver

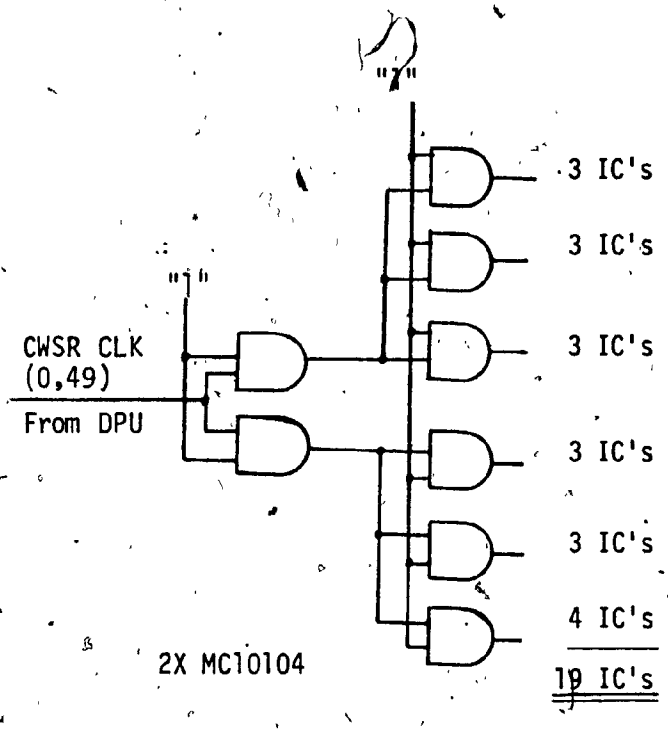



Code Word Shift Register

Fig.4.2.1

Fig.4.2.1 shows the construction of the decoder CWSR. The 75 bit ($u_0, u_1 .. u_{74}$) register is formed by wiring 19 MC 10141 IC's in cascade. Only the first 3 bits are used on the 19th IC. This register shifts right continuously at the code clock rate (CWSR clock(0,49)) and is programmed by hardwiring the S1 input to "0", and S2 to "1". The code word output from the comparator is connected to the data input on the 1st IC,

at pin 5. The buffered parallel outputs Q3,Q2 etc. are hard wired to the corresponding pins of the MLD.

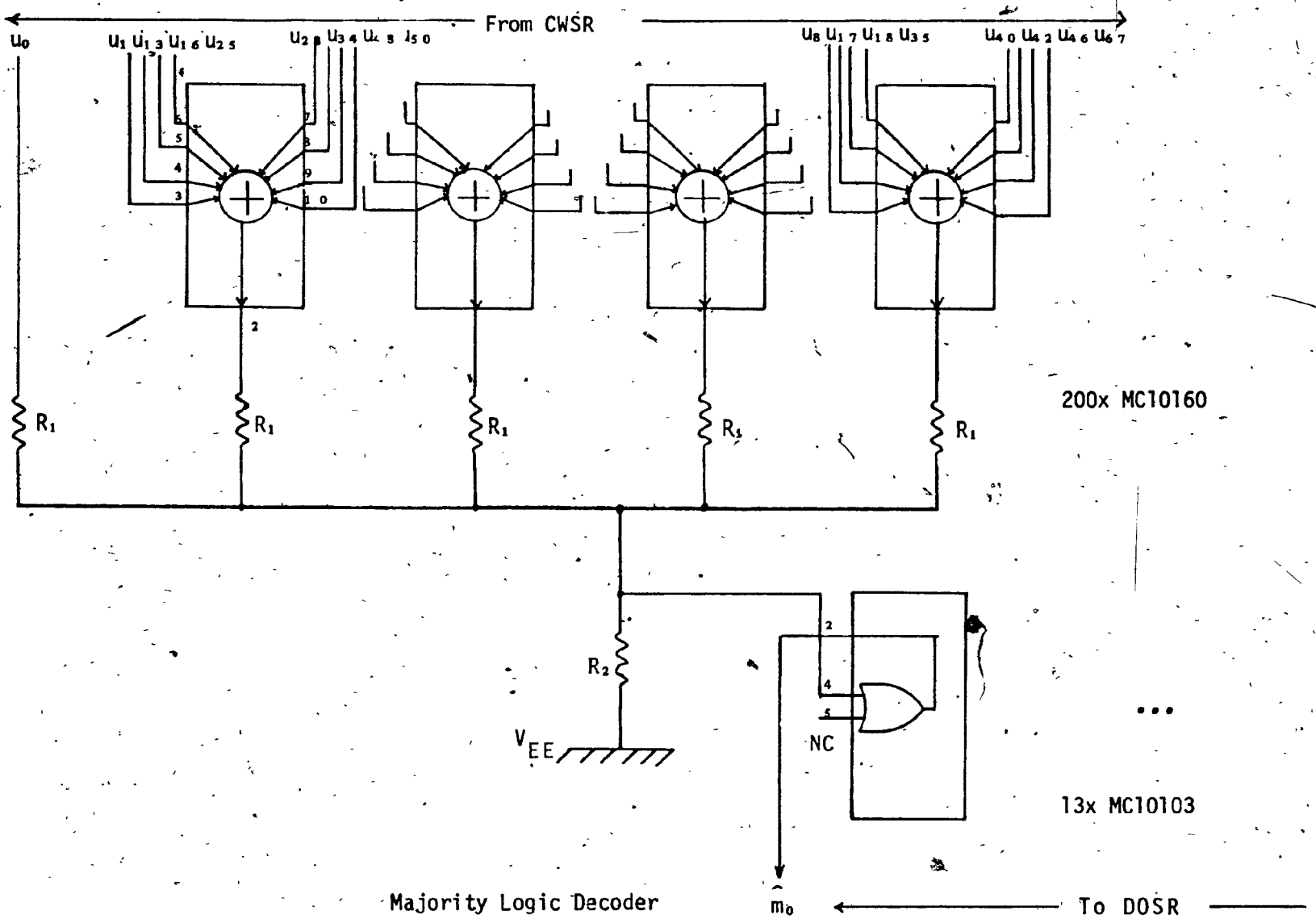



CWSR Driver
Fig.4.2.2

Fig.4.2.2 shows the clock driver for the CWSR. The code clock from the DPU feeds a balanced tree structure formed by 2xMC10104 IC's.

## 4.3 Majority Logic Decoder (MLD)

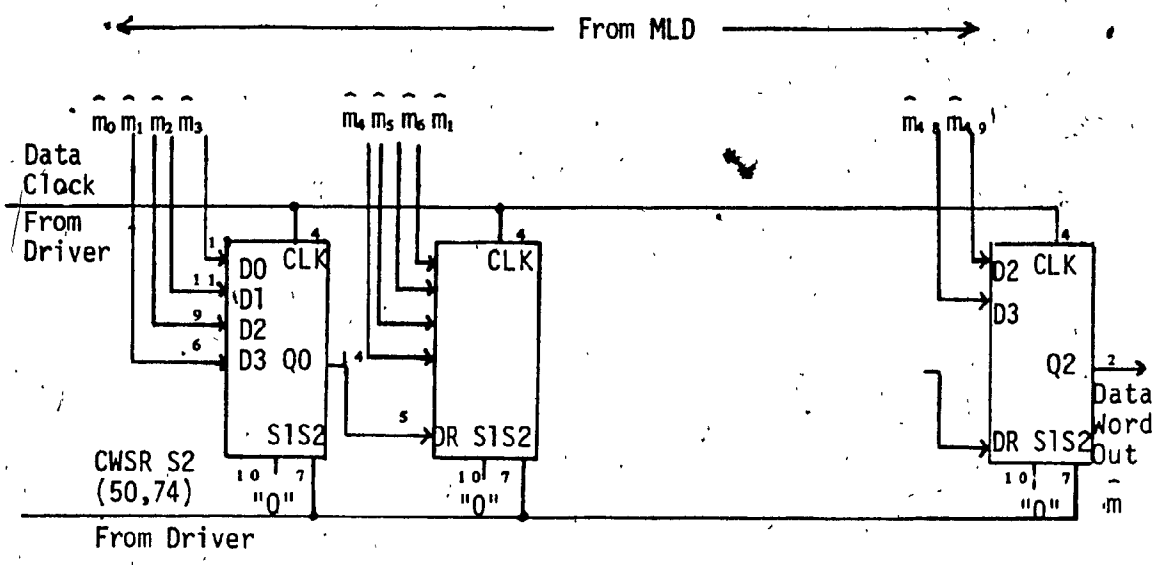
The MLD circuit is shown in Fig.4.3. It is a straightforward hardware implementation of the decoder equations given in Appendix II, using the majority logic gate as designed in Appendix IV. The R1 and R2 values may have to be experimented with, depending on the particular board layout. (Ratio must be .24, recommend using combination of a fixed resistor and quality variable pot for R2 to set threshold exactly). There are 4xMC10160 EX-OR's and 1xMC10103 per decoded bit, or a total of 200 MC10160 and 13xMC10103 IC's. Any ECL gate could be used for the majority logic gate (MC10107 used in Appendix IV), however, the MC10103 has 4 gates per package, and the use of the "OR" function allows the 2nd gate input to be left unconnected. The outputs of the majority gates parallel load into the DOSR.




Majority Logic Decoder

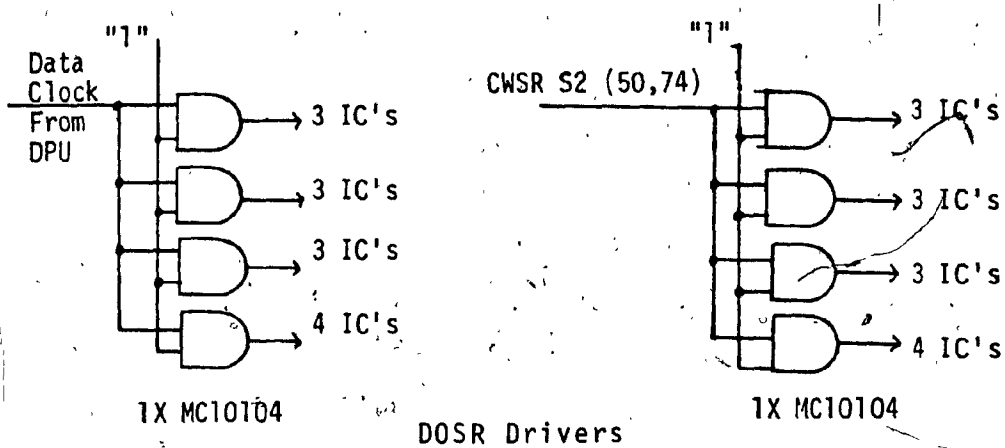
Fig.4.3

## 4.4 Data Output Shift Register (DOSR)




Data Output Shift Register
Fig.4.4.1

Fig.4.4.1 shows the data output shift register (DOSR). It consists of 13 MC10141 IC's wired in cascade. The register normally shifts right at the data rate, except upon receipt of the S2 (50,74) command, whereupon it parallel loads the 50 estimated message bits $\hat{m}_0, \hat{m}_1$, etc. from the MLD. Only the first 2 bit positions are used on the last IC.




DOSR Drivers

Fig.4.4.2

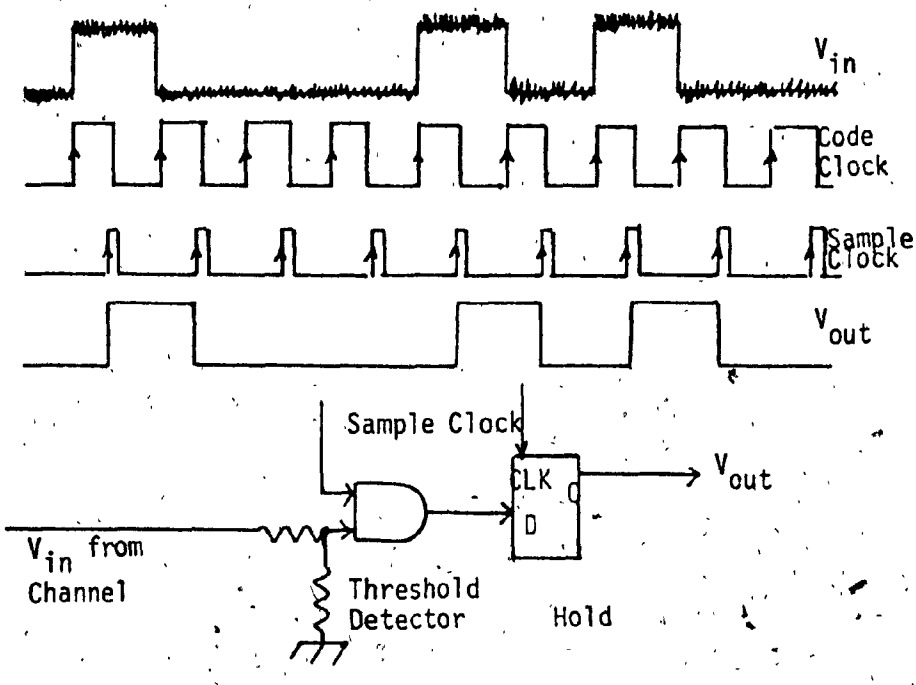
Fig.4.4.2 shows the clock & S2 drivers. Each driver is formed by a single MC10104 IC, each gate driving 3 IC's, with the fourth driving 4 IC's.

## 4.5 Decoder Processing Unit (DPU)

The DPU provides an interface between the various waveforms received from the encoder and the channel and the decoder modules CWSR, MLD & DOSR. It allows various delays in the channel to be matched by incorporating variable delay lines on all received waveforms (tapped coaxial cable).

Normally the Codec would be used in conjunction with a modem to pass data over the channel. In this simulation, the codec is used alone and the process is carried out at base-band. Thus a module must be provided to "clean" up the received waveform from the channel. The comparator serves this purpose.

Fig.4.5 shows the schematic of a typical comparator. The input waveform corrupted by noise is passed to a threshold gate (similar to a majority logic gate, n=2,App.IV) where it is sampled at the code clock rate, and stored for the complete clock period in a flip flop.




Comparator Circuit

Fig.4.5

As the added noise in the channel increases, correspondingly more random decision errors will be made by the threshold detector etc.

# CHAPTER 5

# CONCLUSION

## 5.1 Report Summary

Chapter 1 and 2 have presented the background requirements for construction of high speed forward error correcting codecs. An actual high speed encoder was constructed based on these requirements and its performance and design described in chapter 3. The complete design for a matching decoder was given in chapter 4.

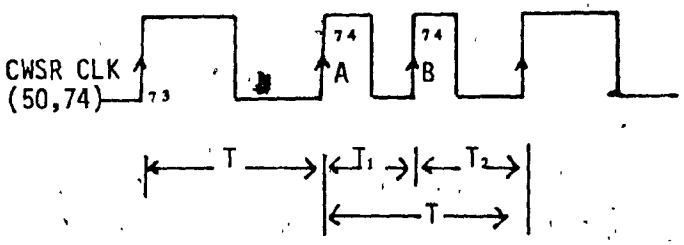
The encoder was set up to operate at a data speed of 1.5 Mbit/sec. This was due only to measurement limitations. (225 MHZ monitoring bandwidth). The designs as presented are limited by:

a) Time required to calculate and load the encoder parity bits.

b) Time required to decode and load the decoder estimated bits.

Table 5.1.1 lists the limitations of the encoder design as implemented for ECL10K logic, and also if MECL III logic were used. All figures quoted are the min/max values taken from tables 2.2.1 and 2.2.2.

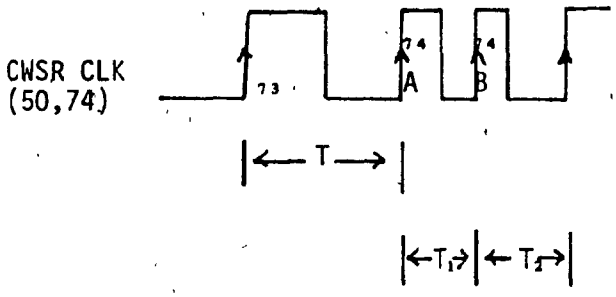
Table 5.1.2 lists the limitations of the decoder design.




| | MECL 10K Min | Max | | MECL III Min | Max | |
|---|---|---|---|---|---|---|
| $T_1$ (ns) | 1.8 | 3.8 | 11 Load MC10141 | 1 | 3 | 11 Load MC1694 |
| | 2 | 7.5 | Pdly MC10160 | 2 | 7.5 | Pdly MC10160 |
| | 1 | 1 | Pdly Wire | 1 | 1 | Pdly Wire |
| | 4.8 | 12.3 | | 4 | 11.5 | |
| $T_2$ (ns) | 1.8 | 3.8 | 11 Load MC10141 | 1 | 3 | 11 Load MC1694 |
| | 1 | 1 | Pdly Wire | 1 | 1 | Pdly Wire |
| | 2.8 | 4.8 | | 2 | 4 | |
| $T=T_1+T_2$ | 7.6 | 17.1 | | 6 | 15.5 | |
| f Code M bit/sec | 131.6 | 58.5 | | 166.7 | 64.5 | |
| f Data (fc/1.5) M bit/sec | 87.7 | 39 | | 111 | 43 | |

Encoding Time Limitations

Table 5.111




| | MECL 10K | | | MECL III | | |
|---|---|---|---|---|---|---|
| | Min | Max | | Min | Max | |
| $T_1$ (ns) | 1.8 | 3.8 | Pdly MC10141 | 1 | 3 | Pdly MC1694 |
| $T_2$ (ns) | 2 | 7.5 | Pdly MC10160 | 2 | 7.5 | Pdly MC10160 |
| | 1 | 2.9 | Pdly MC10103 | .6 | 1.7 | Pdly MC1664 |
| | 2 | 2 | Pdly Wire | 2 | 2 | Pdly Wire |
| | 5 | 12.4 | | 4.6 | 11.2 | |
| $T=T_1+T_2$ (ns) | 6.8 | 16.2 | | 5.6 | 14.2 | |
| f Code M bit/sec | 147.1 | 61.7 | | 178.6 | 70.4 | |
| f Data (fc/1.5) M bit/sec | 98 | 41.2 | | 119.1 | 47 | |

Decoding Time Limitations

Table 5.1.2

Typical costs for the codec package might be as follows (equipment only):

| MECL 10K | | MECL III | |
|---|---|---|---|
| 64 x MC10141 | = $512 | 64 x MC1694 | = $3760 |
| 225 x MC10160 | = $900 | 225 x MC10160 | = $900 |
| 13 x MC10103 | = $13 | 13 x MC1664 | = $130 |
| 40 misc. gates ($2.50 each avg.) | = $100 | 40 misc. gates ($25 each avg.) | = $1000 |
| 3 x Logic Panels | = $3000 | 3 x Logic Panels | = $3000 |
| 2 x Metal Chassis | = $100 | 2 x Metal Chassis | = $100 |
| Misc. Hardware | = $200 | Misc. Hardware | = $200 |
| Power Supplies $2 \times V_{EE}$/ $2 \times V_{TT}$ | = $1500 | Power Supplies | = $1500 |
| | $6325 | | $10590 |

Due to the factor of 10 cost of MECL III over MECL 10K, the MECL III codec is approximately twice as expensive. These costs are prototype costs, and production costs would be significantly lower. Thus a codec with coding gain as given by Fig.1.4.3 and data capability well over 40 Mbit/sec can

be built for about $6,000 and one close to 100 Mbit/sec for about $10,000. The codec may provide a great cost savings by yielding the required system gain vs another alternative (larger tx power on satellite etc.).

## 5.2 Recommendations

It would be very straightforward to build a decoder prototype as outlined in this report. All the materials have been purchased, except the gates required for the MLD (13 x MC10103), and miscellaneous gates, drivers etc. The channel equipment could also be set up very easily. Since there are many modules in common between the encoder and decoder, it might be worthwhile considering building the decoder into the same chassis as the encoder. In a real-life situation, data will usually be bidirectional, that is, a decoder will be required adjacent to the encoder. Thus by combining the two, the operational requirement is satisfied, along with considerable electrical and mechanical savings. Error-correction is simply checked by inserting the required "channel" into the loopback, between encoder and decoder.

# REFERENCES

## Coding


[1] V.K. Bhargava, "Some Codes of Rahman and Blake for Computer Applications," IEEE Trans. No.8, Aug.1978. pp.765-767

[2] M.Y. Hsiao, "Incomplete Block Design Codes for Ultra High Speed Computer Applications," Discrete Math, Vol.3, 1972. pp.89-108.

[3] R.L. Townsend, E.J. Weldon, Jr., "Self-Orthogonal Quasi-Cyclic Codes," IEEE Trans. Infom. Theory, Vol.IT-13, No.2, Apr.1967, pp.183-195.

[4] M. Avni, V.K. Bhargava, "On Improving The Bit Error Rate of Various Modulation Systems by Block Coding," Nat. Telecom. Conf. 1980, Conf. Rec. 52.3.1, 52.3.5.

[5] J.S. Synder, T.Muratani, "Forward Error Correction for Satellite TDMA in the Intelsat V Era," Am. Inst. Aero and Astro Conf. 1980, Conf. Rec. pp. 674-683.

[6] J.E. Medlin, N.J. Bryg, "A 105 Mbit/s Error Correcting Codec," IEEE Trans. Comm. Vol. C-26, No.10, Oct. 1978.

[7] J. Clark, A. Antoniou, V.K. Bhargava, "A High Speed Forward Error Correction Codec," IEEC & E,1979, Conf.Rec. pp.170-171.

[8] A. Brind'Amour, K. Feher, "Design and Evaluation of a Convolutional Codec in Additive White Gaussian Noise, Sinusoidal Interference and Intersymbol Interference Environment," IEEE Trans. Comm. Vol.C-28, No.3, Mar,1980.

[9] A.M. Walker, "High Data Rate PSK Modems for Satellite Communications," Telecommunications, July 1976. pp.27-30.

[10] H.S.V. Reeves, "140 Mbit/s Digital Line System for Coaxial Cable," Electrical Communication, Vol.53, No.2, 1978. pp. 173-177.

[11] T. Røste, N. Holte, S.J. Knapskog, "A 140 Mbit/s Digital Transmission System for Coaxial Cable Using Partial Response Class 1 Line Code with Quantized Feedback," IEEE Trans. Comm. Vol. C-28, No.8, Aug.1980.

[12] J.E. Dunn, "A 50 Mbit/s Multiplexed Coding System for Shuttle Communications," IEEE Trans. Comm. Vol.C-26, No.11, Nov.1978.

[13] S. Lin, "An Introduction to Error Correcting Codes," Prentice-Hall Inc., New Jersey, 1970. p.37.

# REFERENCES

Coding

[14] R.C. Dixon, "Spread Spectrum Systems," Wiley, New York, 1975. p.81.

[15] D. Chase, "Capacity Limits for Binary Codes in the Presence of Interference," IEEE Trans. Comm. Vol.C-27, No.2, Feb.1979.

[16] V.V. Zyablov, "Piecewise Cyclic Codes and their Majority Decoding Schemes," Problemy Peredachi Informatsii, Vol.4, No.2, 1968. pp.31-37.

[17] A.S. Acampora, R.P. Gilmore, "Analog Viterbi Decoding for High Speed Digital Satellite Channels," IEEE Trans. on Comm. Vol. Com-26, No.10, Oct. 1978.

[18] A.M. Michelson, "The Calculation of Post Decoding Bit Error Probabilities for Binary Block Codes," National Telecommunications Conference, 1976.

[19] M. Avni, "Coding/Modulation Tradeoffs in Digital Communications," Concordia University Masters Thesis, 1978.

[20] V.K. Bhargava, D. Haccoun, R. Matyas and P. Nuspl, "Digital Communications by Satellite," Wiley, New York, 1981. pp.46-48.

[21] J.D. Oetting, "A Comparison of Modulation Techniques for Digital Radio," IEEE Trans. on Comm. Vol. Com-27, No.12, Dec. 1979.

# REFERENCES

Electronics


[1] "MECL High Speed Integrated Circuits," Handbook, Motorola Inc., 1978.

[2] MECL Application Notes a) AN-701,"Understanding MECL10K DC/AC Specs".
b) AN-270,"nsec Pulse Handling Techniques".

[3] N.L. Bragg, "ECL for the TTL Designer," Electronic Engineering, Oct.1980. pp.41-52.

[4] M. Grossman, "Fast Logic No Problem for Wire Wrapped Boards," Electronic Design 25, Dec.6, 1978. pp.34-36.

[5] L. Doucet, "Who Said You Can't Wire Wrap ECL?", Electronic Products Magazine, Sept.16, 1974. pp. 13-15.

[6] L.H. Anderson, "Counters and Weights," Ham Radio Magazine, Aug.1979. pp. 66-69.

[7] L.H. Anderson, "Flip Flop Internal Structure," Ham Radio Magazine, April 1979. pp.86-88.

[8] "General Aspects of Interfaces," CCIR Orange Book, Vol.III-2, Rec.G.703, Geneva 1976.

[9] "Theory and Winding Data RF Toroids," Revision A, Amidon Associates, North Hollywood, California, 1974.

[10] F.A. Blum, "Gigabit Logic: Prospects for GaAs LSI," Microwave System News, February 1979.

[11] S. Muroga, "Threshold Logic and Its Applications," Wiley Interscience, New York, 1971.

[12] A. Schwarz, "Wallace Tree as Majority Vote Logic Circuit," Electronic Engineering, September 1977. p.33.

# APPENDIX I

## Parity Check Matrix H and Encoder Equations

The parity check matrix H is of the form $[P^T, I_{25}]$ as given in Section 1.3. The $P^T$ matrix is shown in entirety in Fig.AI-1. Note that it is simply formed by the element matrices a,b,c,d,e,v,w,x,y,z arranged in quasi-cyclic fashion as per Section 1.3. The G matrix is derived directly from inspection of H. Each row of G contains 5 1's (1x1 from identity matrix, and 4 1's from P matrix). The code minimum distance is 5, and hence error correcting ability = (5-1)/2 = 2.

Calculation of the jth parity bit is formed by taking the inner product of $\bar{m}$ with the jth row of $P^T$. Listed below are the complete set of encoder equations for the 25 parity bits. Also listed are the matching IC locations and logic pinout as discussed in Section 3.1.

+ = EX-OR

$U_{50} = M_0 + M_1 + M_{13} + M_{16} + M_{25} + M_{28} + M_{34} + M_{48}$

C7/D3 A10/Q3 A10/Q2 A7/Q2 A6/Q3 B9/Q2 B8/Q3 B7/Q1 C8/Q3

$U_{51} = M_1 + M_2 + M_{14} + M_{17} + M_{26} + M_{29} + M_{30} + M_{49}$

C7/D2 A10/Q2 A10/Q1 A7/Q1 A6/Q2 B9/Q1 B8/Q2 B8/Q1 C8/Q2

$U_{52} = M_2 + M_3 + M_{10} + M_{18} + M_{25} + M_{27} + M_{31} + M_{45}$

C7/D1 A10/Q1 A10/Q0 A8/Q1 A6/Q1 B9/Q2 B9/Q0 B8/Q0 C9/Q2

$U_{53} = M_3 + M_4 + M_{11} + M_{19} + M_{26} + M_{28} + M_{32} + M_{46}$

C7/D0 A10/Q0 A9/Q3 A8/Q0 A6/Q0 B9/Q1 B8/Q3 B7/Q3 C9/Q1

$U_{54} = M_0 + M_4 + M_{12} + M_{15} + M_{27} + M_{29} + M_{33} + M_{47}$

C6/D3 A10/Q3 A9/Q3 A7/Q3 A7/Q0 B9/Q0 B8/Q2 B7/Q2 C9/Q0

$U_{55} = M_5 + M_6 + M_{18} + M_{21} + M_{28} + M_{30} + M_{33} + M_{39}$

C6/D2 A9/Q2 A9/Q1 A6/Q1 B10/Q2 B8/Q3 B8/Q1 B7/Q2 B6/Q0

$U_{56} = M_6 + M_7 + M_{19} + M_{22} + M_{29} + M_{31} + M_{34} + M_{35}$

C6/D1 A9/Q1 A9/Q0 A6/Q0 B10/Q1 B8/Q2 B8/Q0 B7/Q1 B7/Q0

$U_{57} = M_7 + M_8 + M_{15} + M_{23} + M_{25} + M_{30} + M_{32} + M_{36}$

C6/D0 A9/Q0 A8/Q3 A7/Q0 B10/Q0 B9/Q2 B8/Q1 B7/Q3 B6/Q3

$U_{58} = M_8 + M_9 + M_{16} + M_{24} + M_{26} + M_{31} + M_{33} + M_{37}$

D10/D3 A8/Q3 A8/Q2 A6/Q3 B9/Q3 B9/Q1 B8/Q0 B7/Q2 B6/Q2

+ = EX-OR

$U_{59}$ = $M_5$ + $M_9$ + $M_{17}$ + $M_{20}$ + $M_{27}$ + $M_{32}$ + $M_{34}$ + $M_{38}$
D10/D2 A9/Q2 A8/Q2 A6/Q2 B10/Q3 B9/Q0 B7/Q3 B7/Q1 B6/Q1

$U_{60}$ = $M_1$ + $M_{10}$ + $M_{11}$ + $M_{23}$ + $M_{33}$ + $M_{35}$ + $M_{38}$ + $M_{44}$
D10/D1 A10/Q2 A8/Q1 A8/Q0 B10/Q0 B7/Q2 B7/Q0 B6/Q1 C9/Q3

$U_{61}$ = $M_2$ + $M_{11}$ + $M_{12}$ + $M_{24}$ + $M_{34}$ + $M_{36}$ + $M_{39}$ + $M_{40}$
D10/D0 A10/Q1 A8/Q0 A7/Q3 B9/Q3 B7/Q1 B6/Q3 B6/Q0 C10/Q3

$U_{62}$ = $M_3$ + $M_{12}$ + $M_{13}$ + $M_{20}$ + $M_{30}$ + $M_{35}$ + $M_{37}$ + $M_{41}$
D9/D3 A10/Q0 A7/Q3 A7/Q2 B10/Q3 B8/Q1 B7/Q0 B6/Q2 C10/Q2

$U_{63}$ = $M_4$ + $M_{13}$ + $M_{14}$ + $M_{21}$ + $M_{31}$ + $M_{36}$ + $M_{38}$ + $M_{42}$
D9/D2 A9/Q3 A7/Q2 A7/Q1 B10/Q2 B8/Q0 B6/Q3 B6/Q1 C10/Q1

$U_{64}$ = $M_0$ + $M_{10}$ + $M_{14}$ + $M_{22}$ + $M_{32}$ + $M_{37}$ + $M_{39}$ + $M_{43}$
D9/D1 A10/Q3 A8/Q1 A7/Q1 B10/Q1 B7/Q3 B6/Q2 B6/Q0 C10/Q0

$U_{65}$ = $M_3$ + $M_6$ + $M_{15}$ + $M_{16}$ + $M_{38}$ + $M_{40}$ + $M_{43}$ + $M_{49}$
D9/D0 A10/Q0 A9/Q1 A7/Q0 A6/Q3 B6/Q1 C10/Q3 C10/Q0 C8/Q3

$U_{66}$ = $M_4$ + $M_7$ + $M_{16}$ + $M_{17}$ + $M_{39}$ + $M_{41}$ + $M_{44}$ + $M_{45}$
D8/D3 A9/Q3 A9/Q0 A6/Q3 A6/Q2 B6/Q0 C10/Q2 C9/Q3 C9/Q2

$U_{67}$ = $M_0$ + $M_8$ + $M_{17}$ + $M_{18}$ + $M_{35}$ + $M_{40}$ + $M_{42}$ + $M_{46}$
D8/D2 A10/Q3 A8/Q3 A6/Q2 A6/Q1 B7/Q0 C10/Q3 C10/Q1 C9/Q1

+ = EX-OR

$U_{68}$ = $M_1$ + $M_9$ + $M_{18}$ + $M_{19}$ + $M_{36}$ + $M_{41}$ + $M_{43}$ + $M_{47}$

D8/D1 A10/Q2 A8/Q2 A6/Q1 A6/Q0 B6/Q3 C10/Q2 C10/Q0 C9/Q0

$U_{69}$ = $M_2$ + $M_5$ + $M_{15}$ + $M_{19}$ + $M_{37}$ + $M_{42}$ + $M_{44}$ + $M_{48}$

D8/D0 A10/Q1 A9/Q2 A7/Q0 A6/Q0 B6/Q2 C10/Q2 C9/Q3 C8/Q3

$U_{70}$ = $M_8$ + $M_{11}$ + $M_{20}$ + $M_{21}$ + $M_{29}$ + $M_{43}$ + $M_{45}$ + $M_{48}$

D7/D3 A8/Q3 A8/Q0 B10/Q3 B10/Q2 B8/Q2 C10/Q0 C9/Q2 C8/Q3

$U_{71}$ = $M_9$ + $M_{12}$ + $M_{21}$ + $M_{22}$ + $M_{25}$ + $M_{44}$ + $M_{46}$ + $M_{49}$

D7/D2 A8/Q2 A7/Q3 B10/Q2 B10/Q1 B9/Q2 C9/Q3 C9/Q2 C8/Q0

$U_{72}$ = $M_5$ + $M_{13}$ + $M_{22}$ + $M_{23}$ + $M_{26}$ + $M_{40}$ + $M_{45}$ + $M_{47}$

D7/D1 A9/Q2 A7/Q2 B10/Q1 B10/Q0 B9/Q1 C10/Q3 C9/Q2 C9/Q0

$U_{73}$ = $M_6$ + $M_{14}$ + $M_{23}$ + $M_{24}$ + $M_{27}$ + $M_{41}$ + $M_{46}$ + $M_{48}$

D7/D0 A9/Q1 A7/Q1 B10/Q0 B9/Q3 B9/Q0 C10/Q2 C9/Q1 C8/Q3

$U_{74}$ = $M_7$ + $M_{10}$ + $M_{20}$ + $M_{24}$ + $M_{28}$ + $M_{42}$ + $M_{47}$ + $M_{49}$

D6/D3 A9/Q0 A8/Q1 B10/Q3 B9/Q3 B8/Q3 C10/Q1 C9/Q0 C8/Q0

$P^T$ MATRIX

fig AI-1

# APPENDIX II

## One Step Majority Logic Decoding and Decoder Equations

Let $\bar{u}$ represent the received code vector as in Section 1.3. It is of the form:

$$\bar{u} = [u_0, u_1, u_2 \ldots u_{74}]$$

$$\bar{u} = [u_0, u_1 \ldots u_{49}, c_0, c_1 \ldots c_{24}]$$

The first 50 bits of $\bar{u}$ represent the received values of $\bar{m}$, the original message (code in systematic form). The last 25 bits represent the received values of the parity bits. Due to the imperfect nature of the channel, errors will occur, and $\bar{u}_j$ $j=0,49$ will not necessarily be equal to the original message sent, $\bar{m}_j$ $j=0,49$. If $\bar{u}$ is a valid code word, then $\bar{u}$ must be contained in the row space of G, or alternatively

$$\bar{u} \cdot H^T = 0$$

Examining $H^T$, there are 9 1's in every column, and 4 1's in every row. This means for every bit $u_j$, $j=0,49$, 4 simultaneous equations may be written from the relation $\bar{u} H^T = 0$, each equation involving 9 terms. For bit $u_0$ the following 4 relations may be written:

$$u_0+u_1+u_{13}+u_{16}+u_{25}+u_{28}+u_{34}+u_{48}+c_0 = 0$$

$$u_0+u_4+u_{12}+u_{15}+u_{27}+u_{29}+u_{33}+u_{47}+c_4 = 0$$

$$u_0+u_{10}+u_{14}+u_{22}+u_{32}+u_{37}+u_{39}+u_{43}+c_{14} = 0$$

$$u_0+u_8+u_{17}+u_{18}+u_{35}+u_{40}+u_{42}+u_{46}+c_{17} = 0$$

For one step majority logic decoding, each one of the previous independent equations is used as an estimate of the original bit sent, together with the actual received bit itself. A majority vote is taken between these 5 relations, and the estimated or decoded bit set appropriately.

$$m_{0\ \text{decoded}} = \text{maj} \begin{bmatrix} u_0 \\ u_1 + u_{13} + u_{16} + u_{25} + u_{28} + u_{34} + u_{48} + c_0 \\ u_4 + u_{12} + u_{15} + u_{27} + u_{29} + u_{33} + u_{47} + c_4 \\ u_{10} + u_{14} + u_{22} + u_{32} + u_{37} + u_{39} + u_{43} + c_{14} \\ u_8 + u_{17} + u_{18} + u_{35} + u_{40} + u_{42} + u_{46} + c_{17} \end{bmatrix}$$

Example:

$$m_{0\ \text{decoded}} = \text{maj} \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} = 1$$

The complete set of decoder equations for the 50 information bits are as follows:

## Parity Check Equations

$$\hat{m}_0 = \text{maj}\begin{bmatrix} m_0 \\ m_1 + m_{13} + m_{16} + m_{25} + m_{28} + m_{34} + m_{48} + C_0 \\ m_4 + m_{12} + m_{15} + m_{27} + m_{29} + m_{33} + m_{47} + C_4 \\ m_{10} + m_{14} + m_{22} + m_{32} + m_{37} + m_{39} + m_{43} + C_{14} \\ m_8 + m_{17} + m_{18} + m_{35} + m_{40} + m_{42} + m_{46} + C_{17} \end{bmatrix}$$

$$\hat{m}_1 = \text{maj}\begin{bmatrix} m_1 \\ m_0 + m_{13} + m_{16} + m_{25} + m_{28} + m_{34} + m_{48} + C_0 \\ m_2 + m_{14} + m_{17} + m_{26} + m_{29} + m_{30} + m_{49} + C_1 \\ m_{10} + m_{11} + m_{23} + m_{33} + m_{35} + m_{38} + m_{44} + C_{10} \\ m_9 + m_{18} + m_{19} + m_{36} + m_{41} + m_{43} + m_{47} + C_{18} \end{bmatrix}$$

$$\hat{m}_2 = \text{maj}\begin{bmatrix} m_2 \\ m_1 + m_{14} + m_{17} + m_{26} + m_{29} + m_{30} + m_{49} + C_1 \\ m_3 + m_{10} + m_{18} + m_{25} + m_{27} + m_{31} + m_{45} + C_2 \\ m_{11} + m_{12} + m_{24} + m_{34} + m_{36} + m_{39} + m_{40} + C_{11} \\ m_5 + m_{15} + m_{19} + m_{37} + m_{42} + m_{44} + m_{48} + C_{19} \end{bmatrix}$$

$$\hat{m}_3 = \text{maj}\begin{bmatrix} m_3 \\ m_2 + m_{10} + m_{18} + m_{25} + m_{27} + m_{31} + m_{45} + C_2 \\ m_4 + m_{11} + m_{19} + m_{26} + m_{28} + m_{32} + m_{46} + C_3 \\ m_{12} + m_{13} + m_{20} + m_{30} + m_{35} + m_{37} + m_{41} + C_{12} \\ m_6 + m_{15} + m_{16} + m_{38} + m_{40} + m_{43} + m_{49} + C_{15} \end{bmatrix}$$

$$\hat{m}_4 = \text{maj}\begin{bmatrix} m_4 \\ m_3 + m_{11} + m_{19} + m_{26} + m_{28} + m_{32} + m_{46} + C_3 \\ m_0 + m_{12} + m_{15} + m_{27} + m_{29} + m_{33} + m_{47} + C_4 \\ m_{13} + m_{14} + m_{21} + m_{31} + m_{36} + m_{38} + m_{42} + C_{13} \\ m_7 + m_{16} + m_{17} + m_{39} + m_{41} + m_{44} + m_{45} + C_{16} \end{bmatrix}$$

$$\hat{m}_5 = \text{maj}\begin{bmatrix} m_5 \\ m_6 + m_{18} + m_{21} + m_{28} + m_{30} + m_{33} + m_{39} + C_5 \\ m_9 + m_{17} + m_{20} + m_{27} + m_{32} + m_{34} + m_{38} + C_9 \\ m_2 + m_{15} + m_{19} + m_{37} + m_{42} + m_{44} + m_{48} + C_{19} \\ m_1 + m_{22} + m_{23} + m_{26} + m_{40} + m_{45} + m_{47} + C_{22} \end{bmatrix}$$

$$\hat{m}_6 = maj\begin{bmatrix} m_6 \\ m_5+m_{18}+m_{21}+m_{28}+m_{30}+m_{33}+m_{39}+c_5 \\ m_7+m_{19}+m_{22}+m_{29}+m_{31}+m_{34}+m_{35}+c_6 \\ m_3+m_{15}+m_{16}+m_{38}+m_{40}+m_{43}+m_{49}+c_{15} \\ m_{14}+m_{23}+m_{24}+m_{27}+m_{41}+m_{46}+m_{48}+c_{23} \end{bmatrix}$$

$$\hat{m}_7 = maj\begin{bmatrix} m_7 \\ m_6+m_{19}+m_{22}+m_{29}+m_{31}+m_{34}+m_{35}+c_6 \\ m_8+m_{15}+m_{23}+m_{25}+m_{30}+m_{32}+m_{36}+c_7 \\ m_4+m_{16}+m_{17}+m_{39}+m_{41}+m_{44}+m_{45}+c_{16} \\ m_{10}+m_{20}+m_{24}+m_{28}+m_{42}+m_{47}+m_{49}+c_{24} \end{bmatrix}$$

$$\hat{m}_8 = maj\begin{bmatrix} m_8 \\ m_7+m_{15}+m_{23}+m_{25}+m_{30}+m_{32}+m_{36}+c_7 \\ m_9+m_{16}+m_{24}+m_{26}+m_{31}+m_{33}+m_{37}+c_8 \\ m_0+m_{17}+m_{18}+m_{35}+m_{40}+m_{42}+m_{46}+c_{17} \\ m_{11}+m_{20}+m_{21}+m_{29}+m_{43}+m_{45}+m_{48}+c_{20} \end{bmatrix}$$

$$\hat{m}_9 = maj\begin{bmatrix} m_9 \\ m_8+m_{16}+m_{24}+m_{26}+m_{31}+m_{33}+m_{37}+c_8 \\ m_5+m_{17}+m_{20}+m_{27}+m_{32}+m_{34}+m_{38}+c_9 \\ m_1+m_{18}+m_{19}+m_{36}+m_{41}+m_{43}+m_{47}+c_{18} \\ m_{12}+m_{21}+m_{22}+m_{25}+m_{44}+m_{46}+m_{49}+c_{21} \end{bmatrix}$$

$$\hat{m}_{10} = maj\begin{bmatrix} m_{10} \\ m_2+m_3+m_{18}+m_{25}+m_{27}+m_{31}+m_{45}+c_2 \\ m_1+m_{11}+m_{23}+m_{33}+m_{35}+m_{38}+m_{44}+c_{10} \\ m_0+m_{14}+m_{22}+m_{32}+m_{37}+m_{39}+m_{43}+c_{14} \\ m_7+m_{20}+m_{24}+m_{28}+m_{42}+m_{47}+m_{49}+c_{24} \end{bmatrix}$$

$$\hat{m}_{11} = maj\begin{bmatrix} m_{11} \\ m_3+m_4+m_{19}+m_{26}+m_{28}+m_{32}+m_{46}+c_3 \\ m_1+m_{10}+m_{23}+m_{33}+m_{35}+m_{38}+m_{44}+c_{10} \\ m_2+m_{12}+m_{24}+m_{34}+m_{36}+m_{39}+m_{40}+c_{11} \\ m_8+m_{20}+m_{21}+m_{29}+m_{43}+m_{45}+m_{48}+c_{20} \end{bmatrix}$$

$$\hat{m}_{12} = maj\begin{bmatrix} m_{12} \\ m_0 + m_4 + m_{15} + m_{27} + m_{29} + m_{33} + m_{47} + c_4 \\ m_2 + m_{11} + m_{24} + m_{34} + m_{36} + m_{39} + m_{40} + c_{11} \\ m_3 + m_{13} + m_{20} + m_{30} + m_{35} + m_{37} + m_{41} + c_{12} \\ m_9 + m_{21} + m_{22} + m_{25} + m_{44} + m_{46} + m_{49} + c_{21} \end{bmatrix}$$

$$\hat{m}_{13} = maj\begin{bmatrix} m_{13} \\ m_0 + m_1 + m_{16} + m_{25} + m_{28} + m_{34} + m_{48} + c_0 \\ m_3 + m_{12} + m_{20} + m_{30} + m_{35} + m_{37} + m_{41} + c_{12} \\ m_4 + m_{14} + m_{21} + m_{31} + m_{36} + m_{38} + m_{42} + c_{13} \\ m_5 + m_{22} + m_{23} + m_{26} + m_{40} + m_{45} + m_{47} + c_{22} \end{bmatrix}$$

$$\hat{m}_{14} = maj\begin{bmatrix} m_{14} \\ m_1 + m_2 + m_{17} + m_{26} + m_{29} + m_{30} + m_{49} + c_1 \\ m_4 + m_{13} + m_{21} + m_{31} + m_{36} + m_{38} + m_{42} + c_{13} \\ m_0 + m_{10} + m_{22} + m_{32} + m_{37} + m_{39} + m_{43} + c_{14} \\ m_6 + m_{23} + m_{24} + m_{27} + m_{41} + m_{46} + m_{48} + c_{23} \end{bmatrix}$$

$$\hat{m}_{15} = maj\begin{bmatrix} m_{15} \\ m_0 + m_4 + m_{12} + m_{27} + m_{29} + m_{33} + m_{47} + c_4 \\ m_7 + m_8 + m_{23} + m_{25} + m_{30} + m_{32} + m_{36} + c_7 \\ m_3 + m_6 + m_{16} + m_{38} + m_{40} + m_{43} + m_{49} + c_{15} \\ m_2 + m_5 + m_{19} + m_{37} + m_{42} + m_{44} + m_{48} + c_{19} \end{bmatrix}$$

$$\hat{m}_{16} = maj\begin{bmatrix} m_{16} \\ m_0 + m_1 + m_{13} + m_{25} + m_{28} + m_{34} + m_{48} + c_0 \\ m_8 + m_9 + m_{24} + m_{26} + m_{31} + m_{33} + m_{37} + c_8 \\ m_3 + m_6 + m_{15} + m_{38} + m_{40} + m_{43} + m_{49} + c_{15} \\ m_4 + m_7 + m_{17} + m_{39} + m_{41} + m_{44} + m_{45} + c_{16} \end{bmatrix}$$

$$\hat{m}_{17} = maj\begin{bmatrix} m_{17} \\ m_1 + m_2 + m_{14} + m_{26} + m_{29} + m_{30} + m_{49} + c_1 \\ m_5 + m_9 + m_{20} + m_{27} + m_{32} + m_{34} + m_{38} + c_9 \\ m_4 + m_7 + m_{16} + m_{39} + m_{41} + m_{44} + m_{45} + c_{16} \\ m_0 + m_8 + m_{18} + m_{35} + m_{40} + m_{42} + m_{46} + c_{17} \end{bmatrix}$$

$$\hat{m}_{18} = maj\begin{bmatrix} m_{18} \\ m_2+m_3+m_{10}+m_{25}+m_{27}+m_{31}+m_{45}+c_2 \\ m_5+m_6+m_{21}+m_{28}+m_{30}+m_{33}+m_{39}+c_5 \\ m_0+m_8+m_{17}+m_{35}+m_{40}+m_{42}+m_{46}+c_{17} \\ m_1+m_9+m_{19}+m_{36}+m_{41}+m_{43}+m_{47}+c_{18} \end{bmatrix}$$

$$\hat{m}_{19} = maj\begin{bmatrix} m_{19} \\ m_3+m_4+m_{11}+m_{26}+m_{28}+m_{32}+m_{46}+c_3 \\ m_6+m_7+m_{22}+m_{29}+m_{31}+m_{34}+m_{35}+c_6 \\ m_1+m_9+m_{18}+m_{36}+m_{41}+m_{43}+m_{47}+c_{18} \\ m_2+m_5+m_{15}+m_{37}+m_{42}+m_{44}+m_{48}+c_{19} \end{bmatrix}$$

$$\hat{m}_{20} = maj\begin{bmatrix} m_{20} \\ m_5+m_9+m_{17}+m_{27}+m_{32}+m_{34}+m_{38}+c_9 \\ m_3+m_{12}+m_{13}+m_{30}+m_{35}+m_{37}+m_{41}+c_{12} \\ m_8+m_{11}+m_{21}+m_{29}+m_{43}+m_{45}+m_{48}+c_{20} \\ m_7+m_{10}+m_{24}+m_{28}+m_{42}+m_{47}+m_{49}+c_{24} \end{bmatrix}$$

$$\hat{m}_{21} = maj\begin{bmatrix} m_{21} \\ m_5+m_6+m_{18}+m_{28}+m_{30}+m_{33}+m_{39}+c_5 \\ m_4+m_{13}+m_{14}+m_{31}+m_{36}+m_{38}+m_{42}+c_{13} \\ m_8+m_{11}+m_{20}+m_{29}+m_{43}+m_{45}+m_{48}+c_{20} \\ m_9+m_{12}+m_{22}+m_{25}+m_{44}+m_{46}+m_{49}+c_{21} \end{bmatrix}$$

$$\hat{m}_{22} = maj\begin{bmatrix} m_{22} \\ m_6+m_7+m_{19}+m_{29}+m_{31}+m_{34}+m_{35}+c_6 \\ m_0+m_{10}+m_{14}+m_{32}+m_{37}+m_{39}+m_{43}+c_{14} \\ m_9+m_{12}+m_{21}+m_{25}+m_{44}+m_{46}+m_{49}+c_{21} \\ m_5+m_{13}+m_{23}+m_{26}+m_{40}+m_{45}+m_{47}+c_{22} \end{bmatrix}$$

$$\hat{m}_{23} = maj\begin{bmatrix} m_{23} \\ m_7+m_8+m_{15}+m_{25}+m_{30}+m_{32}+m_{36}+c_7 \\ m_1+m_{10}+m_{11}+m_{33}+m_{35}+m_{38}+m_{44}+c_{10} \\ m_5+m_{13}+m_{22}+m_{26}+m_{40}+m_{45}+m_{47}+c_{22} \\ m_6+m_{14}+m_{24}+m_{27}+m_{41}+m_{46}+m_{48}+c_{23} \end{bmatrix}$$

$$\hat{m}_{24} = maj \begin{bmatrix} m_{24} \\ m_8 + m_9 + m_{16} + m_{26} + m_{31} + m_{33} + m_{37} + C_8 \\ m_2 + m_{11} + m_{12} + m_{34} + m_{36} + m_{39} + m_{40} + C_{11} \\ m_6 + m_{14} + m_{23} + m_{27} + m_{41} + m_{46} + m_{48} + C_{23} \\ m_7 + m_{10} + m_{20} + m_{28} + m_{42} + m_{47} + m_{49} + C_{24} \end{bmatrix}$$

$$\hat{m}_{25} = maj \begin{bmatrix} m_{25} \\ m_0 + m_1 + m_{13} + m_{16} + m_{28} + m_{34} + m_{48} + C_0 \\ m_2 + m_3 + m_{10} + m_{18} + m_{27} + m_{31} + m_{45} + C_2 \\ m_7 + m_8 + m_{15} + m_{23} + m_{30} + m_{32} + m_{36} + C_7 \\ m_9 + m_{12} + m_{21} + m_{22} + m_{44} + m_{46} + m_{49} + C_{21} \end{bmatrix}$$

$$\hat{m}_{26} = maj \begin{bmatrix} m_{26} \\ m_1 + m_2 + m_{14} + m_{17} + m_{29} + m_{30} + m_{49} + C_1 \\ m_3 + m_4 + m_{11} + m_{19} + m_{28} + m_{32} + m_{46} + C_3 \\ m_8 + m_9 + m_{16} + m_{24} + m_{31} + m_{33} + m_{37} + C_8 \\ m_5 + m_{13} + m_{22} + m_{23} + m_{40} + m_{45} + m_{47} + C_{22} \end{bmatrix}$$

$$\hat{m}_{27} = maj \begin{bmatrix} m_{27} \\ m_2 + m_3 + m_{10} + m_{18} + m_{25} + m_{31} + m_{45} + C_2 \\ m_0 + m_4 + m_{12} + m_{15} + m_{29} + m_{33} + m_{47} + C_4 \\ m_5 + m_9 + m_{17} + m_{20} + m_{32} + m_{34} + m_{38} + C_9 \\ m_6 + m_{14} + m_{23} + m_{24} + m_{41} + m_{46} + m_{48} + C_{23} \end{bmatrix}$$

$$\hat{m}_{28} = maj \begin{bmatrix} m_{28} \\ m_0 + m_1 + m_{13} + m_{16} + m_{25} + m_{34} + m_{48} + C_0 \\ m_3 + m_4 + m_{11} + m_{19} + m_{26} + m_{32} + m_{46} + C_3 \\ m_5 + m_6 + m_{18} + m_{21} + m_{30} + m_{33} + m_{39} + C_5 \\ m_7 + m_{10} + m_{20} + m_{24} + m_{42} + m_{47} + m_{49} + C_{24} \end{bmatrix}$$

$$\hat{m}_{29} = maj \begin{bmatrix} m_{29} \\ m_1 + m_2 + m_{14} + m_{17} + m_{26} + m_{30} + m_{49} + C_1 \\ m_0 + m_4 + m_{12} + m_{15} + m_{27} + m_{33} + m_{47} + C_4 \\ m_6 + m_7 + m_{19} + m_{22} + m_{31} + m_{34} + m_{35} + C_6 \\ m_8 + m_{11} + m_{20} + m_{21} + m_{43} + m_{45} + m_{48} + C_{20} \end{bmatrix}$$

$$\hat{m}_{30} = maj \begin{cases} m_{30} \\ m_1 + m_2 + m_{14} + m_{17} + m_{26} + m_{29} + m_{49} + c_1 \\ m_5 + m_6 + m_{18} + m_{21} + m_{28} + m_{33} + m_{39} + c_5 \\ m_7 + m_8 + m_{15} + m_{23} + m_{25} + m_{32} + m_{36} + c_7 \\ m_3 + m_{12} + m_{13} + m_{20} + m_{35} + m_{37} + m_{41} + c_{12} \end{cases}$$

$$\hat{m}_{31} = maj \begin{cases} m_{31} \\ m_2 + m_3 + m_{10} + m_{18} + m_{25} + m_{27} + m_{45} + c_2 \\ m_6 + m_7 + m_{19} + m_{22} + m_{29} + m_{34} + m_{35} + c_6 \\ m_8 + m_9 + m_{16} + m_{24} + m_{26} + m_{33} + m_{37} + c_8 \\ m_4 + m_{13} + m_{14} + m_{21} + m_{36} + m_{38} + m_{42} + c_{13} \end{cases}$$

$$\hat{m}_{32} = maj \begin{cases} m_{32} \\ m_3 + m_4 + m_{11} + m_{19} + m_{26} + m_{28} + m_{46} + c_3 \\ m_7 + m_8 + m_{15} + m_{23} + m_{25} + m_{30} + m_{36} + c_7 \\ m_5 + m_9 + m_{17} + m_{20} + m_{27} + m_{34} + m_{38} + c_9 \\ m_0 + m_{10} + m_{14} + m_{22} + m_{37} + m_{39} + m_{43} + c_{14} \end{cases}$$

$$\hat{m}_{33} = maj \begin{cases} m_{33} \\ m_0 + m_4 + m_{12} + m_{15} + m_{27} + m_{29} + m_{47} + c_4 \\ m_5 + m_6 + m_{18} + m_{21} + m_{28} + m_{30} + m_{39} + c_5 \\ m_8 + m_9 + m_{16} + m_{24} + m_{26} + m_{31} + m_{37} + c_8 \\ m_1 + m_{10} + m_{11} + m_{23} + m_{35} + m_{38} + m_{44} + c_{10} \end{cases}$$

$$\hat{m}_{34} = maj \begin{cases} m_{34} \\ m_0 + m_1 + m_{13} + m_{16} + m_{25} + m_{28} + m_{48} + c_0 \\ m_6 + m_7 + m_{19} + m_{22} + m_{29} + m_{31} + m_{35} + c_6 \\ m_5 + m_9 + m_{17} + m_{20} + m_{27} + m_{32} + m_{38} + c_9 \\ m_2 + m_{11} + m_{12} + m_{24} + m_{36} + m_{36} + m_{39} + m_{40} + c_{11} \end{cases}$$

$$\hat{m}_{35} = maj \begin{cases} m_{35} \\ m_6 + m_7 + m_{19} + m_{22} + m_{29} + m_{31} + m_{34} + c_6 \\ m_1 + m_{10} + m_{11} + m_{23} + m_{33} + m_{38} + m_{44} + c_{10} \\ m_3 + m_{12} + m_{13} + m_{20} + m_{30} + m_{37} + m_{41} + c_{12} \\ m_0 + m_8 + m_{17} + m_{18} + m_{40} + m_{42} + m_{46} + c_{17} \end{cases}$$

$$\hat{m}_{36} = maj\begin{bmatrix} m_{36} \\ m_7+m_8+m_{15}+m_{23}+m_{25}+m_{30}+m_{32}+c_7 \\ m_2+m_{11}+m_{12}+m_{24}+m_{34}+m_{39}+m_{40}+c_{11} \\ m_4+m_{13}+m_{14}+m_{21}+m_{31}+m_{38}+m_{42}+c_{13} \\ m_1+m_9+m_{18}+m_{19}+m_{41}+m_{43}+m_{47}+c_{18} \end{bmatrix}$$

$$\hat{m}_{37} = maj\begin{bmatrix} m_{37} \\ m_8+m_9+m_{16}+m_{24}+m_{26}+m_{31}+m_{33}+c_8 \\ m_3+m_{12}+m_{13}+m_{20}+m_{30}+m_{35}+m_{41}+c_{12} \\ m_6+m_{10}+m_{14}+m_{22}+m_{32}+m_{39}+m_{43}+c_{14} \\ m_2+m_5+m_{15}+m_{19}+m_{42}+m_{44}+m_{48}+c_{19} \end{bmatrix}$$

$$\hat{m}_{38} = maj\begin{bmatrix} m_{38} \\ m_5+m_9+m_{17}+m_{20}+m_{27}+m_{32}+m_{34}+c_9 \\ m_1+m_{10}+m_{11}+m_{23}+m_{33}+m_{36}+m_{44}+c_{10} \\ m_4+m_{13}+m_{14}+m_{21}+m_{31}+m_{36}+m_{42}+c_{13} \\ m_3+m_5+m_{15}+m_{16}+m_{40}+m_{43}+m_{49}+c_{15} \end{bmatrix}$$

$$\hat{m}_{39} = maj\begin{bmatrix} m_{39} \\ m_5+m_6+m_{18}+m_{21}+m_{28}+m_{30}+m_{33}+c_5 \\ m_2+m_{11}+m_{12}+m_{24}+m_{34}+m_{36}+m_{40}+c_{11} \\ m_0+m_{10}+m_{14}+m_{22}+m_{32}+m_{37}+m_{43}+c_{14} \\ m_4+m_7+m_{16}+m_{17}+m_{41}+m_{44}+m_{45}+c_{16} \end{bmatrix}$$

$$\hat{m}_{40} = maj\begin{bmatrix} m_{40} \\ m_2+m_{11}+m_{12}+m_{24}+m_{34}+m_{36}+m_{39}+c_{11} \\ m_3+m_6+m_{15}+m_{16}+m_{38}+m_{43}+m_{49}+c_{15} \\ m_0+m_8+m_{17}+m_{18}+m_{35}+m_{42}+m_{46}+c_{17} \\ m_5+m_{13}+m_{22}+m_{23}+m_{26}+m_{43}+m_{47}+c_{22} \end{bmatrix}$$

$$\hat{m}_{41} = maj\begin{bmatrix} m_{41} \\ m_3+m_{12}+m_{13}+m_{20}+m_{30}+m_{35}+m_{37}+c_{12} \\ m_4+m_7+m_{16}+m_{17}+m_{39}+m_{44}+m_{45}+c_{16} \\ m_1+m_9+m_{18}+m_{19}+m_{36}+m_{43}+m_{47}+c_{18} \\ m_6+m_{14}+m_{23}+m_{24}+m_{27}+m_{46}+m_{48}+c_{23} \end{bmatrix}$$

$$\hat{m}_{42} = maj\begin{bmatrix} m_{42} \\ m_4 + m_{13} + m_{14} + m_{21} + m_{31} + m_{36} + m_{38} + c_{13} \\ m_0 + m_8 + m_{17} + m_{18} + m_{35} + m_{40} + m_{46} + c_{17} \\ m_2 + m_5 + m_{15} + m_{19} + m_{37} + m_{44} + m_{48} + c_{19} \\ m_7 + m_{10} + m_{20} + m_{24} + m_{28} + m_{47} + m_{29} + c_{24} \end{bmatrix}$$

$$\hat{m}_{43} = maj\begin{bmatrix} m_{43} \\ m_0 + m_{10} + m_{14} + m_{22} + m_{32} + m_{37} + m_{39} + c_{14} \\ m_3 + m_6 + m_{15} + m_{16} + m_{38} + m_{40} + m_{49} + c_{15} \\ m_1 + m_9 + m_{18} + m_{19} + m_{36} + m_{41} + m_{47} + c_{18} \\ m_8 + m_{11} + m_{20} + m_{21} + m_{29} + m_{45} + m_{48} + c_{20} \end{bmatrix}$$

$$\hat{m}_{44} = maj\begin{bmatrix} m_{44} \\ m_1 + m_{10} + m_{11} + m_{23} + m_{33} + m_{35} + m_{38} + c_{18} \\ m_4 + m_7 + m_{16} + m_{17} + m_{39} + m_{41} + m_{45} + c_{16} \\ m_2 + m_5 + m_{15} + m_{19} + m_{37} + m_{42} + m_{48} + c_{19} \\ m_9 + m_{12} + m_{21} + m_{22} + m_{25} + m_{46} + m_{49} + c_{21} \end{bmatrix}$$

$$\hat{m}_{45} = maj\begin{bmatrix} m_{45} \\ m_2 + m_3 + m_{10} + m_{18} + m_{25} + m_{27} + m_{31} + c_2 \\ m_4 + m_7 + m_{16} + m_{17} + m_{39} + m_{41} + m_{44} + c_{16} \\ m_8 + m_{11} + m_{20} + m_{21} + m_{29} + m_{43} + m_{48} + c_{20} \\ m_5 + m_{13} + m_{22} + m_{23} + m_{26} + m_{40} + m_{47} + m_{22} \end{bmatrix}$$

$$\hat{m}_{46} = maj\begin{bmatrix} m_{46} \\ m_3 + m_4 + m_{11} + m_{19} + m_{26} + m_{28} + m_{32} + c_3 \\ m_0 + m_8 + m_{17} + m_{18} + m_{35} + m_{40} + m_{42} + c_{17} \\ m_9 + m_{12} + m_{21} + m_{22} + m_{25} + m_{44} + m_{49} + c_{21} \\ m_6 + m_{14} + m_{23} + m_{24} + m_{27} + m_{41} + m_{48} + c_{23} \end{bmatrix}$$

$$\hat{m}_{47} = maj\begin{bmatrix} m_{47} \\ m_0 + m_4 + m_{12} + m_{15} + m_{27} + m_{29} + m_{33} + c_4 \\ m_1 + m_9 + m_{18} + m_{19} + m_{36} + m_{41} + m_{43} + c_{18} \\ m_5 + m_{13} + m_{22} + m_{23} + m_{26} + m_{40} + m_{45} + c_{22} \\ m_7 + m_{10} + m_{20} + m_{24} + m_{28} + m_{42} + m_{49} + c_{24} \end{bmatrix}$$

$$\hat{m}_{48} = \mathrm{maj}\begin{bmatrix} m_{48} \\ m_0 + m_1 + m_{13} + m_{16} + m_{25} + m_{28} + m_{34} + c_0 \\ m_2 + m_5 + m_{15} + m_{19} + m_{37} + m_{42} + m_{44} + c_{19} \\ m_8 + m_{11} + m_{20} + m_{21} + m_{29} + m_{43} + m_{45} + c_{20} \\ m_6 + m_{14} + m_{23} + m_{24} + m_{27} + m_{41} + m_{46} + c_{23} \end{bmatrix}$$

$$\hat{m}_{49} = \mathrm{maj}\begin{bmatrix} m_{49} \\ m_1 + m_2 + m_{14} + m_{17} + m_{26} + m_{29} + m_{30} + c_1 \\ m_3 + m_6 + m_{15} + m_{16} + m_{38} + m_{40} + m_{43} + c_{15} \\ m_9 + m_{12} + m_{21} + m_{22} + m_{25} + m_{44} + m_{46} + c_{21} \\ m_7 + m_{10} + m_{20} + m_{24} + m_{28} + m_{42} + m_{47} + c_{24} \end{bmatrix}$$

# APPENDIX III

## Encoder Wiring Schematics

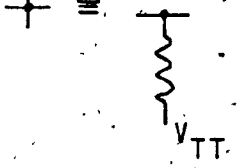


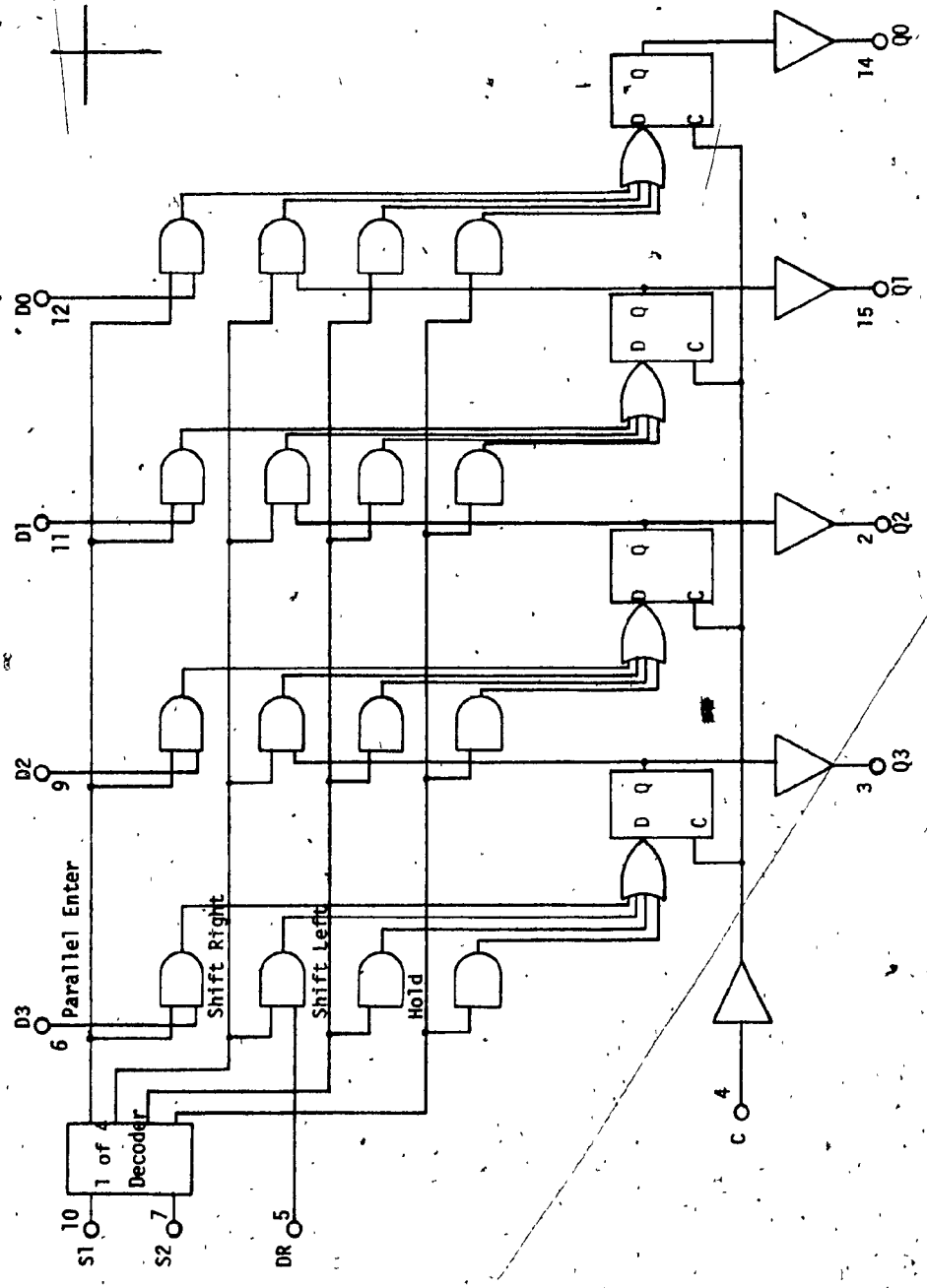
### Legend

$V_{CC1} = V_{CC2} = 0V$ (Red Wire)

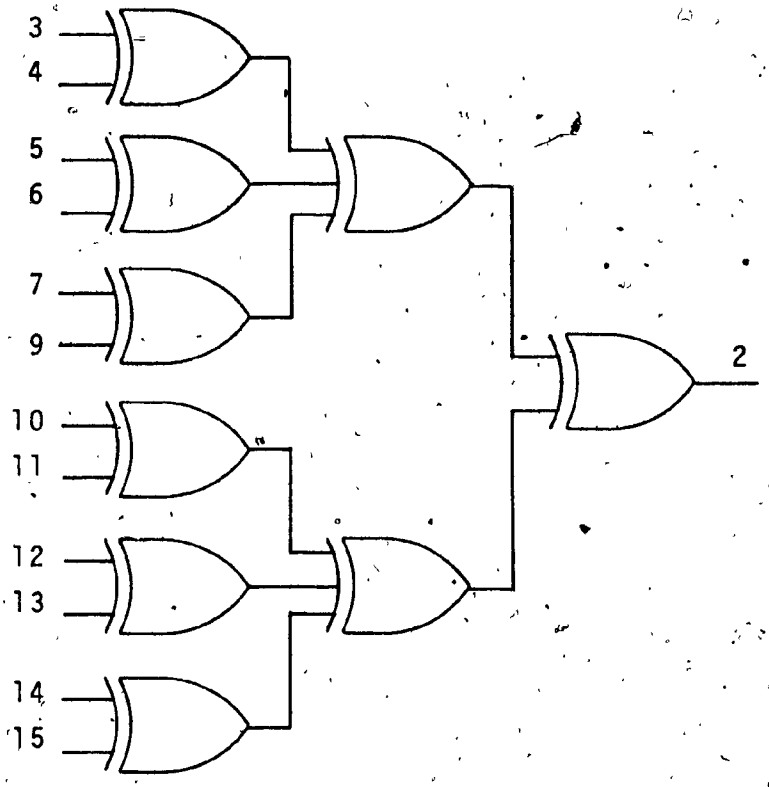
$V_{EE} = -5.2V$ (White Wire, ac ground)

(Blue/Black wire, signal)

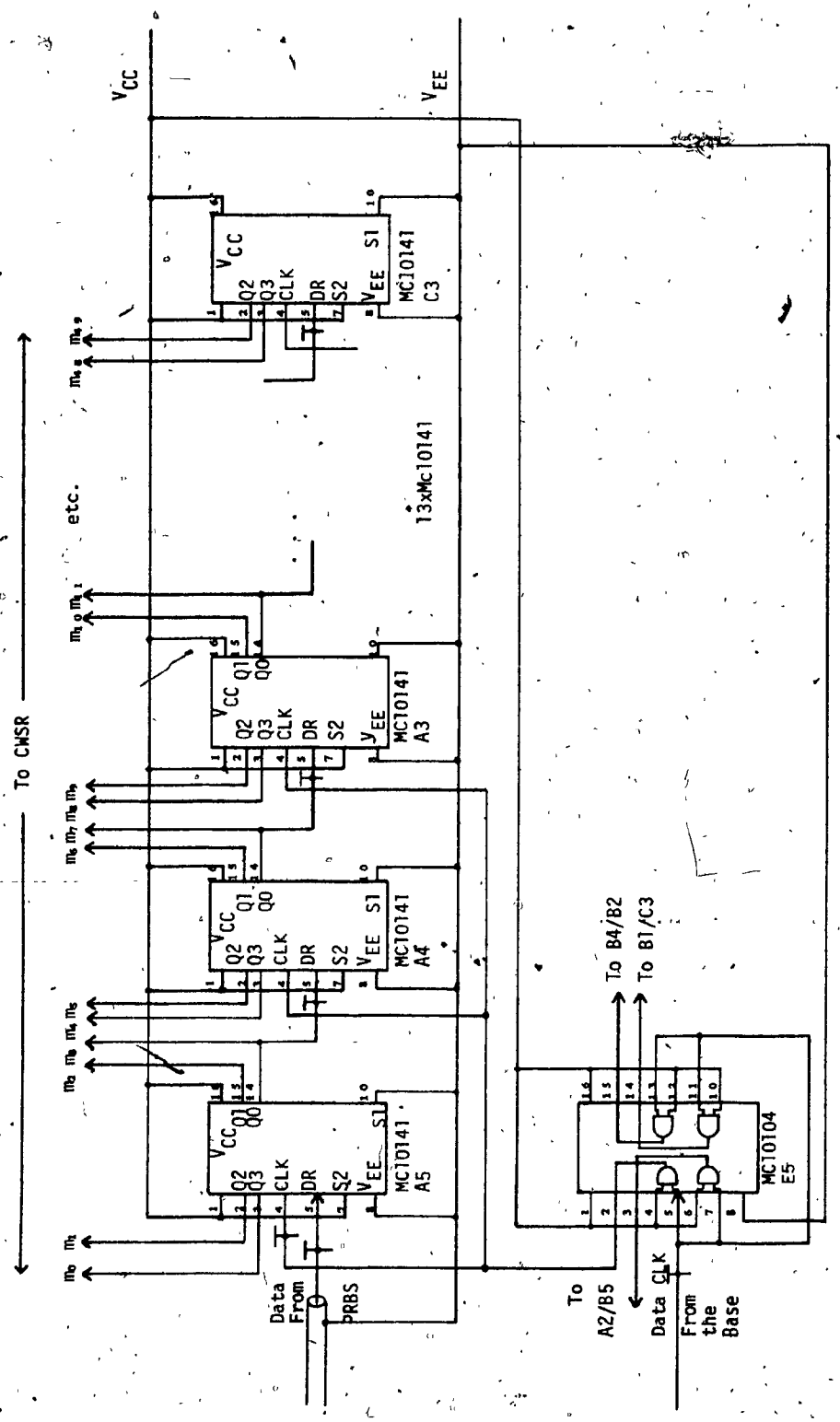




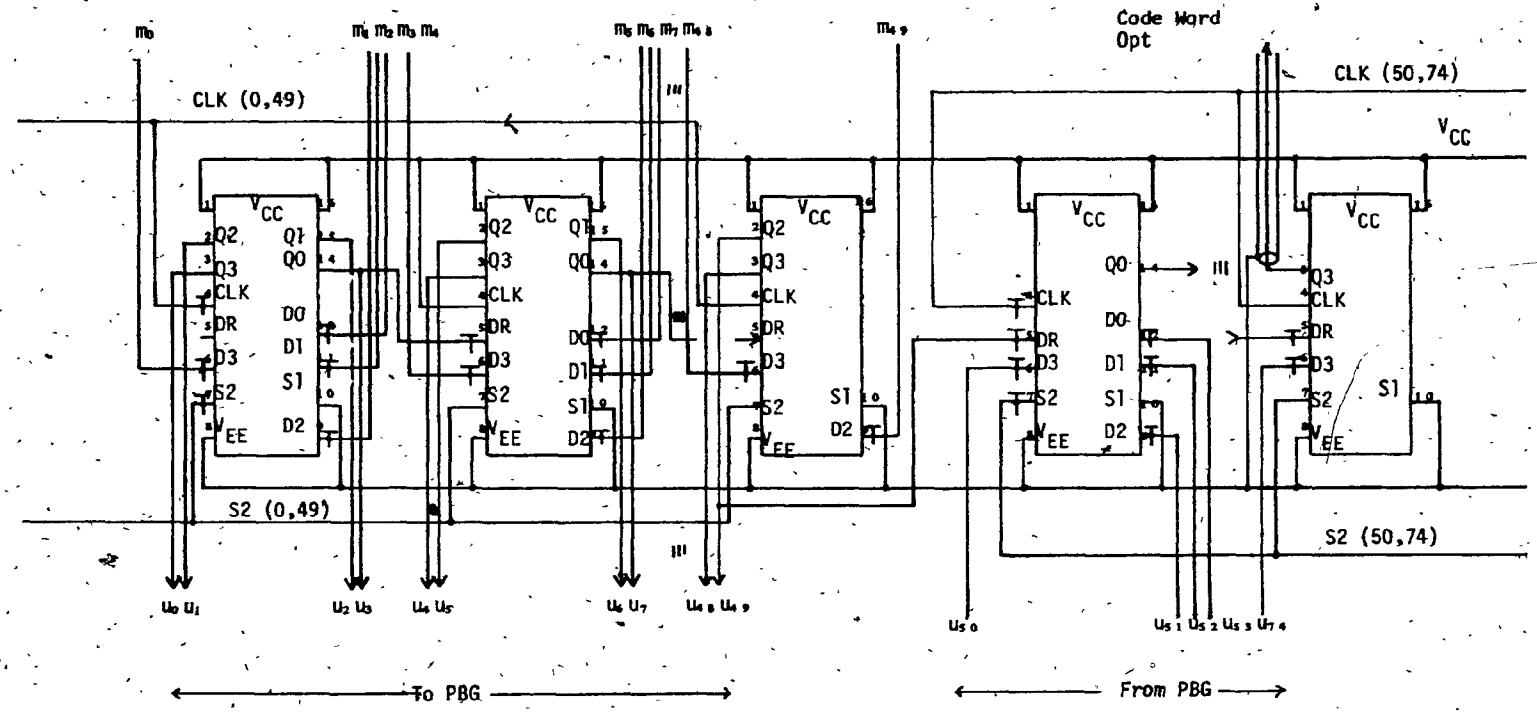

MC 10141 IC
Fig.AIII-1




MC 10160
Fig.AIII-2

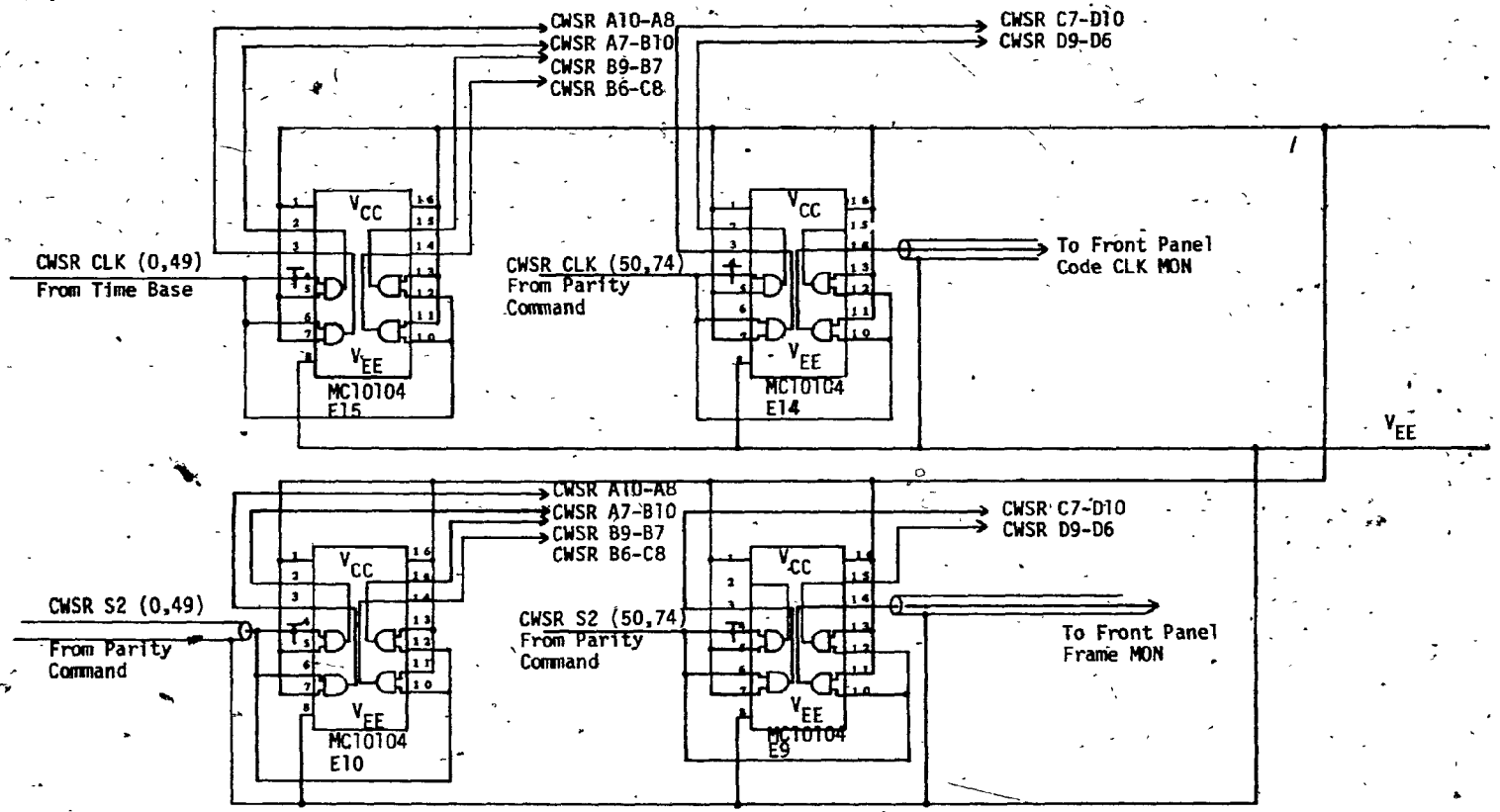



Data Input Shift Register and Clock Driver
Fig. AIII-3

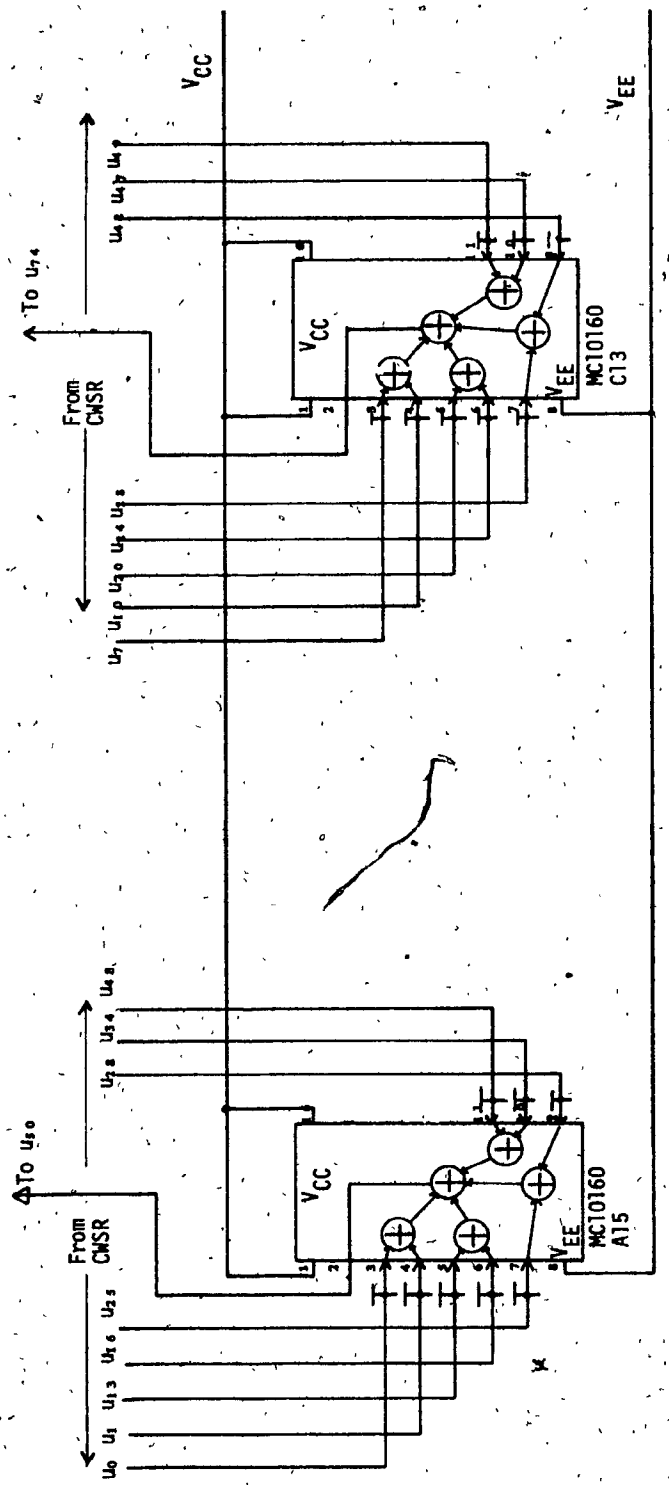



Code Word Shift Register

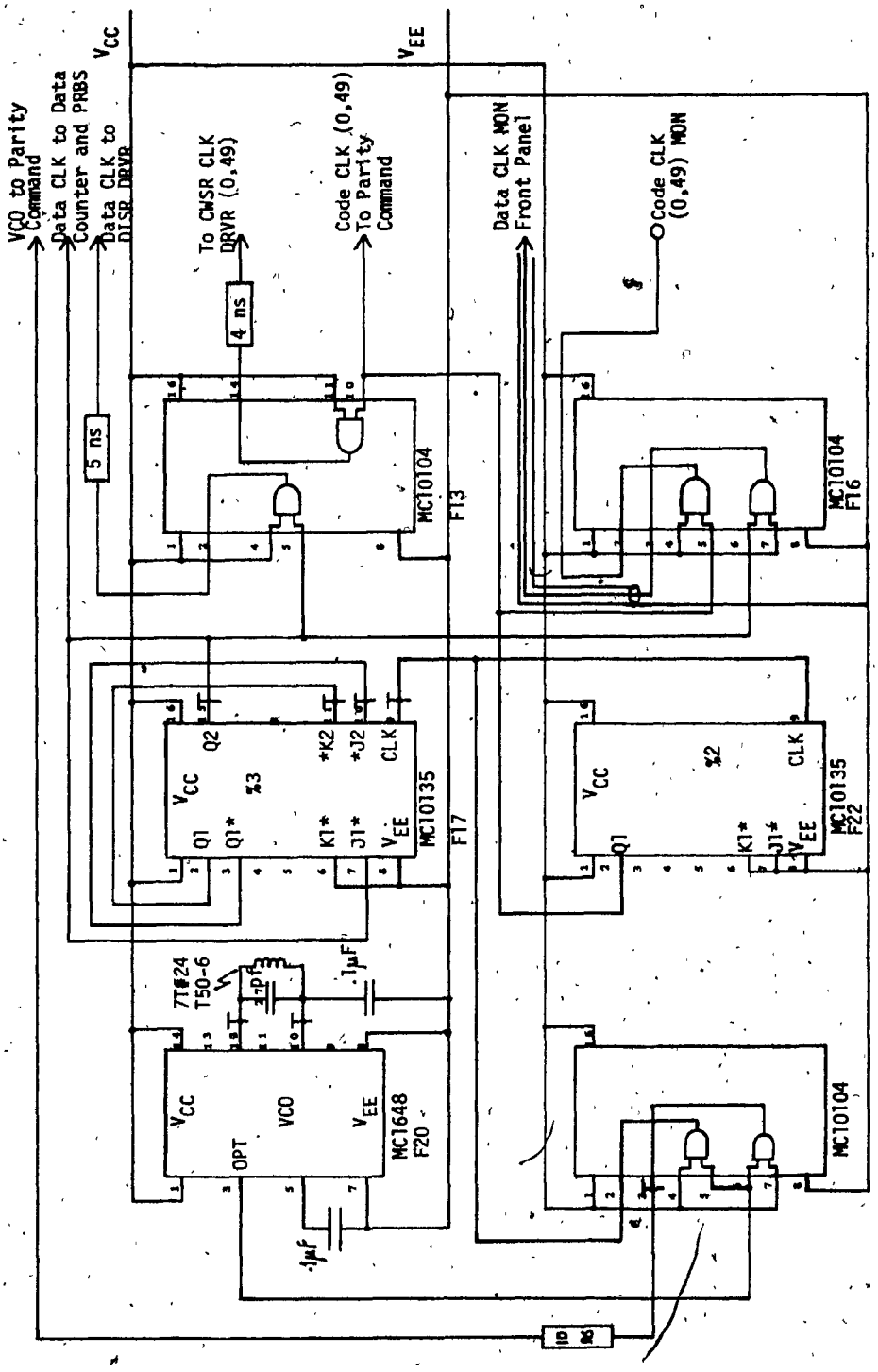
Fig.AIII-4



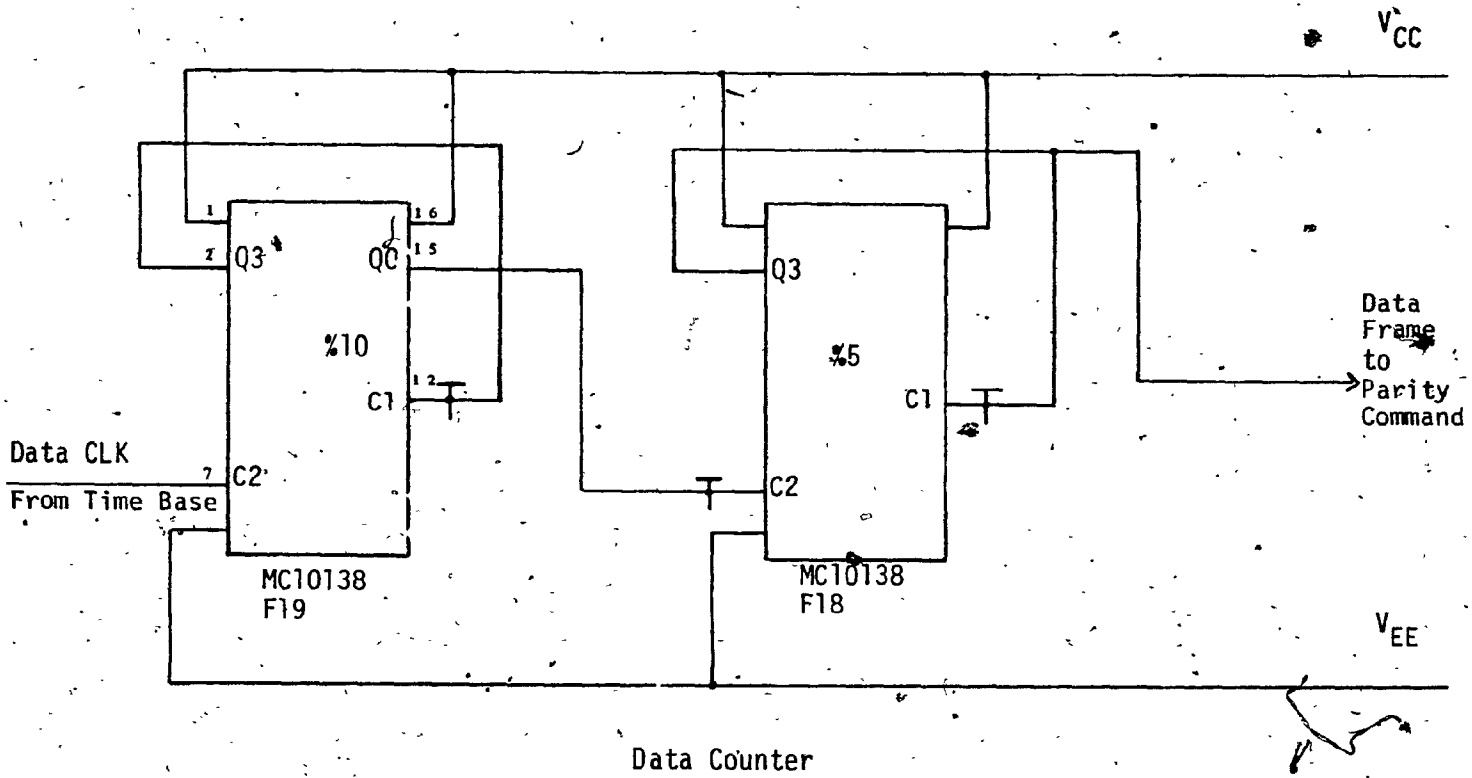

Code Word Shift Register Drivers
Fig.AIII-5




Parity Bit Generator
Fig.AIII-6

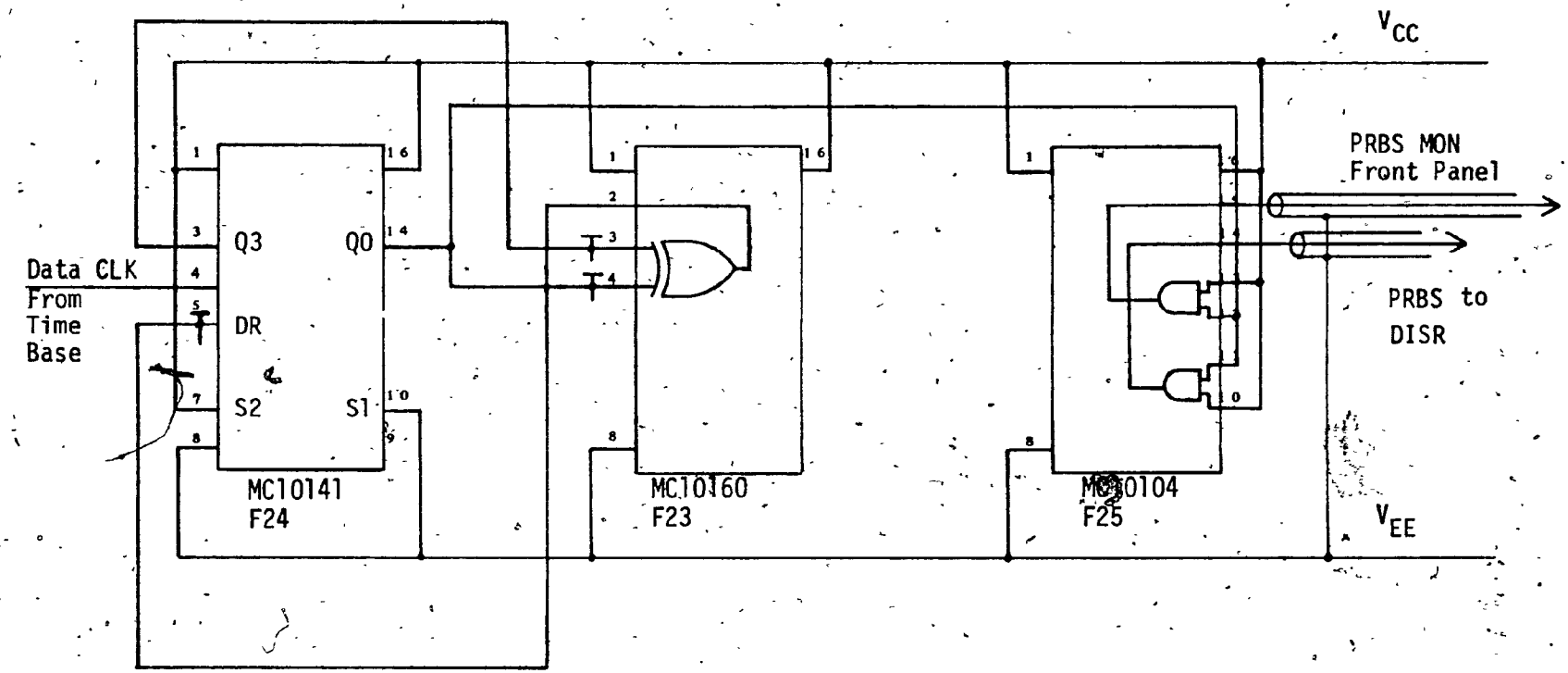



Time Base
Fig.AIII-7



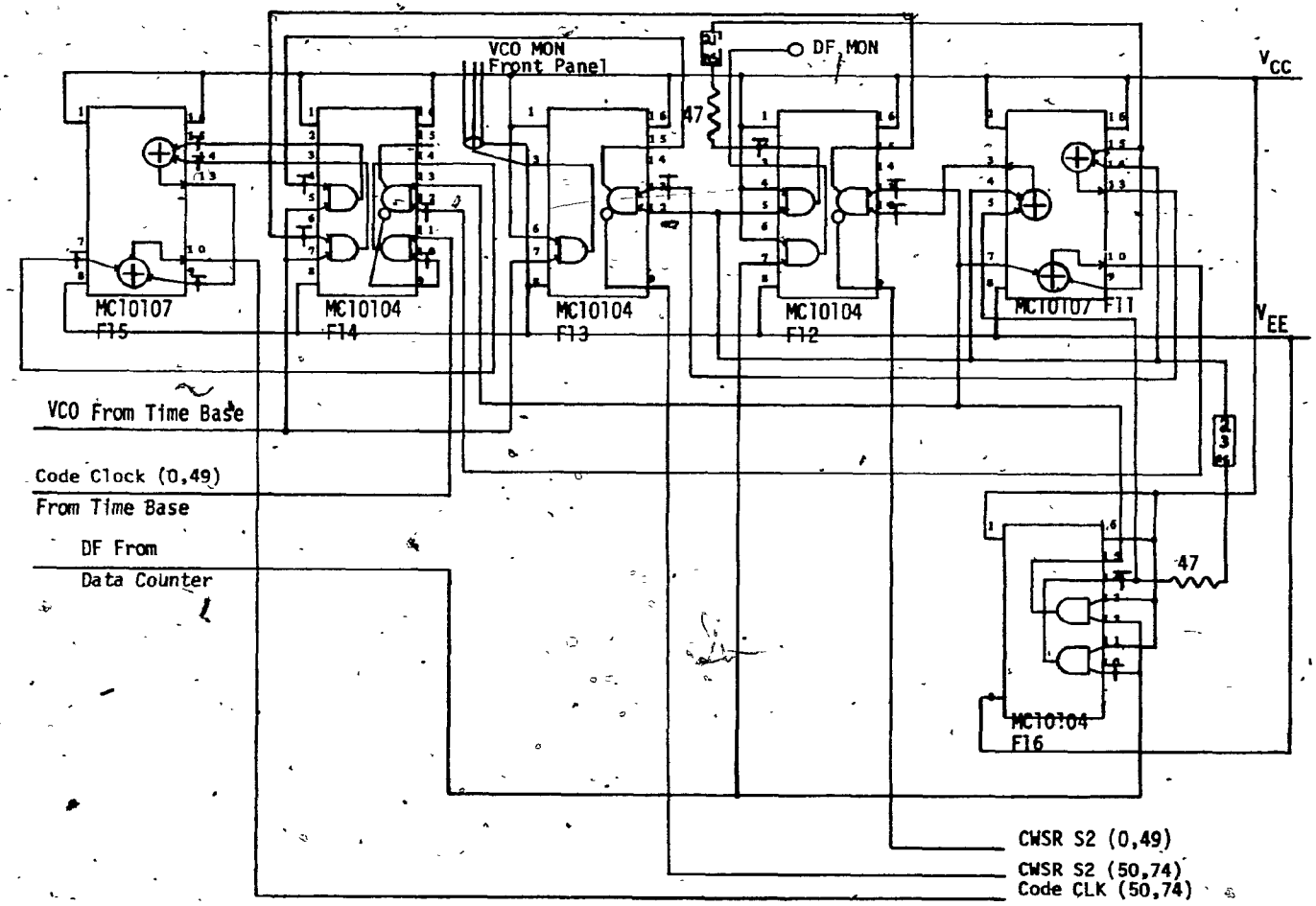

Data Counter

Fig.AIII-8




Pseudo Random Binary Sequence Generator
Fig.AIII-9

| S1 | S2 | |
|---|---|---|
| 0 | 1 | → Shift Right |

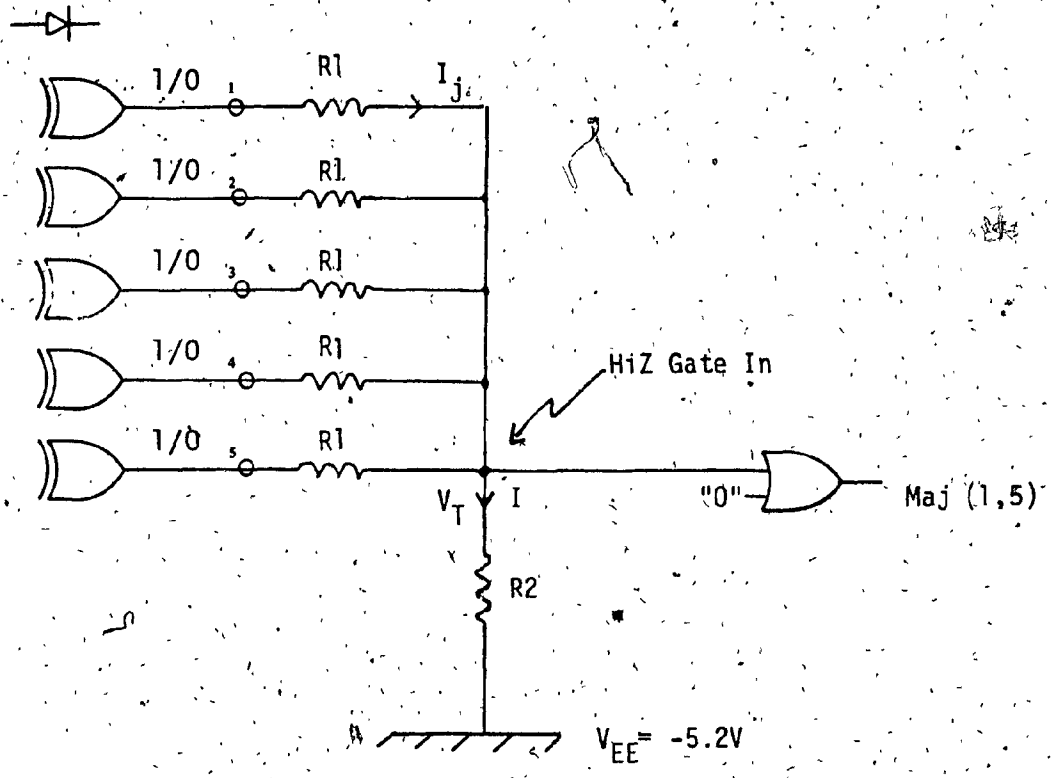



Parity Command Unit Fig.AIII-10

# APPENDIX IV

## Majority Logic Gate Construction

The construction of the 5 input majority logic gate is more complicated than most modules of the codec, but fairly straightforward. Concrete designs for this type of gate are very rarely seen in the current literature.




Majority Logic Gate
Fig.AIV-1

Fig.AIV-1 shows the construction of a 5 input majority gate. Each input feeds a resistance of R1Ω, all terminated at the same point and brought to ground by R2Ω. The junction point feeds the input to an ECL gate. The junction voltage $V_T$ is given by:

1 Input High

$$-.9V - (-5.2V) = I_j R1 + I R2 \quad , \quad I_j = I \text{ "1"} = -.9V \text{ typ.}$$

$$V_T = 4.3V \; \frac{R2}{R1+R2}$$

n Inputs High

$$V_T = 4.3V \; \frac{R2}{R1/n+R2} \quad , \quad nI_j = I$$

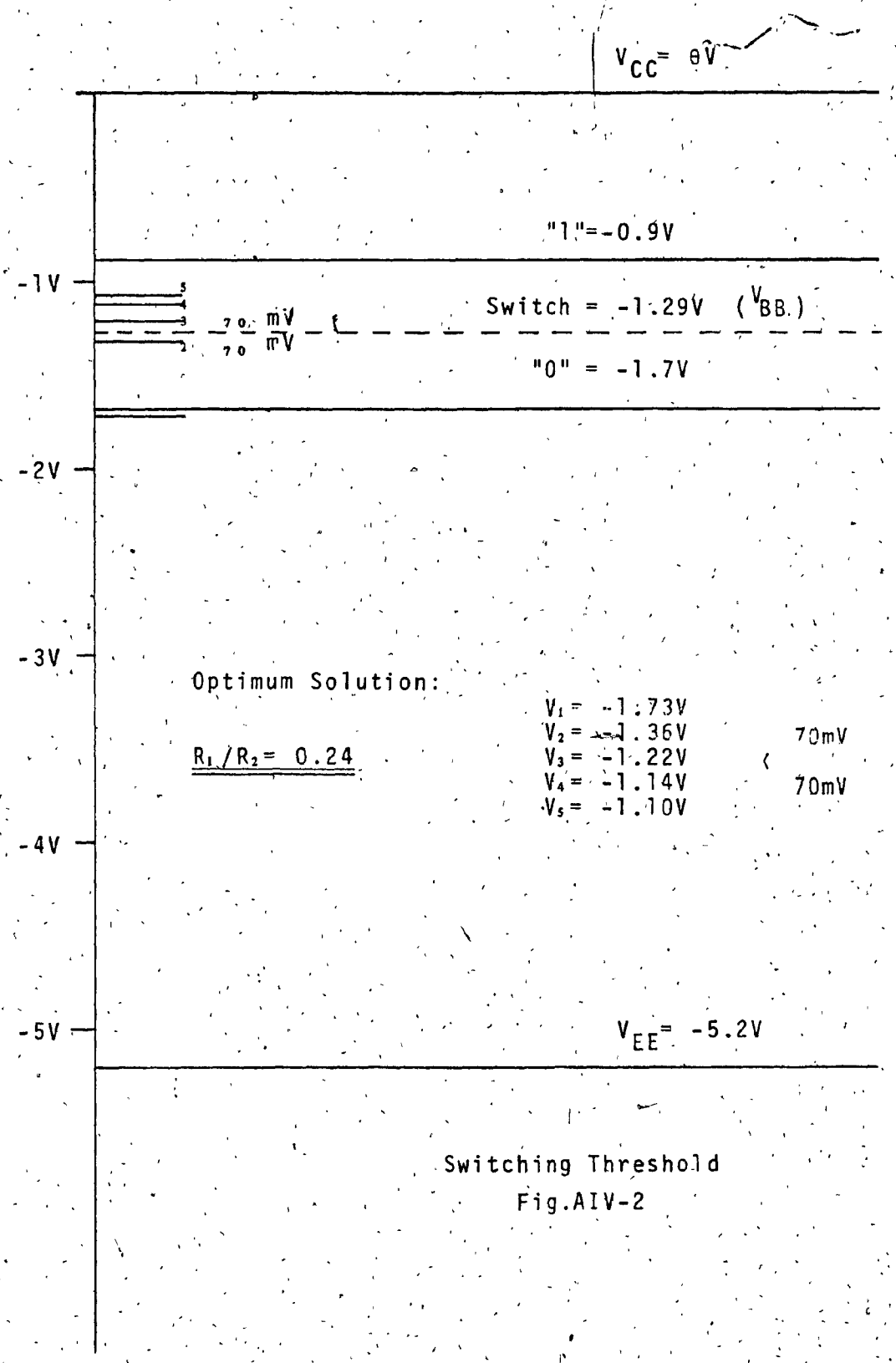
Note that any ECL "OR" output (see Fig.2.3.1) has an in-series diode, so that if the output is low, it will have no effect on the resistance tree structure, as $I_j$ in that branch = 0.

The design problem is to choose R1 & R2 so that the 2 inputs high and 3 inputs high are equally spaced about the switching threshold of an ECL gate, = -1.29V ($V_{BB}$). This is optimum for R1/R2 = 0.24, and gives a switching margin of 70mV. This is shown in Fig.AIV-2.

Although 70 mV is a small margin, once set up properly, the circuit works well. It may be implemented and tested using 2xMC10107 IC's (3xEX-OR; 1/3 as maj. gate & 1 2/3 as inputs). It is essential to match all input resistors R1

to within 1%. R1 is shown as 34.9Ω, and R2 is chosen as 85Ω fixed + 100Ω wirewound potentiometer. The pot is adjusted to yield the exact switching point. (Above or below ratio .24, circuit oscillates when 2 inputs high, or 3 inputs high). Any values may be chosen for R1 & R2, as long as R1+R2 < 470Ω, otherwise, the ECL inputs will not be sufficiently matched or pulled down.

Other structures are possible for majority logic gates ex: [E11] & [E12], however, indirect implementation with adders etc. must be avoided in this application due to duration of operation (EX: using MC 10180 2 bit adder, min 4 required at 4.5 nsec each > 18 nsec!)




Switching Threshold
Fig.AIV-2

16-12-81