BLL ID NO: DX 72751/87

This book was bound by
Badminton Press
18 Half Croft, Syston, Leicester, LE7 8LD
Telephone: Leicester (0533) 602918.

IMAGE TRANSMISSION OVER THE CAMBRIDGE RING

by

Bu Sung Lee, B.Sc.

A Doctoral Thesis

Submitted in partial fulfilment of the requirements for the award of the degree of Doctor of Philosophy of the University of Technology, Loughborough.

July, 1986

Supervisor : Professor J.W.R. Griffiths

Department of Electronic and Electrical Engineering

Loughborough University

England

Abstract

Local Area Networks (LAN) are destined to play a rapidly increasing part in the transmission and distribution of a wide range of information and this thesis describes the study of the problems concerning the transmission of coloured images over a particular network, the Cambridge Ring. A colour image station has been developed for the use on the Cambridge Ring. It provides two main services - a high resolution freeze frame transmission and a medium resolution slow-scan image transmission.

The initial part of the project was concerned mainly with the design, development and construction of a versatile and fast access framestore, the LUT framestore. This was based on a Graphic Display Processor (GDP), which provides vector and character drawing facilities to the user. To ensure compatibility with existing equipment, a standard multibus was used to interface to the host system.

Due to the limited data bandwidth available on the local area network, image data compression techniques were required when transmitting images over the network. A number of different interframe coding techniques, particularly motion predictors, were investigated. The displacement estimation algorithms used for the motion predictors were of two main types: differential algorithms and block matching algorithms, and the results showed that the block matching algorithms can compress the data more efficiently than the differential algorithms. A new block matching algorithm, the Basic Simplex search, was developed. This technique gave a 50% reduction in computation overhead when compared with the better known 2D-Logarithmic search.

An image station was designed around the LUT framestore and

it comprised the framestore, a single board computer and a Cambridge ring interface. Simple but resilient protocols were developed to ensure that the system operated correctly over the network. Flow control mechanisms were used to ensure that data overflow did not occur at the receiver station. Multimode coding was adopted for slow-scan image transmission so that the image station coding technique could adapt to the picture and network statistics. Successful image transmission experiments have been carried out using two of these image stations on the Universe project Cambridge Ring at Loughborough University.

Acknowledgement

I wish to express my sincere thanks to my supervisor Professor J.V.R. Griffiths, for his guidance throughout this work. My thanks also goes to Dr. D.J. Parish, Mr. M.J. Fairfield, Mr. A.D. Goodson and Mr. Tony Erwood for their assistance and technical help.

Many thanks to Mr. and Mrs. Chris Carey-Smith, Dr. T.N. Chen, Ms. Sheila Clarson and Mr. Timothy Rodgers for their patience and time in reading the final draft of this thesis.

I wish to express my sincere gratitude to my parents, my sister and brother, and also to Ms. C.E. Goh, for their encouragement and moral support throughout my studies.

Finally, I gratefully acknowledge the financial support provided by the British Telecom for me to carry out this work.

ABBREVIATIONS

ADC	Analog-to-Digital Converter
ADR	Address
BB	Basic Block
BRAM	Buffer memory
BSP	Byte stream protocol
CAS	Column address signal
CLK	Clock
CRTC	Cathode ray tube controller
DAC	Digital-to-Analog Converter
D^i	'i' displacement estimate vector
Dx	Horizontal displacement estimate
Dy	Vertical displacement estimate
DFD	Displace frame difference
DPCM	Differential Pulse Coded Modulation
ECL	Emitter-collector logic
FDIF	Frame-to-frame difference
GDP	Graphic display processor
I(x,y,t)	The pixel intensity at spatial location (x,y) and at frame 't'
I/O	Input/Output
LUT	Loughborough University of Technology
MA	Moving areas
MAE	Mean absolute error
MSE	Mean square error
PCM	Pulse coded Modulation
PROM	Programmable read only memory
R/W	Read/Write signal
RAM	Random access memory
RAS	Row address signal
SSP	Single shot protocol
TTL	Transistor-transistor logic
VRAM	Video memory
CR	Cambridge Ring

CONTENTS

ABSTRACT

ACKNOWLEDGEMENT

ABBREVIATIONS

CONTENTS

CHAPTER 1 : INTRODUCTION

CHAPTER 2 : CONDITIONAL REPLENISHMENT CODING

CONTENTS

CHAPTER 3 : DISPLACEMENT ESTIMATION ALGORITHMS

CHAPTER 4 : FRAMESTORE

CONTENTS

CHAPTER 5 : NETWORK PROTOCOL ARCHITECTURE

CHAPTER 6 : IMAGE STATION

CONTENTS

CHAPTER 7 : Discussion and Suggestions for Future work

APPENDIX

CHAPTER ONE

INTRODUCTION

1.1. INTRODUCTION

Over the past decade, the range of applications of Local Computer Networks has steadily increased. It is envisaged that they will play an essential role in office automation. Much interest has been focused on real time applications such as image communication.

The video coder/decoder (codecs) market, for videoconferen--cing, can be broadly divided into two main groups: high bandwidth, full motion, e.g. COST 211 system [76], and slow-scan [6], e.g. Netec-XD systems. The two systems are distinguished from each other by their frame update rate. The former have a frame update rate of broadcast television, i.e. 25 frames per second, while the latter can take as long as a few seconds to update a frame.

One of the most significant costs in owning a videoconference system is the cost of the line connection [58]. The cost of having a video connection is very high, e.g. the 1.544 MHz. video connection in the US would cost around US$1,500 per hour [6]. This cost can be reduced by transmitting the digitised video data over computer networks which already exist in most large companies. Thus, a special video connection is not required. Furthermore, provision of this service will not degrade existing services, in terms of response time, as there is usually some excess bandwidth in most computer networks.

An immediate problem encountered in transmitting digitised video data over a computer network is its high bit-rate requirement. Taking the example of a picture with a resolution of 256 by 256 digitised picture elements (pixels), each pixel being represented by 16 bits, a system data transfer rate of 25 Mbits per second is required if 25 frames are transmitted per second. As baseband computer networks, e.g. the Cambridge ring, have a bandwidth of less than 10 MHZ., image compression techniques [40,52,63,66] are required to reduce the bit-rate.

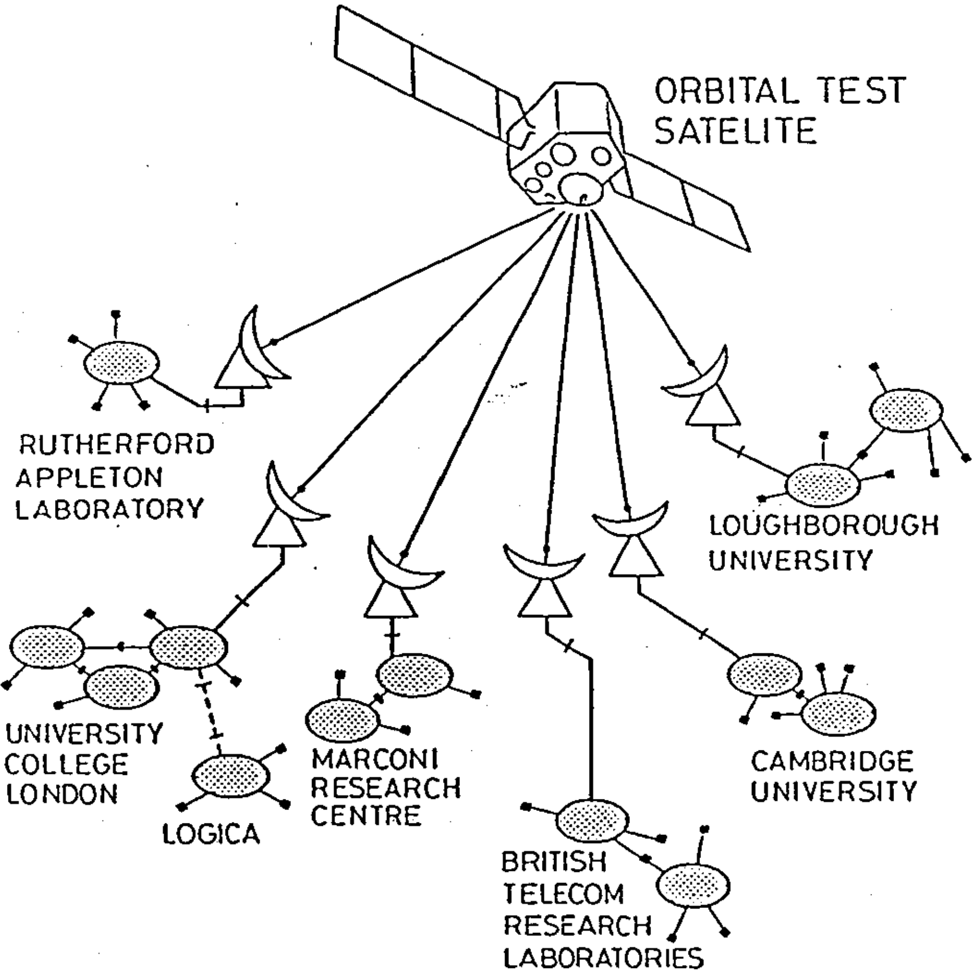


Figure 1.1 : Project Universe infrastructure

Project Universe [16], started in mid-1981, has provided the infrastructure for carrying out experiments on video transmission over a computer network. The Universe network consisted of seven Cambridge ring sites linked by the Orbital

Test Satellite(OTS) as shown in Figure 1.1.. Various forms of data: computer data, digitised video and speech, were transmitted around the ring. Loughborough University was mainly involved in the development of a slow-scan system, image station, [25,26] and coded speech. Experiance obtained from experiments carried out in the Universe project have helped in the design of the new image station which will be discussed in chapter 6.

This thesis relates to the investigation of Conditional replenishment coding, Displacement estimation algorithms and the design and implementation of a video codec for the Cambridge ring. Section 1.2. will give a brief review of interframe predictive coding techniques. The last section will describe the organisation of the thesis.

1.2. Interframe predictive coding - A brief review

Interframe coding [24,29,60] exploits the following redun--dancies of an image sequence:

(a) the similarity between successive frames of images
(b) the resolution requirements of moving areas.

One method of exploiting the similarity between successive frames is to reduce the number of frames transmitted per second. At the receiver the frames are repeated at 25 frames per second, to avoid flicker on the display, until a new frame arrives. The reduction in bit-rate depends on the ratio between the number of transmitted frames per second and frame display rate, 25 frames. R.C.Brainard et. al. [11] recommended a frame update rate of 15 frames per second. Lower frame update rates produce jerky motion.

Another form of frame repeat, proposed by R.C. Brainard [11], systematically scanned and updated a pixel once every N frames. The unupdated pixels are repeated at the receiver.

It suffers from two main defects;

(a) ragged edges of the moving object
(b) superimposition of patterns

A more efficient form of interframe coding is Conditional Replenishment coding, initially proposed by Mounts[59]. In this only those pixels showing significant changes since the previous frame are coded and transmitted. The pixels are classified as moving area pixels if their frame-to-frame difference(FDIF) i.e. the luminance difference between a pixel and its corresponding pixel in the previous frames, exceeds the given threshold. Buffers are required to average out the data rate, as the instantaneous transmission rate of the system fluctuates.

In Mounts[59] system the monochrome image luminance signal of moving area pixels are transmitted as 8-bits Pulse Coded Modulation(PCM). The system has an average rate of 1 bit per pixel. Due to the fluctuation in instantaneous transmission, large buffers are required to average out the data rates.

J.C.Candy [17] proposed two methods of reducing the fluctuation in the instantaneous transmission rate:

(a) moving area pixels within a scanline are clustered into runs
(b) moving area pixels are ignored if they are preceded and followed by two unchanged pixels along a scanline.

These two techniques decrease the fluctuations of the instantaneous transmission rate, thus reducing the size of the buffer required. In adverse conditions, i.e. high percentage of changed data, the coder switched to other modes and reduced the spatial resolution. The switching of modes is determined by the fullness of the buffer.

In the system proposed by J.C. Candy [17] the FDIF signals of moving area pixels are coded using 4 bits. Further reduction in bit-rate can be achieved by using the Conditional replenishment system in conjunction with other coding techniques. Hiroshi Yasuda [83] suggested the use of one dimensional Differential Pulse Coded Modulation(DPCM) for coding all the frame-to-frame differences of moving area pixels in a run. The prediction errors, i.e. the difference in FDIF between adjacent pixels, are then coded with variable wordlength coding. The three stages of coding are shown in the figure below.

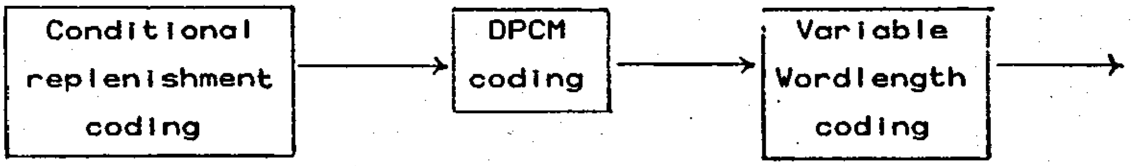


Figure 1.2. Coding stages

Displacement compensated coding utilises the knowledge of the displacement between successive frames, to reduce the bit-rate of the system. It replaces the second phase, DPCM coding, of the coding process shown above. Its principle can be illustrated in the figure below.

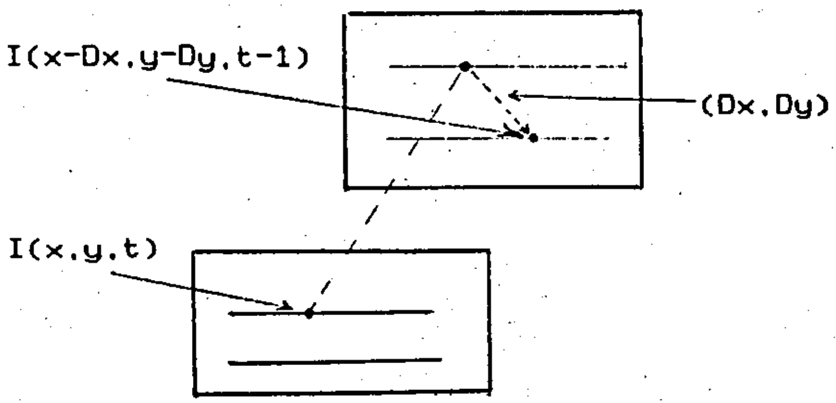


Figure 1.3. Displacement Compensated coding principle

where 'x' and 'y' are the horizontal and vertical spatial coordinates respectively. and 'Dx' and 'Dy' are the horizontal and vertical displacement vector. The luminance difference between the present pixel at spatial location (x,y) and the estimated position in the previous frame (x-Dx,y-Dy), is known as Displace Frame Difference(DFD). The DFD is then coded with a variable wordlength coder. Displacement estimation algorithms can be broadly divided into two main groups: pixel recursive algorithms [68] and block matching algorithms [73].

The pixel recursive algorithms use the pixel luminance differences in the time and spatial domain to estimate the displacement on a pixel by pixel basis. Netravali [62] uses the steepest descent algorithm to find the displacement vector. This algorithm gives a 20% reduction in bit-rate when compared with the Frame-to-Frame Difference Conditional Replenishment system proposed by Candy [17]. D.R. Walker [77] proposed a more efficient algorithm to find the displacement vector which uses Newton's method.

The block matching algorithms first segment the image into fixed rectangular blocks. The best match block in the previous frame is located for each of the blocks in the present frame that have changed significantly. The vector representing the spatial displacement of the best match block relative to its changed block is regarded as the displacement vector of the changed block. The computational overhead in testing every possible displacement location is extremely high. Search techniques have thus been used to reduce the number of tests required to locate the best match block. Jain [41] suggested the use of the 2-dimensional logarithmic search technique. This reduces the computational overhead by a factor of 5. Rani Srinivasan [73] reduced the computational overhead even further by using the Univariate search technique.

A new block matching algorithm, Basic Simplex, was developed. It is based on the simplex search technique. It gives a 50% reduction in computation overhead when compared with the more well known 2-dimensional logarithmic search.

1.3. Organisation of the thesis

Chapter 1 gives a brief review of interframe predictive coding algorithms.

Chapter 2 gives an introduction to the Conditional Replenishment Coder. In this chapter, multimode coding strategies are also discussed.

Chapter 3 is allocated to the discussion of displacement estimation algorithms: pixel recursive and block matching algorithms.

Chapter 4 discusses the design of the Loughborough University of Technology(LUT) framestore. This chapter also provides information on how to use the system and includes brief discussions of the circuits.

Chapter 5 describes the general protocols used on the Universe network. The aim of this chapter is to provide background knowledge of the protocols used on the network. The ring performance, in terms of data throughput and flow control, is also discussed.

Chapter 6 describes the operation of the LUT image station over the Cambridge ring. This includes the description of the coding algorithms, control strategy, and the protocol used.

The thesis concludes with Chapter 7, which reviews the achievements as well as proposing new areas of work.

CHAPTER TWO

CONDITIONAL REPLENISHMENT CODER

2.1. INTRODUCTION

In most video-conference scenes, the amount of movement is fairly small. Thus, only a small number of pixels would have changed significantly [84]. An efficient form of coding, conditional replenishment coding, exploits this characteris--tic by only transmitting those pixels that have changed significantly. It is usually used in combination with other coding techniques, e.g. predictive coding and variable wordlength coding.

Figure 2.1 shows a block diagram of a conditional replenishment coder which uses conditional replenishment coding in combination with motion compensated predictive coding and variable wordlength coding. For each changed pixel, $I(x,y,t)$, the displacement predictor reads the pixel luminance in the reference memory, $I(x-Dx,y-Dy,t-\tau)$, where x and y represent the spatial coordinate and Dx and Dy are their respective displacement estimates. The differential value, $I(x,y,t)-I(x-Dx,y-Dy,t-\tau)$, is then coded with a variable wordlength coder. The output of the variable wordlength coder and the addressing information are then fed into a buffer to await transmission, while the new pixel value is stored in the reference memory.

2.2. Simulation system

The laser disc player, Pioneer LD-1100, is used as the image source when studing interframe coding, e.g. conditional replenishment coding. Figure 2.2 shows the simulation system setup. The laser disc produces composite

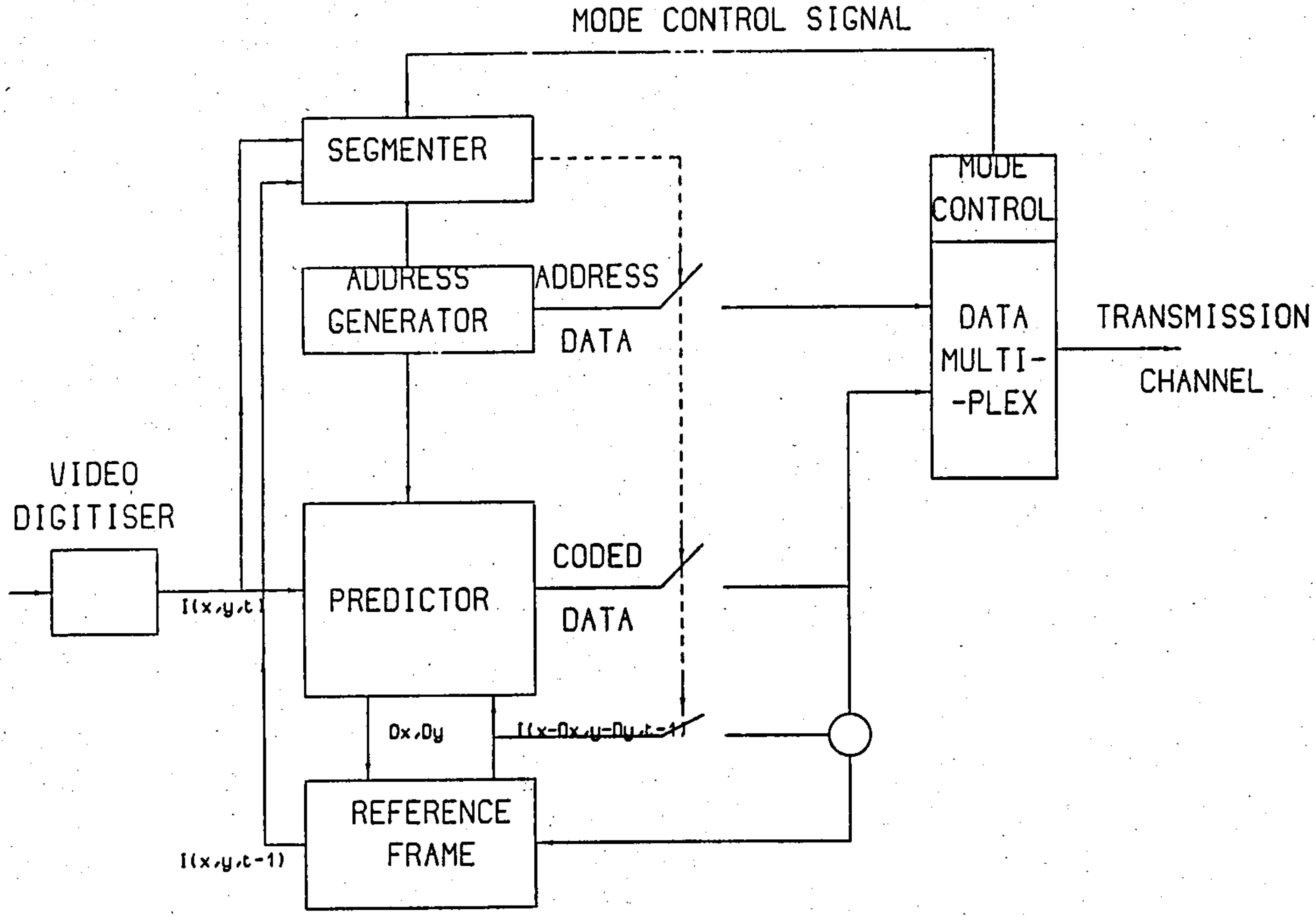


Figure 2.1 : Conditional replenishment coder

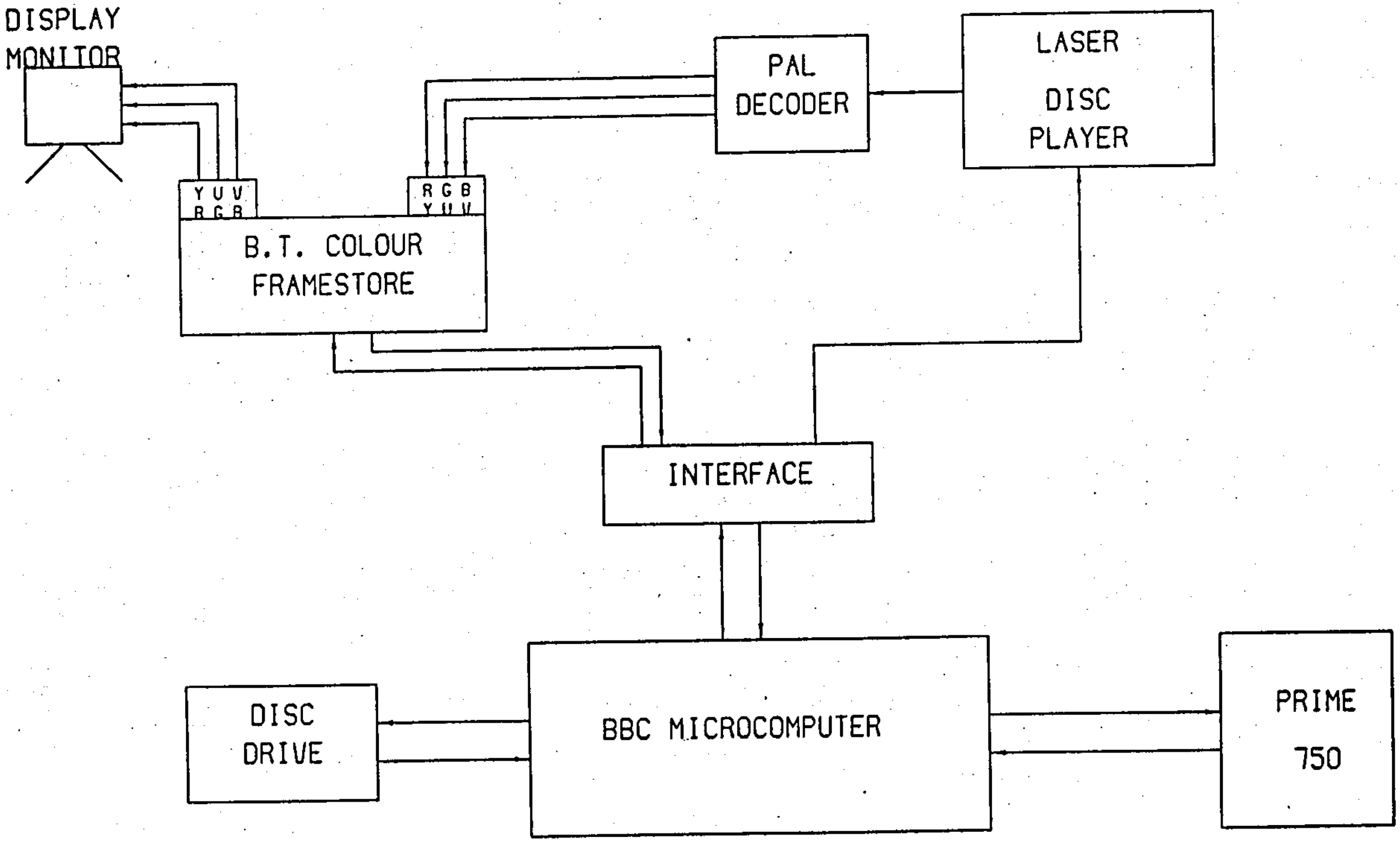


Figure 2.2 : Simulation system setup

Phase Alternating Lines(PAL) output, 25 frames per second, which is decoded and transformed into the luminance and chrominance signals. Its remote control input is interfaced to the BBC microcomputer 1MHz. bus; enabling the BBC microcomputer to step through the image sequence one frame at a time. The images from the laser disc player are then captured and stored in a colour framestore, consisting of 3 British Telecom R16.4.2. framestores. The colour framestore allocates 6 bits to luminance(L) and 4 bits to each chrominance(U,V). The data bus of the framestore is also interfaced to the BBC microcomputer 1 MHz. bus. Circuit details of the interface board and control commands for the laser disc are found in appendix B.

The BBC microcomputer is also used for transferring digitised video data between the framestore and the PRIME 750 minicomputer, located at the University Computer Centre. Two sets of image sequences:

(a) A1- Head and shoulder with detailed background, ten frames
(b) A2 - Half body image, four frames

were used in the interframe coding experiments. The digitised images were stored on the PRIME 750 computer and on BBC microcomputer floppy disc. Most of the simulation software was written on the PRIME because of the limited computing power of the BBC microcomputer.

2.3. Segmenter

The task of the segmenter is to divide the picture into those areas which have changed significantly, known as moving areas, and those which have not changed significantly, known as non-moving areas. Changes in luminance due to noise in the input signal should be suppressed to avoid unnecessary data transmission. Segmenting techniques can be classified into pixel and block segmenters. The pixel segmenters classify individual pixels as either moving area or non-moving area pixels, while block segmenters classify blocks, n by m pixels, as either moving area or non-moving area blocks.

2.3.1. Pixel segmenters

Two pixel segmenters were investigated:-

(a) Individual frame difference(FDIF)
The simplest method of segmenting the picture into moving and non-moving areas is to use the individual pixel frame-to-frame difference(FDIF), i.e.

$$FDIF(x,y)= I(x,y,t)-I(x,y,t-\tau)$$

where 'I(x,y,t)' represents the luminance at spatial coordinate (x,y) and τ represents the frame interval. A pixel is classified as belonging to a moving area if its FDIF is above the specified threshold. The performance of this technique is critically dependent on the setting of the threshold value. Too high a value will cause the system to ignore actual changes resulting in a 'dirty window' effect, while too low a value will result in transmission of pixel changes due to noise. Figure 2.3 shows the FDIF magnitude distribution of the first two images of the image sequence A1. As indicated in the figure there is a sharp drop in the FDIF distribution curve at FDIF=4. Thus, two FDIF thresholds were investigated, 4 and 5. Figure 2.4(b) shows the moving area pixel, represented as white pixels, between the first two images of sequence A1 with FDIF threshold values of 4 and 5. With a threshold of 4 there are still a lot of pixels updated, not due to the movement of the subject. A threshold of 5 seems to have suppressed most FDIF changes due to noise.

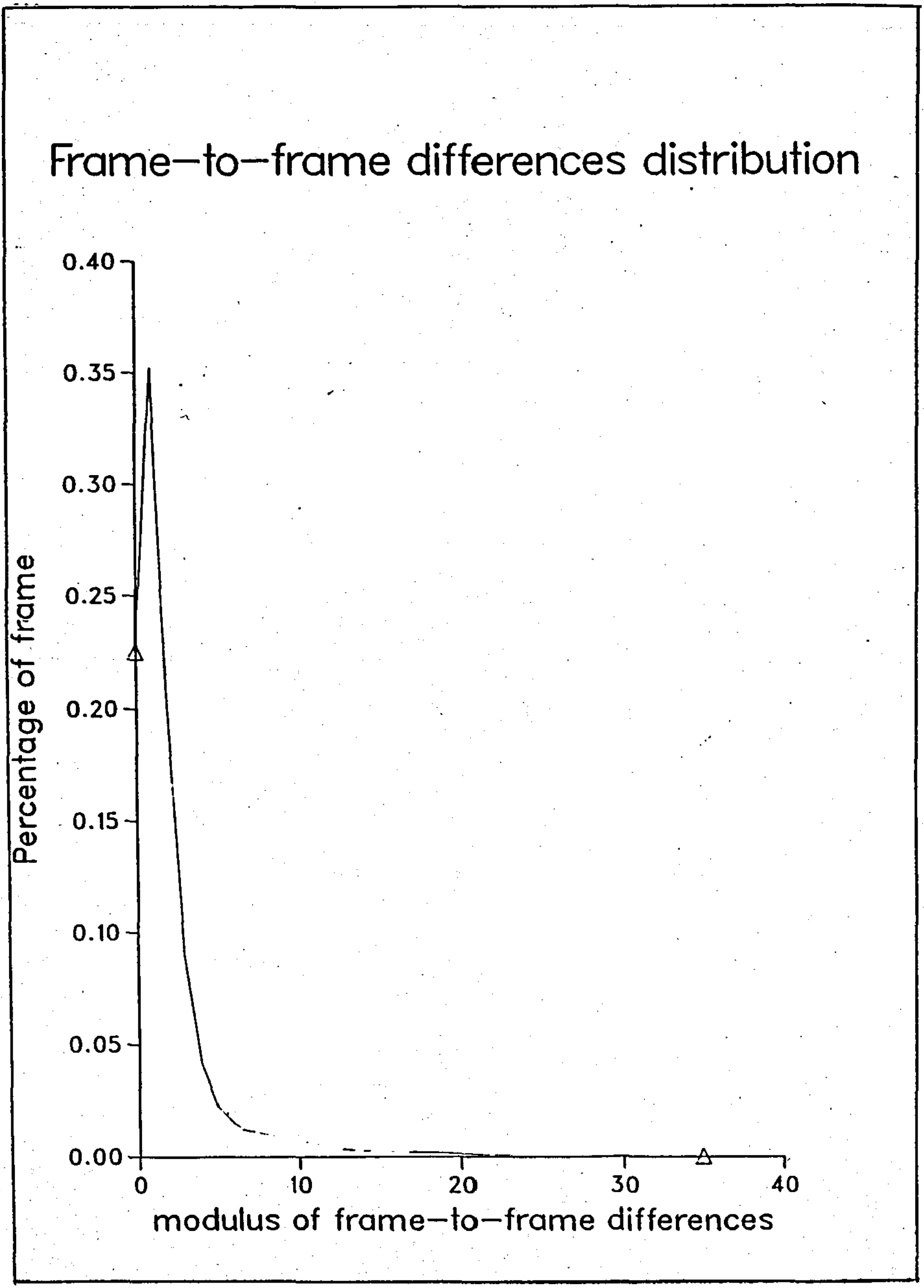


Figure 2.3 : Frame-to-frame difference distribution

FRAME 0

FRAME 1

(a) Original images

FDIF threshold = 4

FDIF threshold = 5

(b) Individual FDIF segmenter

FDIF threshold = 3

FDIF threshold = 4

(c) Cluster segmenter

Figure 2.4 : Pixel segmenter

(b) Cluster segmenter

The cluster segmenter uses the knowledge that most pixel FDIF changes due to movement occur in clusters [29], to classify the pixels as either moving area or non-moving area pixel. To do this, an additional condition is set - on top of the requirement that the pixel must have FDIF greater than the threshold - for segmenting the pixels which requires that at least one out of the two adjacent pixels along a scanline is above the threshold.

Figure 2.4(c) shows the moving area pixels, represented by white pixels, of the first two images of sequence A1 using a cluster segmenter with threshold of 3 and 4. As noted most of the moving area pixels are located around the edges of the speakers head and shoulders. This is a much more efficient way of segmenting the image than the individual frame difference segmenter.

2.3.2. Block segmenters

A block segmenter first divides the entire image into fixed blocks and each block is then individually tested and classified either as a moving area or non-moving area block. Two factors infuence the efficiency of the segmenter: block size and the segmenter change criteria.

Too large a block will produce a 'dirty window' effect on the image. Too small a block reduces the number of bits saved from the addressing overhead. An 8 by 8 pixel block has been found to be the best compromise.

Three threshold detection criteria were investigated :

(a) Maximum pixel FDIF in the block
(b) Average FDIF magnitude of the block
(c) the combination of both the maximum and FDIF of the block

Figure 2.5 shows the percentage of image updated when the maximum pixel FDIF criteria threshold is varied from 0 to 8 for frame 0 and frame 1 of image sequence A1. The figure shows a sharp drop in the percentage of frame updated when the threshold changes from 4 to 5. This could mean that most of the changes due to noise are suppressed and only changes due to the movement of the subject are updated. Figure 2.6(b) shows the areas of change with a maximum FDIF threshold value of 5.

Figure 2.7 shows the relationship between the percentage of frames updated and the threshold set for the average FDIF magnitude of the block. There is a sharp drop in the number of blocks updated when the average block threshold is raised from 1 to 2. Figure 2.6(c) shows the moving areas of the images using average block thresholds of 2 and 3.

Ave. Max.	0	1	2	3	4	5	6	7	8	9 & above
0	1	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0
2	11	2	0	0	0	0	0	0	0	0
3	82	129	0	0	0	0	0	0	0	0
4	42	226	0	0	0	0	0	0	0	0
5	5	128	4	0	0	0	0	0	0	0
6	48	16	1	0	0	0	0	0	0	0
7	0	25	25	1	0	0	0	0	0	0
8	0	0	11	7	3	0	0	0	0	0
9 & above	0	11	56	39	26	17	6	5	9	24

Table 2.1 : Block Average and Maximum FDIF distribution. Frame 0 and 1 of image sequence A1.

The third threshold criterion has been previously used by Kazuyuki Matsui [57]. Table 2.1 shows the number of blocks which satisfy both criteria. It is noted that only a

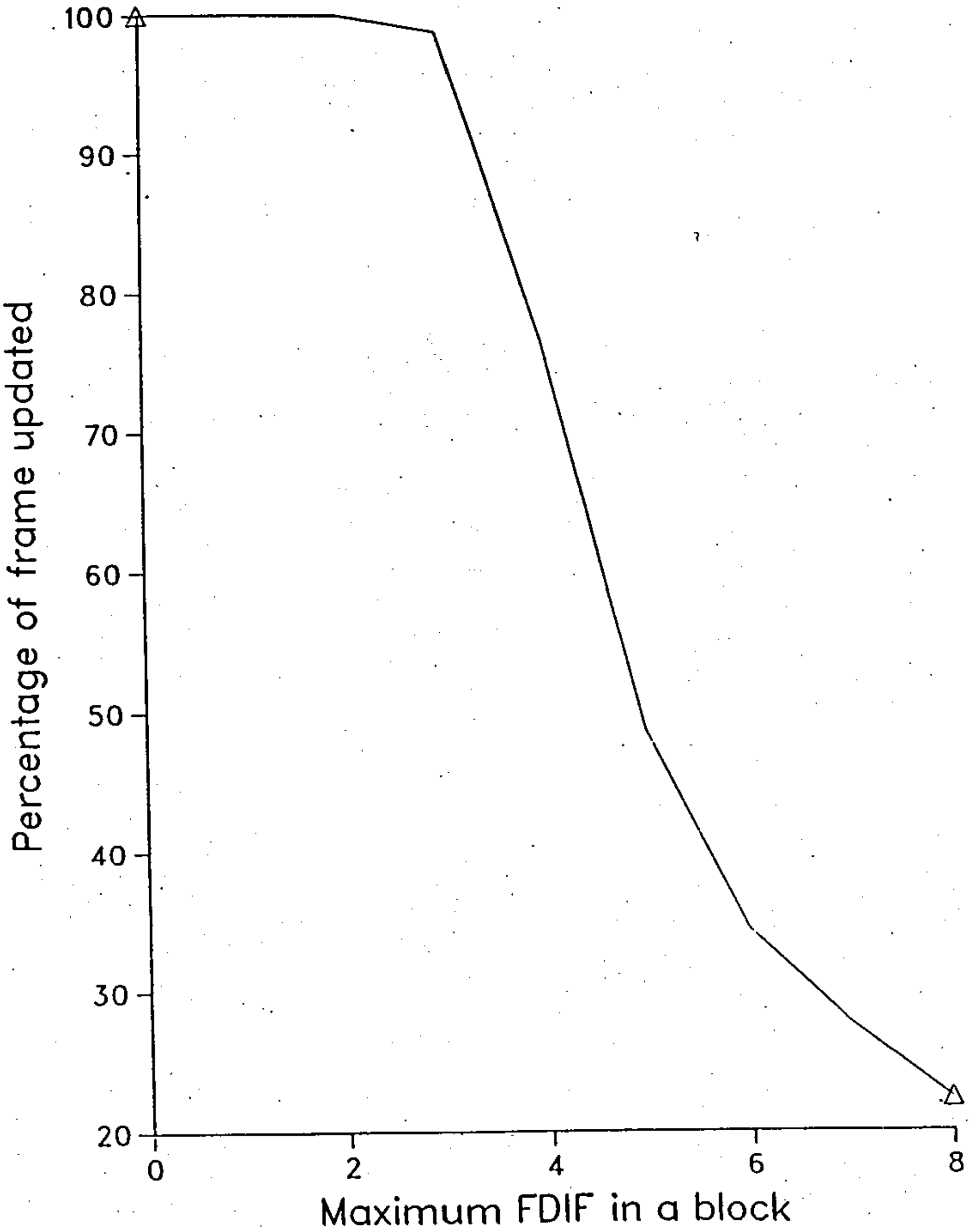
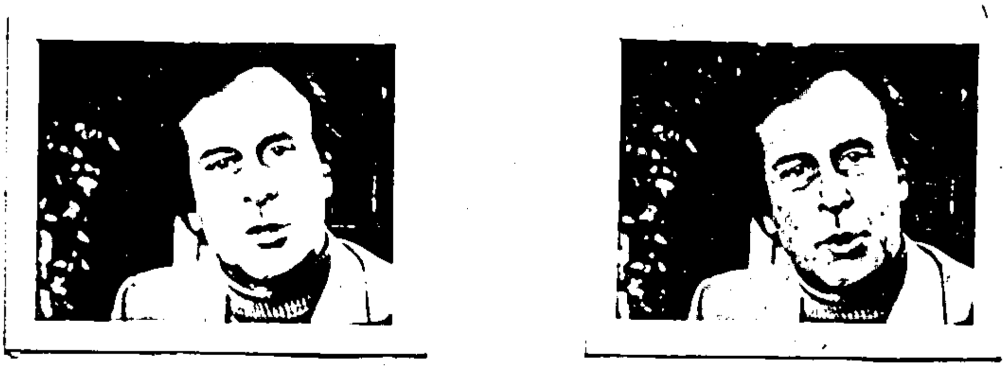


Figure 2.5 : Percentage of frame updated Vs

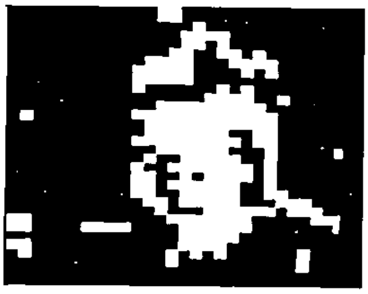
Maximum FDIF in a block.



FRAME 0

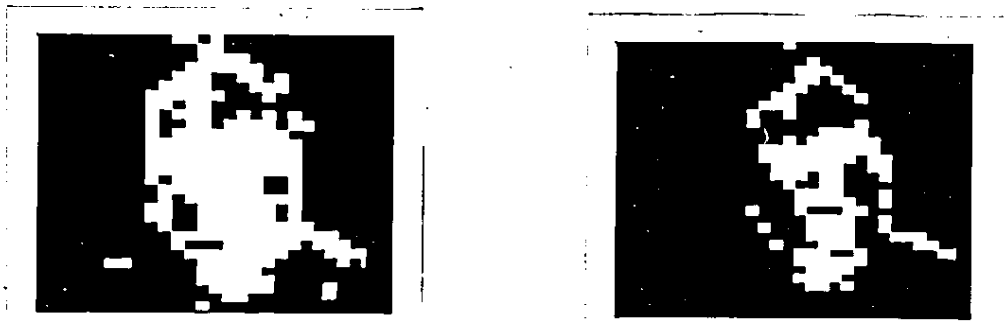
FRAME 1

(a) Original images



(b) Maximum FDIF block segmenter

threshold = 5



Threshold = 2

Threshold = 3

(c) Average FDIF block segmenter

Figure 2.6 : Block segmenter

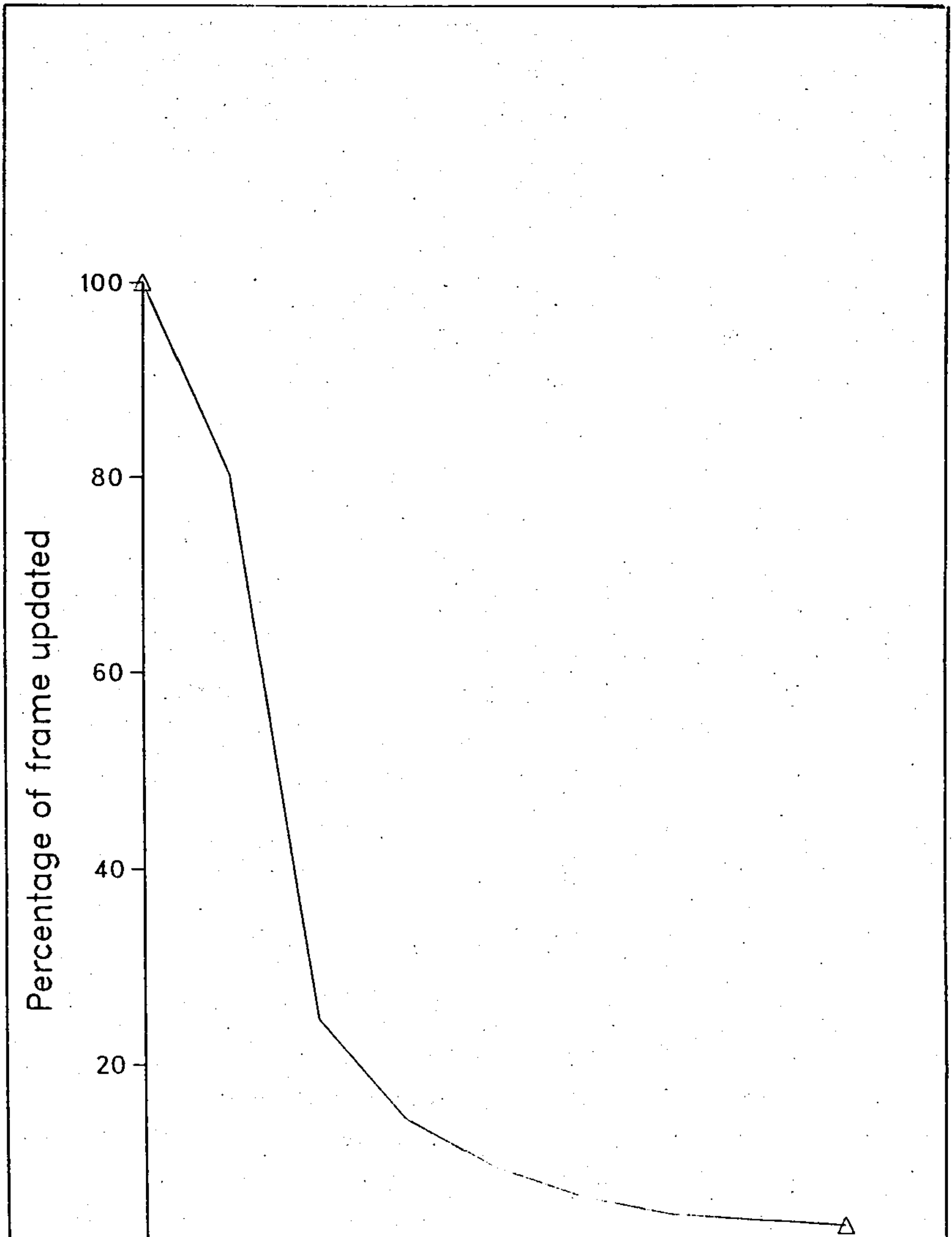


Figure 2.7 : Percentage of frame updated Vs

Average FDIF of a block.

small percentage of blocks would have maximum FDIF, of greater than 6, when the average FDIF of the block is less then 3.

2.4. Data addressing

Due to the fact that with Conditional replenishment coding, only moving areas pixel data are transmitted, data addressing is required to inform the receiver station as to the location of the pixel on the screen. Two commonly used addressing techniques in conditional replenishment coders are:

a) Cluster addressing [17]
b) Runlength addressing [62]

In cluster addressing the moving area pixels along a scanline are transmitted as clusters starting with the x and y addresses, followed by the predictive errors of each pixel in the cluster in the format shown below.

<Xadd><Yadd><Pred.error>..<PreD. error><stop
code><Xadd><Yadd>...

Figure 2.8: Cluster addressing format

The stop oode is necessary to indicate the end of the cluster. Additional bit saving can be achieved by transmitting a special code to indicate the beginning of a scanline. Thus, each cluster would be transmitted with its x coordinate only. Another method of reducing the bit rate is to coalesce cluster runs, in the same scan line which are separated by less than 'K' pixels, into a single cluster. This technique is known as 'bridging'. The value of 'K' can be obtained by dividing the number of bits in the stop code by the average word length.

Experiments by Haskell [31] have shown that the positions of clusters are correlated along a scanline and between the lines. Thus, rather than addressing a cluster with respect to the beginning of a scanline it could be addressed with respect to a corresponding cluster in the previous line, gives additional saving in bit rate. This form of addressing is known as Differential addressing.

Runlength addressing is very similar to the cluster addressing. The main difference is that while the previous method transmits only moving area pixels the runlength addressing methods transmits the entire picture as a one dimensional run. The format of the transmission is as shown below.

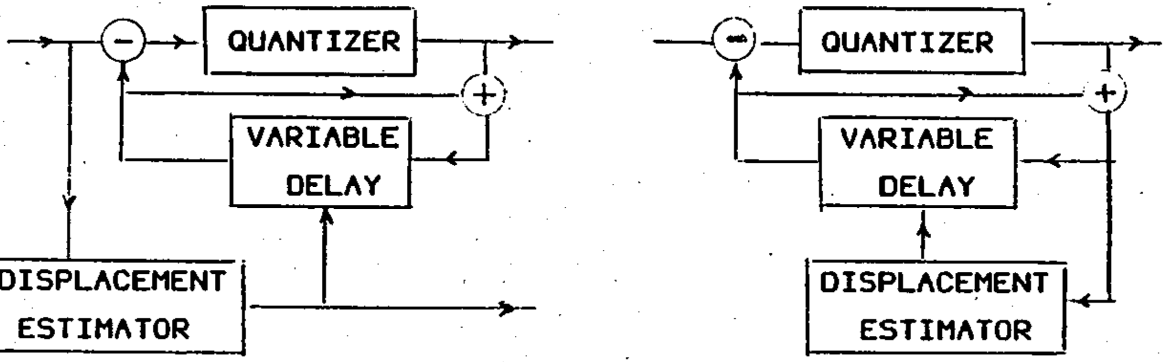
<runtype><runlength><data,data....>	moving area run
<runtype><runlength>	non-moving area run

Figure 2.9: Runlength addressing format

At the start of each run are the type and data length fields. The type field defines the data type, either moving area or non-moving area, of the run. The length of the run is specified in the data length field. If the area is a moving area, the prediction error is stored in the data field. No data field exists if the data type is for non-moving areas.

2.5. Motion compensated predictor

It has long been recognised that using the pixel in the same spatial position in the previous frame, $I(x,y,t-\tau)$, to predict the moving object pixel, $I(x,y,t)$, is not a good predictor when the object has moved. The best prediction of the pixel would be the pixel on the same position of the moving object in the previous frame, i.e. if Dx and Dy are the displacement estimate of $I(x,y,t)$ then its predicted value would be $I(x-Dx,y-Dy,t-\tau)$. Interpolation techniques are used when the displacement estimate is non-integer. This form of coding is known as motion compensated predictive coding. Prabhu [67] found that it was sufficient to use only luminance signal to find the displacement in colour images. The same displacement is then used for coding the chrominance signal.



(a) (b)

Figure 2.10: Configuration of displacment compensated coder [37]. (a) forward action (b) backward action

Figure 2.10 shows the block diagrams of the two types of displacment compensated predictor: forward action predictor and backward action predictor. As shown in the figure, the predictor is a variable delay determined by the displacement estimator. Thus, the displacement estimator (which will be discussed in detail in chapter 3) lies at the heart of the predictor. In the case of the forward action predictor, the displacement estimate vector is transmitted together with the prediction error.

2.6. Variable wordlength coding

The first order statistical redundancy of the predictive error data can be exploited by coding it with a variable wordlength(VWL) coder[22]. The principle of the VWL is simple. It assigns shorter wordlengths to more probable output levels and longer wordlengths to less probable output levels.

There are two well known variable wordlength coders, the Shannon-Fano [27] code and Huffman [33] code. For both methods, the first step is to calculate the image histogram from which the probability of occurrance of each possible output level is determined. Next, a binary tree is constructed with the output levels and their associated probabilities as leaves. The two branches from each node are arbitrarily labelled 0 and 1, and the code word for each output level is formed by concatenating these labels from the root to the appropriate leaf as shown in figure 2.11.

The entropy of the data, represents the lower limits in terms of bits required to encode it. The entropy of the data is defined as

$$E = - \sum_{i=0}^{=N} P(i)\log(P(i))$$

where P(i) is the probability of value 'i' occuring. Given a variable word length code table the average word length is given by

$$L = \sum_{i=0}^{=N} P(i).b(i)$$

where b(i) is the number of bits assigned to output level 'i'.

2.7. Multimode coders[24]

Conditional replenishment coders produce a fluctuating bit rate, depending on the number of changed pixels between the

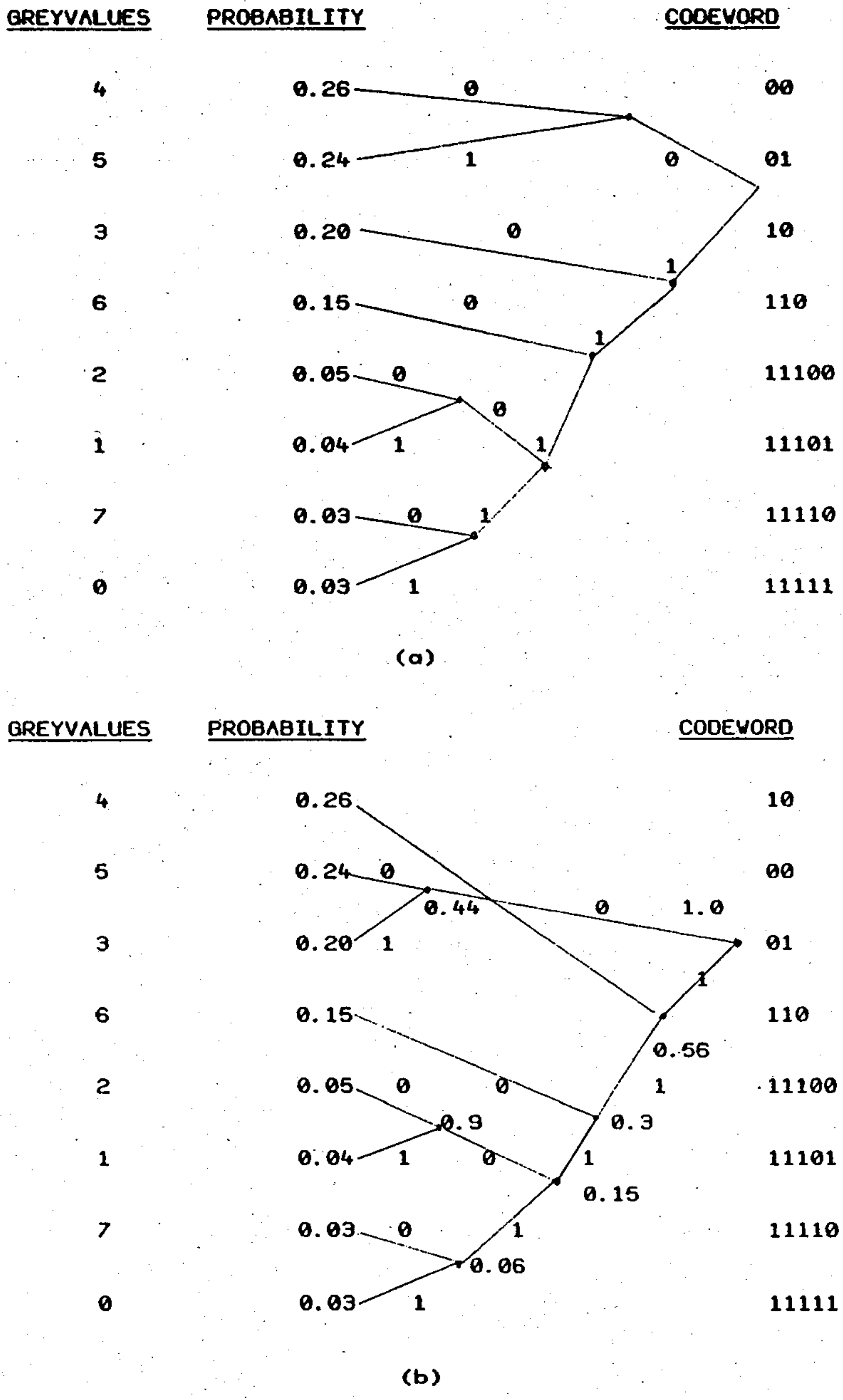


Figure 2.13: Variable wordlength coding. (a) Shannon-Fano coding (b) Huffman coding

two frames. A buffer is thus required to smooth the flow of information if the data is transmitted over a fixed-rate channel. The size of the buffer is limited by the perceptual effect of delay, a maximum of 0.3 seconds [29]. Due to the limited buffer size, data may overflow in the buffer during periods of high activity between frames. To avoid data overflow, it is necessary to adapt the coder mode of operation with the amount of motion, providing for a gradual and graceful degradation of quality as the amount of activity increases.

The criteria most frequently used for switching between the coding modes is the buffer memory occupancy. Usually a hysteresis is built into the switching process to prevent the coder from oscillating between modes. Figure 2.12 [30] shows an example of using the buffer occupancy for switching between the different coding modes.

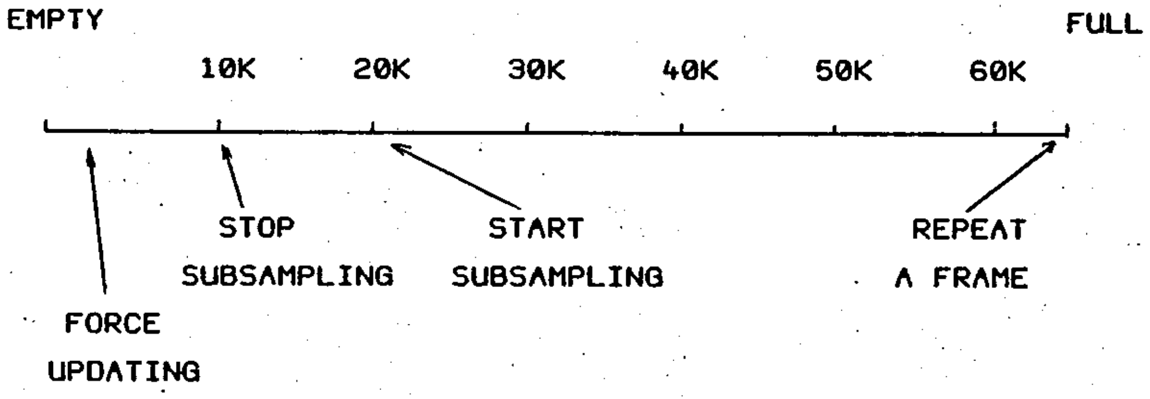


Figure 2.12: Coding mode controlled by buffer occupency

The different coding modes can be divided into three categories according to the buffer occupancy: underflow (force updating), normal and overflow mode (subsampling and repeat a frame). The main mode of operation, normal mode, is designed for "typical" image statistics. It gives full available resolution and best picture quality. The 'underflow mode' , is required to ensure that the buffer does not empty by transmitting a few lines of image data when the buffer occupancy falls below a set value. The subsampling mode and the repeat frame mode are referred to as 'overflow modes', and have been invoked successively in periods of increased picture activity as shown in figure 2.13. A few lines of image data are unconditionally undated during each frame period to prevent propagation of channel errors.

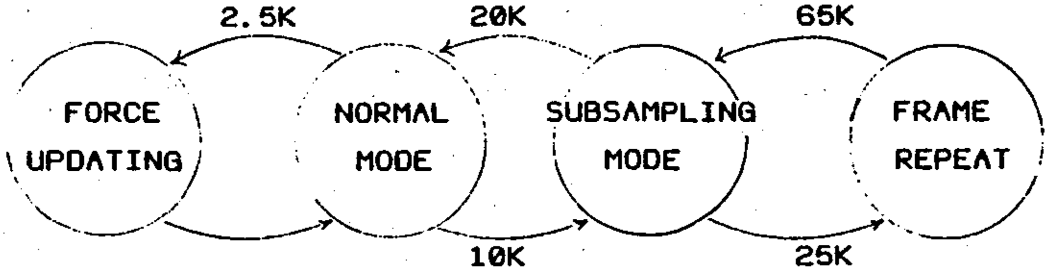


Figure 2.13 : Coding state diagram

An additional factor which needs to be considered when transmitting data over a local area network is network traffic. This is because the transmission channel bit rate varies according to the amount of traffic on the network. The criteria used for switching from one coding mode to another by the image station developed at Loughborough University is based on two parameters: frame update rate and percentage of change between two successive frames. The latter parameter provides information about the image activity. The network traffic can be measured by dividing the amount of image data transferred by the frame update time.

Unlike fixed bit rate channels, where each line is allocated to a specific user, the bandwidth of the LAN is shared by all the systems on the network. Thus, the bandwidth which one transmitter does not use will be used by other systems on the network. This means that it is not necessary to have an 'underflow' coding mode.

Data congestion on the bridge, the system linking two LAN's, may arise when transmitting images over a wide area network. This is due to the fact that the network traffic on each site is independent of each other. If the receiver station network has a much higher traffic load than the transmitter station network, data is loaded into the bridge, by the transmitter, faster than they are being read out, by the receiver, from the bridge. This will cause the bridge to become congested after a period of high activity which will eventually lead to the discarding of data by the bridge. The receiver station can detect that the data transfer is limited by checking the frame update rate. If the frame update rate, at the receiver, is higher than specified it would indicate that the data transfer is limited by either the transmitter LAN or receiver LAN. Thus, the receiver station should be used to control the coding mode of the transmitter station when image data is transferred over a wide area network.

2.8. Discussion

Conditional Replenishment coding is an efficient form of interframe coding. The amount of saving in bit-rate depends on the number of changed pixels between successive frames, which, in normal situations, is about 5% of the total image when using individual FDIF pixel segmenting. One major problem with conditional replenishment coding is the high fluctuation in bit rate. To overcome this problem the data needs to be buffered and multimode coding is used to ensure that the buffer does not overflow. The parameter used for switching between the coding modes in most existing systems is buffer occupancy. A better criterion which was proposed for coding control when transmitting image data over a network is based on the computer network traffic and the percentage of frame updated. It was also proposed that the coding mode should be controlled by the receiver station.

CHAPTER THREE

DISPLACEMENT ESTIMATION ALGORITHMS

3.1. Introduction

One method of reducing the bit-rate of Conditional replenishment coders, described in the previous chapter, is by displacement compensation. The displacement coding technique estimates the translation of moving objects in the picture and uses it to code the moving area pixels. An essential part of this coding technique is the displacement estimation algorithm. Displacement estimation algorithms can be broadly classified into two main groups [73]: pixel recursive algorithms and block matching algorithms. Pixel recursive algorithms estimate pixel displacement on a pixel by pixel basis. Block matching algorithms estimate the displacement of a block of pixels are at a time and assume that the motion is uniform within the block.

3.2. Pixel recursive algorithms

This group of algorithms uses the optimisation methods derived from the expansion of the Taylor series. The displacement estimate in the horizontal and vertical directions are represented by Dx and Dy respectively. A displaced frame difference(DFD) is defined as

$$DFD(x,y,Dx,Dy) = I(x,y,t)-I(x-Dx,y-Dy,t-\tau) \quad ....<3.1>$$

If there is no error in the displacement estimate, i.e. the estimated displacement is equivalent to the actual displacement of the pixel, the DFD should be zero, since ideally the pixel values are identical. To ensure that the function is positive definite the square of the

DFD(x,y,Dx,Dy) is minimised by the algorithms. The pixel recursive algorithms calculate the new displacement estimate using the previously estimated displacement value as shown in the equation below:

$$D^{i} = D^{i-1} - (\text{correction factor}) \qquad \ldots.\langle 3.2\rangle$$

where D^{i} and D^{i-1} represent the present displacement estimates and the previous displacement estimate. The difference between different pixel recursive algorithms is used to obtain the 'correction factor'.

For a Conditional replenishment coder which uses the pixel recursive algorithm all the pixels in a frame are first classified into moving and non-moving area pixels. The pixels are classified using the cluster segmenter with a frame-to-frame difference magnitude (FDIF) threshold of 4. The displacement estimation calculation is then carried out on the moving area pixels progressing along the scan direction and is disabled in the non-moving areas of the picture. According to the result of the prediction error, DFD, the moving area pixels are further divided into two groups:

(i) compensatable, where the displacement estimation has adequatly compensated for the motion. No update information need be sent.

(ii) non-compensatable pixels which require transmission of predictive error, i.e. DFD value.

Figure 3.1 shows a flowchart of the way the pixels in a frame are classified.

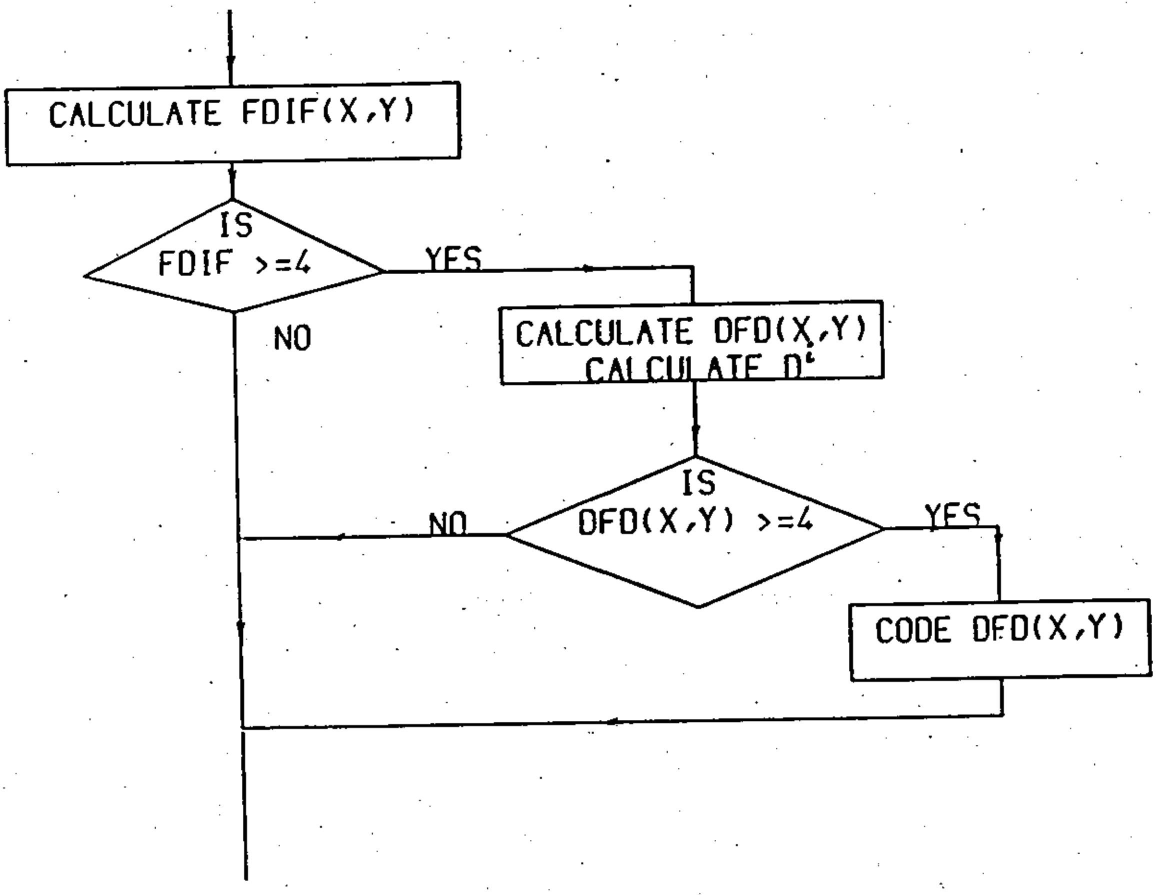


Figure 3.1 : Pixel recursive segmenter

The intensity gradient in the x and y axes, EDIF and LDIF respectively, are approximated as follows (refer to figure 3.2):

$$\begin{aligned} EDIF &= (D - F)/2 \\ LDIF &= (B - H)/2 \end{aligned} \qquad \ldots\ldots\langle 3.3\rangle$$

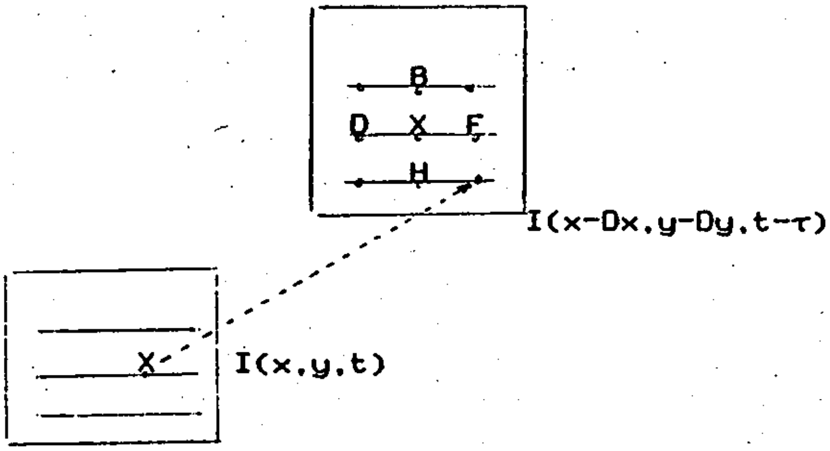


Figure 3.2 : Intensity gradient calculation

For non-integer displacements, the luminance of the pixel in the previous frame is obtained by linear interpolation. Refer to figure 3.3 for the discussion on interpolation below.

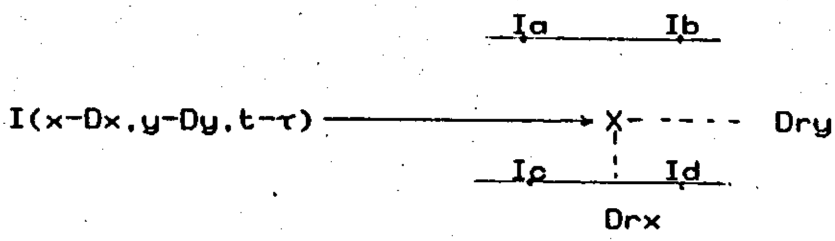


Figure 3.3 : Interpolation

The location of the pixel in the previous frame is first rounded to the nearest integer. This gives the location of the nearest neighbor pixel, Id. By testing the sign of the rounding errors the locations of two other neighbouring pixels are located. The following formula is then used for interpolation:

$$I = Id + Drx(Ic - Id) + Dry(Ib - Id) \qquad ....<3.4>$$

where I represents the luminance and Drx and Dry are the rounding errors in the x and y directions respectively. The resolution of the interpolation is limited to 1/8 pel/field.

3.2.1. Steepest Descent method

The first pixel recursive algorithm was suggested by Netravali [62]. This algorithm uses the steepest descent method [82] to find the minimum value of the square of the DFD. It uses the first order differential to locate the direction of steepest descent. The algorithm is as shown below.

$$\begin{aligned} D^i &= D^{i-1} - \varepsilon/2.\nabla\,[DFD(X,D^{i-1})]^2 \\ &= D^{i-1} - \varepsilon.DFD(X,D^{i-1}).\nabla\,DFD(X,D^{i-1}) \end{aligned} \qquad ......<3.5>$$

where D^i and D^{i-1} are the displacement estimates of the present and previous pixels respectively.

$$\nabla\,(DFD(X,D^{i-1})) = \nabla\,I(X - D^{i-1},t-\tau) \qquad ....<3.6>$$

This gives

$$D^i = D^{i-1} - \varepsilon.DFD(X,D^{i-1}).\nabla\,I(X-D^{i-1},t-\tau) \qquad ....<3.7>$$

ε is a positive scalar constant and ∇ is the spatial gradient of the image intensity. Looking at the second component on the right hand side of the equation above, the vector which is added to the old displacement estimate is parallel to the direction of the spatial gradient of the image intensity and its magnitude is proportional to the DFD. The setting of ε is very important as a small value of ε will result in slow convergence of the the algorithm while too large a value will result in a noisy estimator.

A simplified form of equation <3.6> is suggested to reduce the computational overhead. The new algorithm is given by the equation below:

$$D^i = D^{i-1} - \varepsilon \, \mathrm{sign}(DFD(X,D^{i-1})). \, \mathrm{sign}(\nabla I(X-D^{i-1},t-\tau)) \quad \ldots .<3.8>$$

where

$$\mathrm{sign}(x) = 0, \text{ if } x=0$$
$$= x/|x|, \text{ otherwise}$$

3.2.2. Newton's method

D.R. Valker [77] suggested using the Newton method [82] to find the displacement estimate of the pixel. This method uses the second order differential to increase the rate of convergence. The method is as shown below:

$$D^i = D^{i-1} - \frac{2.\nabla DFD(x,y,D^{i-1})^2}{\nabla^2 DFD(x,y,D^{i-1})^2} \quad \ldots\ldots <3.9>$$

The numerator of the second component in the right hand side of the equation is the same as the second component in the steepest descent method above. The denominator of the equation can be evaluated as follows.

$$\begin{aligned}
\nabla^2[DFD(X,D^{i-1})]^2 &= \nabla[\nabla(DFD(X,D^{i-1})^2)] \\
&= \nabla[2.DFD(X,D^{i-1}).\nabla I(X-D^{i-1},t-1)] \\
&= 2.[\nabla DFD(X,D^{i-1}).\nabla I(X-D^{i-1},t-1) \\
&\qquad + DFD(X,D^{i-1}).\nabla^2 DFD(X,D^{i-1})] \\
&= 2.[(\nabla DFD(X,D^{i-1}))^2+\nabla^2 DFD(X,D^{i-1})]
\end{aligned}$$
$$\ldots\ldots <3.10>$$

Replacing $\nabla^2[DFD(X,D^{i-1})]$ and $\nabla[DFD(X,D^{i-1})]$ in in equation <3.9>,

$$D^i = D^{i-1} - \frac{[DFD(X,D^{i-1}).\nabla^2 I(X-D^{i-1})]}{[(\nabla I(X-D^{i-1},t-\tau)^2+(DFD(X,D^{i-1}).\nabla^2 DFD(X,D^{i-1})]} \quad ......<3.11>$$

A simplified form of this equation has been proposed by D.R. Walker [77]. It assumes that $\nabla^2 DFD(X,D^{i-1}) = 0$, so that the equation above becomes

$$D^i = D^{i-1} - \frac{[DFD(X,D^{i-1}).\nabla^2 DFD(X,D^{i-1})]}{[\nabla I(X-D^{i-1},t-\tau)]^2} \quad ...<3.12>$$

3.3. Block matching algorithms

Another method of estimating the displacement of pixels is to use matching techniques which try to find the best matched block of pixels in the previous frame for the block of pixels in the present frame. Initially the present frame is divided into a fixed grid of blocks of N by M pixels. Each block of pixels is then compared with blocks of pixels within a search area (S) of the previous frame to locate the best match as shown in figure 3.4 'p' represents the maximum displacement

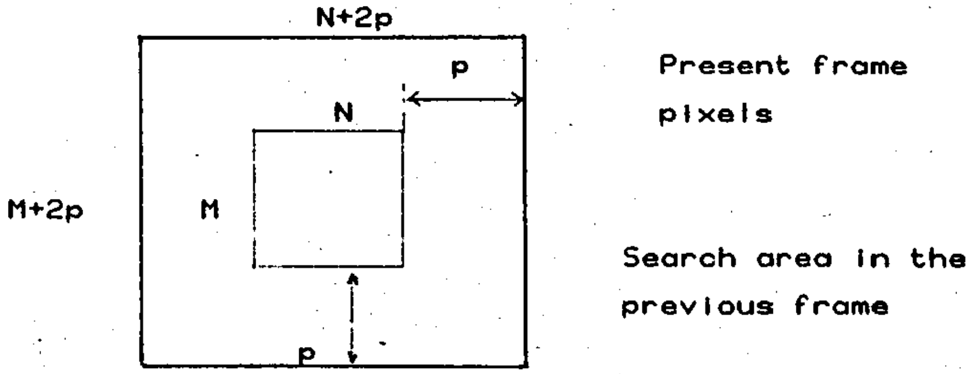


Figure 3.4 : Geometry for manipulation of (M x N) in the present frame.

permissible. The best match is based on some specified criteria such as minimum mean square error(MSE) or minimum mean absolute error(MAE) as defined below:

(a) Mean Square error (MSE)

$$M(Dx,Dy) = \sum_{n=1}^{=N} \sum_{m=1}^{=M} (I(x,y,t)*I(x-Dx,y-Dy,t-\tau))^2/(N*M) \qquad \ldots\ldots\langle 3.13\rangle$$

(b) Mean Absolute error (MAE)

$$M(Dx,Dy) = \sum_{n=1}^{=N} \sum_{m=1}^{=M} (I(x,y,t)*I(x-Dx,y-Dy,t-^{1}))/(N*M) \qquad \ldots\ldots\langle 3.14\rangle$$

where Dx and Dy represent the horizontal and vertical displacement of the block in the present frame from its spatial location (x,y) and τ is the time interval between two successive frames. When estimating the displacement vector using block displacement estimation algorithms we assume that all the pixels within the block have a uniform motion and that the motion is translational or can be approximated by piecewise translation. Once the displacement vector has been determined, all the pixels in the block in the present frame are predicted from their corresponding pixels in the best matching block in the previous frame and their prediction error transmitted to the receiver station.

The computational overhead of estimating the displacement vector by 'brute force', i.e. evaluating the response of all the possible locations, is extrememly high, e.g. if p= 4 a total of 81 points are evaluated per block. Search algorithms [23,80,82,19] are used to reduce the average number of points evaluated per block which will result in reduced computation overhead.

In searching for the best matched block one has to decide either to search intensively around the present area or to continue exploring so as to provide useful information for locating the minima. Thus, throughout the search we must be continually deciding either to carry out an intensive search, in which case one assumes that it is in the vicinity of the minima or to explore further.

Another factor that helps to reduce the average number of points evaluated per block is the high degree of continuity in motion from frame to frame [79]. Thus, by positioning the initial search in the vicinity of the displacement vector of its adjacent block we can reduce the average number of points evaluated per block even further.

The following conventions are adopted for the discussion below:

(a) the displacement vector is known as Direction of minimum Distortion(DMD)

(b) the matching function, e.g. MSE,MAE, is known as distortion function

(c) the result of the matching function M(i,j) is known as the response value of the displacement (i,j)

3.3.2.1. 2D-Logarithmic search

The 2D-logarithmic search technique [41] is an extension of the one dimensional logarithmic search. The active search area, i.e. the area in which future tests or evaluations are to be carried out, is successively reduced after each search step. Each step consists of evaluating the response value of five locations which contain the centre of the area, and points between the centre and the four boundaries of the area along the axes passing through the centre. This is an exploratory search step and will continue until the centre of the active search area remains unaltered after a search step.

When this happens, i.e. centre of the active search area remains unaltered after a search step, the active search area is reduced to a 3 by 3 displacement window around the center of the present active search area. All nine locations are evaluated and the location corresponding to the minimum distortion is the DMD. The number of search points evaluated per block using the 2D-logarithmic search ranges from 13-19.

A flowchart of the 2D-logarithmic search is found in figure 3.5. The following conventions are used in the figure:

name	the location vector (Dx,Dy)
Rname	the response value of the location specified by 'name'
CTR	the centre of the search
DCHG	set = TRUE if the centre of search has changed since the last search step.

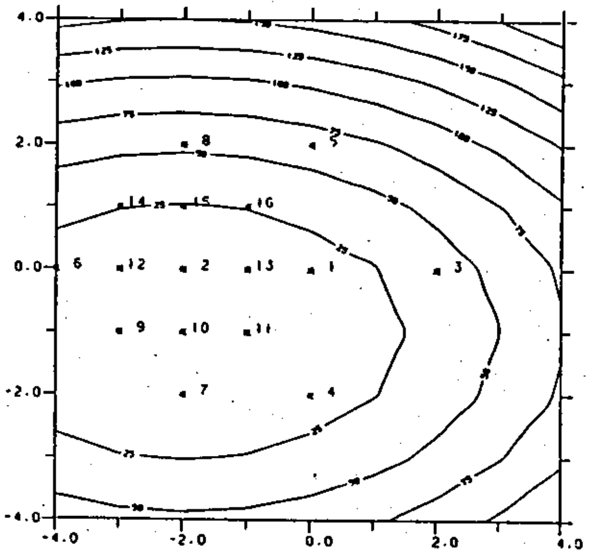


Figure 3.6 : 2D-logarithmic search.
DMD = (-2,-1)

Figure 3.6 shows an example of the 2D-logarithmic search. The numbers in the figure represent the order in which the points were evaluated. Initially five points: (0,0), (2,0),

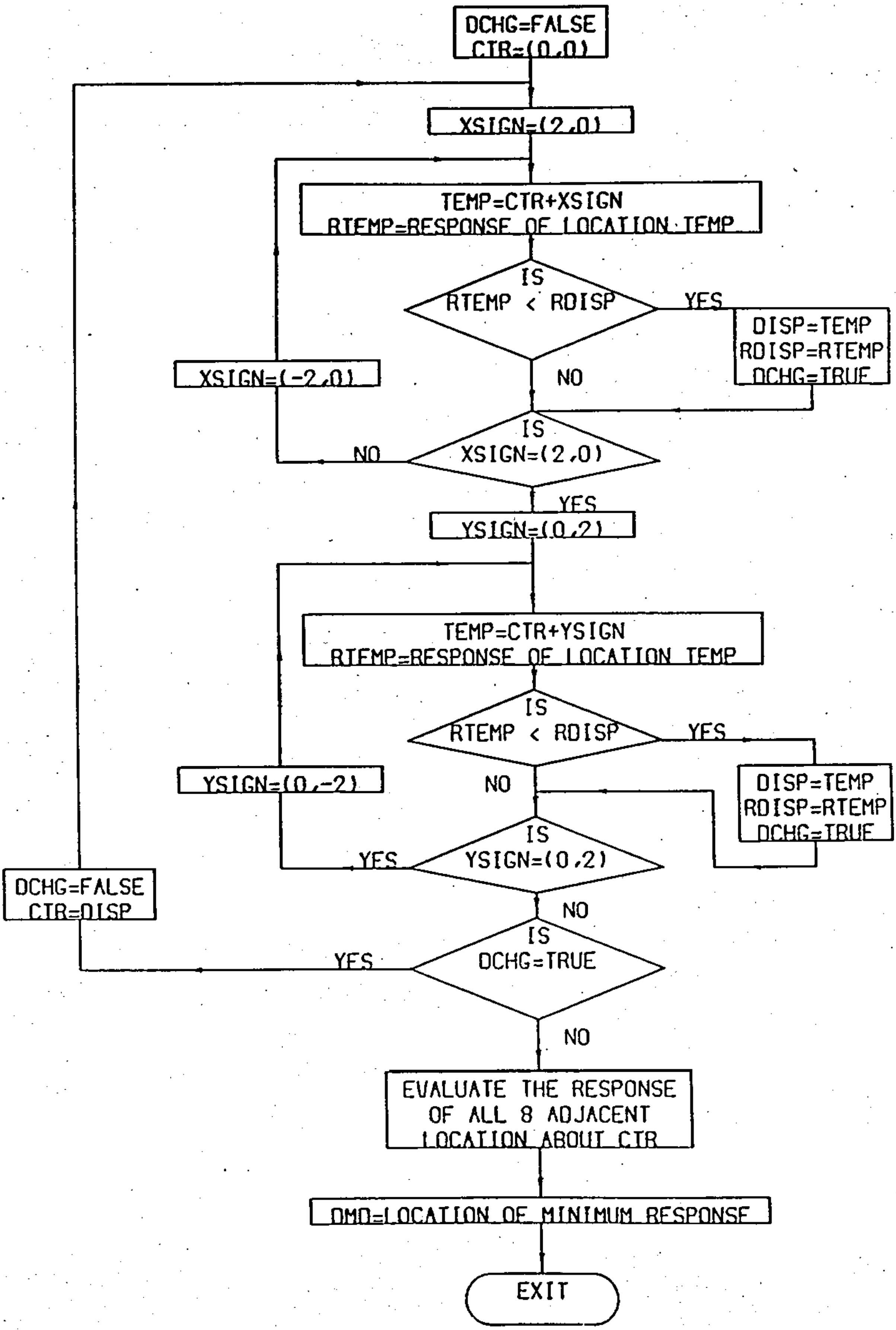


Figure 3.5 : 2D-logarithmic search flowchart

(-2,0), (0,2) and (0,-2) were evaluated. Location (-2,0) was found to give the least response value. Thus, the next search centre will be (-2,0). Locations (-4,0), (-2,2) and (-2,-2) were then evaluated. In this case location (-2,0) still had the least response value. This causes the algorithm to resort to an intensive search about (-2,0), where all the adjacent locations are evaluated. The DMD is then set to (-2,-1) which has the least response value of the intensive search step.

3.3.2.2. Univariate search

The univariate search algorithm [73] is one of the simplest search algorithms. The basic principal of the univariate search is as follows: each variable is minimised in turn, by adjusting each variable by a small predefined step until its minima is located. The minima of each variable is defined as the point lying between two other points with higher response values. The search is terminated if the minima of the two variables, Dx and Dy, are situated at the same location.

A flowchart of the univariate search is shown on figure 3.7. The following conventions are used in the figure:

name	the location vector (Dx,Dy), e.g. CTR, NEG, POS
Rname	the response value of the location specified by 'name'
DCHG	set = TRUE only during the first search along the Dx variable
x	is a scalar multiply

The univariate search requires between 5-23 displacement points to be evaluated for 'p' = 4. As would be expected, the least number of search points, 5, occurs when the minima lies at location (0,0) as shown in figure 3.8(a). The search becomes quite inefficient when the minima lies at

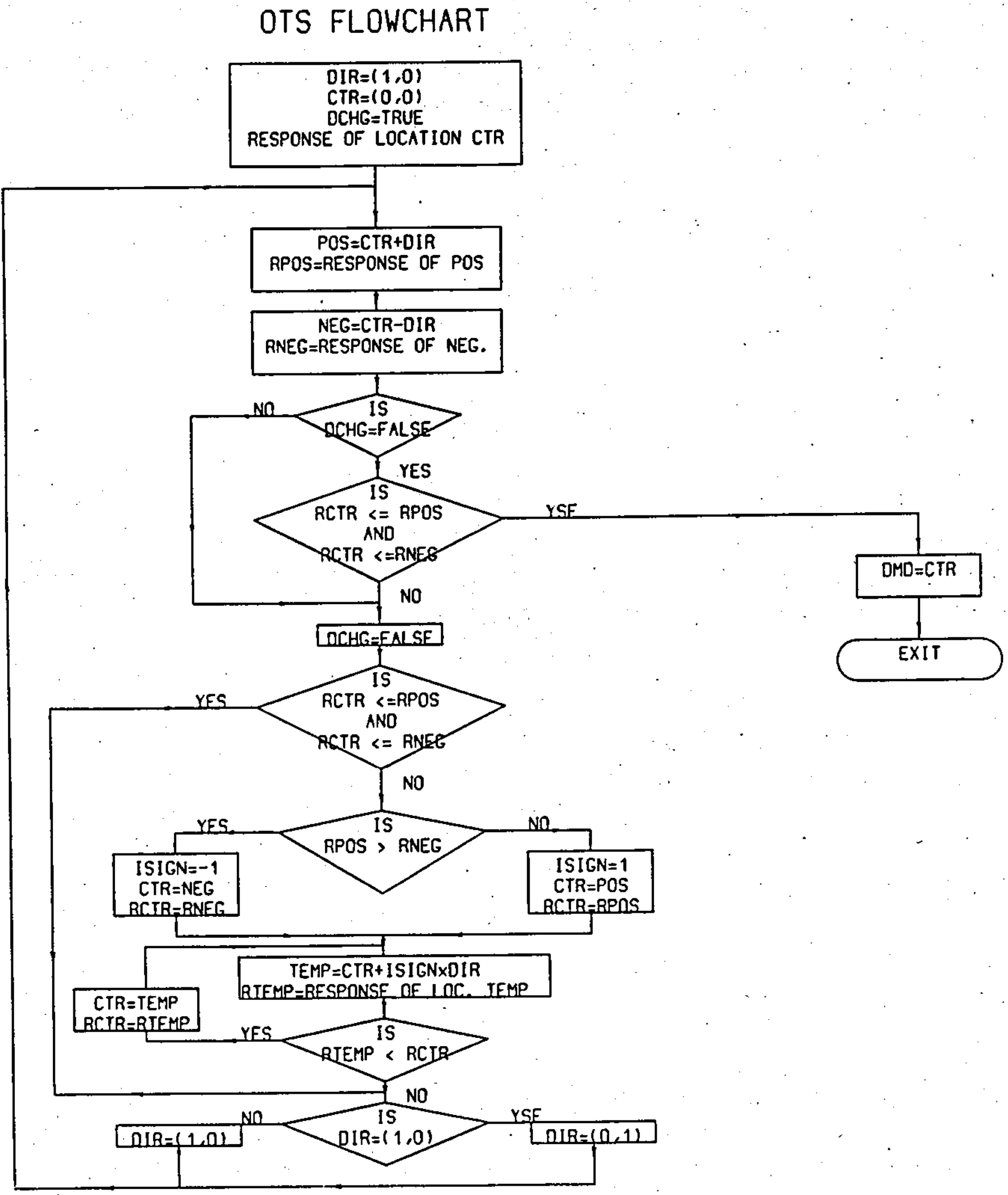


Figure 3.7 : Univariate search flowchart

a diagonal to the axis and the variables are closely related as shown in figure 3.8(b) where the response value is simulated as an ellipse.

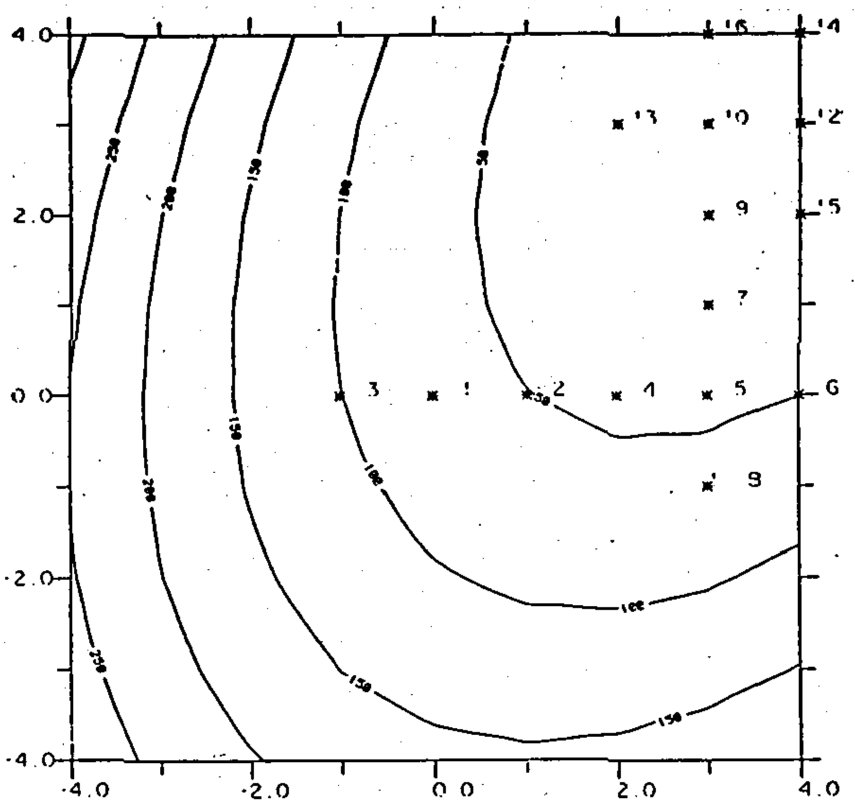
3.3.2.3. Simplex search

The simplex technique was first suggested by Spendley, Hext and Himsworth [72] for locating the maxima or minima of a plane. The simplex search techniques have the great advantage of not requiring the response function to be continuous and they approach the minima quite quickly. A simplex consists of a pattern with at least n+1 vertices, for an n-dimensional plane, e.g. a simplex for a two dimensional plane is a triangle.

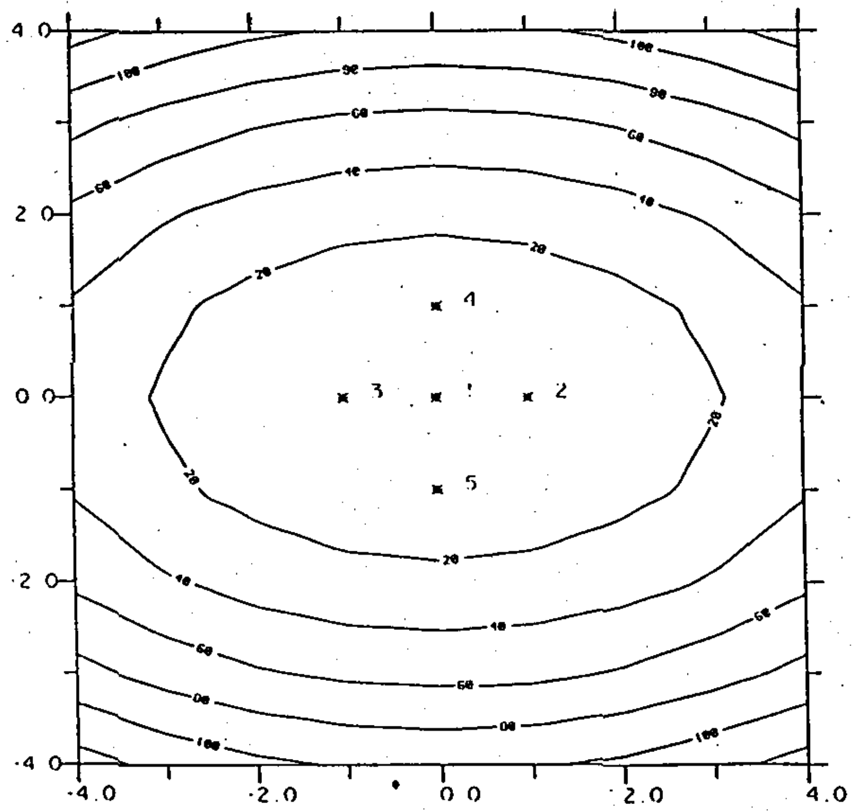
The essence of the simplex algorithm is as follows. The response value of each vertex of the simplex is evaluated and the vertex having the highest response value is replaced by a new point with a lower response value. This is done in such a way that 'the simplex adapts itself to the local landscape, and contracts on to the minimum' [61]. There are three types of movement to find the new point: reflection, expansion, and reduction. On top of these movements is the contraction movement which reduces the size of the simplex in all dimensions. Only two types of movement: reflection and contraction, were used in the simplex techniques discussed below.

The vertices of the simplex are labelled (V1,V2 and V3) according to their response values (R1, R2 and R3 respectively) such that R3 >= R2 >= R1. During the reflection movement of vertex V3, the vertex V3 is reflected about its reflection centre, rc. The coordinate of the reflection centre of V3 is calculated as follows:

$$rc = (V1 + V2) / 2 \quad ....<3.15>$$



(b) DMD = (4,4)



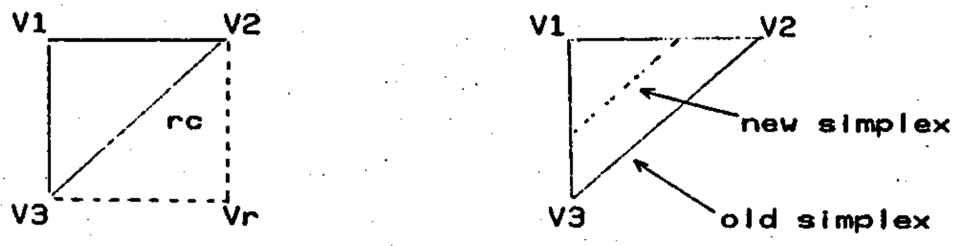
(a) DMD = (0,0)

Figure 3.8 : Univariate search examples

while the coordinates of the new point, known as reflected vertex (Vr), is obtained by the equation below:

$$Vr = V3 + 2 \times (rc - V3) \quad .....<3.16>$$

Figure 3.9(a) shows an example of the reflection movement.



(a) reflection (b) contraction

Figure 3.9 : Simplex movements

The contraction movement reduces the size of the simplex by a specified factor β. The simplex is contracted about the V1 vertex, i.e. V1 is maintained while the other two vertices are replaced by new locations. The vertices, other than V1, of the new simplex are obtained using the equation below:

$$Vi = V1 + (Vi - V1)/\beta \quad .....<3.17>$$
$$\text{for } i = 2,3$$

where β is the factor by which the size of the simplex is to be reduced as shown in figure 3.9(b). After a contraction movement the vertices of the new simplex are reordered according to their response values in the manner described previously.

Two factors influence the rate of convergence of the simplex search algorithms: size and orientation of the initial simplex and the rules for generating subsequent simplexes.

The rate of advance from the initial simplex is maximum when the direction of movement lies in one of the "preferred directions", i.e. the centre of the successive simplexes and the individual vertices as shown in figure 3.10. A problem might occur if the direction of the movement is perpendicular to one of the faces of the simplex, which immediately leads to a most unfavourable situation in which all the responses except one are equally high as shown in figure 3.11. This phenomenon is known as 'spiralling'.

To ensure that the DMD lies in one of the 'preferred directions' of the initial simplex, two vertices of the initial simplex are fixed at displacement location (0,0), (2,0) while the third vertex is set directly below whichever of the previous two vertices has a higher distortion value, e.g. if location (0,0) has a higher response value than location (2,0) the third vertex will be positioned at (0,-2). This method of generating the initial simplex is adopted for all the simplex techniques investigated below.

A new simplex search algorithm, the Basic simplex search, has been developed. Both the Basic simplex search and Spendley simplex search will be discussed below.

3.3.2.3.1. Basic simplex search

A new set of rules has been developed for locating the minima using the simplex search technique. Since we are trying to move as far away as possible from a high response area, we will reflect V3 through its reflection centre as given in equation <3.15>. If the response value VR (RR) located by equation <3.16>, is less than the response value of V2 (R2), then VR will replace V3. The vertices of the new simplex are then labelled according to their response values

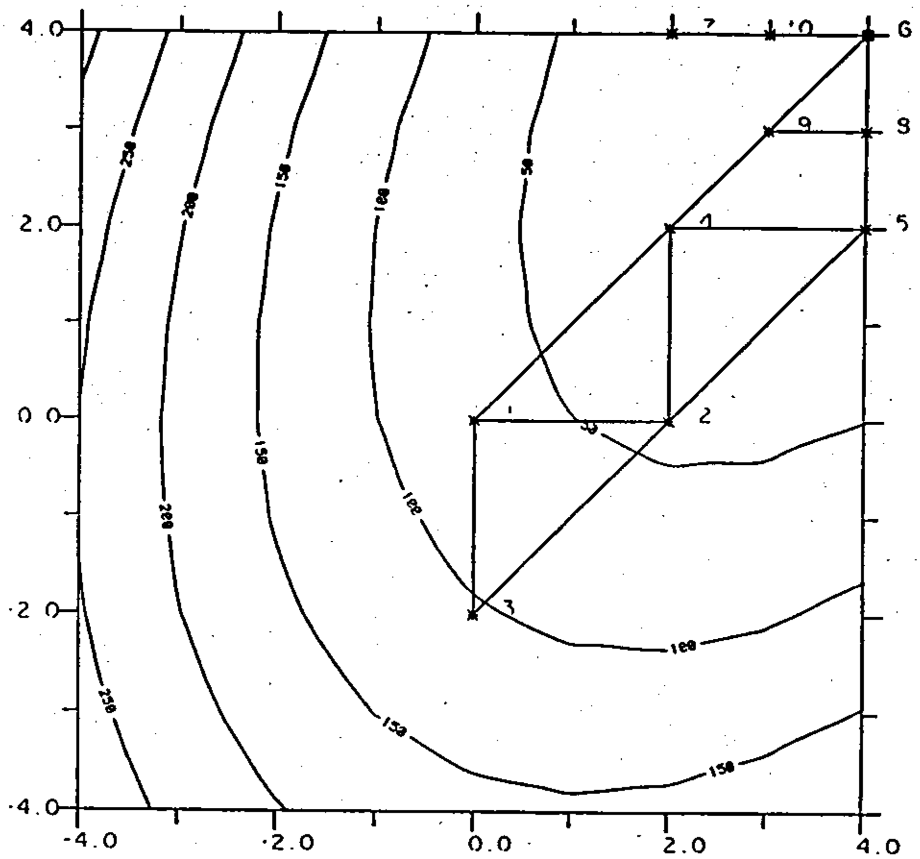


Figure 3.10 : Preferred direction movement

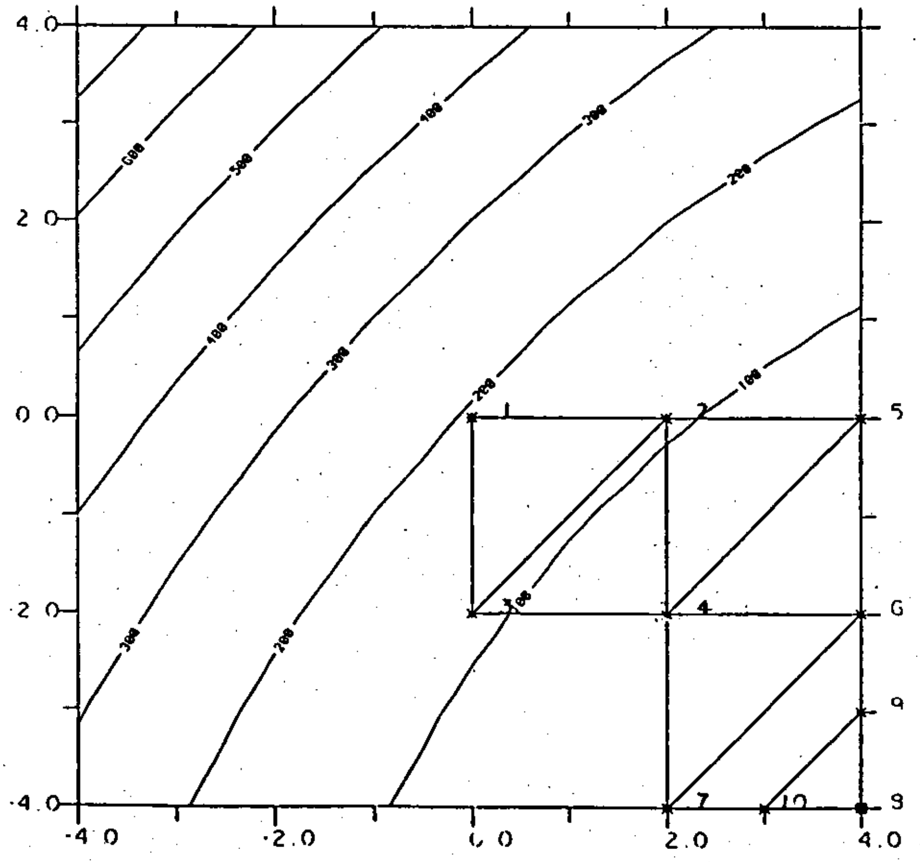


Figure 3:11 : Spiralling occurs at displacement location (2,-2)

as described previously and the reflection process continues with the new simplex. If RR >= R3 the old simplex will be maintained. It is then followed by a contraction movement where β is equal to 2, thus reducing the simplex vector size by 2. The contraction movement is necessary to obtain a more precise value of the displacement. If RR >= R2 the reflection movement is immediately followed by the contraction movement on the new simplex.

After the contraction movement, using equation <3.17>, the vertices of simplex are again reordered and labelled according to their response values. The reflection movement is then applied to the contracted simplex until the reflection of V3 is unsuccessful, i.e. RR >= R2. The search is then terminated and the DMD is set equal to the location V1. After each simplex movement, contraction or reflection, the response values of V1 and V3 are compared. If they are equal to each other the search is terminated. This is to stop the search when a flat response surface has been encounted. A flowchart of the basic simplex search algorithm is shown in figure 3.12. The following conventions are used in the figure:

USIMPX	set = TRUE if the simplex have been contracted before.
name	represents the position of the point 'name' in the displacement plane
Rname	represents the response of the point 'name'
VCHK(A,RA)	is a procedure for reordering and labelling the the vertices of the simplex with the new location 'A' which have the response 'RA'.

Figure 3.13 shows an example of the basic simplex search technique. The numbers in the figure represent the order in which the locations are evaluated. The vertices of the initial simplex are (0,0), (2,0) and (0,-2). As shown in the figure, the simplex converges to the minima very quickly.

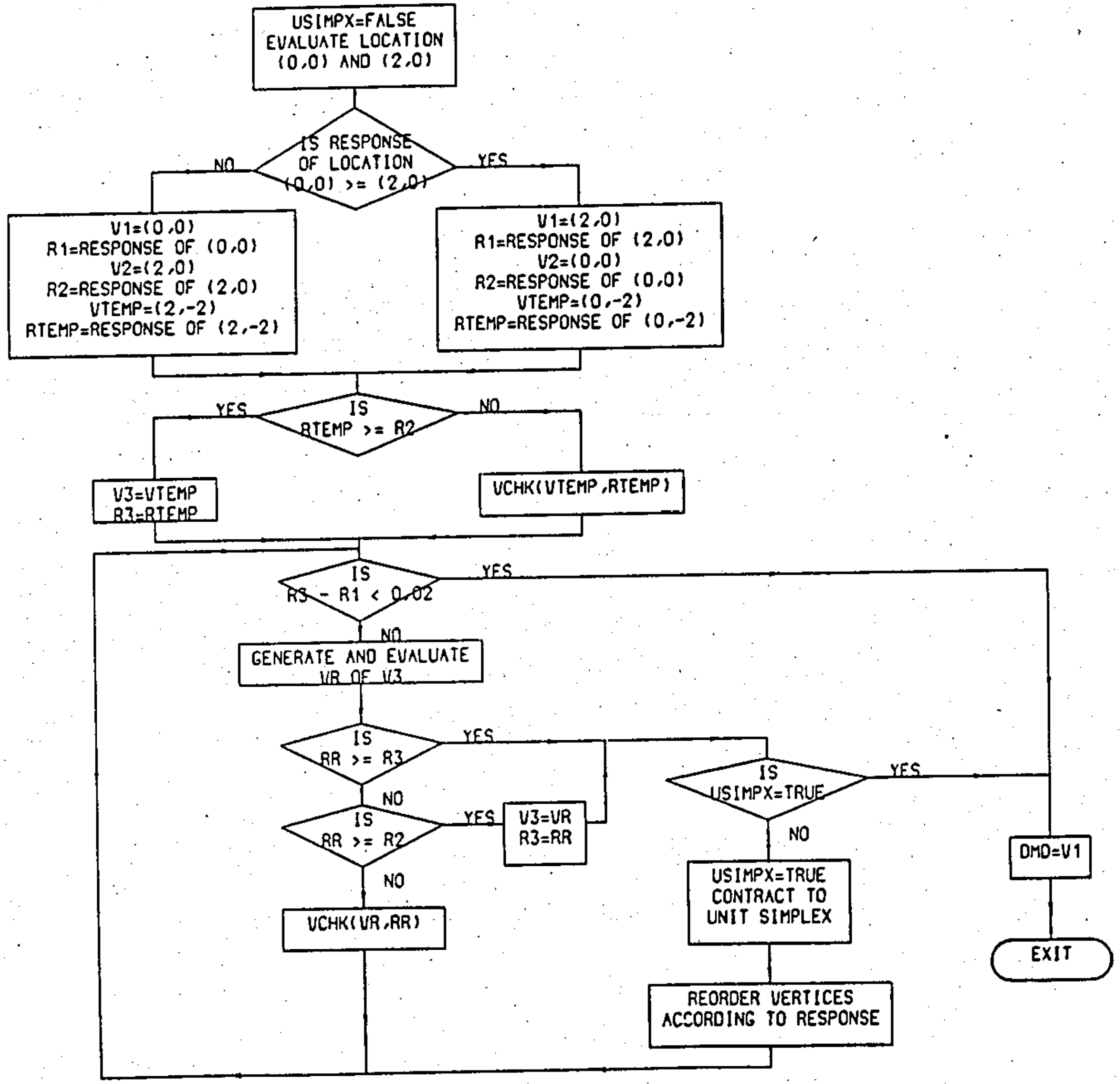


Figure 3.12 : Basic simplex search flowchart

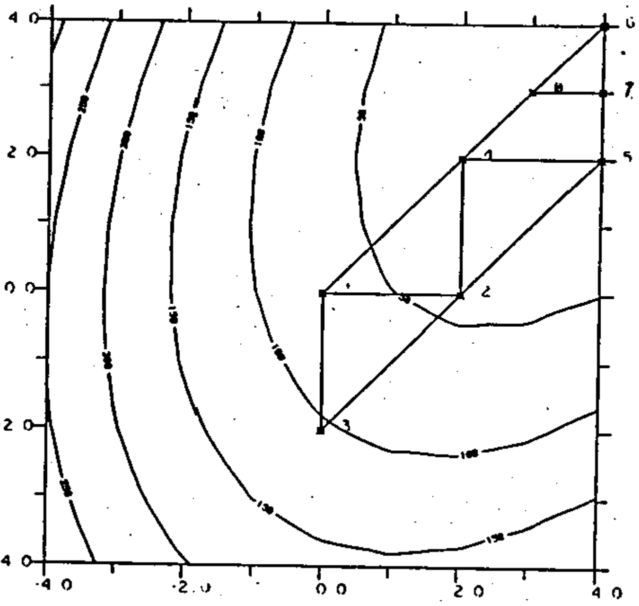


Figure 3.13: Basic simplex search.
DMD = (4,4)

3.3.2.3.2. Spendley simplex

The Spendley [23] simplex search algorithm is similar to the basic simplex search algorithm described above. The difference is that it has an additional step; if the reflection of V3 is unsuccessful, i.e. response value of VR is higher than the response of V2, vertex V2 will be reflected about its own reflection center rc obtained by the equation below:

rc = (V1 + V3)/2 ....<3.18>

A flowchart of the Spendley simplex generating procedure is found in figure 3.14. The same convention is used in the flowchart as in the basic simplex search algorithm flowchart.

Figure 3.15 shows an example of the search technique, where the simplex is superimposed on the contour map of a function typifying the response plane. The numbers represent

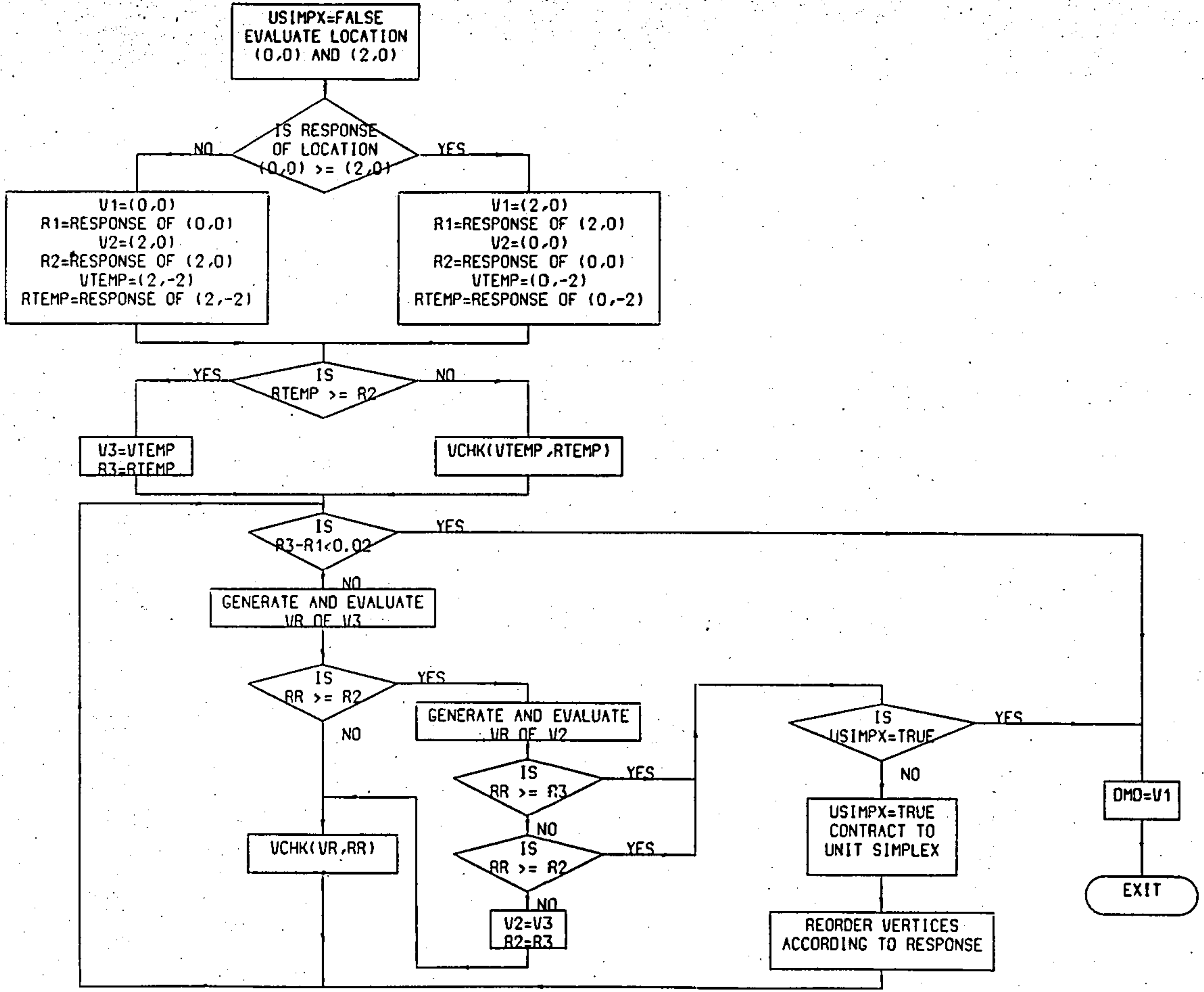


Figure 3.14 : Spendley simplex search flowchart

the order in which the locations are tested. The vertices of the initial simplex are (0,0), (2,0) and (0,-2).

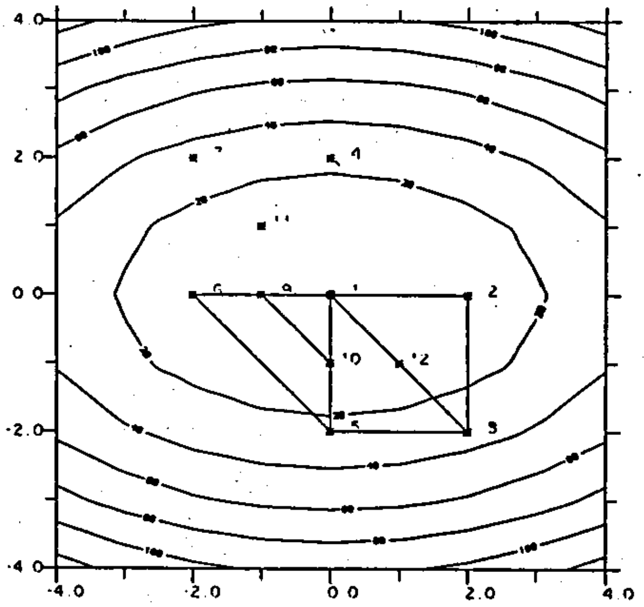


Figure 3.15 : Spendley simplex search
DMD = (0,0)

3.4. Results

The performance of the different search algorithms discussed above were compared with each other in terms of predictive error entropy and the average number of search points evaluated per changed block. The entropy gives an indication of the number of bits required to transmit the video signal if an optimum variable wordlength coder is used to code the prediction error. Two image sequences entitled "A1" and "A2" , showing head and shoulder pictures with detailed background and a halfbody image respectively, were chosen for the study.

(a) Pixel recursive algorithms

Table 3.1 shows results of the predictive error entropy of the moving area pixels. The pixel recursive algorithms give approximately 4-5% improvement over the frame-to-frame difference predictive error entropy. As expected the Newton method gives the best result as it can converge to the displacement location at a faster rate than the other two algorithms.

Algorithm	Predictive error entropy
Frame-to-frame difference (reference)	4.152
Steepest descent method (equation <3.7>)	4.097
Simplified steepest descent method (equation <3.8>)	4.022
Newton's method (equation <3.12>)	3.968

Table 3.1 : Pixel recursive algorithms predictive error entropy for image sequence A1.

(b) Block matching algorithms

Since conditional replenishment coders only transmit pixel data of those areas that have changed significantly, we need only estimate the DMD of those blocks that have changed significantly. Results obtained in chapter 2 show that a good criterion for determining whether a block has changed significantly since the last frame is average frame-to-frame difference magnitude of the block with the threshold set to 2, i.e. blocks which have average FDIF magnitude of >= 2 are considered as moving area blocks. Figure 3.16 shows a flowchart of the classification of the blocks in a picture.

Initially, it was necessary to determine the difference in performance, in terms of predictive error entropy, between the two distortion functions, MSE and MAE. The results of the 'Brute force' search for the DMD of image sequence A1, using MSE and MAE distortion functions, are shown in table 3.2. The table shows that there is no significant difference in performance between MSE and MAE as the distortion function. Since the MAE function requires less computation overhead than the MSE function, the MAE function was used for the experiments that followed.

Table 3.3 shows the average predictive error entropy of image sequence A1 for different search algorithms. The 'Brute force' technique gives an indication of the lowest entropy expected of the search. The table indicates that the predictive error entropy of the frame-to-frame difference can be reduced by 20-25% by using block matching algorithms to estimate the displacement. The small difference in predictive error entropy between the 'Brute force' technique and the other search algorithms indicates that the search algorithms are able to locate the position of minimum response successfully in most cases. Furthermore there is not much difference between the search algorithms. Thus, the choice of search algorithm would depend on the amount of computational overhead involved for each search algorithms. The computational overhead can be measured in terms of average number of search points evaluated per moving area block.

The third column in table 3.3 shows the average number of points evaluated per moving area block. The results show that the Basic simplex search has the lowest number of search average points per block. It saves about one search point for each moving area block when compared with the univariate

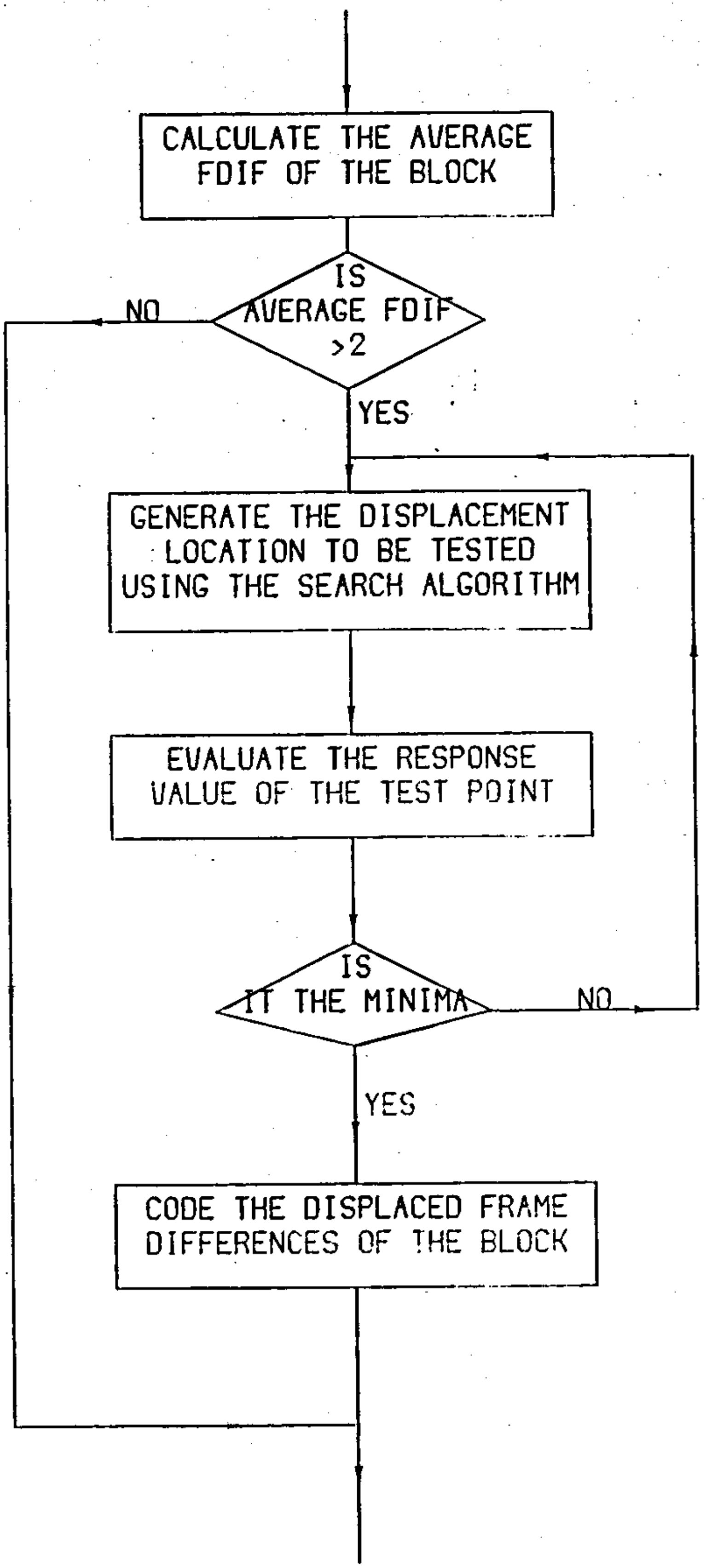

CALCULATE THE AVERAGE FDIF OF THE BLOCK
IS AVERAGE FDIF >2
NO
YES
GENERATE THE DISPLACEMENT LOCATION TO BE TESTED USING THE SEARCH ALGORITHM
EVALUATE THE RESPONSE VALUE OF THE TEST POINT
IS IT THE MINIMA
NO
YES
CODE THE DISPLACED FRAME DIFFERENCES OF THE BLOCK

Figure 3.16 : Block classification flowchart

Distortion function	Average predictive error entropy
Mean square error	3.1056
Mean absolute error	3.1057

Table 3.2: Distortion function performance comparison. Image sequence A1.

Search algorithm	Average predictive error entropy	Average no. of search pts.
frame-to-frame difference(reference)	4.305	0
Brute force	3.106	81
2D-logarithmic search	3.164	15.826
Univariate search	3.252	9.824
Spendley simplex	3.294	11.32
Basic Simplex	3.351	8.824

Table 3.3: Search algorithm comparison using image sequence A1.

search.

There are circumstances in which the univariate search performs better than the Basic simplex search. To find the conditions in which the univariate search performs better than the Basic Simplex search and vice versa the DMD distribution of frame 0 and frame 1 of image sequence A1 and frame 1 and frame 2 of image sequence A2 (shown in figure 3.17), found using 'Brute force' search, were tabulated in table 3.4. The Basic simplex search gives a better performance than the Univariate search for the first pair of frames while the reverse is true for the second pair of frames. In table 3.4(a) the number of occurrences of DMD peaks near the origin, i.e. displacement location (0,0), while table 3.4(b) shows a peak at displacement location (0,-3). This is the reason for the difference in performance, i.e. the Univariate search performs better than the basic simplex search for image sequence A2, as the univariate search only requires 5 search points if the minima lies at the origin while the Basic simplex search requires 7 search points. The DMD distribution in table 3.4(a) is not typical as it shows a person rotating his body. When the results were averaged over a sequence the Basic simplex search is the best search algorithm to use.

Table 3.4(b) shows another fact: most blocks have similar DMD locations. To exploit this DMD distribution characteristic the initial simplex was located in the vicinity of the DMD of the previous block. The DMD is set to the origin, location (0,0), if the adjacent block is not a moving area block. The first vertex of the initial simplex was located using the function below.

$$\begin{aligned} ISD(z) &= 0, \text{ if } |z| <= 1 \qquad \ldots\ldots\ldots<3.19> \\ &= 2.\text{sign}(Z) \text{ , elsewhere} \end{aligned}$$

The second vertex, in the horizontal direction, is located by image processing and its application

1986 - London

281769/01 - Z 621.3819598

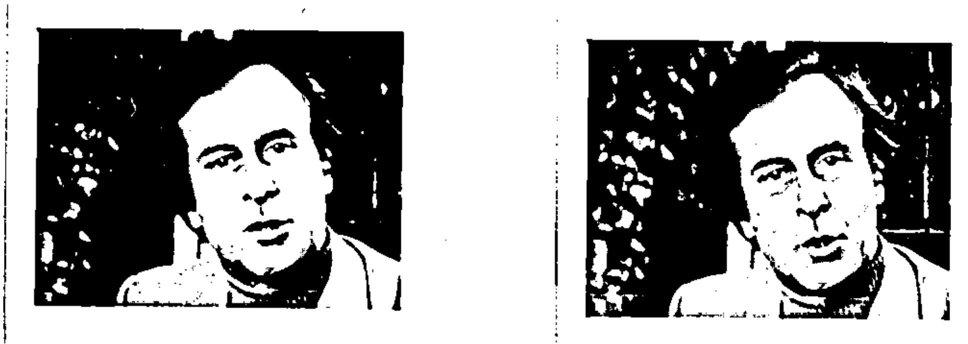
621.3819598

621.381

image Transmission over the

Cambridge Airy (Lee → B.S)

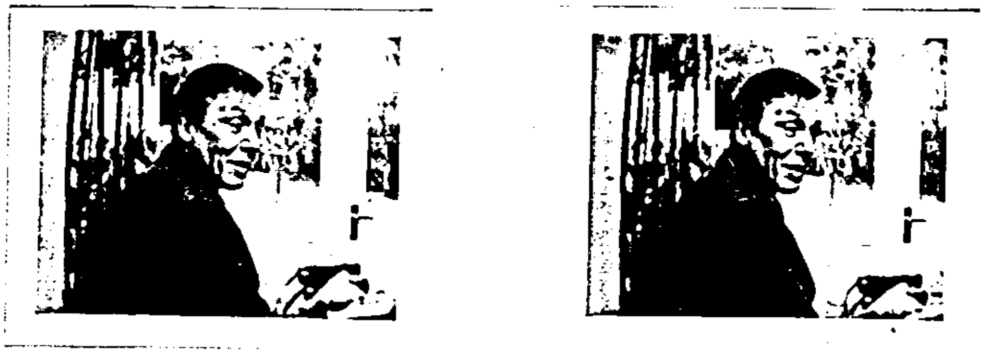
1986 (Thesis 000287/01



FRAME 0

FRAME 1

(a) Image sequence A1



FRAME 1

FRAME 2

(b) Image sequence A2

Figure 3.17 : Reference images

Dx Dy	-4	-3	-2	-1	0	1	2	3	4
-4	0	2	2	2	6	0	0	0	0
-3	2	0	1	1	0	0	0	0	0
-2	3	2	1	0	0	1	1	0	1
-1	7	6	3	4	4	3	0	1	1
0	11	7	4	3	52	23	7	1	1
1	4	5	1	3	11	17	8	0	1
2	3	3	0	1	2	5	0	0	0
3	1	0	0	1	2	0	1	0	0
4	2	2	3	6	9	3	6	4	14

Table 3.4(a): DMD distribution of frame 0 and frame 1 of image sequence A2.

Dx Dy	-4	-3	-2	-1	0	1	2	3	4
-4	0	0	0	0	1	0	0	0	0
-3	0	0	0	0	0	0	0	0	0
-2	0	0	0	0	0	0	0	0	0
-1	0	1	1	1	1	0	0	0	0
0	0	0	0	7	4	1	0	1	1
1	1	0	1	4	26	2	0	0	0
2	1	0	0	2	14	4	1	0	2
3	0	1	2	11	49	7	2	1	1
4	2	1	2	14	40	15	3	4	6

Table 3.4(b): DMD distribution of frame 1 and frame 2 of image sequence A1.

subtracting the Dx value of the first vertex from the Dx value of the DMD of the previous changed block. If the value is negative then the second vertex is placed on the left side of the first vertex otherwise it will be located on the right side of the first vertex. The location of the third vertex is directly below or above depending on which of the two previous vertices have the higher response. If the DMD of the previous block is (-3,1), the first vertex will be located at (-2,0). The second vertex will be located at (0,0) so that the Dx of the DMD lies between the first two vertices. Likewise the third vertex is located at (-2,2) so that the Dy of the DMD lies between the first and third vertices.

Location of the initial simplex	Average predictive error entropy	Average no. of search pts.
origin(0,0)	3.351	8.824
DMD of previous block	3.273	8.087

Table 3.5: Comparison of Basic simplex search with different method of setting the initial simplex.

Table 3.5 shows the results of the Basic simplex search technique with the initial search position at the origin and in the vicinity of the DMD of the previous block as described above. The results show that the Basic simplex search technique performs better, both in terms of entropy and average number of search points per frame, when locating the initial simplex in the vicinity of the DMD of the previous block rather than at the origin.

3.7. Conclusion

In this chapter we have investigated different displacement estimation algorithms: pixel recursive algorithms and block matching algorithms. Of the three pixel recursive algorithms: the steepest descent method, the simplified steepest descent method and Newton's method, Newton's method gives the lowest predictive error entropy. This is because the rate of convergence of Newton's method is better than that of the other two algorithms as it uses both the first and second derivatives of the square of the DFD signal to determine the step size. However the improvement in predictive error entropy is very small, 4-5%.

Four search algorithms were used to reduce the computational overhead of searching for the displacement estimate, DMD. A new search algorithm, the Basic simplex search algorithm, was developed. On the whole the Block matching algorithm gives about 20-25% reduction in predictive error entropy when compared with the frame-to-frame difference. There was very little difference between the algorithms in terms of predictive error entropy. The computation overhead, measured in terms of average number of search points per moving area block, shows that the Basic simplex search is the best. It reduces the number of search points evaluated by 45% when compared with the more well known 2D-logarithmic search. Further reduction in computation overhead was achieved by placing the initial simplex of the Basic simplex search in the vicinity of the DMD of its adjacent block.

CHAPTER FOUR

L.U.T. FRAMESTORE

4.1. Introduction

The Loughborough University of Technology (LUT) framestore was designed as a general purpose framestore with a multibus interface. Coupled with a multibus host computer a realtime videoprocessing or high-resolution graphics system is available to the user. The L.U.T. framestore consists of three main boards: controller board, two memory boards and an analogue board.

The controller board interfaces directly to the multibus system occupying a block of 256 I/O addresses and 64 Kbytes of the system memory space. The controller board is built around a Thompson CSF Graphic display processor (GDP) chip, EF9365. The GDP has inbuilt vector and character generation circuitry so that digitised picture elements (pixels) can either be accessed individually or in groups, using the GDP commands.

Each memory board consists of sixty-four, 64K by 1 bit, dynamic memory chips capable of storing a maximum of two frames of 512 pixels by 512 lines, each pixel being represented by 8 bits. Two memory boards are required if colour images are to be stored.

The analogue board contains both the analogue-to-digital converters (ADC) and the digital-to-analogue converters(DAC) and their relevant circuitry. It can accommodate up to a maximum of 3 ADC and DAC chips.

4.2. Framestore Description

The framestore will be described in three sections, namely:

(a) framestore registers
(b) data transfer between framestore and host system
(c) frame acquisition and display

4.2.1. Framestore registers

The framestore occupies 256 input/output locations of the multibus system, base address specified by switch 1, as shown in figure 4.1. The functions performed by the individual registers are as follow:

(a) GDP registers
A brief description of the individual GDP register functions is given below. A more detailed explanation is found in the GDP data sheet in reference [10].

(i) X and Y address registers
The X and Y address registers are 12 bit read/write registers, only 9 bits are valid. They point to the current position of the GDP 'pen', i.e. the position of the next pixel to be modified.

(ii) DELTAX and DELTAY registers
The DELTAX and DELTAY registers are 8 bit read/write registers. They indicate the size of the vector to be plotted on the x and y coordinate axes respectively.

(iii) CSIZE register
The CSIZE register is an 8 bit read/write register. It indicates the scaling factors of characters to be drawn. The lower 4 bits represent the scaling factors of the x axis while the upper 4 bits represent the scaling factor of the y axis.

A13 - AC	A8	A7	A6	A5	A4 - A0	REGISTERS
SWITCH 1	0	0	0	0	X	GRAPHIC DISPLAY PROCESSOR
	0	0	0	1	X	RESOLUTION SELECT
	0	0	1	0	X	BORDER
	0	1	0	1	X	BUFFER ADDRESS
	0	1	1	0	X	READ/WRITE
	0	1	1	1	X	BUFFER RESET

A4	A3	A2	A1	A0	REGISTERS
0	0	0	0	X	CMD/STATUS
0	0	0	1	X	CTRL 1
0	0	1	0	X	CTRL 2
0	0	1	1	X	CSIZE
0	1	0	1	X	DELTAX
0	1	1	1	X	DELTAY
1	0	0	0	X	XHIGH
1	0	0	1	X	XLOW
1	0	1	0	X	YHIGH
1	0	1	1	X	YLOW

Figure 4.1 : LUT framestore register address

(iv) CTRL1 register

The CTRL1 register is a 7 bit read/write register. It controls the general operation of the GDP, e.g. pixel access mode (normal or fast), organisation of screen, etc.

(v) CTRL2 register

The CTRL2 register is a 4 bit register. It specifies the line type of the vector drawn, i.e. continous, dotted, dashed, dash-dotted, and the tilt and orientation of the character drawn.

(vi) CMD/STATUS register

The CMD/STATUS register is an 8 bit read/write register. It acts as command register when data is written to it. Details of the commands available, e.g. vector plotting, character plotting etc., are found in reference [10]. Data read from the register represents the status of the GDP, e.g. is the GDP ready to receive the next command.

(b) Read/write register

This is a 2 bit write only register. The least significant bit, D0, specifies whether the next pixel access is a read or write operation. 'D1' is used to specify if the input to the video memory is from either the digitised external video input or the buffer memory as shown below.

D7 - D2	D1	D0	
X	0	0	Buffer memory write
X	0	1	Buffer memory read
X	1	0	Digitised video write

Figure 4.2: R/W register

(c) Border register

This is a 16 bit write only register. It specifies the colour on both sides of the actual display on a screen. The function of the individual bits is shown in figure 4.3.

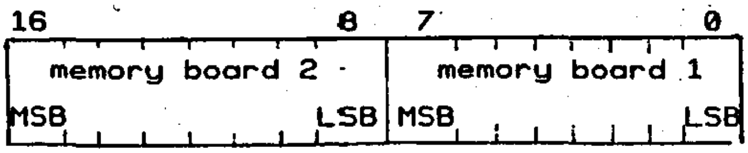


Figure 4.3: Border register

(d) Buffer address reset register

This is a 1 bit write only register. The buffer address counter is reset when this register is accessed.

(e) Resolution select register

This is an 8 bit write only register. It specifies the resolution of the image, display frame number and the host access frame number. The function of the individual bits is shown in figure 4.4. Where V1-V3 represent the frame which is been written to or read from, while M1-M3 represent the display frame number. As shown in the figure, the two least significant bits of the register represent the vertical and horizontal resolution respectively.

MSB LSB

V3	X	X	M3	X	X	0	0	512 by 512
V3	V2	V1	M3	M2	M1	1	0	256 by 512
V3	V2	X	M3	M2	X	0	1	512 by 256
V3	V2	V1	M3	M2	M1	1	1	256 by 256

Figure 4.4: Resolution select register

(f) Buffer address register

This is an 8 bit read only register. It contains the outputs of the buffer address counter, BA2-BA9, as shown in figure 4.5.

MSB							LSB
BA9	BA8	BA7	BA6	BA5	BA4	BA3	BA2

Figure 4.5: Buffer address register

4.2.2. Data transfer

Data transfers are initiated by the host system. Each data transfer over the multibus takes 1.3 microseconds. Two types of data are transferred between the host system and the framestore: command/status data and image data. Command/status data refers to the data used for setting the content of registers on the framestore controller board.

Data transfer from the host to the video memory, or vice versa, is through an 8 Kbytes dual ported buffer memory. The buffer memory is transparent to both the video memory and the multibus, i.e. both of them have access to the video buffer memory all the time without incurring any wait state. The transfer of data between the host system and the video memory consists of two steps:

(a) data transfer between the buffer memory and the host system

(b) data transfer between the video memory and the buffer memory

Step (a), is done via the multibus interface of the controller board, whilst step (b) requires the use of the GDP pixel access commands. The GDP offers two modes of pixel access, fast and normal. Normal read/write mode has an average access time of 1.3 microseconds whilst the fast mode has an access time of 592 nanoseconds. During the fast pixel access mode the display is blanked because the entire frame period, other than the dynamic storage refresh periods, is allocated to pixel access. The command set can be divided into the following categories:

(i) Vector plotting

There are two ways of plotting a vector line from the present x and y position. If the length of the vector is less than three pixels vertically and horizontally, the direction and length of the vector can be specified with a single command code. For long vectors, the horizontal and vertical length of the vector are specified by writing to the respective DELTAX and DELTAY register of the GDP respectively. The maximum length of the vector in the vertical and horizontal direction is 256 pixels. The addressing of the pixels in the vector is as shown in figure 4.6.

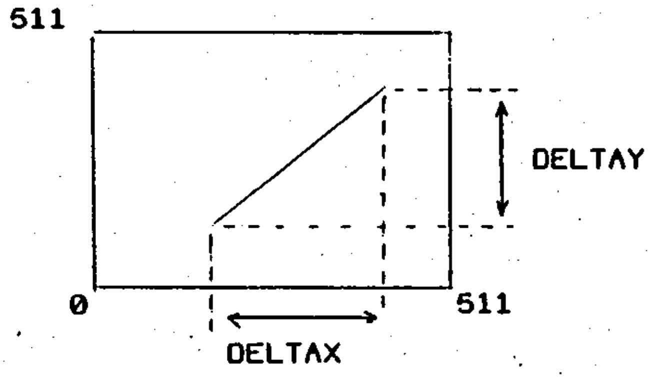


Figure 4.6: Screen addressing

(ii) Character plotting

The GDP has an internal character set of 97 ASCII characters. The minimum size of these characters is 5 by 8 pixels. On top of this there is a special character which fills a square area, with minimum size of 4 by 4 pixels. The character size can be increased to a maximum of 16 times the minimum size. Figure 4.7 shows the order in which the pixels are accessed when the special character is used with a vertical orientation.

64	63	62	61	60	59	58	57	
.	.	.	.	.	.	.	.	
.	.	.	.	.	.	.	.	
.	.	.	.	.	.	.	.	
17	18	19	20	21	22	23	24	
16	15	14	13	12	11	10	9	
1	2	3	4	5	6	7	8	Start address

Figure 4.7: Pixel access sequence

(iii) Screen scanning

This set of instructions (4H, 6H, 07H and 0CH) is used either to clear the entire screen, or set the screen to a specific colour and also to initiate the acquisition of video input to the video memory. The screen scanning starts from the beginning of the next new frame.

Figure 4.8 and figure 4.9 shows the sequence of steps for writing to and reading from the video memory respectively.

4.2.3. Frame acquisition and display

The video input signal is sampled, digitized and stored in the video memory. The resolution of the displayed or captured picture is software controlled and the different options available are:

(i) 2 frames of 512 pixels by 512 lines
(ii) 4 frames of 512 pixels by 256 lines
(iii) 4 frames of 256 pixels by 512 lines

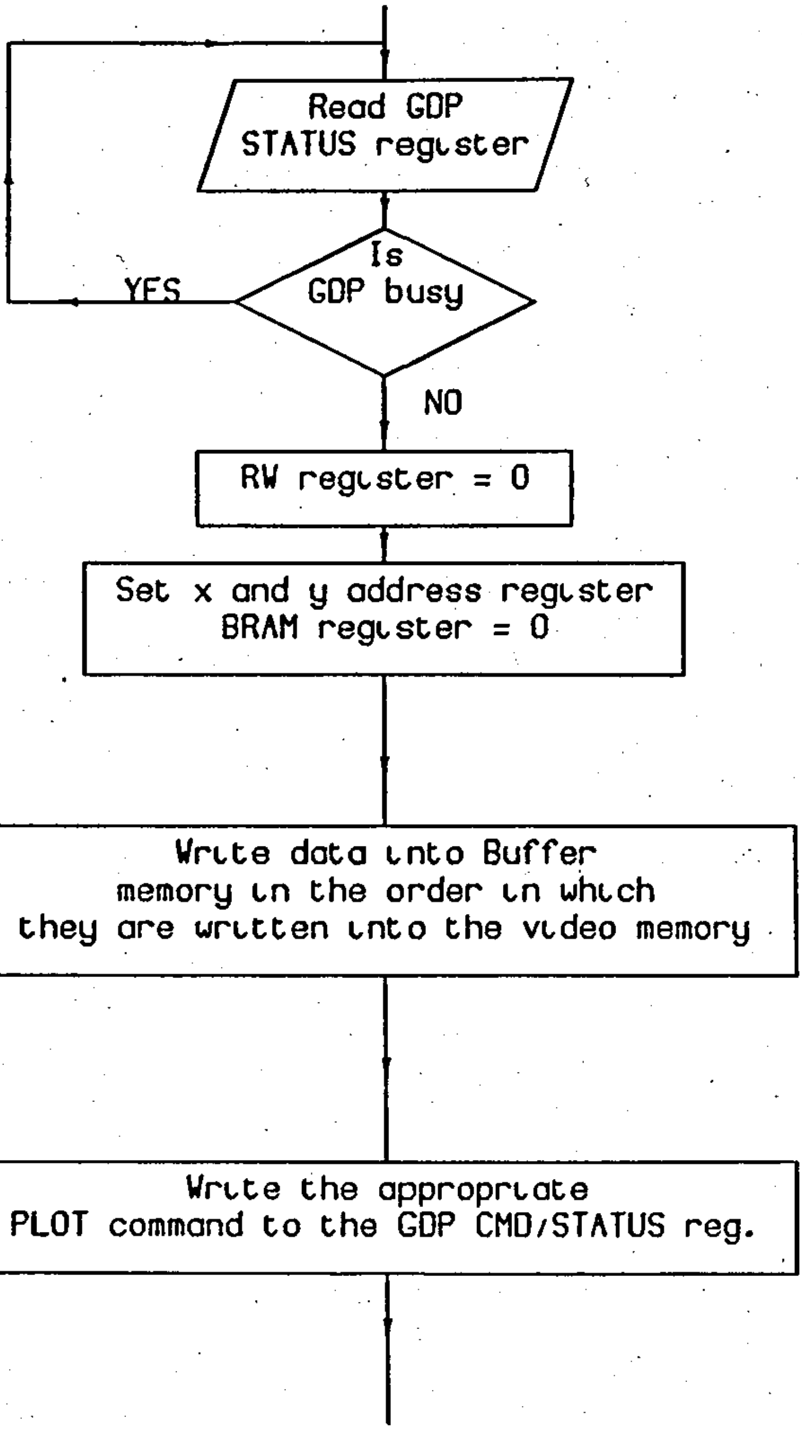

Read GDP
STATUS register
Is
GDP busy
YES
NO
RW register = 0
Set x and y address register
BRAM register = 0
Write data into Buffer
memory in the order in which
they are written into the video memory
Write the appropriate
PLOT command to the GDP CMD/STATUS reg.

Figure 4.8 : Write to video memory flowchart

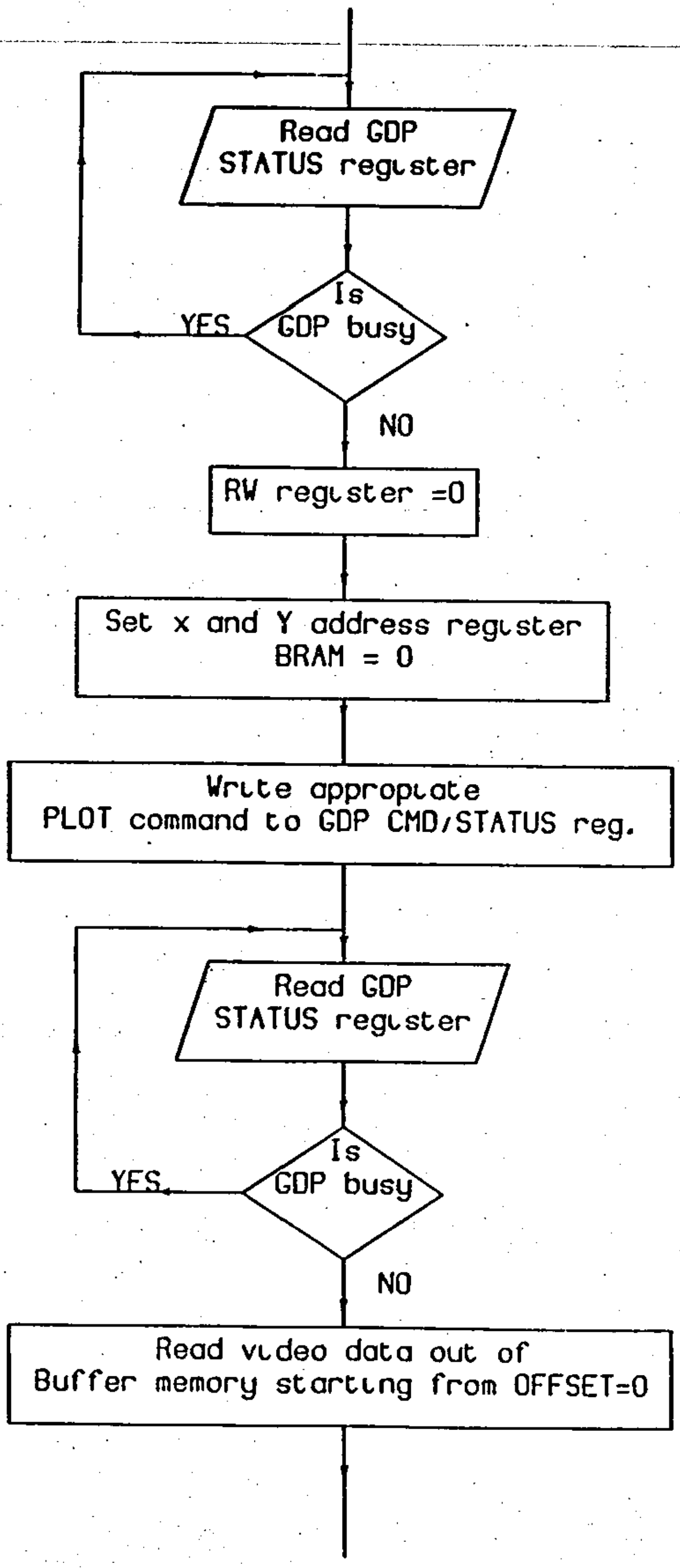


Figure 4.9 : Read from video memory flowchart

(iv) 8 frames of 256 pixels by 256 lines

The aspect ratio of the display is 1:1 unlike the normal 4:3 ratio. This is due to the fact that out of the 64 microseconds in a scanline only 32 microseconds are used for displaying data. This display format has the advantage that when drawing a circle on the display we will get a circle while for a 4:3 ratio system we would get an ellipse.

The pictures are captured at full frame rate, i.e. 25 frames per second. The acquisition of the digitized picture starts from scanline number 38 of the field after the GDP has received the screen scanning command. A noticeable 'jerky' effect occurs on the screen because the digitized video data is written into the video memory during what was previously the active display period of a scanline and it is also outputted to the DAC .

If the vertical resolution is 256, only one field is stored in the memory at a time. Figure 4.10 shows the flowchart of the steps for video acquisition.

4.3. Hardware details

Due to the complexity of the framestore circuitry, each of the three circuit boards will be discussed separately:

(a) video memory board
(b) controller board
(c) analogue board

4.3.1. Video memory board

Each memory board can store up to a maximum of 2 frames with 512 pixels by 512 lines resolution and 8 bits per pixel. It consists of 64, 64K by 1 bit dynamic RAM, divided into 8 memory planes, each containing 8 memory chips. A maximum

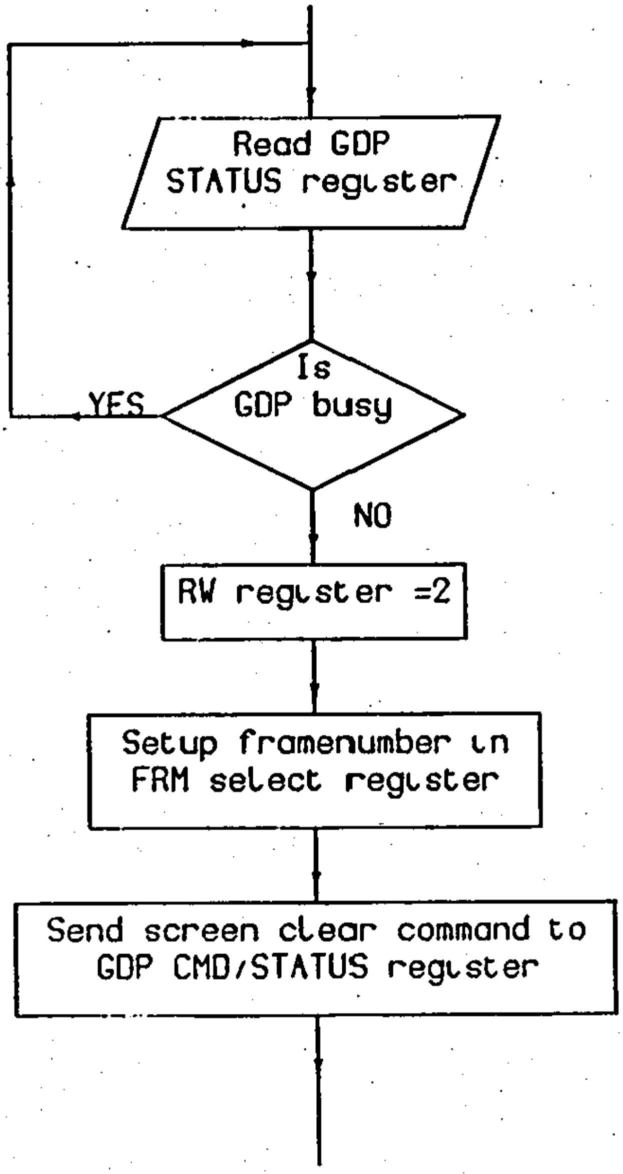


Figure 4.10 : Video acquisition flowchart

of 256 levels are available on each memory card. Figure 4.11(a) shows the circuit diagram excluding the memory banks. Figure 4.11(b) shows one of the bit planes.

The video memory control and address signals are derived from the controller card and are buffered by high fan-out TTL drivers. The individual chips in each plane are selected by MSL0-MSL2, which are decoded by the 3 to 8 decoder, IC73, which supplies the RAS signals. During the display, refresh, and screen scanning period, all the RAS lines are enabled by setting the ALL line low. Only one RAS line will be enabled during the control board read and write period so that when individual pixels are accessed by the GDP. The two column address signals(CAS), CAS ODD and CAS EVEN are used to vary the horizontal resolution of the picture. If the horizontal resolution of the picture is 512 pixels both CAS signals are enabled while only one is enabled when the horizontal resolution is 256 pixels.

Due to the slowness of the memory chips, data demultiplexing at the input and multiplexing at the output is necessary. A universal shift register (SR) with parallel-serial and serial-parallel options, is used to multiplex data to and from the VRAM. Data read out of the memory is latched into the shift registers and shifted out using the CP signal. The outputs of the shift registers are latched by dual output latches, IC78-IC79. They are then directed to either the analogue board (via connector P2) for display or the controller board (via connector P3) by the OE W and OE Y signals respectively.

The EXT OE signal and Foregnd EN signal are used to select the source of the input, either digitised video or buffer memory data, as input to the video memory. They are connected to the output enable pin of the external input register (IC76) and foregound register(IC77) respective.

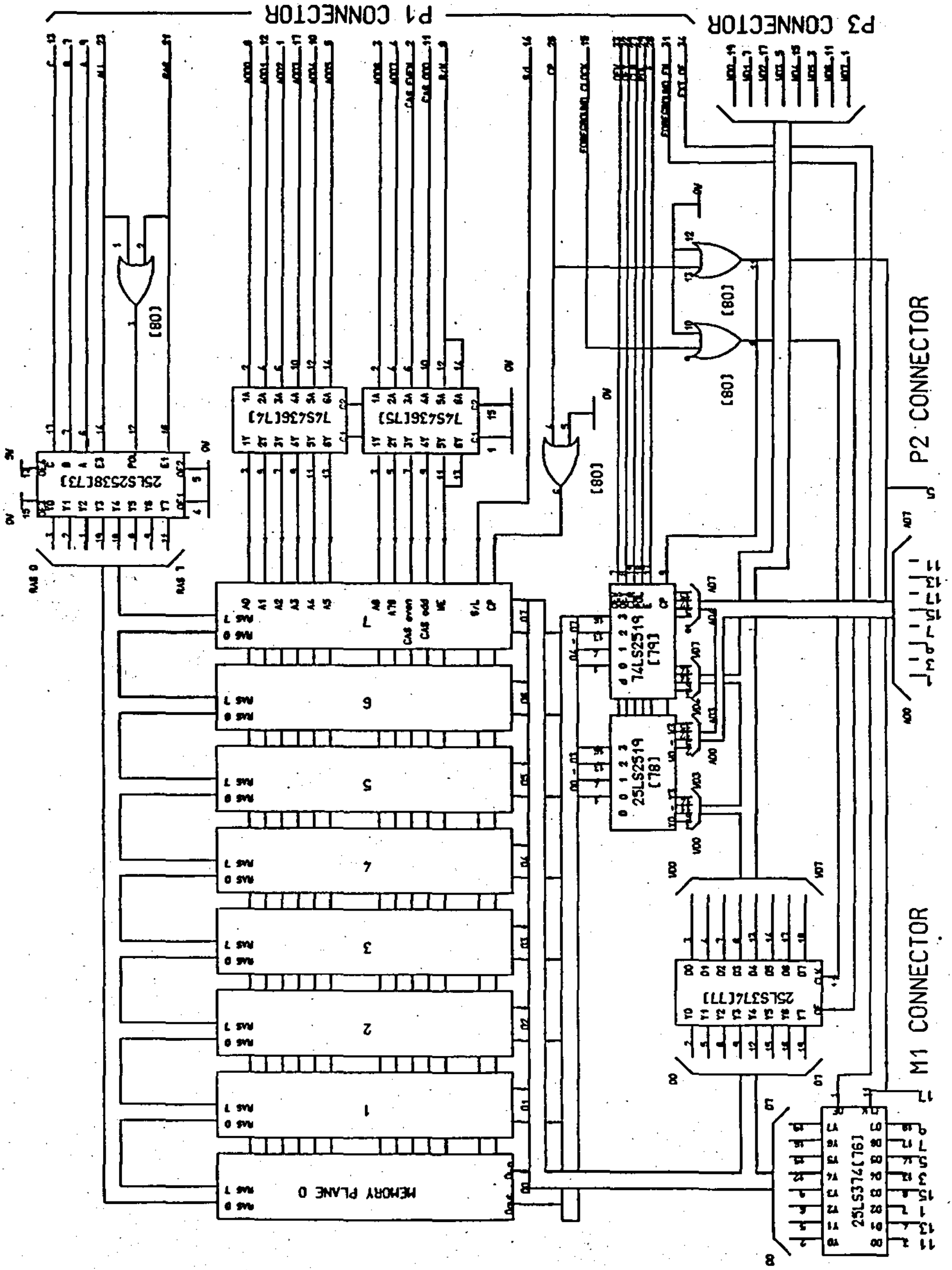


Figure 4.11(a) : Memory board circuit diagram

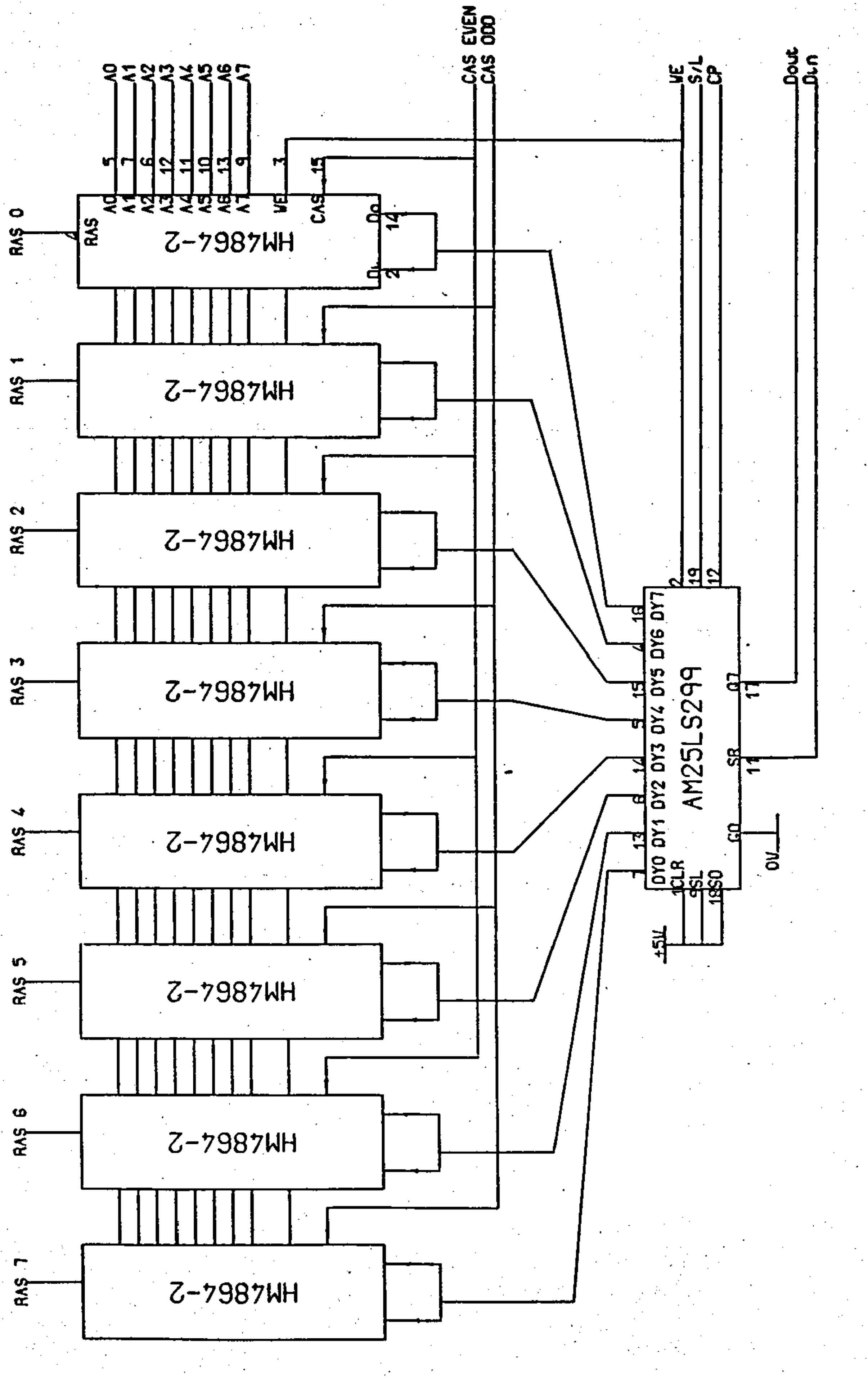


Figure 4.11(b) : Individual memory plane circuit

4.3.1.1. Timing details

The timing details of the video memory board will be discussed according to their functions, consisting of:

(a) memory chip access requirements
(b) video memory display cycle
(c) video memory read cycle
(d) video memory write cycle
(e) video acquisition cycle

4.3.1.1.1. Memory chip access requirements

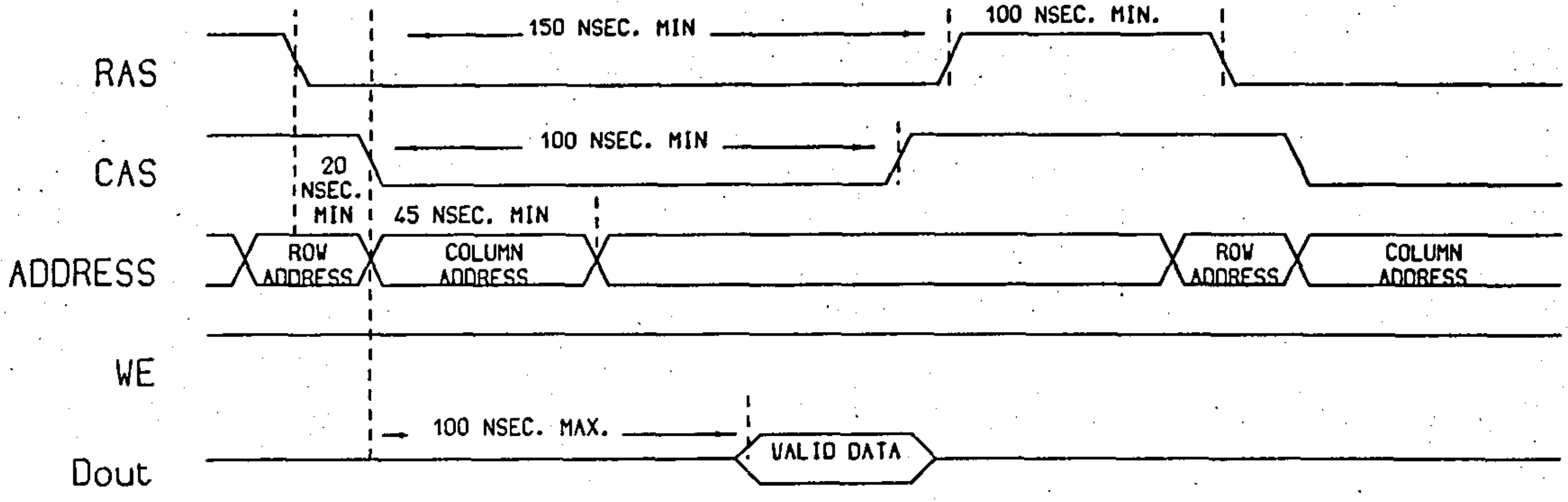
Figure 4.12 shows the read and write access requirements of the memory chips. The mode of operation, i.e. read or write, is determined by the VE signal. The RAS and CAS signals latch the row and column address signals of A0-A7 on its falling edge.

The minimum read or write cycle time of the memory is 270 nanoseconds. During the read mode, the data will appear at the output, Dout, after a minimum of 135 nanoseconds. While in the write mode the data must be available at its data input pin, Din, at the beginning of the CAS.

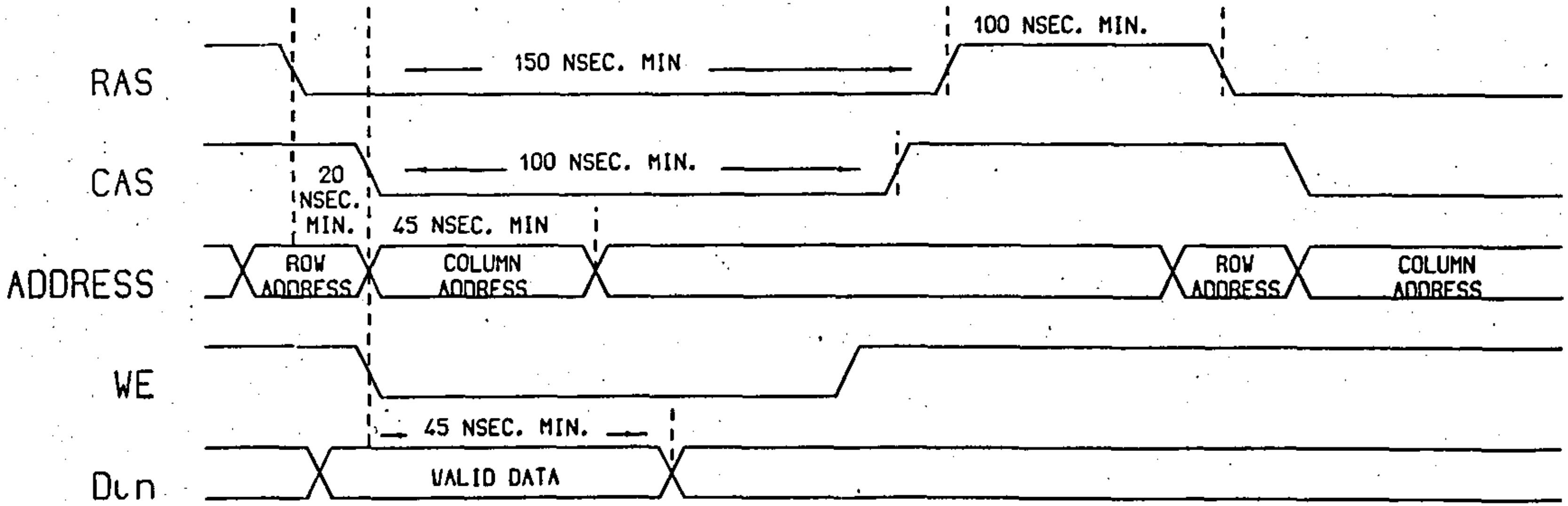
4.3.1.1.2. Video memory display cycle

Figure 4.13 shows the timing waveform of the main signals involved when data is read out of the memory to be displayed on the screen.

During the display cycle, the ALL signal is set low to enable the RAS output signals of all the memory chips in each memory plane. The valid data is loaded into the universal shift registers of each memory plane on the rising edge of the CP clock with the S/L signal high. They are then shifted out of the shift register, by the CP signal, into the dual output



(a) MEMORY READ CYCLE



(b) MEMORY WRITE CYCLE

Figure 4.12 : Memory chips access requirements

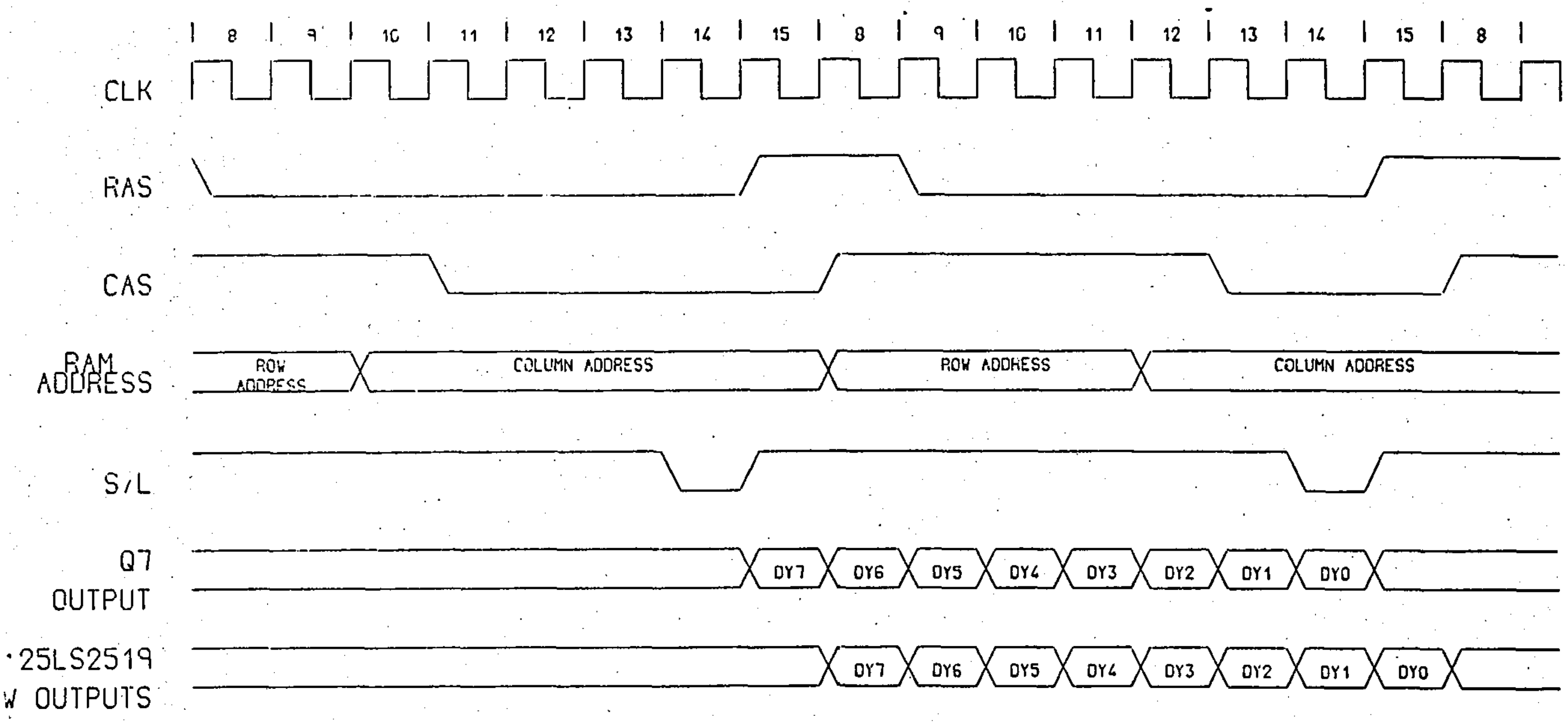


Figure 4.13 : Video memory display cycle timings

port latches, 25LS2519, as indicated by the Q7 output signal in figure 4.13. During this period the OE V is set low so that the data is directed to the analogue board via its V0-V3 outputs.

4.3.1.1.3. Video memory read access cycle

The timing waveform of this cycle is almost the same as the video memory display cycle. But during this cycle only one memory chip per bit plane is selected by MSL0-MSL2. The data output of the memory is loaded in the SR as above. The dual ported latches are only enabled for one CP cycle. When the E signal goes low the data from the shift register Q7 output is latched into the dual ported latches. The data is held until the next access cycle. The data is outputted to the controller board via its Y output port. Figure 4.14 shows an example of the timing when the DY4 data is required.

4.3.1.1.4. Video memory write access cycle

Figure 4.15 shows the timing waveform of the main signals involved in transferring data from the controller board buffer memory to the video memory board.

The foregound register output is first enabled by the foreground OE signal. Data from the controller board is then latched into the foreground register by the foreground clock. It is then shifted into the universal shift registers. The outputs of the shift registers are then latched into the appropriate memory chip, specified by MSL0-MSL2, in each memory plane on the falling edge of the CAS signal.

4.3.1.1.5. Video acquisition cycle

The video acquisition cycle is exactly the same as the video memory write access cycle timing with the following two exceptions:

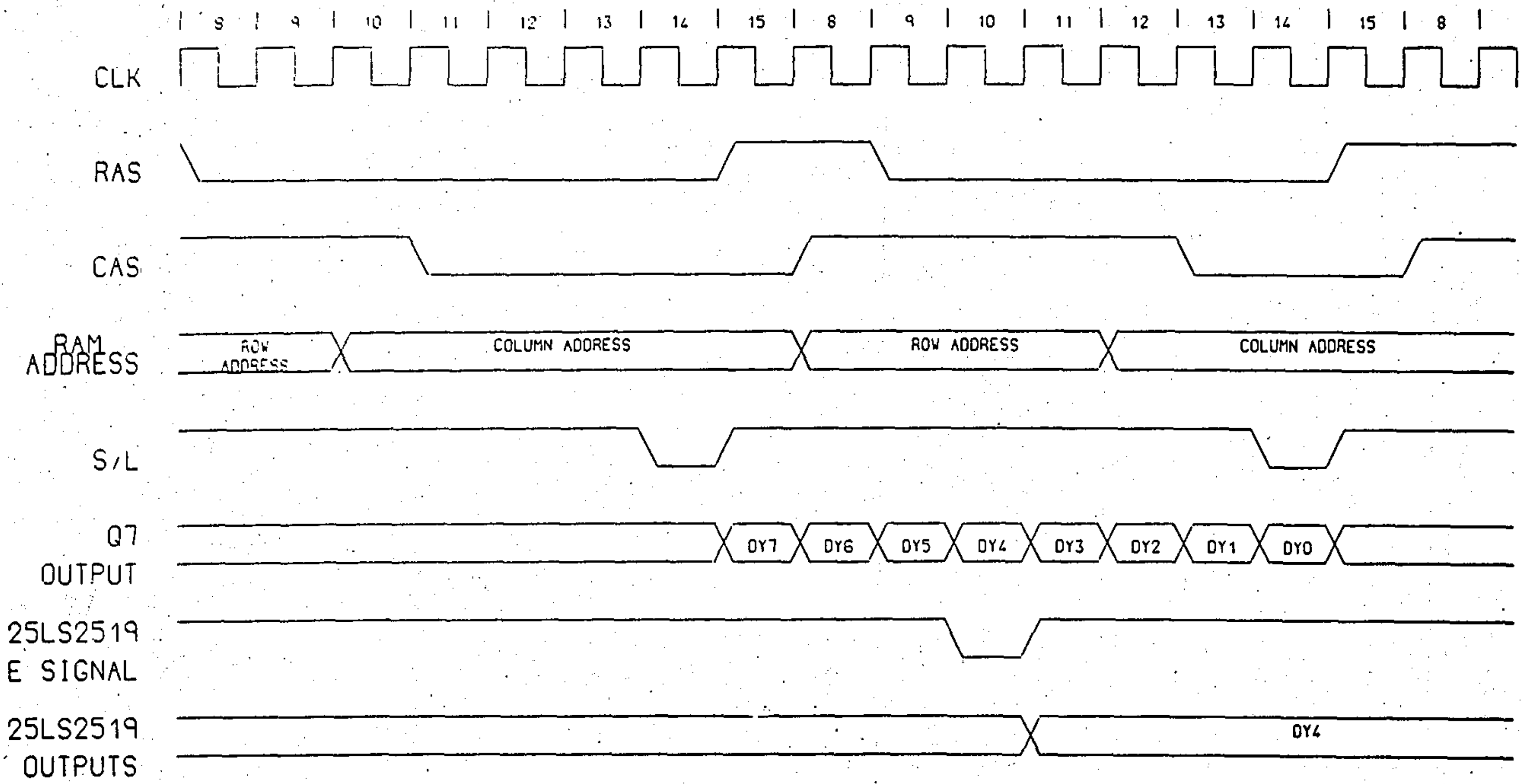


Figure 4.14 : Video memory read cycle timings

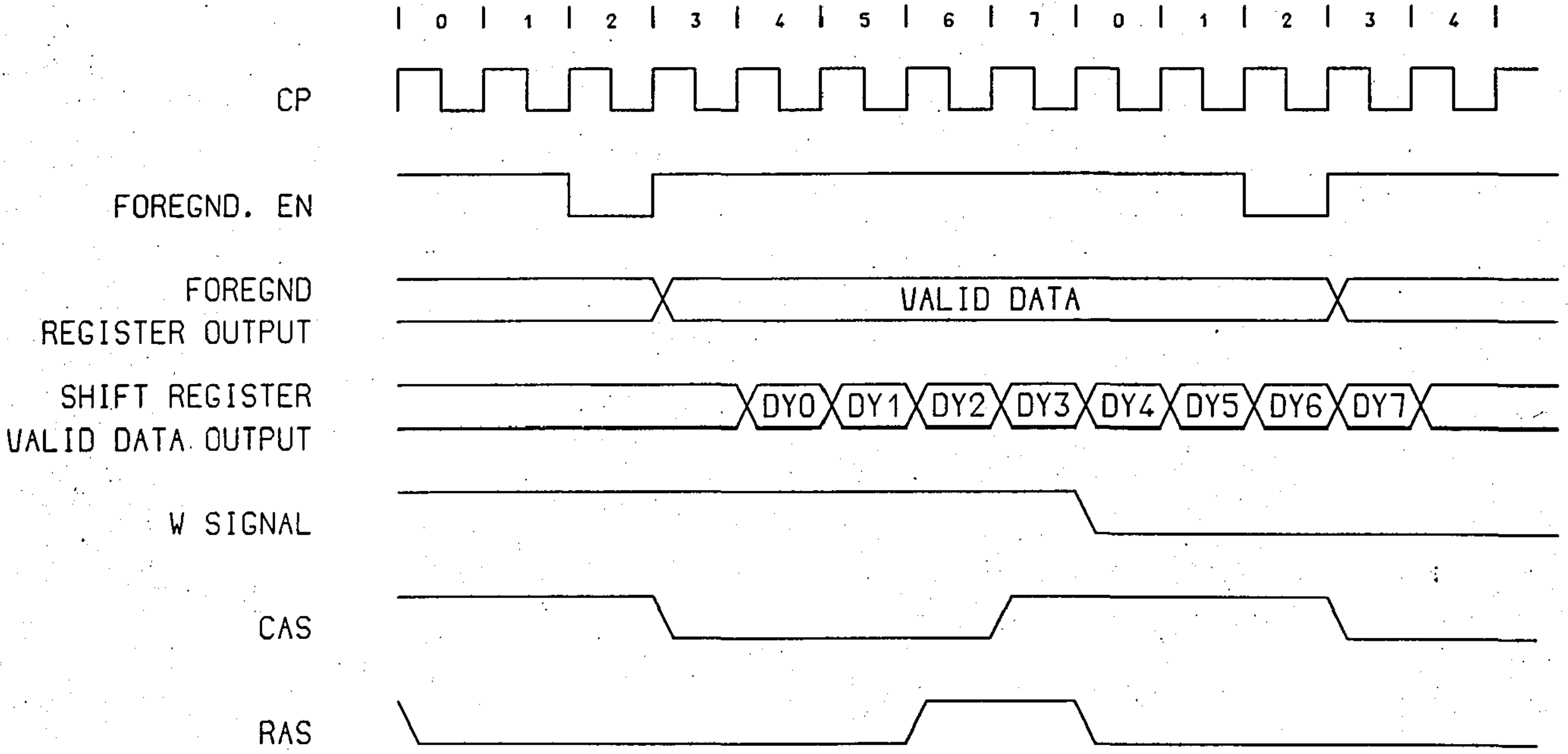


Figure 4.15 : Video memory write cycle timings

(a) the ALL signal is low, which enables all the memory chips

(b)the external input register output is selected, by the EXT EN signal.

Data from the analogue board is latched by the external input register, IC 76. The data is then shifted into the universal shift registers. The outputs of the registers, DY0-DY7, are then latched into all the memory chips in each memory plane on the falling edge of the CAS signal.

4.3.2. Controller board

Due to the complexity of the circuitry the controller board hardware will be discussed under the following headings:

(a) multibus interface
(b) video display control and addressing
(c) video data buffer

4.3.2.1. Multibus interface

The standard multibus [85] bus structure has been selected as the communication bus between the host system and the framestore. The framestore occupies a block of 256 I/O ports, selected by switch 1, and 64 Kbytes of the memory address space, specified by switch 2.

Figure 4.16 shows the circuit details of the multibus interface. The multibus data lines, MD0-MDF, are buffered by 4 bidirectional drivers so that it can drive more TTL devices. The address lines, ADR0-ADRC, are likewise also buffered.

Data transfers over the multibus are initiated by one of the control lines: input/output read(IORD), input/output

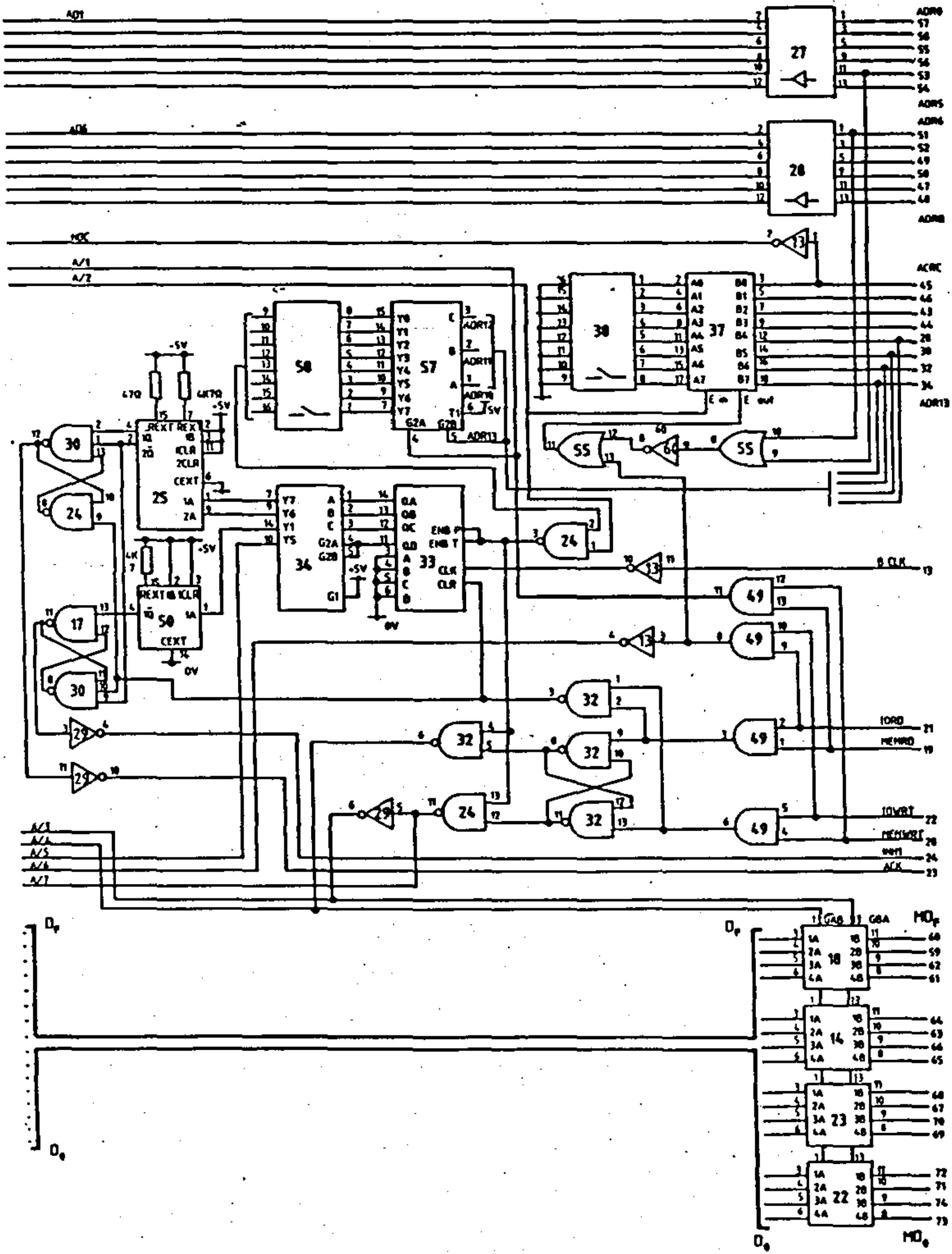


Figure 4.16 : Multibus interface circuitry

write(IOWRT), memory read(MEMRD) and memory write(MEMWRT) signals. The framestore controller board acknowledges the reception of the command by pulsing the ACK and INH1 lines on the multibus. The timing requirements of the multibus are as shown in figure 4.17.

4.3.2.2. Video display control and addressing

Figure 4.18 shows the circuit details of this section of the controller board. Due to its complexity the discussion on its circuitry is divided further into the following sections:

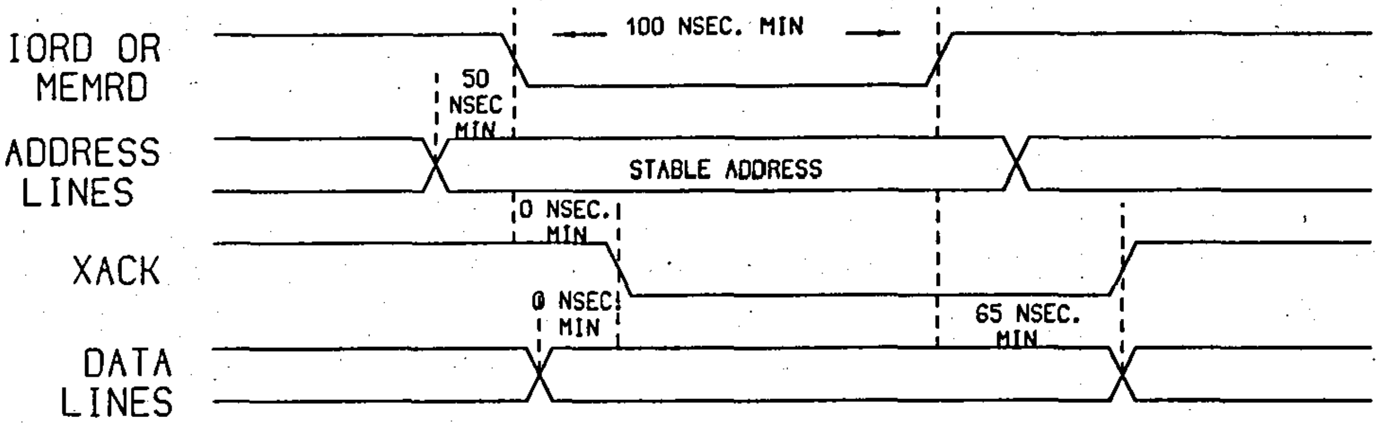
(a) register address decoder
(b) programmable read only memory (PROM) sequencer
(c) graphic display processor and video addressing

4.3.2.2.1. Register address decoder

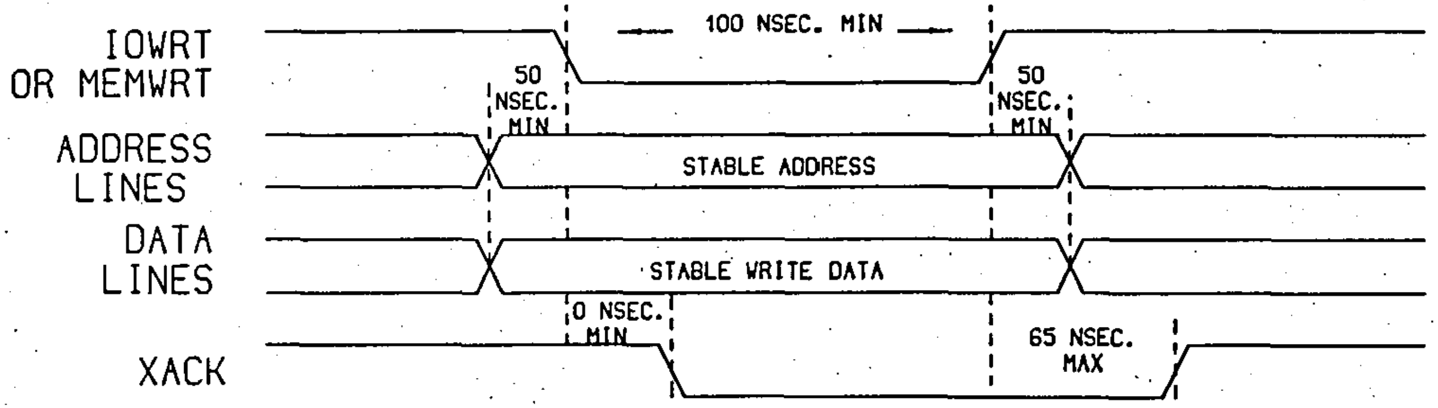
The individual registers on the controller board are selected by the output of the register address decoder (IC20). Multibus address line AD5-AD7 are decoded to form the enable lines of the individual registers. The register address decoder is enabled only by the input/output select line, A/6, from the multibus interface. The registers enabled by the individual output of the decoder are shown in the table below. Some of the output of the resolution select register, Q2-Q8, represents the frame select during the display and host access period are multiplexed by a 2:1 multiplexer, IC2. The input is selected by the BLK signal of the GDP which divides the entire frame period, 40 milliseconds, into display periods (BLK=low) and host access periods (BLK=high).

4.3.2.2.2. PROM sequencer

The PROM sequencer circuitry is located at the bottom left hand corner of figure 4.18. It consists of 3 components:



(a) Multibus read signals



(b) Multibus write signals

Figure 4.17 : Multibus timing requirements

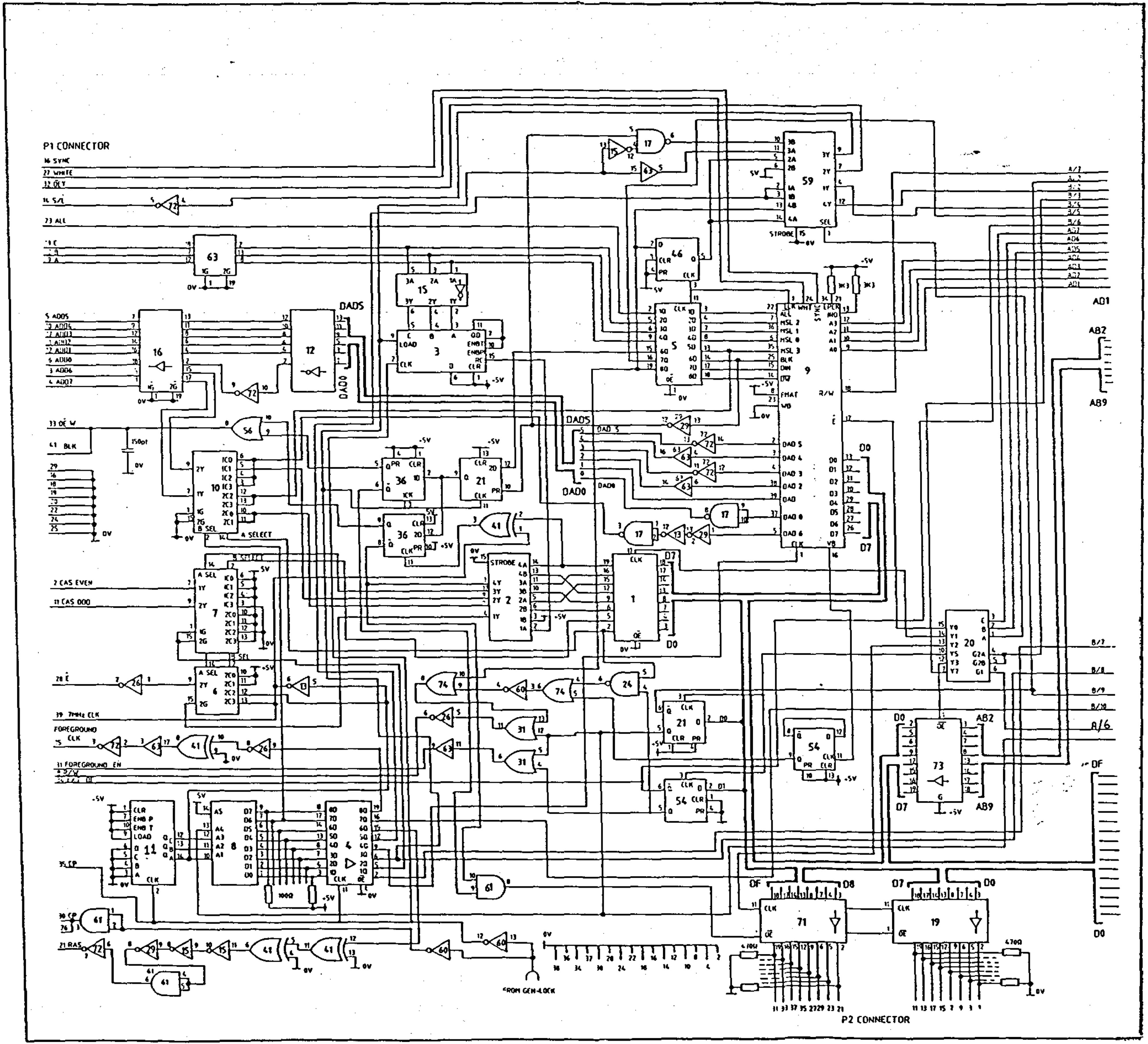


Figure 4.18 : Video display control and addressing circuitry

(a) 4 bit synchronous binary counter (IC11)

(b) 32 by 8 bit word PROM (IC8)

(c) 8 bit latch (IC4)

PROM	MULT PULSE	CAS	S/L	GDP	BADD CLK.	RAS	MULT VRAM	FOREGND CLOCK
ADDRESS (HEX)	D0	D1	D2	D3	D4	D5	D6	D7
0	0	1	1	1	1	0	1	1
1	1	1	1	1	0	0	1	1
2	0	1	1	1	1	0	0	0
3	0	0	1	0	1	0	0	1
4	0	0	1	0	1	0	0	1
5	0	0	1	0	1	0	0	1
6	0	0	0	0	1	1	0	1
7	0	1	1	0	1	1	0	1
10	0	1	1	1	1	0	1	1
11	1	1	1	1	1	0	1	1
12	0	1	1	1	1	0	0	0
13	0	0	1	0	1	0	0	1
14	0	0	1	0	1	0	0	1
15	0	0	1	0	1	0	0	1
16	0	0	0	0	1	1	0	1
17	0	1	1	0	0	1	0	1

Table 4.1 : PROM content

The upper 16 bytes of the PROM are not used,as A5 is set low. A 14 MHZ. signal is connected to the clock pin of the binary counter, forcing it to count from 0 to 15. The count will reset to 0 on the next clock once its output value reaches 15. QA, QB and QC output signals of the counter are connected to PROM address lines, A1-A3 respectively. Thus, the sequence of PROM access is repeated after every 8 counts. The A4 address line of the PROM is connected to the output of the R/W register, splitting the lower 16 bytes of the PROM into two banks, of 8 bytes each. This is necessary because the timing requirements of the two types of video memory access, read from video memory and write to video memory, are different. Table 4.1 shows the content of the PROM. To synchronise the output of the PROM, they are all latched by an 8 bit latch. Figure 4.19 shows the timing waveform output of the latch.

4.3.2.2.3. Graphic display processor and video addressing

(a) addressing signals

The GDP lies at the heart of the framestore, generating most of the required signals needed to drive the video memory. It also generates the signals needed to drive an interlaced raster scan cathode ray tube(CRT) display with the CCIR 625 lines 50 Hz. standard. The GDP also has a microprocessor interface through which it receives commands. A complete data sheet is given in reference [10] and the reader should refer to it for details.

	MSL				DAD						
	0	1	2	3	0	1	2	3	4	5	6
RAS	X0	X1	X2	Y2	X8	X7	X6	X5	X4	X3	Y1
CAS	X0	X1	X2	f	Y8	Y7	Y6	Y5	Y4	Y3	Y0

Table 4.2 : GDP memory addressing

The GDP video memory address signals, DAD0-DAD6 and MSL0-MSL3, are buffered to give high drive over the ribbon cable. The pixel coordinate address represented by the address signals is shown in table 4.2, where 'f' selects frame 0 or frame 1. The DAD6 and MSL3 signals from the GDP are multiplexed with the outputs of the resolution select register by a 4:1 multiplexer (IC10). The input is selected by the least significant bit, i.e. the vertical resolution select bit, of the resolution select register and the MULT/VRAM signal. The Y0 address, shown in the table above, represents the even and odd field signal. This signal is

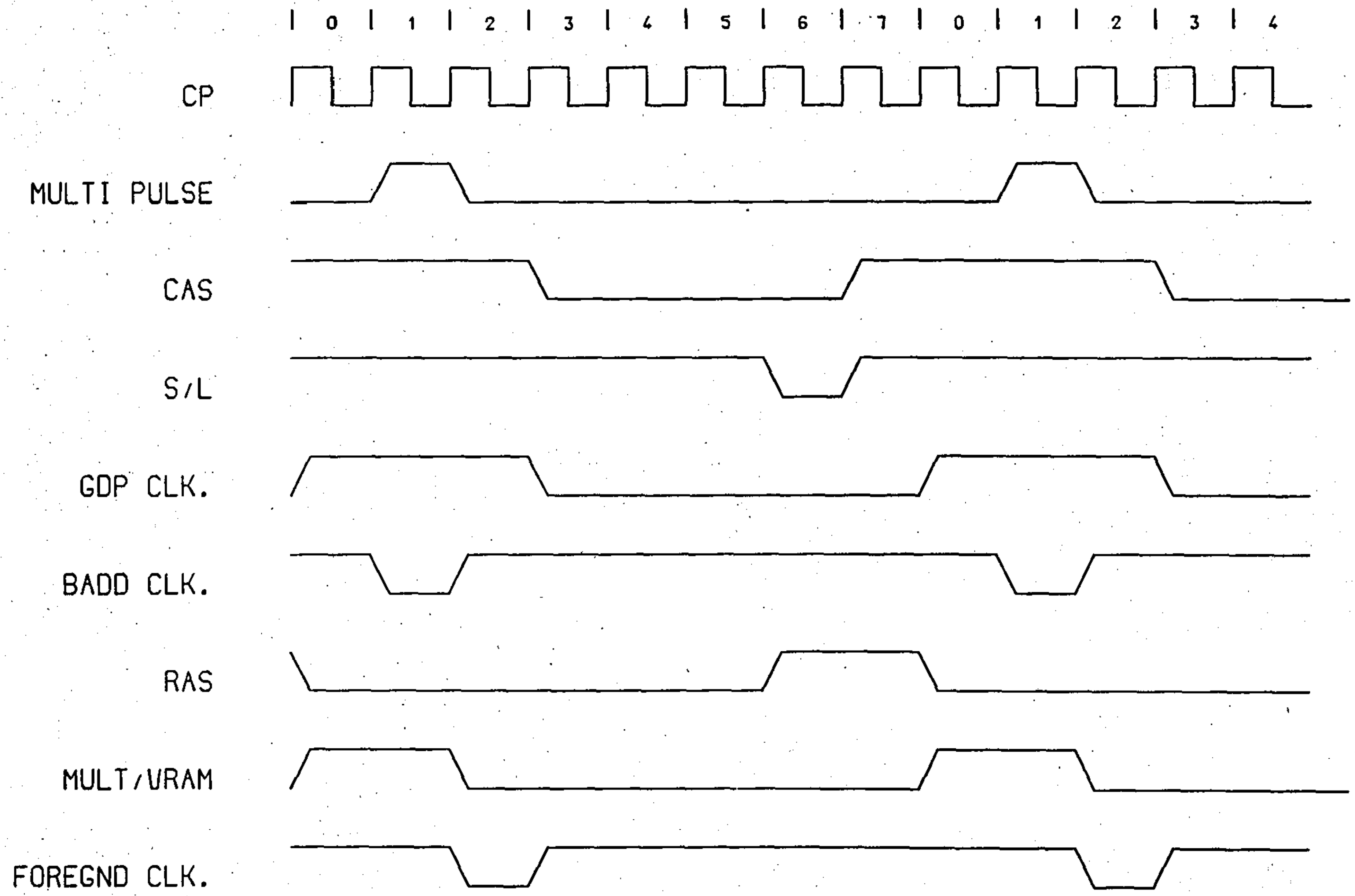


Figure 4.19(a) : PROM write output

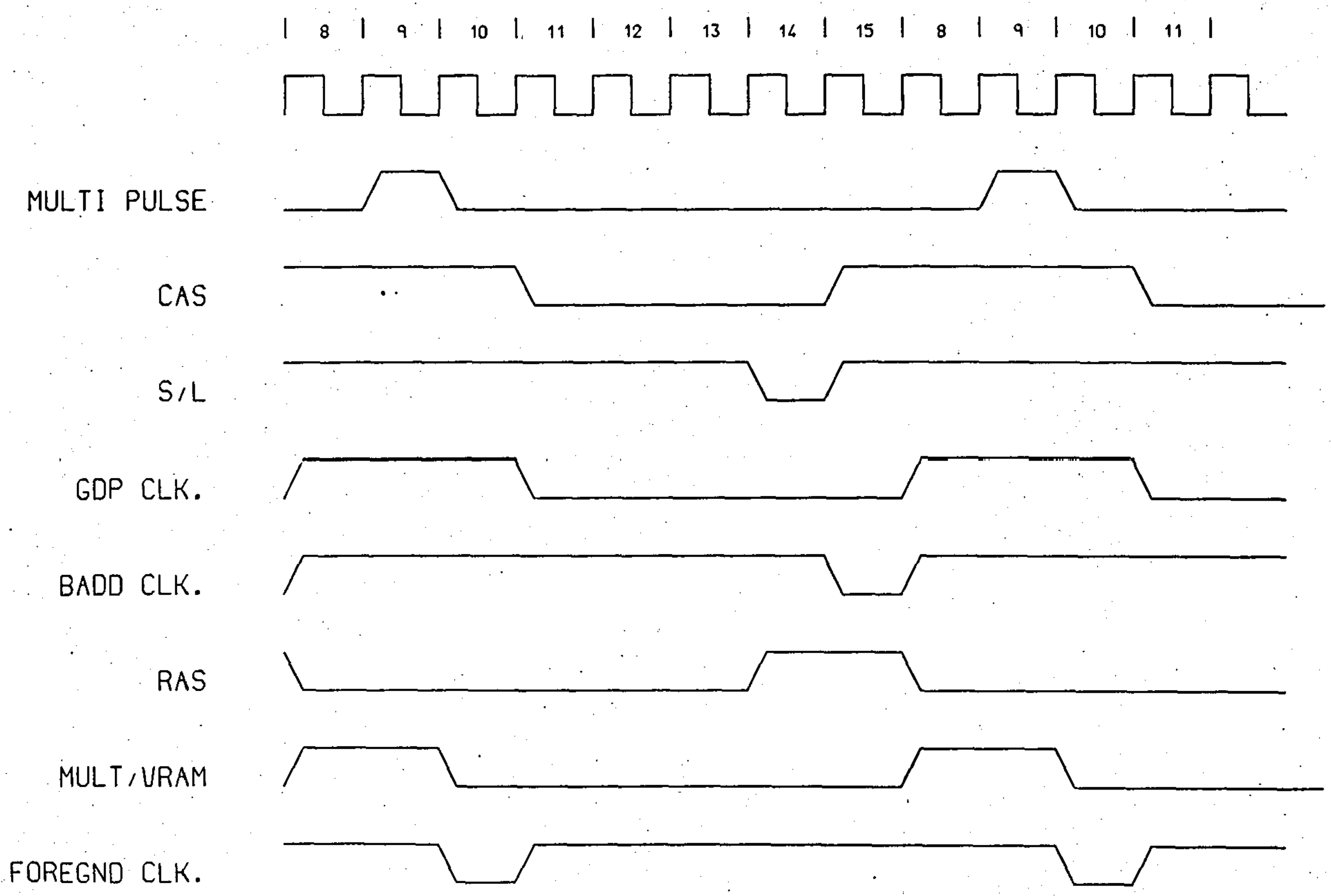


Figure 4.19(b) : PROM read output

multiplexed with the outputs of the resolution select register. If the vertical resolution selected is 256, the multiplexer selects the output of the resolution select register instead of the GDP DAD6 signal as the Y0 value. However, if the vertical resolution is 512 then the multiplexer selects the GDP DAD6 signal as its output all the time.

The MSL3 signal is multiplexed with the resolution select register frame select output, such that the A7 line during RAS period will select one of the two frames, 512 pixel by 512 lines, available on the video memory board.

The horizontal resolution of the picture is controlled by the CAS EVEN and CAS ODD signals. These two signals are connected to the CAS pins of alternate memory chip in each memory plane. If the horizontal resolution is 256 only one CAS signal, CAS ODD or CAS EVEN, is enabled.

(b) control signals

The GDP DW signal will be set low during the video memory access cycle. This signal is 'OR-ed' with the output of the R/W register and the output of the frame pulse generator to generate the WE signal for the video memory board. The output of the R/W register determines whether the memory access cycle is either read or write cycle. The output of the frame pulse generator is enabled only when a vertical resolution of 256 is selected. It is used to enable the WE for only one field during each frame acquisition. This is to ensure that frame acquisition has the same field everytime. The timing waveform of the addressing and control signals is given in reference [10].

(c) microprocessor bus signal

Commands are passed from the host system to the GDP via its microprocessor bus. The GDP internal registers are mapped onto the host system I/O ports, occupying 32 locations. The

GDP data lines, D0-D7, are connected to the lower byte of the multibus data lines. The timing requirements of the GDP microprocessor bus are found in reference [10].

4.3.2.3. Video data buffer

The circuit diagram of the video data buffer is shown in figure 4.20. The video data buffer can be divided into four groups of components:

(a) buffer address counters (IC35, IC40, IC39)
(b) address multiplexers (IC45, IC44, IC43, IC42)
(c) buffer memory (IC47, IC48, IC69, IC70)
(d) data latches (IC65-68, IC51-53, IC62)

The buffer address counters generate the address of the location to which the next video memory data is written into or read from the buffer memory. They are incremented after each GDP video memory access cycle. The counters can be disabled, i.e. stop counting, if DIN output of the GDP, which is controlled by CTRL 1 register, is set low. The counters can be reset by accessing the BRESET register.

The buffer memory is dual ported and its access is transparent to both the VRAM and the multibus, i.e. both have full access to the memory without any time delay. To do this the GDP signal (571 nanoseconds) is divided into two memory access periods: multibus access time and video memory access time, by the MULT/VRAM signal. The output of the buffer address counters, AB0-ABB, and the multibus address lines, AD 1-AD C, are time division multiplexed with each other by the address multiplexers. The outputs of the multiplexers are connected to the address pins of the buffer memory.

The buffer memory consists of four low powered 4K by 4 bits static random access memory IMS1420, which have 70

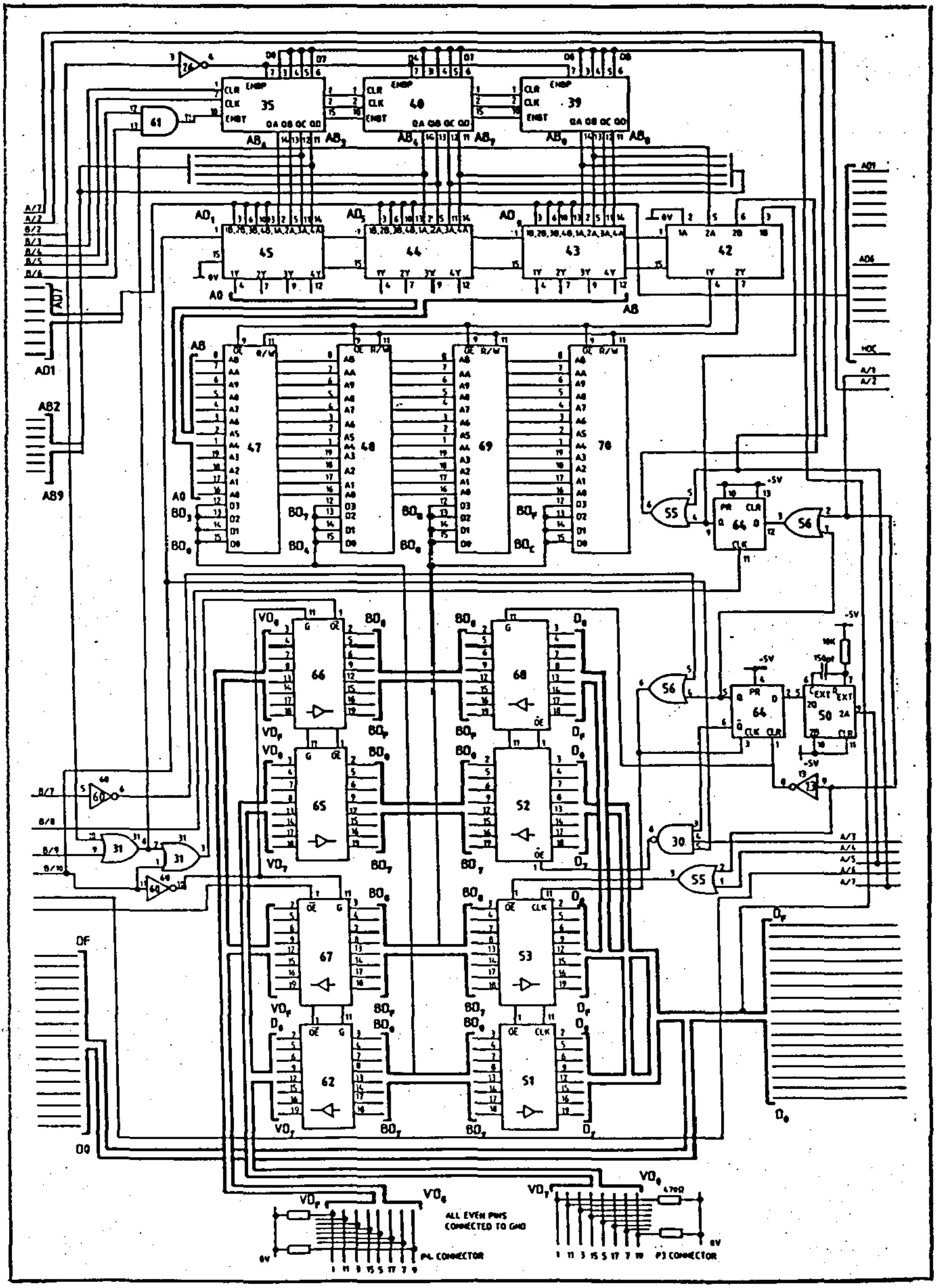


Figure 4.20 : Video data buffer circuitry

nanosecond access time. They are arranged such that the 16 bits word representing a pixel data, stored in the buffer memory, will appear as consecutive address data bytes to the host system.

The data lines between the multibus and the buffer memory are buffered by latches. They are used to latch the data at the appropriate time. The output of the latches are controlled such that only one pair of latches can drive the data lines of the memory at any one time.

The multibus signals: address, data, and control signls are asynchronous to the other signals on the controller board, generated by the PROM sequencer. Thus, additional circuitry is required to generate the respective latch signal and output enable signal of the buffers between the multibus and the buffer memory.

4.3.2.3.1. Timing details

Due to the complexity of the timing involved, the timing diagrams will be discussed according to the function performed by the video data buffer.

(a) Write to video memory
(b) Read from video memory
(c) Multibus read
(d) Multibus write

4.3.2.3.1.1. Write to video memory

The timing waveform of the signals involved in transferring data from the buffer memory to the video memory is shown in figure 4.21. The data from the buffer memory, addressed by the output of buffer address registers, is latched into the buffer on the falling edge of the inverted MULT/VRAM signal. The data is continuously outputted to the memory

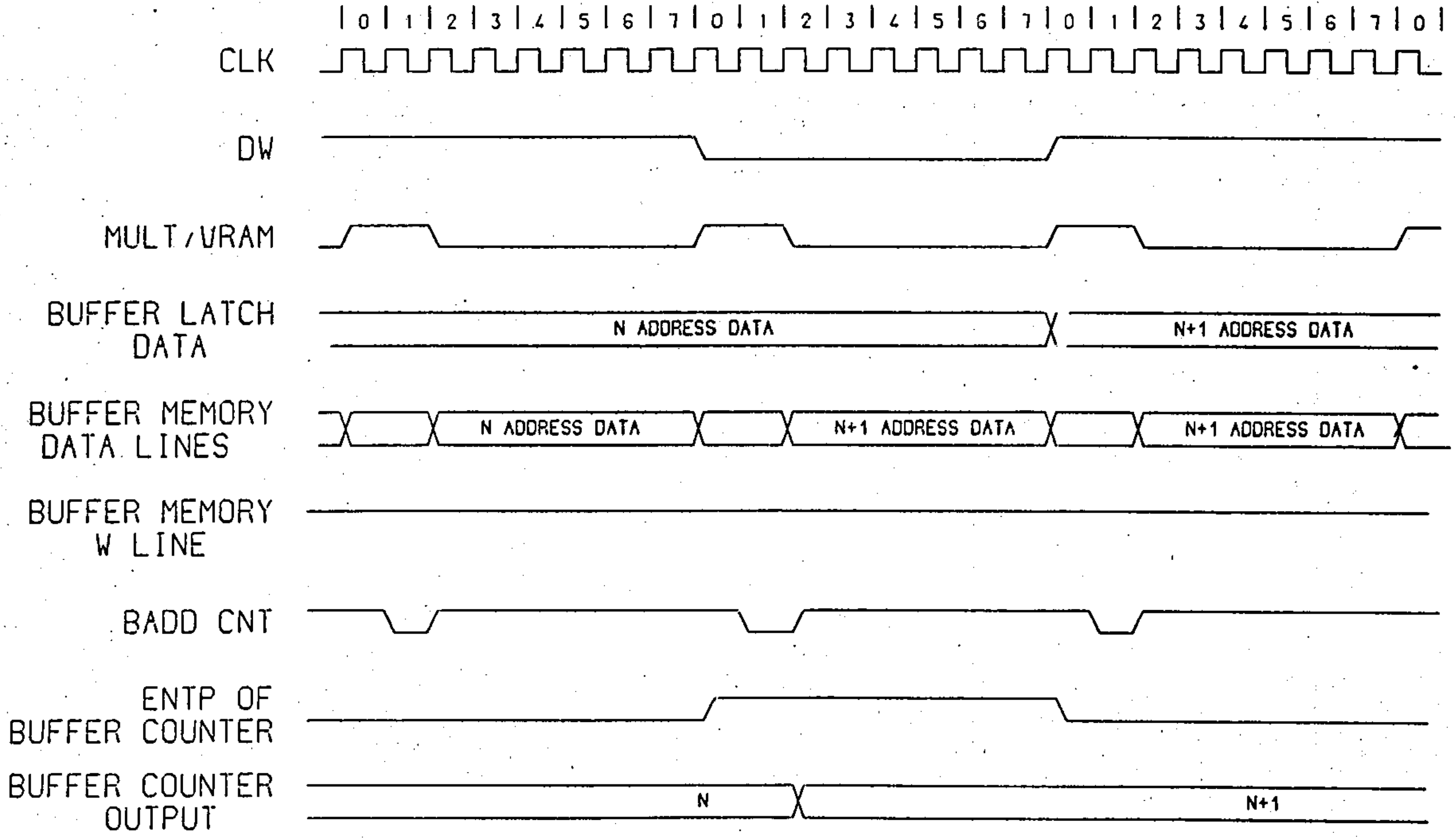


Figure 4.21 : Write to video memory timings

board. The buffer address counter is incremented on the rising edge of the BADD CNT signal. The ENTP signal is the inverted DV signal. It enables the buffer address counter when it is high.

4.3.2.3.1.2. Read from video memory

The timing waveform of the relevant signals involved in the transfer of data from the video memory to the buffer memory is shown in figure 4.22. The data from the video memory is only valid 571 nanoseconds after the rising edge of the GDP access cycle. Thus, the data is only latched by the buffer on the start of the next GDP cycle. It is then written into the buffer memory during VRAM time slot, when MULT/VRAM signal is low. The buffer address counter is incremented by the BADD CLK signal.

4.3.2.3.1.3. Multibus read

Figure 4.23 shows the timing waveform of the relevant signals involved in the transfer of data from the buffer memory to the multibus. When the host computer is reading data from the BRAM, the data output of the buffer memory is latched by the MULT PULSE signal. The MULT PULSE signal is used to disable the buffer latch signal by triggering a monostable 600 nanoseconds after the start of the multibus transfer, indicated by MEMRD. The output of the monostable is latched into a D-type latch, whose Q output is OR-ed with the S/L signal to generate the buffer latch signal. Thus, a high value output on the D-type latch will disable the buffer latch signal.

4.3.2.3.1.4. Multibus write

Figure 4.24 shows the timing waveform of the signals involved in the transfer of data from the multibus to the buffer memory. The transfer is initiated by the multibus MEMWRT

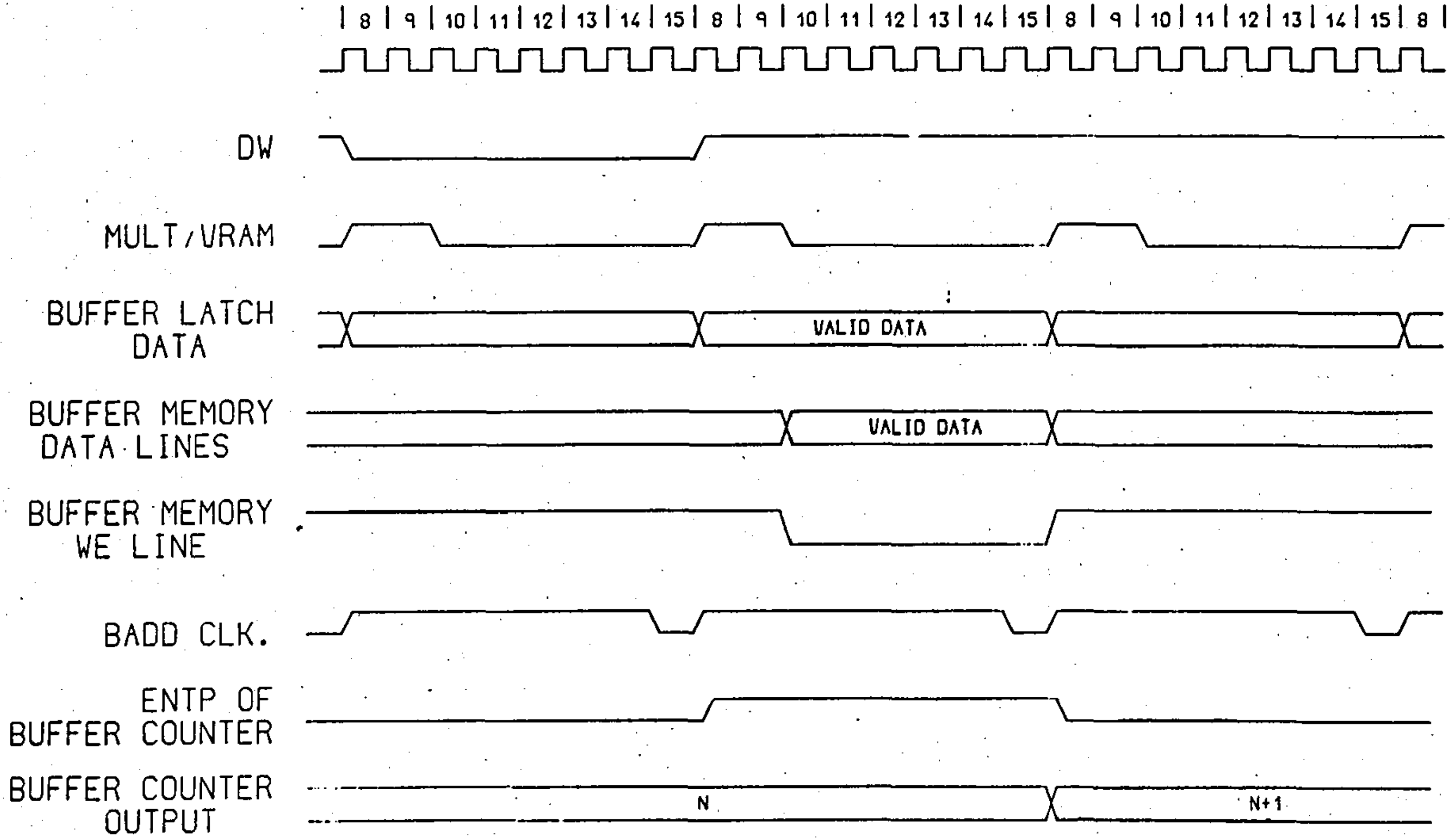


Figure 4.22 : Read from video memory timings

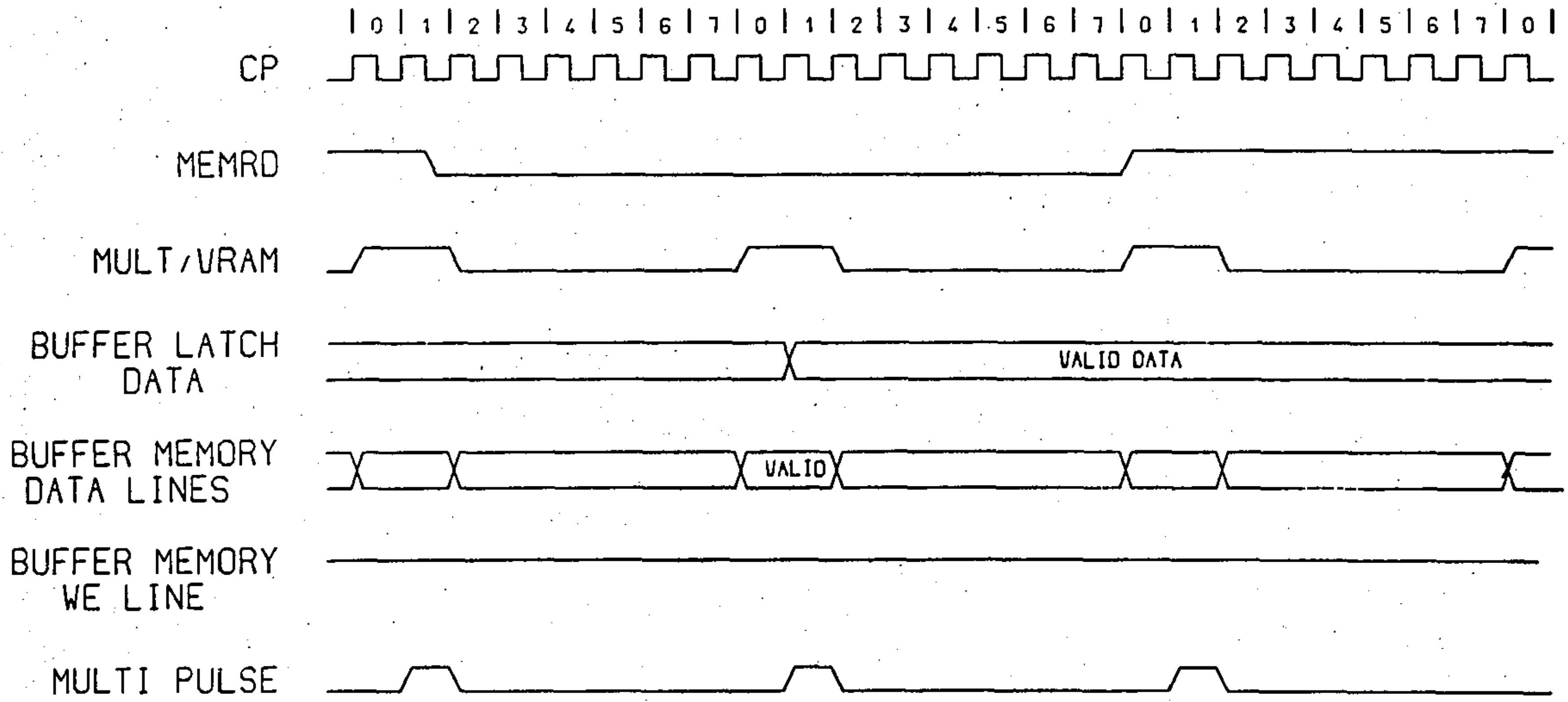


Figure 4.23 : Multibus read timings

signal. Data from the multibus, D0 - DF, is first latched into the multibus write buffers, IC 52 and IC 68. The latched data is written into the buffer memory during the multibus time slot, when MULT/VRAM is high.

4.3.3. Analogue board

A brief discussion of the circuitry will be given in the following paragraphs. For more details refer to reference 65. Figure 4.25 shows a block diagram of the analogue board circuitry.

The video inputs are first amplified. The amplified signals are then fed into a buffer prior to the ADC analogue input. The buffered Y signal is also fed to the input of a sample and hold circuitry. This circuitry samples and holds the video black level for the entire scanline. This value represents the lowest voltage to the input to ADC. It is buffered and used to drive the bottom end of the converter resistor ladder.. The sampling frequency, obtained from the memory board of the ADC is 14 MHz.. 6 or 8 bits flash converters can be used. The outputs of the ADC are then buffered by latches before going to the memory board.

On the DAC side, data from the memory board is first latched. Since the digital input to the DAC is ECL logic, the TTL signals are converted to ECL signals. 6 or 8 bits DAC can be used. This circuitry is replicated for the other two channels, U and V. The sync output from the controller board is added to the analogue signal in the Y channel.

4.4. Discussion

The framestore was designed around the Thompson CSF graphic chip, EF9365. It has a data transfer rate of 1.5 Mbytes per second over the multibus. The design makes use of the

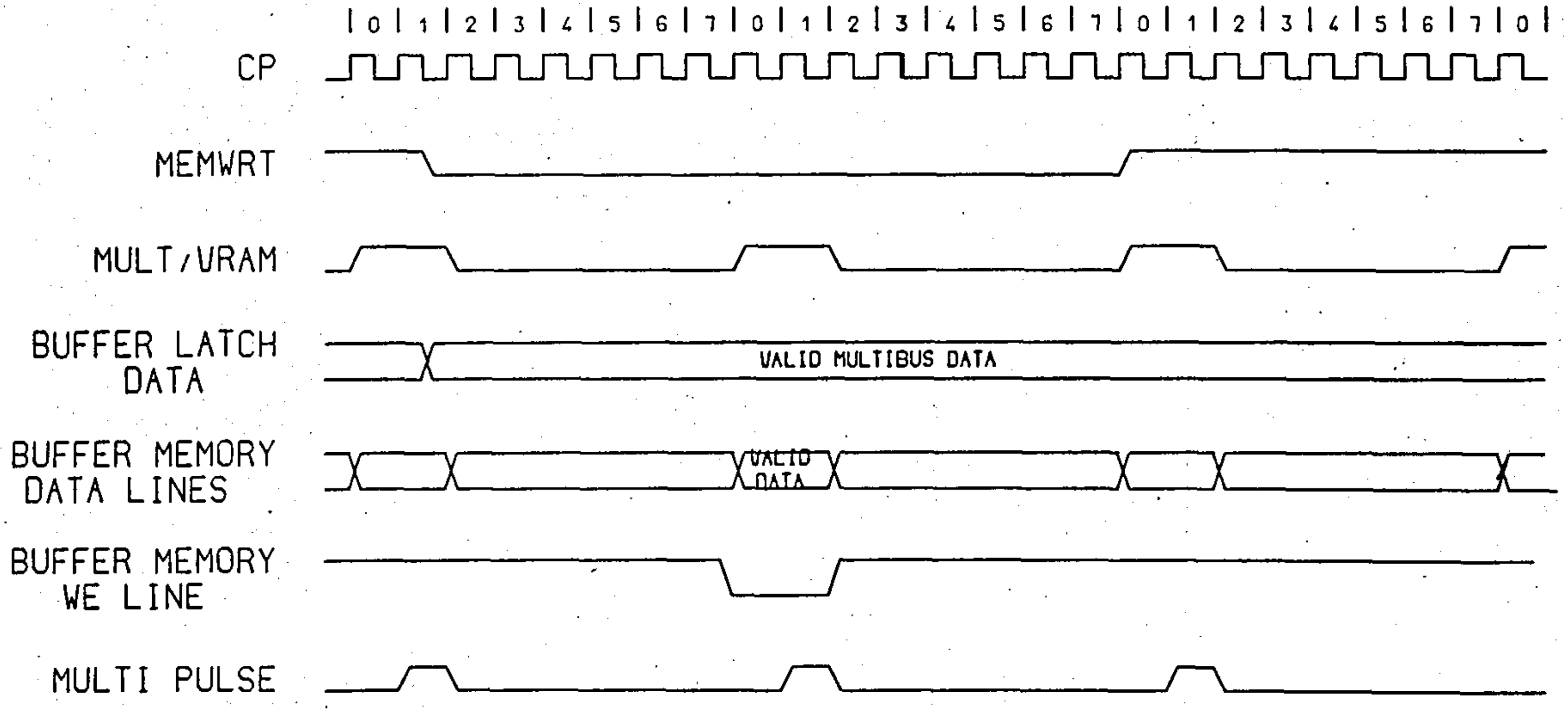


Figure 4.24 : Multibus write timings

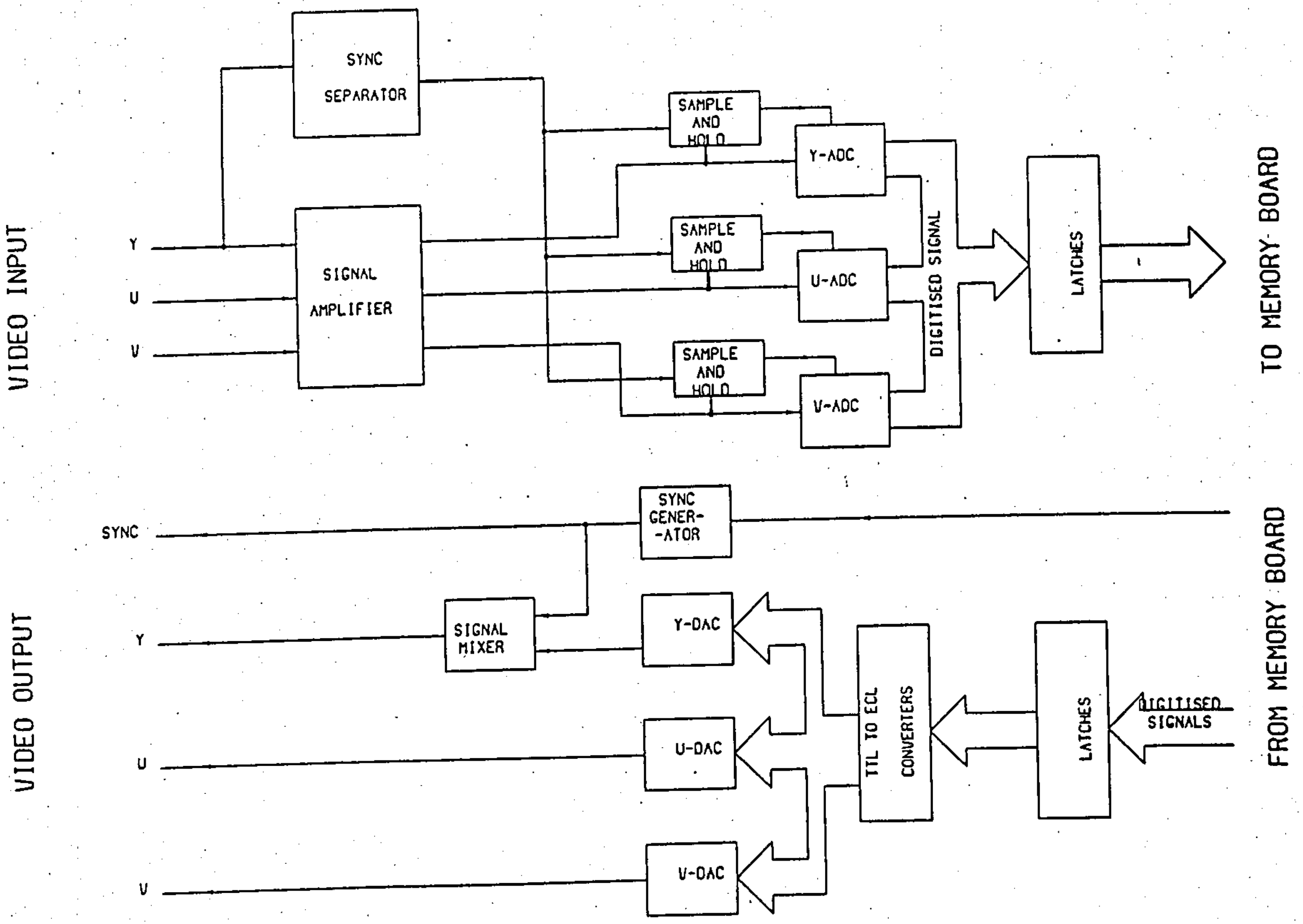


Figure 4.25 : Analogue board block diagram

graphic chip inbuilt vector drawing and character generator circuitry to access the video memory at a fast rate. This helps to reduce the access set up overhead, i.e. the number of registers that need to be setup per pixel access, as a group of pixels can be accessed with a single GDP command. This also enables the framestore to be used as a graphic display terminal, as the framestore can be set such that data written into the video memory comes from a single location on the buffer memory.

Data transfer between the framestore and the host is via an 8 Kbytes buffer memory. This enables the host system to work concurrently with the GDP.

4.5 Note on publication

A paper entitled "A video frame store for Teleconferencing application in Computer Networks.", was presented in the IERE Conference in 1985 held at Loughborough University.

CHAPTER FIVE

NETWORK PROTOCOL ARCHITECTURE

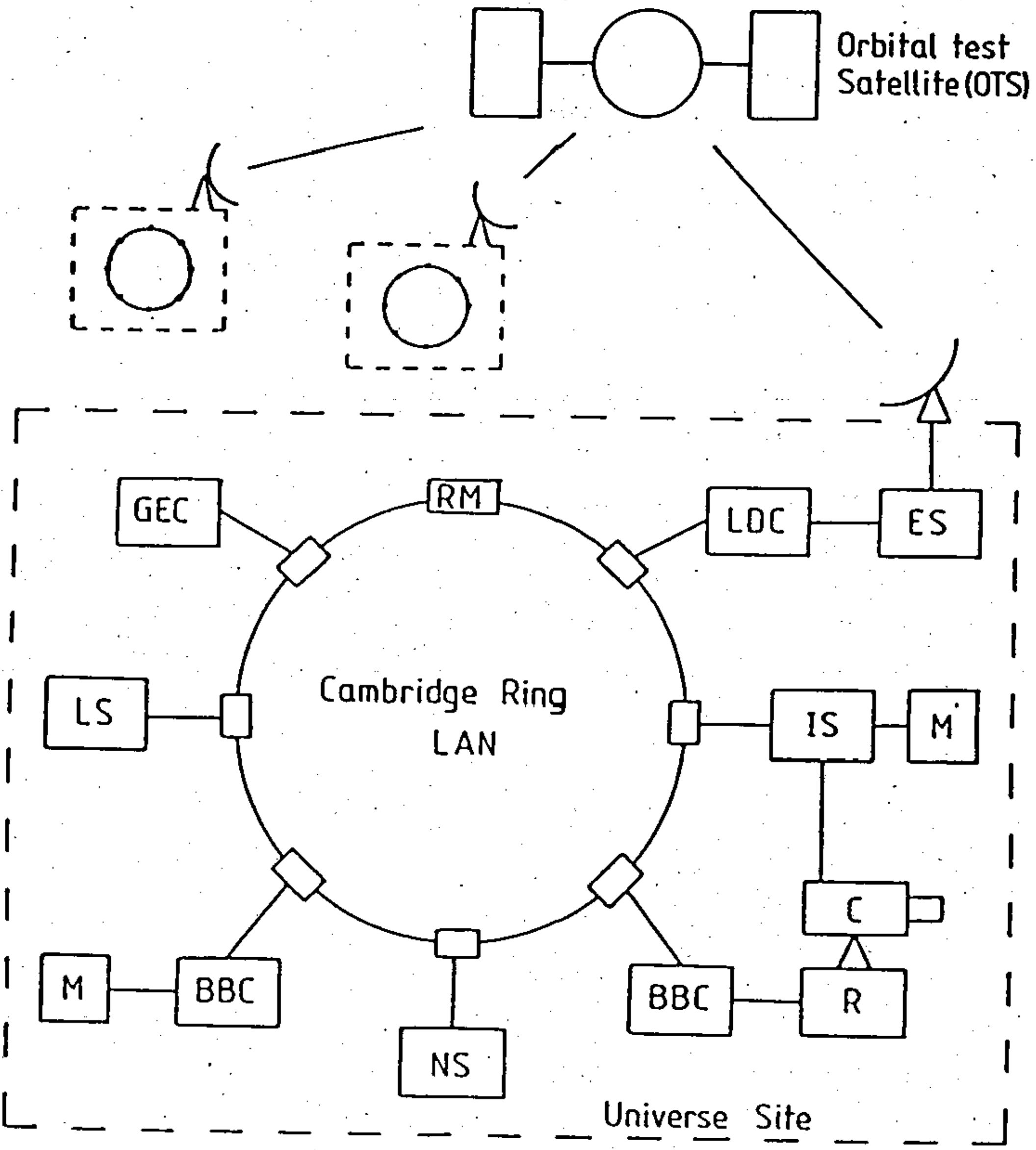
5.1. Introduction

As one of the sites involved in PROJECT UNIVERSE, a Cambridge ring [81] was installed in the Electrical Department at Loughborough University in 1982. The network infrastructure of a Universe site is shown in figure 5.1. The main function of the different stations are as follow:

Nameserver - to provide the name lookup service, giving the ring address of the service required by the user
Bootserver - to provide the bootstrapping facility, loading codes into the user memory
Monitor - to ensure the continued operation of the ring
BBC microcomputer - to provide a common user interface, to image transfer and robot control
Image station - to provide colour image transmission.

The International Standard Organisation(ISO) suggested a Reference Model, the Open Systems Interconnection(OSI) [42] for the exchange of information among systems. The OSI consists of seven layers each layer performing a specific function. The functions of the different layers can be briefly defined as follows:

-the Application layer is the window through which exchange of information occurs between communication users
-the Presentation layer performs certain transformations, e.g. coding
-the Session layer ensures the organisation and synchronisation of data exchange
-the Transport layer ensures the correct end-to-end



KEY

BBC	BBC Microcomputer	LDC	Link Driving Computer
C	TV Camera	M	T.V. Monitor
ES	Earth Station	NS	Name Server
IS	Image Station	R	Robot Camera Mount
GEC	GEC 4090 Computer	RM	Ring Monitor
LS	Universe Large Server		

Figure 5.1 : Individual Universe ring infrastructure

dispatching of information

-the Network layer is in charge of network connection thus allowing the transport layer independence from routing and relaying consideration

-the Data link layer ensures the correct framing of data and sequence control

-the Physical layer is concerned with the transmission of the raw bits over a communicating channel.

Although the protocol supported on the Cambridge ring does not conform exactly with the ISO standard, it can be fitted approximately onto the defined layers as seen below. The Cambridge ring protocol provides standard protocol only up to the Transport layer. Above this layer the protocol is dependent on the service provided. Thus, the protocol discussion will be limited to the lower four layers, below the Session layer.

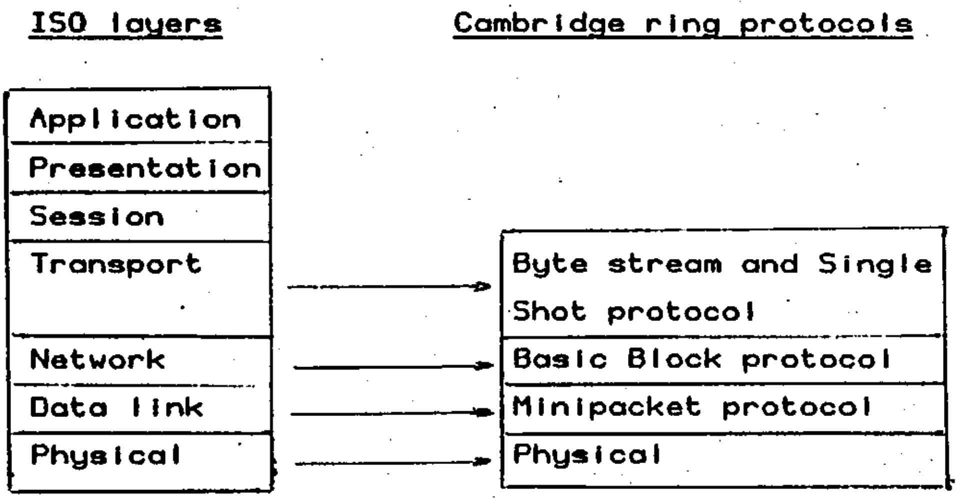


Figure 5.2: Cambridge ring protocol mapping to ISO layers

5.2. Physical Layer

The Cambridge ring is a slotted ring running at a clock rate of 10 Mbits/second. All host systems are interfaced to the ring via a node and they each have a unique address. Each node consists of a repeater and a station unit. The address of the node is stored in the station unit. An access box provides the interface between the node and the device as shown in figure 5.3. Detailed specification of the interface can be found in reference 70.

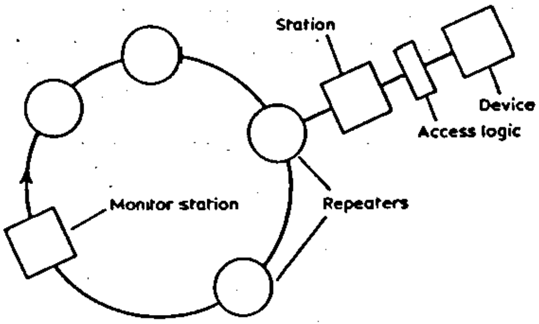


Figure 5.3: Cambridge ring node

5.3. Data link layer

At ring startup, the monitor generates a minipacket and sends it circulating unidirectionally round the ring. The size of the minipacket is 38 bits and the structure of the minipacket is shown in figure 5.4. The first bit is the start bit.

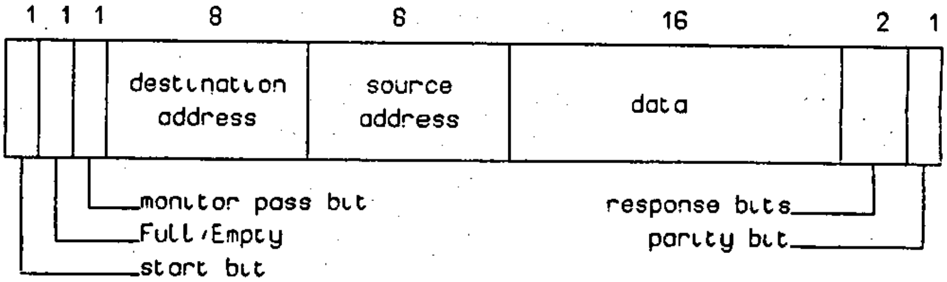


Figure 5.4 : Minipacket structure

It is always set to 1 and is used for the synchronisation of the ring. The FULL/EMPTY bit determines whether a minipacket contains valid information. The monitor pass bit is used for detecting an erroneous minipacket that is continuously circulating the ring. It is set by the transmitting node and is reset when the minipacket passes the monitor. Any minipacket detected as full and has this bit cleared, will automatically be marked empty when it passes the monitor. The transmitting and receiving node addresses are specified by their respective field. The data field contains the true data content of the minipacket. Two bits are allocated to specifying the response of the receiving station. The response is encoded as follows:

Bit nos. 36	37	
1	1	Ignored - No station with the destination address
1	0	Not selected - The destination station does not wish to receive from the source station
0	1	Accept - The destination has successfully read the minipacket
0	0	Busy - The destination is not ready to read the minipacket content

Thus, a successful transfer only occurs when the response bits return a value of 01. The last bit of the minipacket, a parity bit, is used by the receiver station to detect a corrupted minipacket. Corrupted minipacket will be rejected.

When a station has data to transmit, it loads its buffer with the data and waits until its repeater receives an empty minipacket. The repeater will then mark the minipacket as full and fill the minipacket data field with the content of its buffer. The minipacket continues circulating round the ring until it reaches the destination station. The destination repeater reads the data into its buffer and sets the acknowledgement field appropriately. The minipacket is then circulated again until it returns to its sender, which then marks the minipacket as free for other stations to use.

5.4. Network layer

The Basic Block(BB) [54] protocol is a Network layer protocol developed for the Cambridge ring. It is made up of four fields; header, count, data and checksum as seen in figure 5.5.

16 BITS
HEADER
COUNT
PORTNUMBER
DATA
CHECKSUM

Figure 5.5: Basic Block structure

The header is divided into three fields as shown below.

Field A	Field B	Field C
< 1001 >	< two bits>	< ten bits>

Figure 5.6: Basic Block header

The first field, field A, contains a bit pattern of four bits, indicating the start of the block. Field B consists of two bits defining the Basic Block type as shown below:

Bits value	
0	Type 0 - the number of words in the data field minus 1 is found in field C of the header. The checksum field is set to 0.
1	Type 1 - the number of words in the data field minus 1 is found in field C of the header.
2	Type 2 - the data is 10 bits and is found in field C. Its Basic Block consists of only one minipacket. Not used.
3	Type 3 - the data byte count is found in the count field. Field C is then used to define whether or not it is a datagram block.

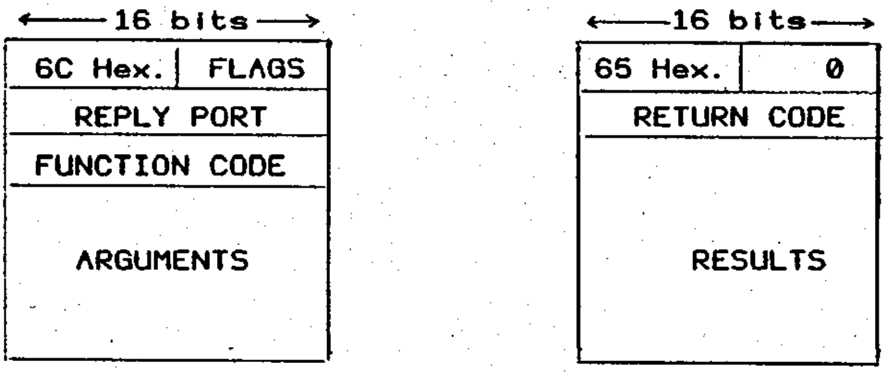
Most sytems on the Universe project use only Type 3 Basic Blocks. The following discussion will be limited to Type 3 Basic Blocks only. The count field specifies the number of bytes of data in the data field minus one. In the case of an odd number of bytes in the block, a pad character is appended to the data stream. The maximum number of bytes per Basic Block is 2048. The route field, byte 4 and 5, consists of 12 valid bits defining the portnumber specified by the receiving system. The top four bits are set to 0.

The last word in the block is a 16 bit checksum. It is produced by taking each word in the block, including the header, count, and port field, and forming the end-around-carry sum of them. The carry bit is then added after each addition. A more complete description of the Basic Block is found in the Logica Manual [54].

5.5. Transport layer protocol

Lying above the Basic Block protocol is the Transport layer protocol. There are two sets of protocol in this layer. Single Shot Protocol(SSP) and Byte Stream Protocol(BSP) [47].

The SSP is used when all the data in a transaction can be contained within a single Basic Block. The station initiating the transaction, i.e. the client, sends an SSP request block, SSPREQ, to the public port of the station that provides the service it wants i.e. service station. The service station will put the data requested in a Basic Block, SSPRPLY, and send it back to the client through the portnumber specified by the reply port field. The format of the SSPREQ and SSPRPLY basic block is shown in figure 5.7. The SSPREQ/SSPRPLY protocol is frequently used for accessing the Server(NS).



SSPREQ SSPRPLY

Figure 5.7: SSPREQ and SSPRPLY basic block

The BSP protocol is a lightweight protocol used when a large amount of data needs to be transferred. The client sends an OPEN basic block to the service station which acknowledges with an OPENACK basic block. The format of the OPEN and OPENACK blocks are shown in figure 5.8. The connection port number, specified in the OPENACK, will be used for future communication between the two stations. Thus, the OPEN/OPENACK sets up a virtual circuit between the two systems which will remain active until closed by the systems.

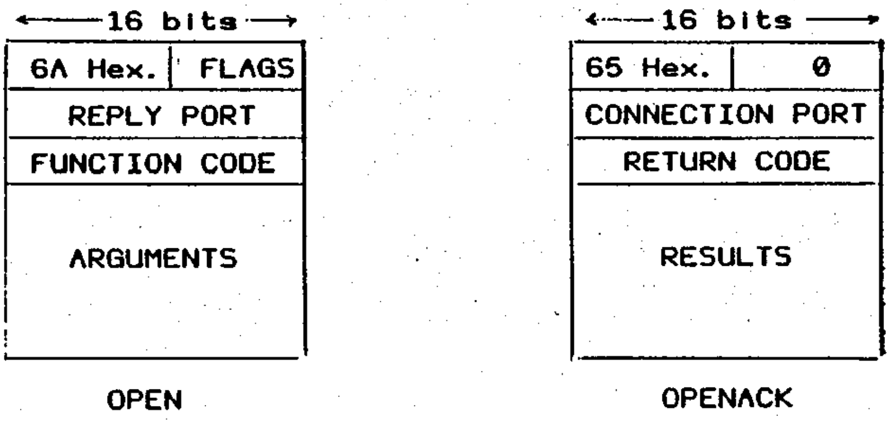


Figure 5.8: OPEN and OPENACK basic block

The FLAGS field in both the SSPREQ and OPEN block provides information about the service necessary for the connection as shown below:

Bits	
0-2	specify the protocol 0 - non standard 1 - OPEN 2 - SSP 3 - the name is a machine name, so that the port and function fields are meaningless. Use by Nameserver only. 4 - Datagram
3	set if the function code is not specified
4	set if an extended timeout, for SSP and OPEN, is needed
5	set if the service is found in the remote site
6	set if the service can receive and transmit Type 3 Basic Block
7	set if the service can receive and transmit Type 0 and 1 Basic Block

The procedure for setting up a virtual link between two image stations will be used to illustrate the SSP and BSP protocols. The image transfer function is initiated by the controller station, BBC A. Assume that Image station A is the receiving station and Image station B is the transmitting station. The steps are as follows:

1. BBC A sends an SSP, containing the name of the Image station A, to the Nameserver.

2. The Nameserver sends an SSPRPLY, containing the address and portnumber of Image station A, back to BBC1.

3. BBC A sends as SSP, containing the function required, to Image station A.

4. Image station A acknowledges the reception of the BB with an SSPRPLY.

5. Image station A sends an SSP to the Nameserver asking for the address of Image station B.

6. The Nameserver replies with an SSPRPLY containing the address of Image station B.

7. Image station A then sends an OPEN to Image station B using the address supplied by the Nameserver.

8. Image station B replies with an OPENACK

The different steps mentioned above can be visualised in figure 5.9. The numbers in the figure represent the step numbers.

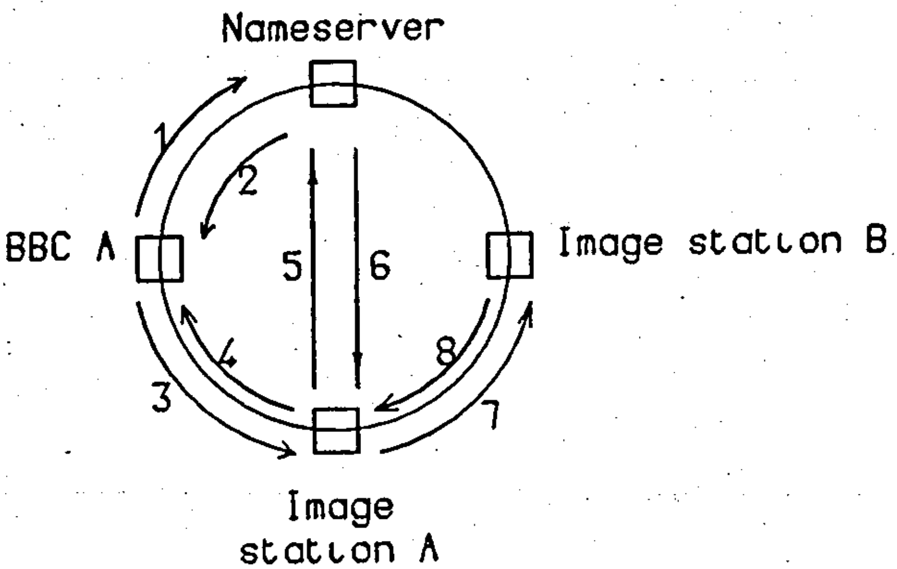


Figure 5.9 : Virtual link setup

5.6. Performance

The performance of the protocol [75] is measured in terms of the data throughput, error recovery and flow control.

5.6.1. Data Throughput

The data throughput of the system refers to its transmission rate of useful data. The Cambridge ring operates at a rate of 10MHz, but there is a lot of overhead in a minipacket. Out of the 38 bits in a minipacket only 16 bits are available for carrying data as shown in figure 5.4. Thus, the ring bandwidth is only (16/38) × 10 Mbits = 4.2 Mbits per second. For a ring with 'n' packets and 'm' active transmitters, each transmitting as fast as possible, and with all receivers accepting packets without delay, the point-to-point bandwidth per transmitter is

$$
\begin{aligned}
Tp &= 4.2M / (n + m) \quad & m > 1 \\
&= 4.2M / (n + 2) \quad & m = 1
\end{aligned}
$$

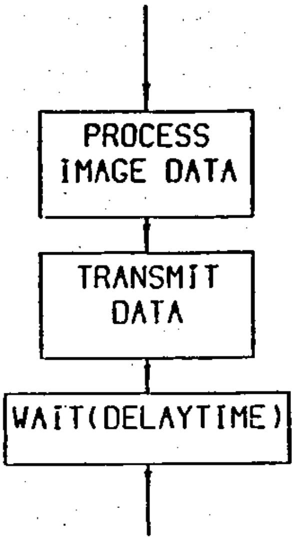
The above equation assumes that the active stations use all the minipackets allocated to them. But in the case where some stations are not as active, the excess bandwidth left by them are shared equally amongst the other active stations. Thus, it permits a form of bandwidth stealing to occur on the network.

5.6.2. Flow control

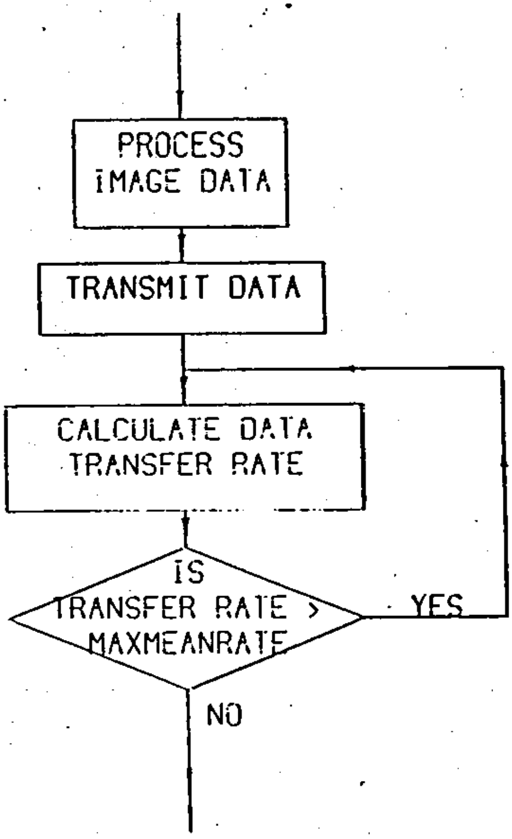
The aim of the flow control mechanism is to try to match the transmitter station's transmission rate with the receiver station's acceptance rate [21,43]. Flow control mechanisms can be broadly divided into : protocol and interface mechanisms.

The minipacket protocol provides a form of flow control using the response bits, bit 36 and 37. If the response bits show an unsuccessful transmission it will try to transmit the minipacket again when it receives the next free minipacket. The Basic Block protocol does not provide any form of flow control.

Two interface flow control mechanisms will be described. The first method sets a time delay between each Basic Block transfer as shown in figure 5.10(a). The time delay is set to ensure that the receiver station is always ready to accept the data. The other flow control mechanism is maxmeanrate as shown in figure 5.10(b). The maxmeanrate is specified by the receiver station. The transmitter checks its transfer rate every time it has transferred N bytes of data. If the transfer rate is higher, it will wait until it is lower, before it continues processing the image data again.



(a) Delaytime flow control mechanism



(b) Maxmeanrate flow control mechanism

Figure 5.10 : Flow control mechanism flowchart

CHAPTER SIX

IMAGE STATION

6.1. Introduction

The image station provides multilevel colour image communication services for the Cambridge ring. Two image transfer services are provided: a medium resolution slow-scan transfer and a high resolution freeze frame transfer. The freeze frame transfer service is mainly used for transmitting still pictures, e.g. diagrams and charts, while the slow-scan transfer service is used basically for transmitting a sequence of images, e.g. face to face communication and surveillance.

The BBC microcomputer is used as the controller station, providing a menu driven user interface for controlling the image station services. The BBC microcomputer is connected to the ring via the SEEL/ORBIS program interrupt interface which provides Basic Block (BB) level protocol in firmware.

6.2. System description

6.2.1. Hardware

The image station consists of three multibus boards: 86/30, VMI-1 and framestore controller card, interconnected as shown in figure 6.2. They are housed, together with the power supplies and fans, in a 6U type cabinet. The VMI-1 ring interface card is connected to a Logica VTS Polynet ring node via a 50 way ribbon cable. The photograph in figure 6.1. shows the setup of the image station.

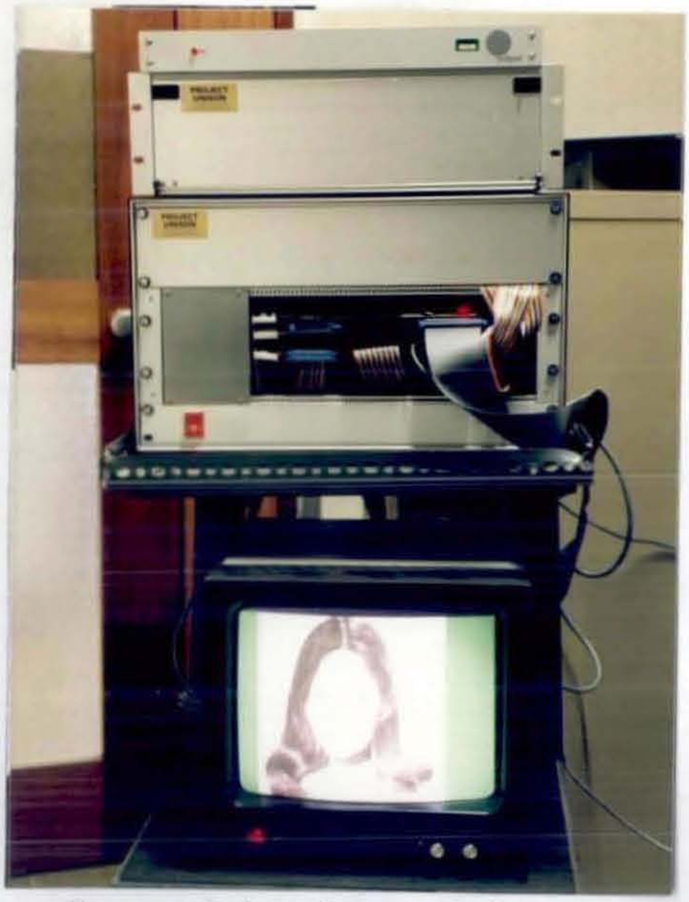


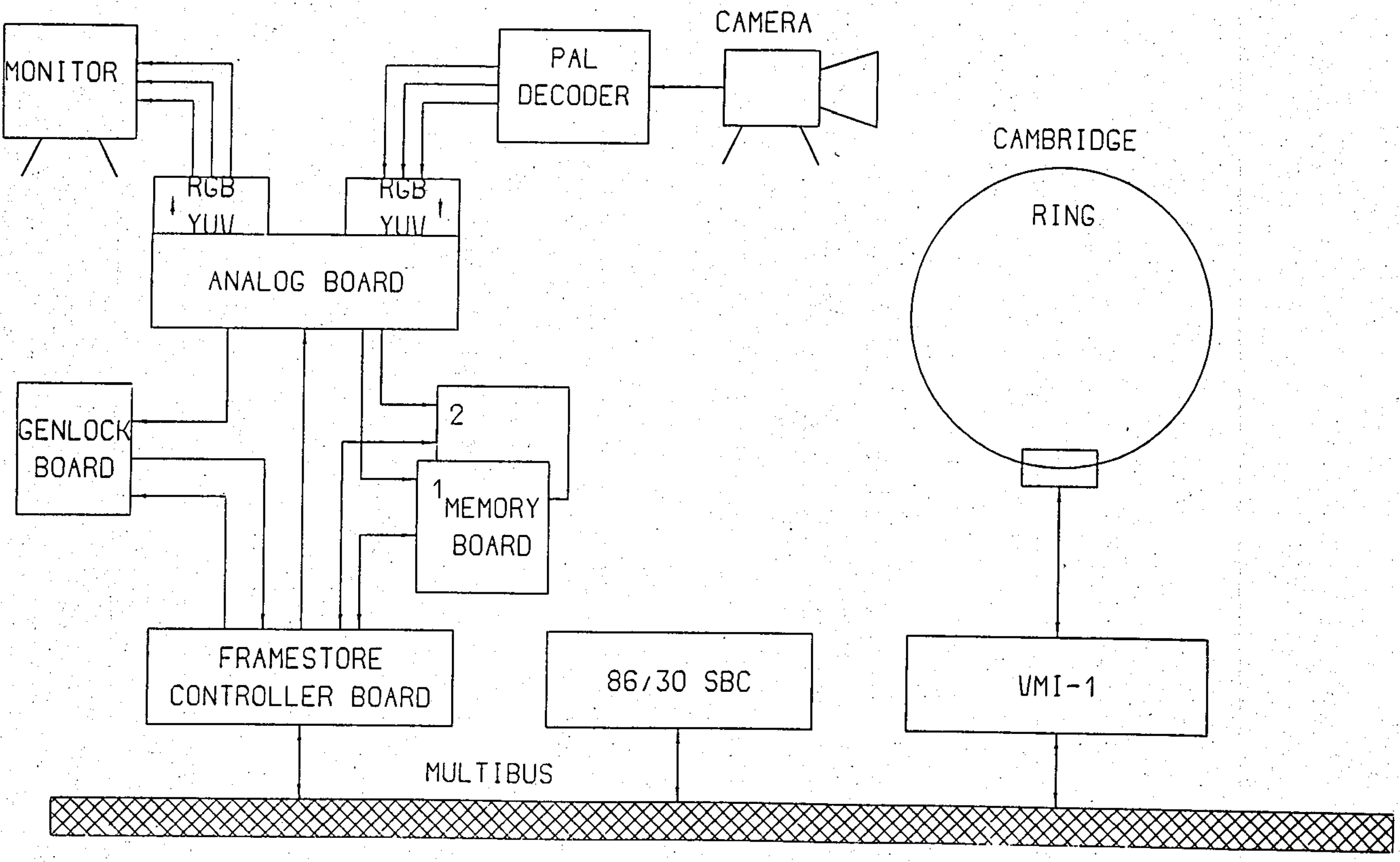
Figure 6.1 : Image station

6.2.1.1. 86/30 single board computer[36]

The 86/30 is a single board computer system designed around the 16 bit 8086-1 microprocessor. The microprocessor has a clock frequency of 8 MHz. It has 256 Kbytes of dual port memory, of which 64 Kbytes are used by the VMI-1 as shared memory, for transferring data to and from the ring. It also has 32 Kbytes of EPROM space for storing the program codes.

The 86/30 also has two independent programmable interval timers, timer 1 and timer 2. The output of timer 1 is used as the baudrate clock for the serial I/O chip while timer 2 is used as a realtime clock to measure the data transfer rate and frame update rate of the image station.

Figure 6.2 : Image station block diagram



6.2.1.2. VMI-1 ring interface

The VMI-1 provides a high performance interface between the 80/30 and a Logica VTS Polynet node. It contains the firmware for supporting the Basic Block protocol. The VMI-1 uses a direct memory access to the 86/30 memory, to transfer data to and from the 86/30 memory. The VMI-1 signals the completion of the previous request by interrupting the 86/30, using interrupt vector 2. Details of the Basic Block setup procedure required by the VMI-1 are found in reference 55.

6.2.1.3. Framestore

The L.U.T. framestore provides the frame acquisition and display facility of the image station. It is configured for colour image display and acquisition. The input video signals: red (R), green (G)and blue (B), are transformed to luminance (L) and chrominance (U,V) signals before being digitized. The transformation matrix for converting RGB signals to YUV signals and vice versa given by the Phase Alternating Lines(PAL) television standards [35] are:

$$
\begin{aligned}
Y &= 0.299R + 0.587G + 0.114B \\
U &= -0.147R - 0.289G + 0.437B \\
V &= 0.615R - 0.515G + 0.100B
\end{aligned}
$$

and

$$
\begin{aligned}
R &= 1.000Y + 1.140V \\
G &= 1.000Y - 0.394U - 0.581V \\
B &= 1.000Y + 2.020U
\end{aligned}
$$

The two main reasons for digitizing the luminance and chrominance signals instead of the three primary colours (RGB) are:

(a) the bandwidth of the chrominance signals, U and V, are lower than the luminance signal. Thus, a 2:1 ratio of luminance to chrominance can be used for transmitting the colour image.

(b) significant changes in chrominance are usually accompanied by significant changes in luminance [49]. Thus, changes can be detected by checking the luminance component only.

The framestore has two video memory boards, enabling it to store up to a maximum of two frames of 512 pixels by 512 lines. Each pixel is represented by 16 bits, 6 bits for luminance and 5 bits for each of the chrominance signals as shown in figure 6.3.

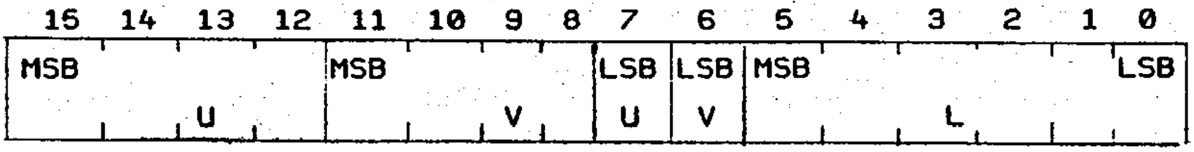


Figure 6.3. Digitised video bit arrangement

A commercially available GEN-LOCK board, CLU603B, is used to synchronise the framestore video signals to the video input signals. It also generates a 14 MHz. signal for the framestore controller board PROM sequencer.

The framestore controller board registers are located between port address 0 to FFH, excluding port addresses between 50H to 60H which are used by the VMI-1. The base address of the framestore buffer memory is set to 80000H.

6.2.2. Software

The system software occupies the top 32 Kbytes, address F8000 - FFFFF Hex., of the 86/30 memory. The software was developed using the 8086 in-circuit emulator attached to the Intel MDS system and can be divided into three sections as follows:

- Ring drivers, which deal with request for transmission and reception of Basic Block
- Framestore drivers, which deal with the transfer of data to and from the L.U.T. framestore as well as the coding and decoding of the image data
- the Main program, which analyses the task to be performed and calls the appropriate routines needed to perform the tasks.

The system software was written in Pascal and 8086 assembly language. General housekeeping tasks, main program and ring drivers, were written in Pascal while the framestore driver routines were written in assembly language. This is necessary to speed up the image data transfer rate, and also to ensure that the main program and ring driver routines need not be rewritten, if a different framestore is used by the image station in the future.

6.3. Image coding

6.3.1. Scanning and Addressing

Different scanning and addressing techniques are used to address pixel data, for freeze frame transfer and slow-scan transfer.

(a)Freeze frame transfer
During freeze frame transfer, the pixels are transmitted a scanline at a time. The scanlines are transmitted in increasing order of y address, i.e. from the bottom to the top of the screen. Each scanline is addressed by the x and y coordinates of the first pixel along a scanline.

(b)Slow-scan frame transfer
During slow-scan transfer the frame is divided into regular blocks, of 8 by 8 pixels. Each block is specified by the x

and y coordinates of the pixel at the bottom right hand corner of the block. The order in which the pixels in the block are coded is shown in the figure below.

7	57	58	59	60	61	62	63	64
.	.	.	.	.	.	.	.	.
2	17	.	.	.	.	.	.	.
1	9	10	11	12	13	14	15	16
line 0	1	2	3	4	5	6	7	8

Figure 6.4. Coding order

The slow-scan transfer has two frame update modes, conditional update and forced update, running concurrently. In the conditional update mode, all the blocks in a frame are tested, and only those blocks which have changed significantly will be coded and transmitted. The blocks are scanned in an outward spiralling manner, starting from the centre of the screen (x coordinate = 256, y coordinate =256) as shown in figure 6.5. This scanning process was adopted because most of the active areas lie at the centre of the screen, and thus can be updated in a more continuous manner than the more conventional top to bottom block scan. The scanning process is restarted after all the blocks in the screen have been tested.

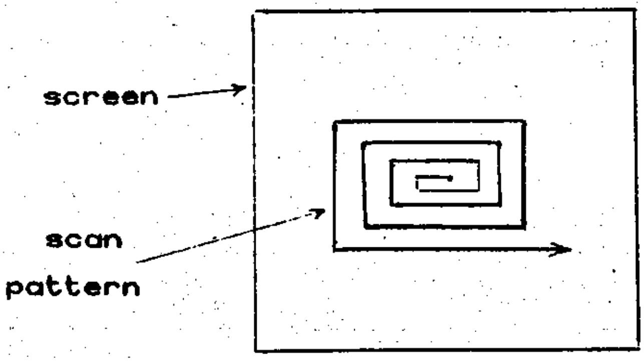


Figure 6.5. Block scanning pattern

In the forced update mode, only a portion of the blocks in a frame are coded and transmitted within a frame update period. The scanning process of the forced update mode was from left to right followed by top to bottom movements.

6.3.2. Coding techniques

The coding technique adopted for coding the image data for the freeze frame and slow-scan transfer are different.

(a) Freeze frame
In the case of the freeze frame image transfer all the data is coded and transmitted using PCM coding, 6 bits for luminance and 5 bits for each of the chrominance signals. This is because the receiver station requires an exact picture of the image in the transmitter station. The resolution of the image is set to 512 pixels by 512 lines to provide as detailed a picture as possible.

(b) Slow-scan
Block conditional replenishment coding technique is used to code the image data during slow-scan image transfer. In this form of coding, the frame is first divided into square grid blocks and each block is individually compared to the block, at the same spatial location, in the reference frame to check if it satisfies the criteria of a changed block. The pixels of those blocks which satisfy the criteria will then be coded and transmitted. A block is classified as a moving area block if it has either an average frame-to-frame luminance difference >= 2 or maximum frame-to-frame luminance difference >= 8 as shown in figure 6.6.

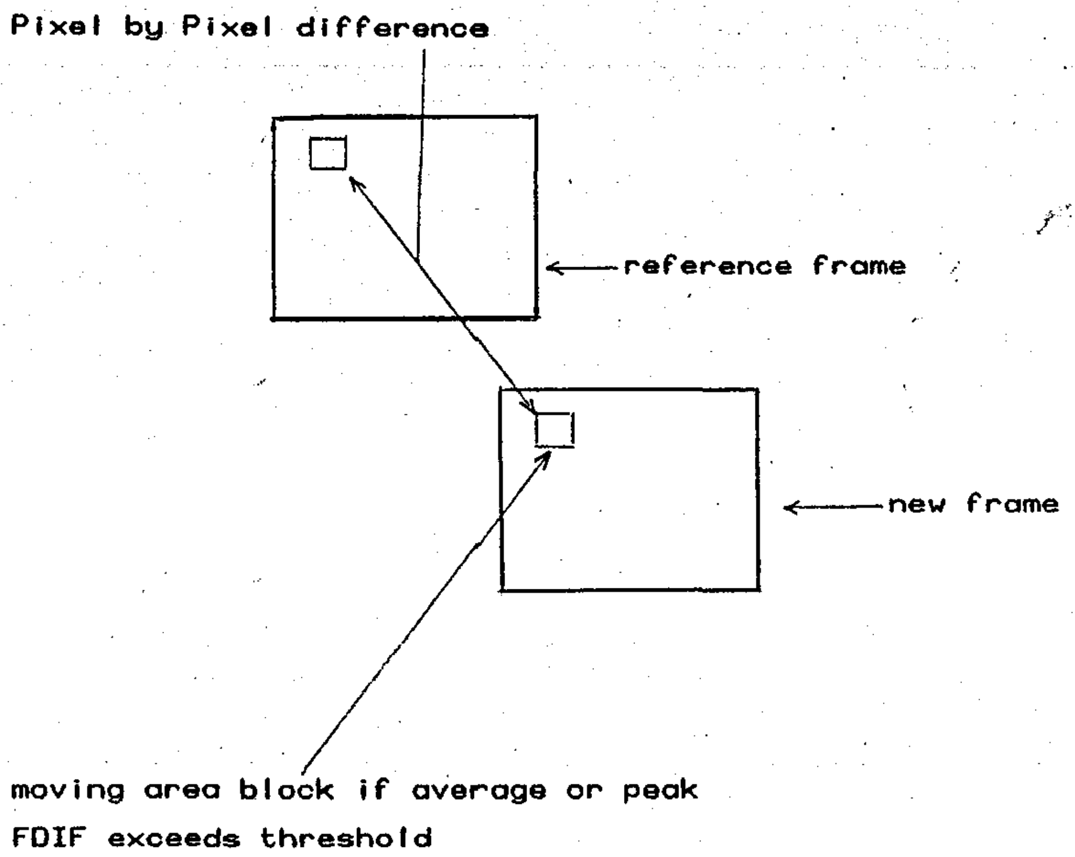

Pixel by Pixel difference
reference frame
new frame
moving area block if average or peak FDIF exceeds threshold

Figure 6.6. Block Conditional Replenishment

The image station slow-scan transfer has three modes for coding the pixels of moving area blocks. The choice of coding is controlled by the receiver station using the coding control strategy described in the next section.

In mode 3, PCM coding (6 bits for luminances and 5 bits for each of the chrominance signals) is used to code the block pixels. Since the bandwidths of the chrominance signals are much lower than the luminance signal, the chrominance signals of alternate pixels along a scanline are transmitted, i.e. even numbered pixel chrominance signals will not be transmitted (refer to figure 6.4). At the receiver the chrominance signals are repeated for those pixels which do not have the accompanying chrominance signals.

In mode 2, 2:1 horizontal subsampling mode [5], the luminance and chrominance signals of alternate pixels along a scanline, i.e. odd numbered pixels (refer to figure 6.4) in the block, are coded with 6 bits for luminance and 5 bits each for the chrominance signals. At the receiver station the missing pixel's luminance signal is restored by taking the average value of the two adjacent pixels along a scanline, while its chrominance data is obtained from the previous pixel along the scanline.

In mode 1, 4:1 subsampling mode, alternate pixels along a scanline, as in the 2:1 subsampling mode, and alternate lines, i.e. odd lines in the block (refer to figure 6.4), are not transmitted. At the receiver, the luminance signal of the missing pixels of the transmitted line are first generated by taking the average value of its two adjacent pixels, the same as in 2:1 subsampling mode. The missing lines luminance data is generated by taking the average value of its adjacent pixels in the y axis.

A forced updating technique, where blocks are unconditionally updated, is used to ensure that data errors on the screen are corrected after a fixed period of time. The number of blocks updated per frame are specified by the controller station. The pixel data is coded with PCM coding, each pixel represented by 16 bits - 6 bits luminance signal and 5 bits for each of the chrominance signals.

6.3.3. Coding Control strategy

The coding control strategy refers only to the coding of slow-scan image data. It determines the coding technique used by the transmitting station. The strategy adopted was to try to maintain a constant frame update rate, given the network condition and the receiver image data acceptance rate.

Image data transferred by the image station during slow-scan transfer service is coded in one of three modes. The choice of coding mode depends on the following factors:

(a) maxmeanrate of the transfer
(b) percentage of frame updated
(c) data lost in transmission
(d) frame update time

The maxmeanrate is defined by either the controller station or the receiver image station, whichever is lower. It specifies the maximum data transfer rate, in terms of number of bytes per 1/40 of a second, of the transmitter station. At the receiver station it is used to set the threshold, in terms of percentage of moving area blocks, to switch between the modes as shown in figure 6.7. The following conventions have been used in the diagram:

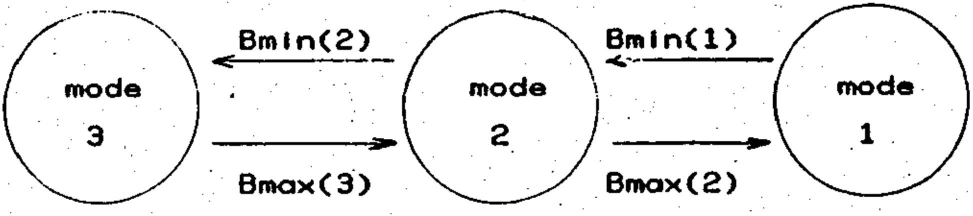


Figure 6.7 : Coding mode switching

-Bmax(n), is the threshold, in terms of number of blocks updated, for switching from coding mode 'n' to a lower coding mode
-Bmin(n), is the threshold, in terms of number of blocks updated, for switching from coding mode 'n' to a higher coding mode.

The Bmax(n) and Bmin(n) values are related to the maxmeanrate and are calculated as follows:

$$Bmax(n) = \frac{maxmeanrate * 40}{(6 + (16*number\ of\ pixels\ in\ a\ block*n)/4)}$$

$$Bmin(n) = \frac{maxmeanrate * 32}{(6 + (16*number\ of\ pixels\ in\ a\ block*n)/4)}$$

Traffic congestion on the ring causes two effects: data lost in transmission and slower frame update rate. The former effect is detected by a break in the number sequence of BBNUMBER while the latter is detected by the timer at the receiver station. When this occurs the image station will switch to a lower mode to reduce the bit-rate. Figure 6.8 shows the flow diagram of the coding control algorithm at the receiver image station.

6.4. Image station protocols

The protocols used by the image station can be classified under three main headings: command protocol, path setup protocol and image transfer protocols. In the following discussion, the following convention will be used:

- Remote station, refers to the image station which transmits image data
- Local station, refers to the image station which receives and displays image data

6.4.1. Command protocol

Command protocol refers to the communication between the controller station and the local image station using the Single Shot protocol (refer to chapter 4). The controller station sends an SSPREQ block to the local image station with the function field containing the service required or task to be carried out as shown below:

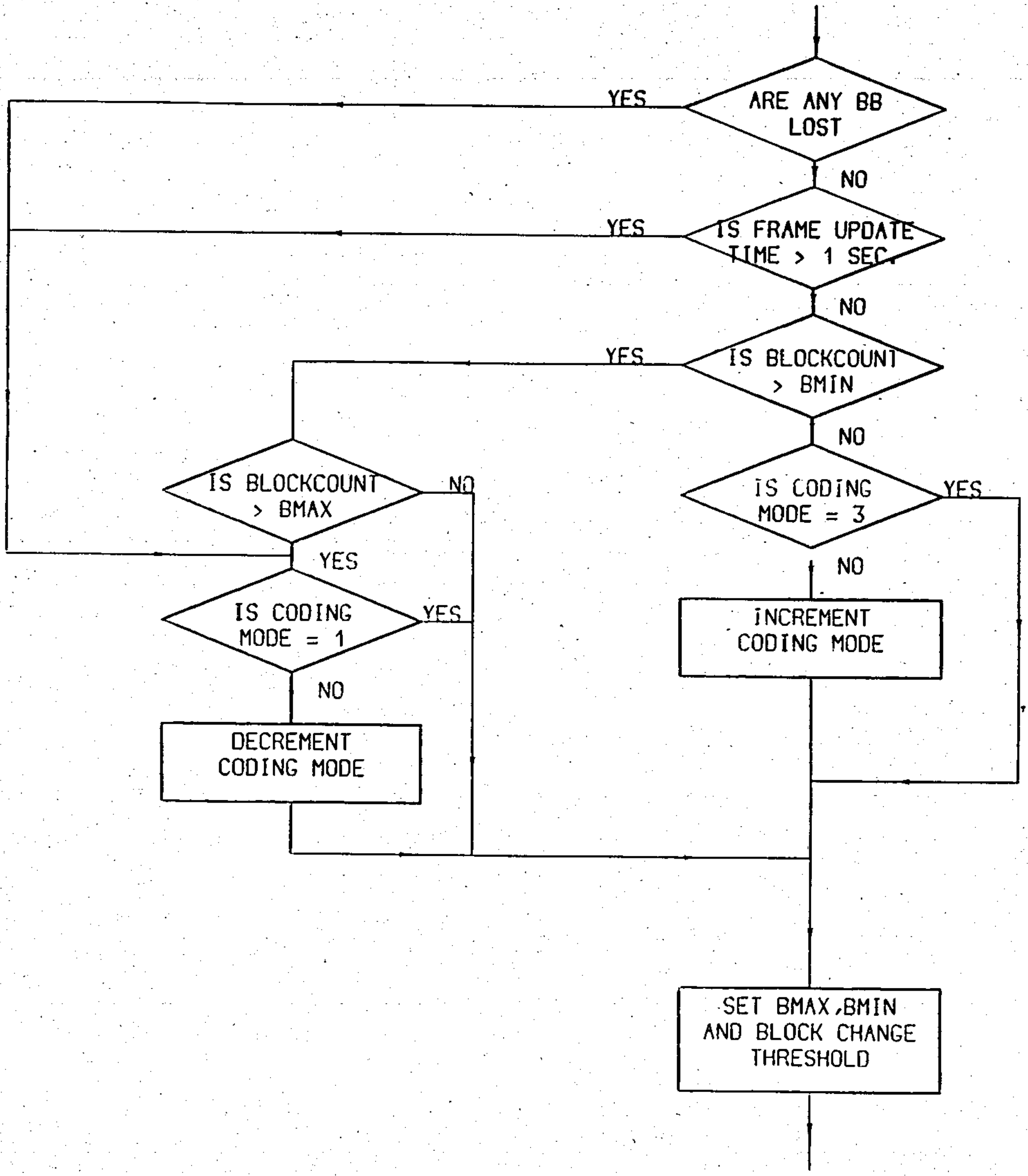


Figure 6.8 : Coding control program flowchart

Function code	Tasks
1	Grab a frame, frame number specified in the argument field
2	Display the frame specified in the argument field
3	Freeze frame transfer
4	Slow-scan transfer
5	Stop slow-scan transfer

The argument field contains the name of the remote station and the specification of the picture to be transferred.

If function 1 or 2 is selected, the local image station will send an SSPRPLY on completion of the task. But if function 3 or 4 is required, the SSPRPLY is sent immediately after it has read the contents in the argument field. Details of the content of the SSP blocks are found in appendix A.

6.4.2. Path setup protocol[2,25]

As the name implies the aim of the protocol is to set up a bidirectional virtual link between the local and remote station. The protocol can be split into two stages: getting the node address of the remote station from the nameserver and establishing the virtual link with the remote station.

When image transfer is required the local station sends the name of the remote station, contained in the SSPREQ block, to the nameserver station using the Single Shot protocol. The nameserver will then send a reply, SSPRPLY block, with the node address of the remote image station in its result field.

Using the node address supplied by the nameserver, the local image station will transmit an OPEN Basic Block to the remote image station. A virtual link will be established once the local image station receives an OPENACK Basic Block from the remote station. This protocol has been described in detail in chapter 5.

6.4.3. Image transfer protocols

The image transfer protocols are divided into freeze frame and slow-scan transfer. Flow control mechanisms and lost data recovery mechanisms used by the protocols will also be discussed.

6.4.3.1. Freeze frame protocol

The freeze frame protocol is used for the transmission of a single frame. The resolution of the image is set to 512 pixels by 512 lines, each pixel represented by 16 bits. Detection and correction of data lost in transmission is essential in this protocol. This is because the picture will be scrutinised more closely as it will be displayed continuously at the receiver image station until the user wants another frame or service. To ensure the reception of all the lines in a frame the local station keeps a tally of all the lines received and will request for retransmission of the missing lines at the end of each frame. The protocol for freeze frame data transmission is as follow:

1. The local image station set up a tally of requested lines.

2. The local image station sends the linenumber of all the lines requested to the remote image station.

3. The remote image station sends image data of all the requested lines to the local image station.

4. The local image station keeps a tally of all lines received

5. When the remote image station has sent all the requested data it sends an End_of_frame Basic Block to the local image

station.

6. The local image station compares the tally of requested lines with that of received lines. If it has not received all the data it goes to step 2 again, otherwise it goes to step 7.

7. The local image station sends stop_transfer block to the remote image station to indicate the end of transmission.

The sequence of the transfer operation is shown in the figure 6.9. The numbers correspond to the numbering of the steps described above.

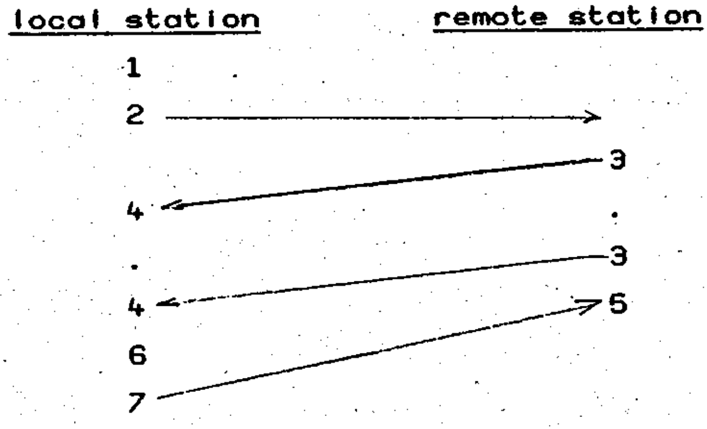


Figure 6.9 : Freeze frame transfer sequence

6.4.3.2. Slow-scan transfer protocol

The requirements of the slow-scan transfer protocol are very different from that of the freeze frame protocol. In the case of slow-scan transfer it is not necessary for the remote station to retransmit image data lost in transmission as the screen is continuously updated. The aim of the slow-scan transfer protocol is to provide the user with a continuous stream of images as fast as possible until the controller station asks it to stop. The procedure is as follows:

1. The local image station sends a special BB, INITF-BB, to the remote image station. The INITF-BB contains the coding mode and the block segmenter thresholds.

2. After receiving the INITF-BB, the remote image station starts sending the coded data. This is repeated until the entire frame has been processed.

3. The remote image station sends an End-of-frame BB to the local image station.

4. The local image station checks port 2 for a command to stop the transfer. It also checks if two consecutive timeouts have occurred. If either conditions has occurred go to step 5 otherwise go to step 1.

5. The local image station sends a stop_transfer BB to the remote image station informing it to stop the transfer immediately.

6.4.4. Flow control

A flow control mechanism is needed to ensure that the transmission rate matches the receiver acceptance rate. This prevents the loss of data at the receiver. Data flow is controlled at two levels, the interface and the protocol level.

Two interface flow control mechanisms, the delay time between BB and the maxmeanrate technique, are used. In the former technique the remote station waits for a fixed time period after each BB transfer before it continues processing. The delay time between each BB is specified by the controller station. This flow control mechanism was used during freeze frame transfer.

The maxmeanrate (maximum transfer rate permitted) can be defined by either the controller station or local station, whichever is lower. The remote station checks its data rate every time it has transferred twice the specified maxmeanrate value. There must be a time interval of 25 milliseconds since the last check before the image station can continue processing the remaining image data. This flow control mechanism was used during slow-scan transfer.

6.4.5. Buffering and Acknowledgement

The buffering policy is closely related to flow control. The local image station has two input buffers, one for the controller station and the other for the remote image station. The buffers are multiplexed: port 2 for the former and port 1 for the latter. The latter buffer consists of 9 listening BBs, used for buffering the image data from the remote station, whilst the former buffer, consisting of only one BB is used for listening to the controller station.

The remote image station has a transmission buffer of 2 BBs. It also allocates 2 listening BBs for the INITF-BB from the receiving image station.

The local image station sends an acknowledgement BB, INITF-BB, on reception of the end-of-frame BB from the remote image station. Two acknowledgement BBs are sent per frame during slow-scan transfer. This is to overcome the round trip delay when transferring over a wide area network, which would otherwise reduce the frame update rate. The round trip delay when transferring data over the OTS is around 0.4 seconds [78].

6.4.6. Lost data recovery mechanism

The Cambridge ring has an error rate of about 10^{-12}. It has a number of mechanisms for detection of data lost or corrupted in transmission. At the data link layer, i.e. minipacket protocol, the transmitter station compares the returned minipacket with the packet it has transmitted. If they are not the same, the station will try to retransmit the minipacket again, for a maximum of 15 times [70].

The Basic Block protocol uses the checksum at the end of each block to detect corrupted blocks, which will be ignored by the receiving station, but it does not inform the sender that the blocks have been corrupted or lost. Lost blocks create one of the following effects:

(a) a deadlock state [46], i.e. both communicating stations wait indefinitely for each other to initiate an action

(b) the receiver image station has a different image to that at the transmitter station

The deadlock state occurs when either the INITF-BB or the end-of-frame BB is lost. A timeout facility provided by the VMI-1, when setting up input buffers of basic block, is used to overcome this problem. The timeout period is set to 2 seconds.

The BBNUMBER attached to the beginning of each block is used to detect lost BB, indicated by a break in the sequence, during slow-scan transfer. Forced update of image data ensures that the errors created by the lost BB will be overwritten after a fixed period of time.

The freeze frame protocol uses a different technique for detecting lost blocks. It keeps a tally of all the lines received and compares it with its own list of requested lines. The list of lines not received will be requested again for a maximum of 5 times.

6.5. Software

The software of the image station can be divided into four stages: initial, task analysis and performance, path setup, and image data transfer as shown in figure 6.10.

6.5.1. Initial stage

On reset the image station will bootup the VMI-1, and initialise the interrupt timer. It then generates a colour bar signal on the screen by writing to frame 0 of the framestore.

After the above reset procedures, the image station will set up two Basic Blocks input buffers, one each for port 1 and port 2, to listen for data from the ring. It will then wait indefinitely for data to arrive on one of these ports.

6.5.2. Task analysis

On receiving a BB the image station will check if it has the correct headers. If it does not have correct headers the image station will return to the initial stage.

If the block is received on port 1, it is from the controller station. The block would have an SSPREQ block structure with the task defined by the 'OPCODE' variable. If the task does not require image data transfer, the local station will perform the task and send an SSPRPLY block back to the controller station. The image station will then return to the initial state, waiting for Basic Blocks to arrive. For freeze frame or slow-scan transfer, the local image station will first read the content in the argument field of the SSPREQ block prior to sending an SSPRPLY block back to the controller.

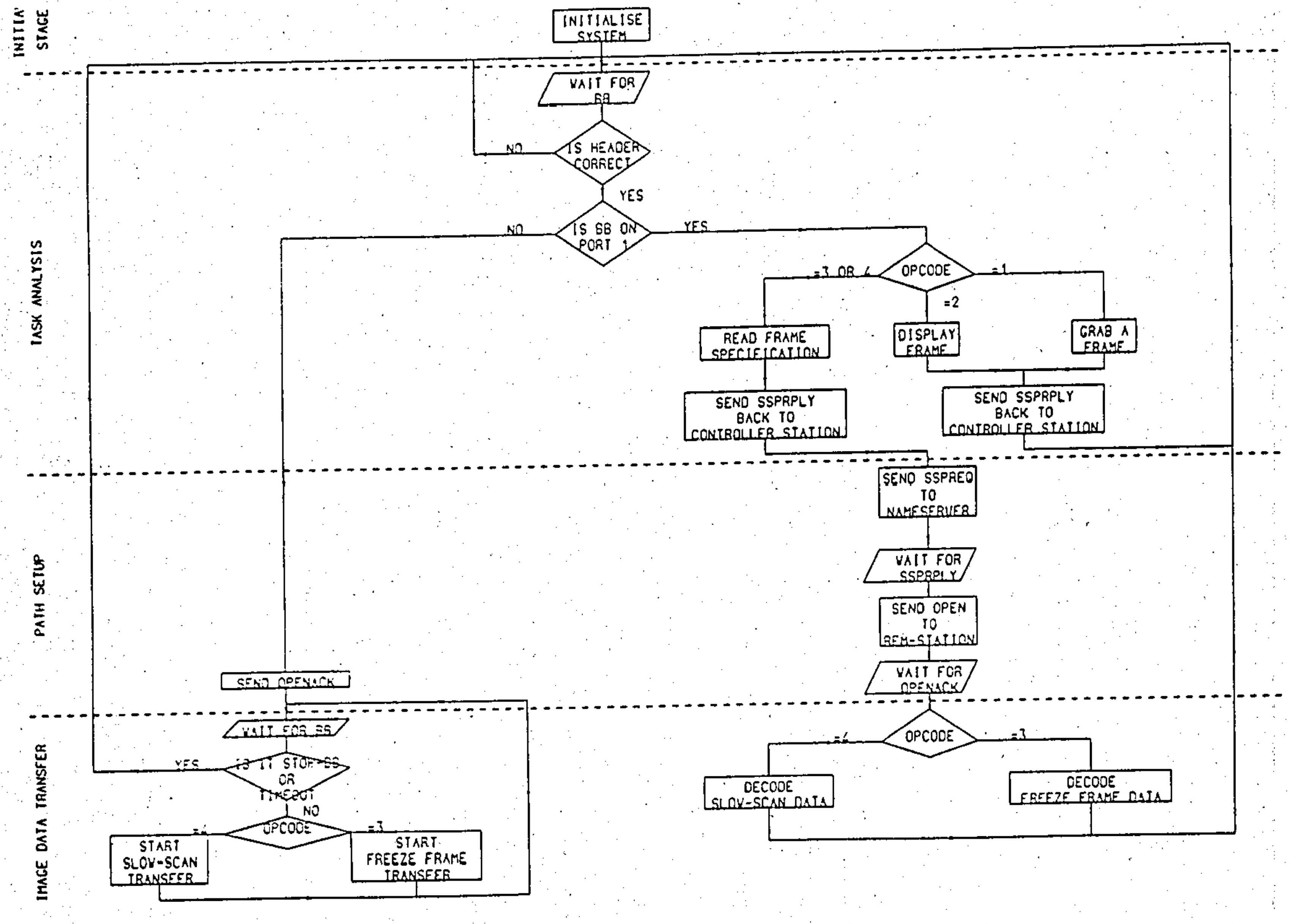


Figure 6.10 : Image station flowchart

6.5.3. Path setup

This stage is only required when the task to be performed is either freeze frame transfer or a slow-scan transfer. The image station uses the name of the remote image station given by the controller station, in the SSPREQ, to access the nameserver address table [39]. The Nameserver will reply with the destination node address and its public portnumber. It then sends an OPEN BB to the remote station, which will reply with an OPENACK BB.

6.5.4. Image data transfer

The flowcharts in figure 6.11 and figure 6.12 show the software of the remote image station during freeze frame and slow-scan image data transfer. In figure 6.11 the following conventions are adopted.

BYTECNT = Number of bytes in the Basic Block
WAIT(N) = Delay routine. Wait for N milliseconds
BYTESPERBLOCK = Maximum number of bytes per block (2048)

For the slow-scan image data transfer flowchart (figure 6.12) the following conventions are adopted.

SCNT = This variable is incremented by the timer interrupt every 25 milliseconds.
NUMBYT = Number of bytes transmitted since the last SCNT check.
UCOUNT = Number of conditional blocks checked since the last unconditionally updated block.

Figure 6.13 shows the flowchart of the receiver image station during image transfer.

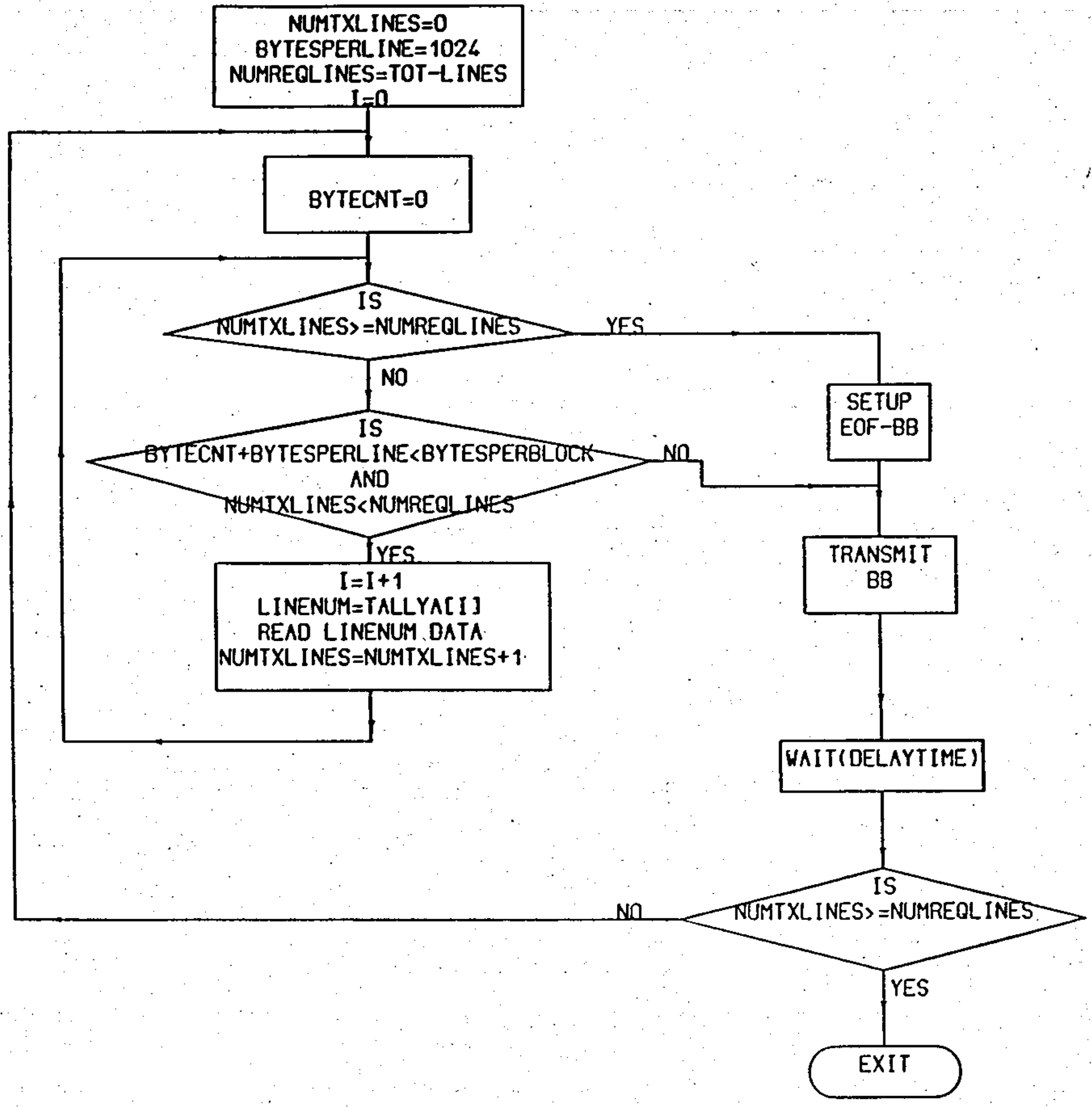


Figure 6.11 : Freeze frame transfer flowchart

(transmitter station)

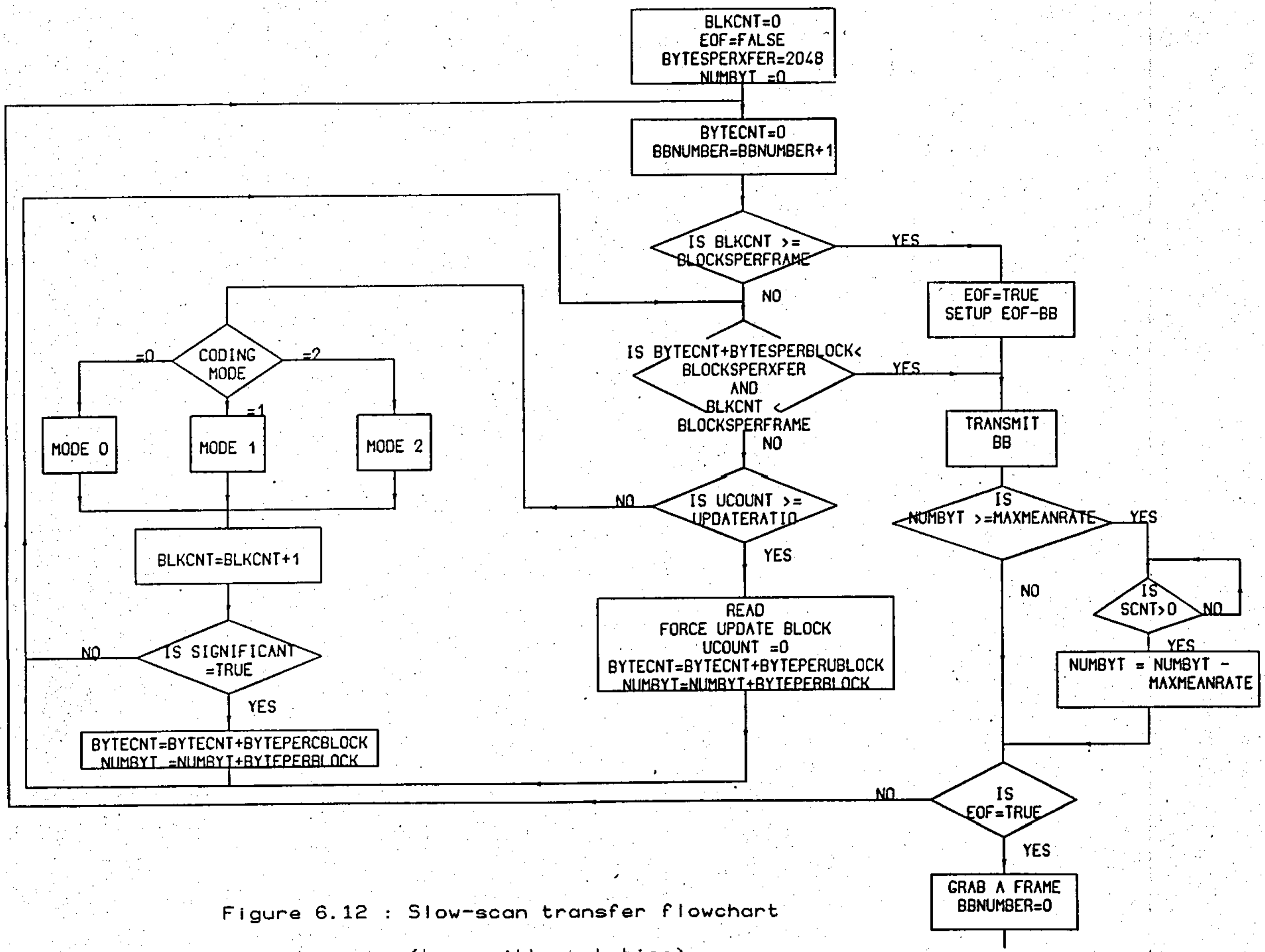


Figure 6.12 : Slow-scan transfer flowchart

(transmitter station)

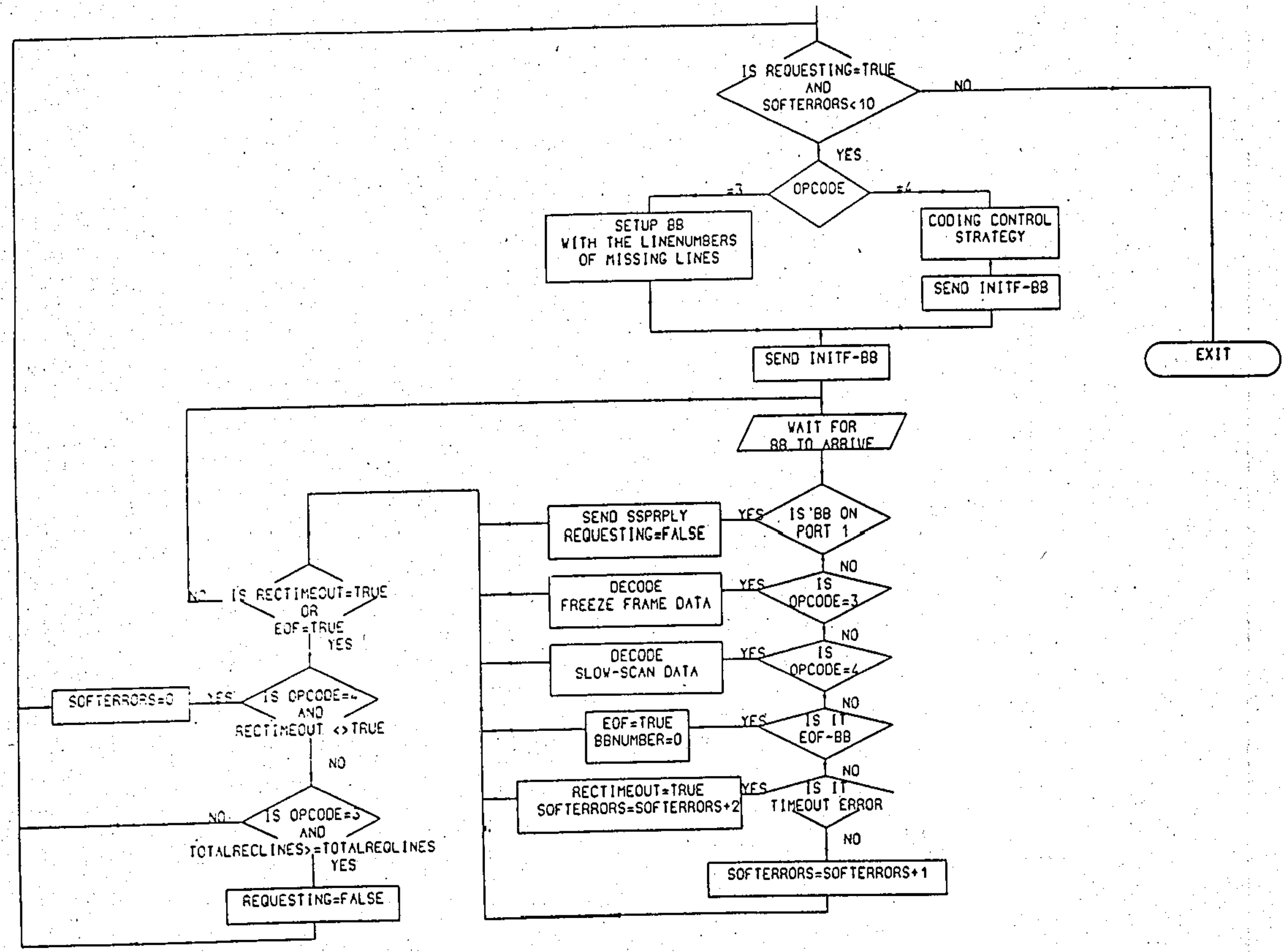


Figure 6.13 : Receiver station flowchart

6.6. Performance

6.6.1. Data rates

Block conditional replenishment coding was used to code the image data during slow-scan transfer. The data transfer rate can be limited by one of the following factors:

(a) processing power of the 8086
(b) VMI-1 data transfer rate
(c) ring data bandwidths

Using the block conditional replenishment coding technique image data rate output to the ring fluctuates, depending on the percentage of changed blocks between two successive frames. The minimum time required by the 86/30 to process a block of 8 by 8 pixels is approximately 620 microseconds, when the block is detected as changed after testing the first line of pixels, and a maximum of 1089 microseconds, when the block has been detected as unchanged.

The VMI-1 has a maximum data transfer rate of approximately 800 Kbits per second. Thus, the maximum data transfer rate of the image station is 800 Kbits/second.

The ring data bandwidth allocated to each station depends on the number of active stations, stations that use all the minipackets allocated to them, on the ring and the number of minipackets circulating round the ring by the equation below

$$\begin{aligned} Tp &= 4.2M \,/\, (n + m) \quad \text{if } m > 1 \\ &= 4.2M \,/\, (n + 2) \quad \text{if } m = 1 \end{aligned}$$

where 'm' represents the number of minipackets on the ring and 'n' represents the number of active stations on the ring. The Universe ring at Loughborough University has only one minipacket on the ring at a time. When data is transferred over a wide area network, the bridge bandwidth is usually the limiting factor. eg. the OTS satellite bridge has a data transfer bandwidth of approximately 260 Kbits/seconds.

6.6.2. Image coding performance

The quality of the pictures obtained using the different modes is as shown in figure 6.14 and figure 6.15. The degradation in image quality is quite noticable when comparing the original image with a 4:1 subsampled picture in figure 6.14. But the degradation is less noticeable in the head and shoulder picture in figure 6.15.

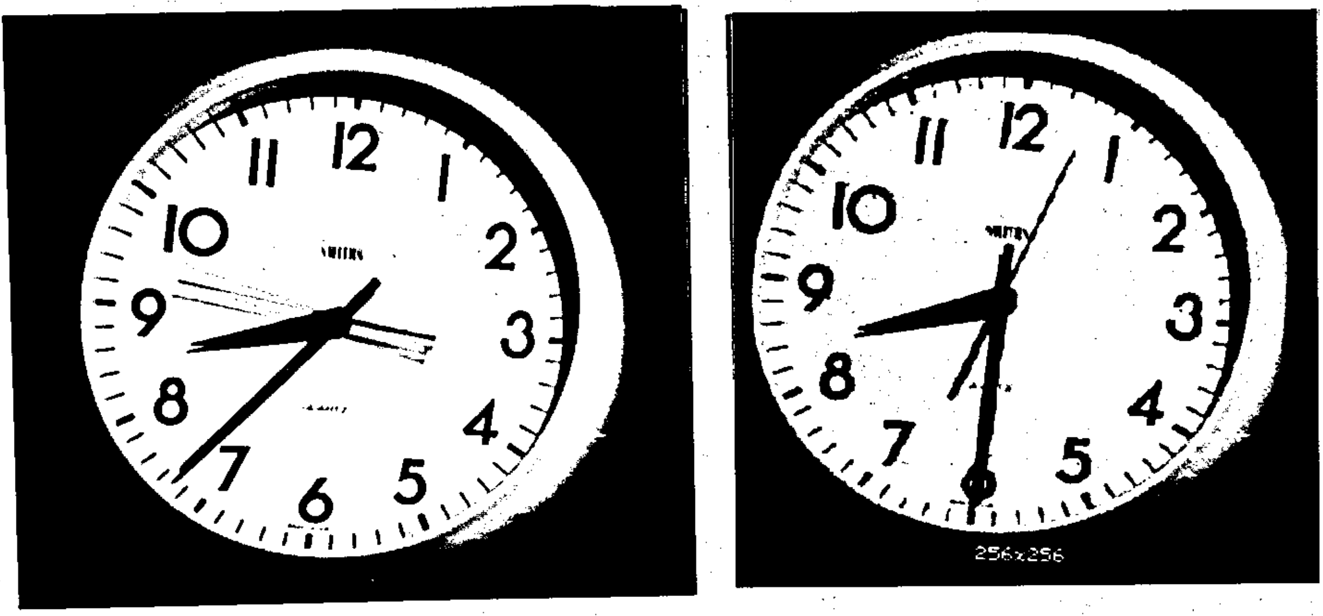
The amount of compression achieved depends on the percentage of changed blocks on the screen and the coding mode employed. On average, between 25% and 40% of the screen changes within a second.

6.7. Conclusion

The image station provides multilevel colour image communication, slow-scan transfer and freeze frame transfer, over the Cambridge ring. High resolution pictures, 512 pixels by 512 lines, were transmitted during freeze frame transfer and medium resolution pictures were transmitted during slow-scan transfer.

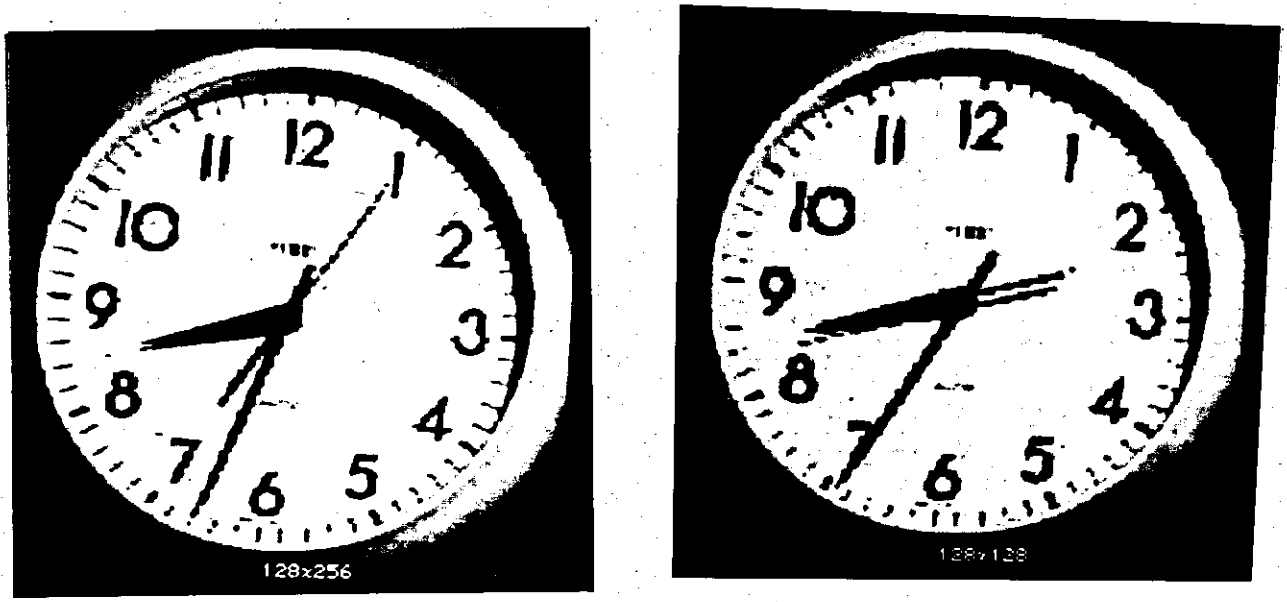
The slow-scan image transfer mode has three modes of coding pixels of changed blocks: PCM, 2:1 subsampling and 4:1 subsampling. A coding control strategy was used to switch between the different modes to maintain a constant frame update rate of one frame per second. This is 15 times faster then the previous image station.

Standard protocols, SSP and BSP, were used by the image station to maintain compatibility with existing service stations. Simple protocols were used where the local image station makes simple requests within a small Basic



Original

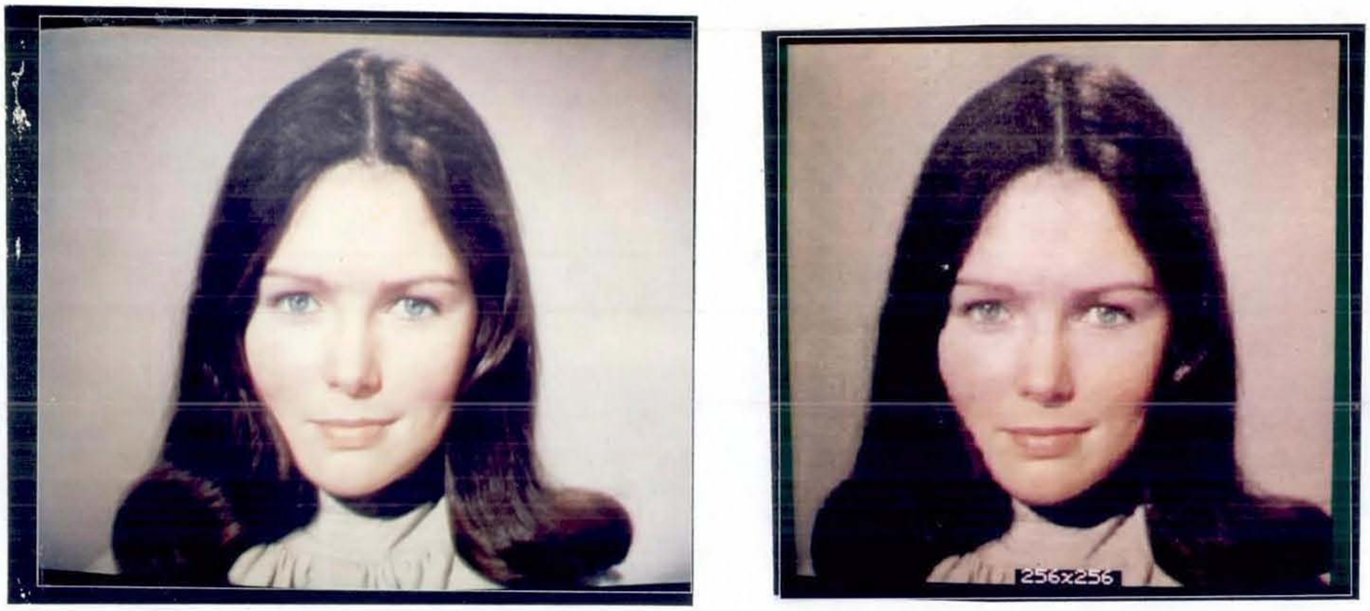
Normal mode(256 by 256)



2:1 subsampling

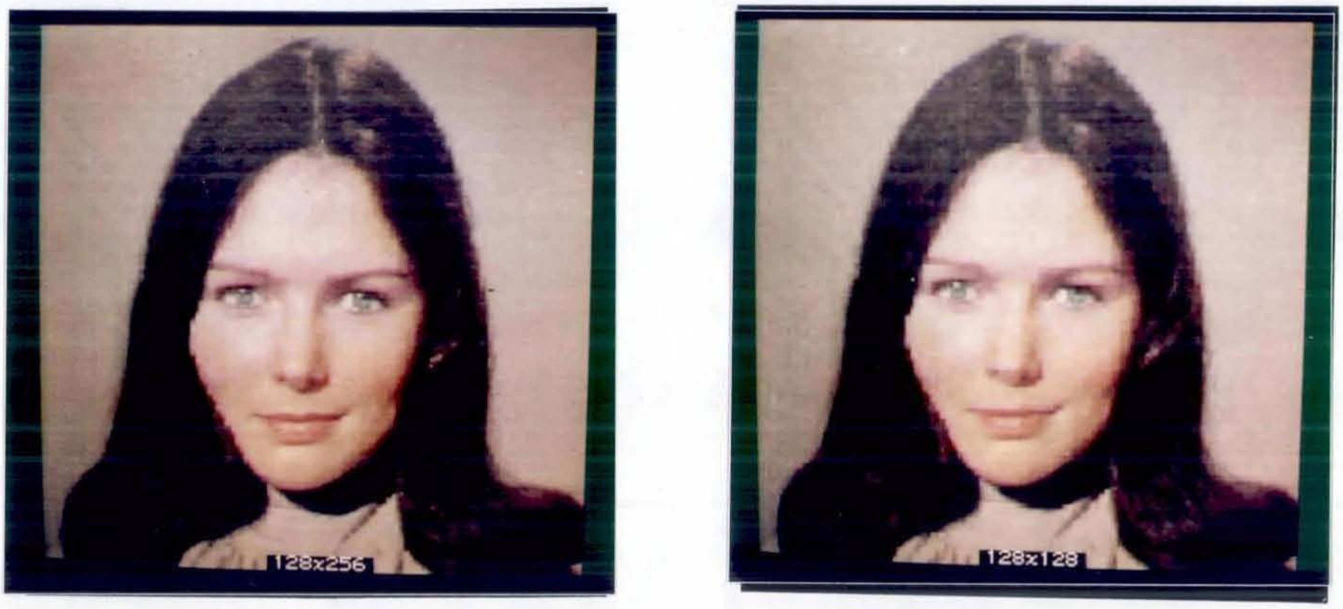
4:1 subsampling

Figure 6.14 : Clock



Original

Normal mode(256 by 256)



2:1 subsampling

4:1 subsampling

Figure 6.15 : Test card

Block to the remote image station, on a frame by frame basis, which responds by sending the image data in the reverse path.

CHAPTER SEVEN

Discussion and Suggestions for future work

7.1. Introduction

As a result of the research described in this thesis two new services have been developed for the Cambridge ring: slow-scan transfer and freeze frame colour image transfer. These two services are provided by an image station which has been built around the LUT framestore. The work performed during the research period consisted of image coding simulation, design and construction of the LUT framestore, and the implementation of the services on the Cambridge ring.

At the start of the project, a significant amount of time was spent on the design and construction of a fast and flexible framestore to be used by the image station for the acquisition and display of images. However, at the same time various interframe image coding techniques were investigated, using the B.T. framestore. These proved a useful guide to the amount of data reduction that could be achieved as well as possible algorithms which could be implemented for the transmission of colour images on the Cambridge ring. It was only in the latter part of the project, that the framestore and the results obtained from the image coding work were brought together to provide the user with the slow-scan transfer and freeze frame transfer services on the image station.

7.2. Discussion

7.2.1. Image coding

Displacement estimation algorithms: recursive pixel and block matching techniques were used to compensate for motion in the picture. Three recursive pixel algorithms were investigated: the steepest descent method, the simplified steepest descent method and Newton's method. Of these three the Newton method gives the highest data compression, i.e. the lowest moving area predictive error entropy.

A number of block matching algorithms were investigated and a new algorithm, the Basic simplex search algorithm, was developed. The Basic simplex search algorithm requires the lowest computational overhead, giving approximately 50% reduction when compared with the 2D-logarithmic search algorithm [41].

Comparing the results of the two groups of displacement estimation algorithms the block matching algorithms give a greater reduction in predictive error entropy than the recursive pixel algorithms (which is directly related to bit-rate).

7.2.2. L.U.T. framestore

The L.U.T. framestore was designed as a general purpose framestore with vector drawing graphic capabilities. The table below gives a comparison between the LUT framestore and two other commercial framestores, R16.4.2 developed by British Telecom [13] and IP512 developed by Imaging Technology Inc.[28]. The table shows some of the specifications of the individual framestores.

Unlike the B.T. framestore, the LUT framestore display resolution is software selectable. This enables the user to vary the display resolution to his/her requirement, e.g. high resolution (512 by 512 pixels) for displaying diagrams while medium resolution (256 by 256 pixels) for continuously updated images.

Framestore		R16	LUT	IP512	unit
Image resolution	(max)	512x512	512x512	512x512	pixel
	(min)	256x256	256x256	256x256	pixel
Resolution select		hardware	software	software	
Pixel depth (bits/pixel)		6	16	8	
Pixel access time	(max)	1.6	1.3	1.2	usec
	(min)	0.8	0.6	0.8	usec
Frame acquisition rate		50	50	50	Hz.
Sampling rate		5 or 10	7 or 14	5 or 10	MHz.

Table 7.1 : Framestore comparison table

Another advantage of the LUT framestore is that it is built as a colour framestore while the IP512 and R16.4.2. framestores are monochrome framestores. To convert the IP512 into a colour framestore, it is necessary to connect at least 2 of them in parallel.

The LUT framestore has an additional advantage of low register setup overhead when accessing blocks of data. It has a block access instruction which reads out a square block of data, from the video memory with a single instruction. In the case of the R16.4.2. framestore, the X address register, Y address register, and the R/W pulse register have to be setup for each pixel access. The IP512 does not have block read instructions but it does have autoincrement and autodecrement facilities. This reduces the register setup overhead but it is still higher than the LUT framestore setup overhead as the Y address needs to be setup at the end of each scanline in a block. By reducing the register setup overhead per pixel, the rate at which data can be read from the framestore is increased. Furthermore the pixel access time of the LUT framestore compares favourably with the other framestores.

The LUT framestore can be set so that only a specific colour is written onto the screen. Using this feature with the GDP inbuilt character and vector drawing capabilities the framestore can be transformed into a graphic display terminal.

The LUT framestore has 8K bytes of buffer memory through which data is passed between the host system and the video memory. This enables the host processor to work concurrently with the GDP, i.e. the host processor can be reading the data out of the buffer memory while it is being filled with image data by the GDP.

7.2.3. Image station

The image station is used for the transmission/reception of digitised colour image data over the Cambridge ring. It has two modes of operation: freeze frame transfer (512 by 512 pixels resolution) and slow-scan transfer (256 by 256 pixels resolution). The former mode is used mainly for transmitting a single high resolution images while the latter is used for continuously updated medium resolution images. The operation is selected by the user via an image station control program on the BBC microcomputer.

(a) Freeze frame transfer
During freeze frame transfer the image data are coded using PCM coding, 6 bits for luminance and 5 bits for each chrominance components. It has a resolution of the image 512 pixels by 512 lines. The data is coded line by line. The 'delaytime' flow control mechanism, is used to match the transmission rate to the remote station data reception rate.

A data recovery mechanism is incorporated into the freeze frame transfer protocol to ensure that the local station has received all the requested data. Data lost will be requested again for a maximum of five times. The data recovery mechanism is essential during freeze frame transfer as the image received will be displayed until another operation is requested by the user.

(b) Slow-scan transfer

Block conditional replenishment coding is used for the transfer of data during slow-scan transfer. It has three coding modes: PCM, 2:1 subsampling and 4:1 subsampling. The coding mode is selected by the receiver station on a frame by frame basis. The choice of coding mode depends on the amount of movement between successive frames and the traffic on the Cambridge ring. This is necessary because the Basic Block protocol does not inform the sender station if data has not been successfully transferred. To overcome this problem the receiver station monitors the first byte of each Basic Block, which contains the Basic Block sequence number (BBNUMBER). A break in the number sequence indicates that data has been lost in transmission and the transmitter data rate should be reduced by switching the coding mode. Traffic congestion is detected by monitoring the frame update rate. The coding mode is switched to a lower bit-rate mode when the frame rate is greater than 1 second.

The slow-scan transfer uses the 'maxmeanrate' mechanism to control the flow of data from the transmitter station. The 'maxmeanrate' is defined by either the BBC microcomputer or the receiving station. This method of flow control is more suitable to the slow-scan transfer than the 'delaytime' method because the Block conditional replenishment coding technique produces a variable bit-rate output.

Problems of long delay between the end of one frame and the start of the next, i.e. round trip delay, arise when transferring slow-scan image data over a wide area network. This problem is solved by requesting one frame in advance, i.e. before the receiver station has received the data for the current frame, it has already requested the next frame.

7.3. Future work

The results of the image coding work have proven the feasibility of bit-rate saving using block matching displacement estimation algorithms. But due to the lack of computational power of the 86/30 SBC, the simplest form of interframe coding, Conditional replenishment/PCM coding, has been implemented on the image station. One way of providing more computational power to the image station is to incorporate a digital signal processor (DSP), e.g. TMS32010, on the framestore controller card. The DSP could take over the task of coding/decoding image data from the 86/30.

The present image station provides colour image communication between two stations only. A possible avenue for the extension of this work is to provide a multipoint colour image communication, i.e. each station could communication with more than one other station at the same time. The display of the monitor could be divided into individual windows to display the data from the different stations. This would help reduce the equipment cost, as only one receiver image station is required by each of the participants in a videoconference. One of the windows on the display could be used for displaying the contents of an electronic blackboard for graphical communication. Figure 7.1. shows an example of how the screen could be divided.

Loughborough University	Cambridge University
Science and Engineering Research council	Electronic Blackboard

Figure 7.1 : Windowed display example

REFERENCES

1. Adam, C.J., et. al., "Voice and Image Communication in Project Universe.", Proc. Networks, 1984, Online 1984.

2. Adam, C.J., "Universe Network Protocol Architecture.", Project Universe report no. 2.

3. Adam, G.C., Leslie, I.M., "Client Protocol-Implementation Specification.", Universe internal report UP/94.2.

4. Ahiya, V., "Routing and Flow control in System Network Architecture.", IBM System Journal, Vol. 18, No. 2, 1979, pp. 298-315.

5. Andrew, H.C., Patterson, C.L., "Digital Interpolation of Digital Images.", IEEE Trans. Computer, Vol. C-25, No. 2, Feb. 1976.

6. Barney, C., "Video conference sees new window.", Electronics, 3 Nov. 1983, pp. 103-104.

7. Bergman, H.C., "Displacement estimation based on the correlation of image segments.", IEE Proceeding International Conf. Electronic Image Processing, York

1982, U.K.

8.	Bergman, H.C.,	"Analysis of Different Displacement Estimation Algorithms for Digital Television Signals.", in <u>Image Sequence Processing and Dynamic Scene Analysis</u> ed. by T.S. Huang, 1983. pp. 215-234.
9.	Blair, G.S.,	"A Performance Study of the Cambridge Ring.", Computer Network, No.6, 1982, pp. 13-20.
10.	Bowden, G.J.,	"A Graphic Display Interface for an 8-bit Microcomputer.", Final Year project report, Electrical and Electronic Dept., Loughborough University, 1982.
11.	Brainard, R.C.,	Mounts, F.W., Prasada, B., "Low-Resolution TV: Subjective Effects of Frame Repetition and Picture Replenishment.", Bell System Technical Journal, Vol. 46, Jan. 1967, pp. 261-271.
12.	Bridge, Alison	"DSP chips multiply fast.", Systems International, June 1986, pp. 37-38.
13.	British Telecom	R16.4.2/RBH Technical Manual.
14.	Brusewitz, H.,	Weiss, P., "Hybrid Coding for Video Conferencing.", Internal report of TTT, TRITA-TTT-8502, Feb. 1985.

15. Burley, D.M.,(ed) Studies in Optimization. Published by International Textbook Company Limited, 1974.

16. Burren, J.V., "Overview of the Project.", Project Universe report no. 1.

17. Candy, J.C., et. al. "Transmitting Television as Clusters of Frame-to-Frame Differences.", Bell System technical Journal, Vol. 50, No. 6, Aug. 1971, pp. 1889-1917.

18. Cafforio, C. Rocca, F., "Methods for Measuring Small Displacements of Television Images.", IEEE Trans. Information Theory, Vol. 22, No. 5 Sept. 1976, pp. 573-579.

19. Cattermule, K.V., O'Reilly, J.J.,(ed) Volume 1: Optimisation Methods in Electronics and Communications. Pentech Press, 1984.

20. Chiariglione, L., Corgnier, L., "System Consideration for Picture Communication.", ICC, 1984, pp. 245-249.

21. Davies, D.V., "The control of congestion in packet switching network.", IEEE Trans. Commun. COM-20, No. 3, June 1972, pp. 546-550.

22. Dick, J., "Optimal Coding.", Electronics

and Computing Monthly, Oct. 1983, pp. 70-71.

23. Dixon, L.C.V., (ed) Nonlinear Optimisation, The English University Press Limited, 1972.

24. Dubois, E., Prasada, B, Sabri, M.S., "Image Sequence Coding.", in Image Sequence Analysis ed. by T. S. Huang, Springer Verlag, Berlin 1981, pp. 229-285.

25. Erwood, A.F., "Protocol for Slow-scan T.V. Demonstration.", UP/450.

26. Erwood, A.F., "Colour Slow-Scan Image Unit Design.", Universe Report UP/449.

27. Griffiths, J.V.R., "Images", Project Universe, Report No. 12, 1984.

28. Hall, Edwin, "Multibus boards build imaging, graphics system.", Electronic Design, 27th May, 1982.

29. Haskell, B.G., Mounts, F.V., Candy, J.C. "Interframe Coding of Videotelephone Pictures.", Proceedings of the IEEE, Vol. 60, No. 7, July 1972, pp. 792-800.

30. Haskell, B.G., Gordon, P.L., "Source Coding of television Signals Using Interframe Techniques.", Efficient Transmission of Pictorial Information, SPIE, Vol. 66, 1975.

pp. 9-22.

31. Haskell, B.G., "Entropy Measurements for Nonadaptive and Adaptive, Frame-to-Frame, Linear-Predictive Coding of Videotelephone Signals.", Bell Systems Technical Journal, Vol. 54, Aug. 1975, pp. 1155-1174.

32. Huang, T.S., Tsai, R.Y., "Motion Estimation.", in <u>Image Sequence Analysis</u> ed. by T. S. Huang, Springer Verlag, Berlin 1981, pp. 1-18.

33. Huffman, D.A., "A method for the Construction of Minimum-Redundency Codes.", Proc. IRE 9, Sept. 1952, 1098-1101.

34. Iinuna, K., et. al. " A 1.5 Mb/s Full Motion Videoconference System.", Digital Satellite Communication, 1983, pp. VII-A-25 - VII-A-30.

35. Independent Broadcasting Authority "Standards for Television and Local Radio station.", IBA Technical review no. 13.

36. Intel iSBC 86/14 and iSBC 86/30 Single Board Computer Hardware Reference Manual.

37. Ishiguro, T. Iinuna, K., "Television Bandwidth Compression Transmission by Motion-Compensated Interframe Coding.", IEEE Commun.,

Vol. 20, 1982, pp. 24-30.

38. Iversen, V.R., "Picture Phones get a new image.", Electronics, 19 Aug. 1985, pp.30-32.

39. Iversen, V.R., "Picture phone cuts time and cost in Teleconferencing.", Electronics, 19 Aug. 1985, pp. 34-36.

40. Jain, A.K., "Image Data Compression:A Review.", Proc. of the IEEE, Vol. 69, No. 3, March 1981, pp. 349-389.

41. Jain, J.R., Jain, A.K., "Displacement Measurement and its Application in Interframe Image Coding.", IEEE Trans. Commun., Vol. 29, Dec. 1981, pp. 1799-1808.

42. Jardins, R., "Overview and Status of the ISO Reference Model of Open System Interconnection, Computer Network 5, 1981, pp. 77-118.

43. Kahn, R.E., Crowther, V.R., "Flow control in a resource sharing computer network.", IEEE Trans. Commun., Vol. 20, pp.539-545.

44. Kappagantula, S., Rao, K.R., "Motion Compensation Interframe Image Prediction.", IEEE Trans. Commun., Vol. COM-33, No. 9, Sept. 1985, pp. 1011-1014.

45. Lee, B.S., "Tektronic 4010 Emulator.", Final

Year project report, Electrical and Electronic Dept., Loughborough University, 1982.

46. Lai, V.S., "Protocol Traps in Computer Networks-A Catalog.", IEEE Trans. Commun., Vol. COM-30, No.6, June 1982, pp. 1434-1449.

47. Leslie, I.M., "Extending the Local Area Network.", Technical Report No. 43, University of Cambridge.

48. Limb, J.O., "A Simple Interframe Coder for Video Telephony.", Bell System Technical Journal, Vol 50, No. 6, Aug. 1971, pp. 1877-1888.

49. Limb, J.O., Rubinstein, C.B., "Plateau Coding of the Chrominance Component of Color Picture Signals.", IEEE Trans. Commun., Vol. COM-22, No. 6, June 1974, pp. 812-826.

50. Limb, J.O., Pease, R.F.V., Walsh, K.A., "Combining Intraframe and Frame-to-Frame Coding for Television.", Bell System Technical Journal, Vol. 53, No. 6, Aug. 1974, pp. 1137-1173.

51. Limb, J.O., Murphy, J.A., "Measuring the speed of Moving Objects from Television Signals.", IEEE Trans. Commun., COM-23, Apr. 1975, pp. 474-478.

52. Limb, J.O., et. al., "Digital Coding of Color Video Signals-A Review.", IEEE Trans. Commun., Vol. COM-25, No. 11, Nov. 1977, pp. 1349-1384.

53. Limb, J.O., Murphy, J.A., "Estimating the velocity of Moving Images in Television Signals.", Computer Graphic and Image Processing, Vol. 4, Apr. 1975, pp. 311-327.

54. Logica VTS "Basic Block Protocol for Polynet."

55. Logica VTS "Interface Unit Manual MULTIBUS Intelligent Interface Unit."

56. Logica VTS "Interface Unit Manual MULTIBUS Program Interrupt."

57. Matsui, K. Achida, M., Fukinuki, T., "High Speed Transmission of Sequential Freeze-Pictures by Extracting Changed Areas.", IEEE Trans. Commun., Vol. 29, Dec. 1981, pp. 1977-1981.

58. Mc. Keracher, Iain "An International Video conferencing standard for full motion at 384Kbps.", Conference on the Developement and Application of Information standard, held at Park Lane Hotel, London, on 13th and 14th March, 1985

59. Mounts, F.V. "A video Encoding System with Conditional Picture-Element Replenishment.", Bell System Technical Journal, No. 48, 1969, pp. 2545-2554

60. Nagel, H., "Analysis Techniques for Image Sequence.", International Joint Conference on Pattern Recognition, 1978, pp. 186-211.

61. Nelder, J.A., Mead, R., "A simplex method for function minimization.", Computer Journal, No. 7, pp.308-313

62. Netravali, A.N. Robbins, J. D., "Motion-Compensated Television Coding : Part 1.", Bell System Technical Journal, Vol. 58, March 1979, pp. 631-670.

63. Netravali, A.N., Limb, J.O, "Picture Coding: A Review.", Proc. of the IEEE, Vol. 68, No. 3, March 1980, pp. 366-406.

64. Nicol, R.C., Duffy, T.S., "A codec system for worldwide videoconferencing.", Professional Video, Nov. 1983, pp. 36-42.

65. Parish, D.J., Fairfield, M.J., Lee, B.S., "A Video Frame Store for Teleconferencing applications via Computer Networks.", IERE Conference, 1985, pp. 353-357.

66. Pratt, W.K., ed. "<u>Digital Image Processing.</u>", John Wiley and Sons, 1978.

67. Prabhu, K.A., Netravali, A.N., "Motion Compensated Component Color Coding.", IEEE Trans. of Comm., Vol COM-30, No. 12, Dec. 1982, pp. 2519-2527.

68. Robbins, J.D., Netravali, A.N., "Recursive Motion Compensation: A Review.", in <u>Image Sequence Processing and Dynamic Scene Analysis</u> ed. by T. S. Huang, Springer Verlag, 1983

69. Rocca, F. Zanoletti, S., "Bandwidth Reduction via Movement Compensation on a Model of the Random Video Process.", IEEE Trans. Commun., Vol. 20, Oct. 1972, pp. 960-965.

70. Sharpe, W.P. Cash, A.R., "Cambridge Ring 82: Interface Specifications.", 1982 SERC.

71. Sorensen, S.A., "Cambridge Ring Performance.", Computer Networks and ISDN Systems, No. 9, 1985, pp. 345-352.

72. Spendley, W., Hext, G.R., Himsworth, F.R., "Sequential Application of Simplex Designs in Optimisation and Evolution Operation.", Technometrics, vol. 4, No. 4,

Nov. 1962, pp. 441-461.

73. Srinivasan, R., Rao, K.R., "Predictive Coding Based on Efficient Motion Estimation.", IEEE Trans. Commun., Vol. 33, No. 8, Aug. 1985, pp. 888-896.

74. Stenger, L., Kremers, Th., Govaerts, R., "Optimization of Coding algorithms by Computer Simulation Studies.", IEEE conference, 1982, pp. D4.2.1.-D4.2.5.

75. Sunshine, C.A., "Efficiency of Interprocess Communication Protocols for Computer Network.", IEEE Trans. Commun., Vol. 25, Feb. 1977, pp. 287-293.

76. Thompson, J.E., "Objectives and results of PROJECT COST 211.", paper D4.1, Globecom 82, Miami, Nov. 1982.

77. Walker, D.R., Rao., K.R., "New technique in pel-recursive motion compensation.", Science, Systems and Services for Communication, IEEE, 1984, pp. 703-706.

78. Waters, A.G., "Satellite Bridge.", Project Universe report no. 17.

79. Wells, S.C., "Efficient Motion Estimation for Interframe Video Coding Applications.", Electronics Letters, Vol. 21, No. 7, March 1985,

pp. 289-290.

80. Wilde, D.J. ed. "Optimum Seeking Methods.", Prentice-Hall Inc., 1964.

81. Wilkes, M.V., "Communication using a Digital Ring.", Proceeding of PACNET Conference, Sendai, Japan, 1975, pp. 47-55.

82. Wolfe, M.A. ed. "Numerical methods for unconstrained optimization: An Introduction.", Van Reinhold Company Ltd., 1978

83. Yasuda, H., et. al. "Transmitting 4-MHz. TV Signals by Combination Difference Coding.", IEEE Trans. Commun., Vol. COM-25, No. 5, May 1977, pp. 508-516.

84. Seyler, A.J., "Probability distribution of Television frame differences,", Proc. IREE (Aust.), pp. 355-366, Nov. 1965.

85. Intel Intel Multibus Specifications

APPENDIX A

Image station protocol

The protocol of the image station can be divided into two Command protocol and Image transfer protocol. The former refers to the interaction between the controller station and one of the image stations. It only uses the SSP protocol for its transection. The Image transfer protocol refers to the communication between two image stations. This involves the setting up of a virtual link between the two image station using the OPEN/OPENACK protocol and the actual transfer of compressed image data. Another protocol that would be described is the Nameserver protocol which is used by the receiver framestore to get the address of the transmitting framestore so that it can establish the virtual link.

1. Command protocol

1.1. Controller to Image station

(a) Frame grab

This function causes the Image station to grab a frame from the video source in the store specified by the store number. The new frame is also displayed.

SSP Port = 1; Function = 1

Basic Block:

Bytes	Contents
0-5	Header
6	Character count of the remote Image station name in the nameserver table.
132,133	Store number

(b) Display frame

This function causes the Image station to display the content of the frame specified.

SSP Port = 1; Function = 2

Basic Block:

Bytes	Contents
0-5	Header
6	Character count of the remote Image station name in the nameserver table.
132,133	Store number

(c) Fetch single frame

This function causes the local Image station to initiate the transfer of a frame from the remote Image station.

SSP Port = 1; Function = 3

Basic Block:

Bytes	Contents
0-5	Header
6	Character count of the remote Image station name in the nameserver table.
7 on	Remote Image station name
130,131	Delaytime
132,133	Local store number
134,135	Remote store number
136,137	Local store - first pixel in line
138,139	Remote store - first pixel in line
140,141	Local store - first line
142,143	Remote store - first line
144,145	Total pixels in each line
146,147	Total lines in a frame

(d) Start slow-scan transfer

This function causes the local Image station to initiate transfer of a continuous stream of frames until requested to stop. The data were coded using Block Conditional

Replenishment coding.

SSP Port=1 Function=4

Basic Block:

Bytes	Contents
0 - 5	Header
6	Character count of the remote Image station name in the nameserver table.
7 on	Remote Image station name
130,131	Delaytime
132,133	Local store number
134,135	Remote store number
136,137	Unconditional to Conditional block update ratio
138,139	Maxmeanrate
140,141	Minimum data block width
142,143	Minimum data block height
144,145	Block scalefactor
146,147	Block start x address
148,149	Block start y address
150,151	Block horizontal overlap
152.153	Block vertical overlap
154,155	Luminance peak threshold
156,157	U colour component peak theshold
158,159	V colour component peak threshold
160,161	Luminance block average threshold
162,163	U colour component block average threshold
164,165	V colour component block average threshold

(e) Stop slow-scan transfer

This function commands the receiving image station to stop the slow-scan transfer.

SSP Port=2 Function=5

Basic Block:

Bytes	Contents
0 - 5	Header

1.2. Image station to Controller station

This is the SSPRPLY to the controller station acknowledging the reception of the SSPREQ.

Basic Block:

Bytes	Contents
0 - 5	Header

2. Image transfer protocol

2.1. Open/Openack protocol

The Open/Openack protocol sets up a bidirectional virtual link between the two image station. The format of the Basic Block is as shown below.

(a) Open Basic Block

OPEN Port=2 Function=3,4

Basic Block:

Bytes	Contents
0 - 5	Header
6,7	0 = monochrome, 1 = colour
8,9	Size of the Basic Block
10,11	Delay between Basic Block transmission or Maxmeanrate
12,13	Display frame
14,15	Number of pixels per line
16,17	Number of lines per frame
18,19	Number of bits allocated to luminance component
20,21	Number of bits allocated the U and V components
22,23	Unconditional block update per frame

24,25	4 = Slowscan image transmission
26,27	Minimum width of the image data block
28,29	Minimum height of the image data block
30,31	Scalefactor of the image data block
32,33	Horizontal image data overlap
34,35	Vertical image data overlap
36,37	Peak luminance threshold value
38,39	Peak U component threshold value
40,41	Peak V component threshold value
42,43	Average luminance threshold value
44,45	Average U component threshold value
46,47	Average V component threshold value
48,49	Maximum transmission rate permitted by the channel or local station depending on whichever is lower.

(b) Openack Basic Block

Bytes	Content
0-5	Header
6,7	0 = monochrome, 1 = colour
8,9	Size of the Basic block buffer size
10,11	Maximum transmission rate of remote station
12,13	Display frame
14,15	Number of pixels per line
16,17	Number of lines per frame
18,19	Number of bits allocated to luminance component
20,21	Number of bits allocated to U and V component

2.2. Data transfer protocol

(a) Freeze frame

The transfer is initiated by the INITF basic block as shown below.

Bytes	Content
0,1	=3
2,3	Remote framestore display frame number
4,5	Number of pixels per line
6,7	Number of lines per frame
8,9	Total requested lines(N/2)
10-10+N	List, as 16 bits integers, of requested lines in the sequence required.

The image data from the remote station is organised as shown below.

Bytes	Content
0,1	Linenumber
2,3	Firstpixel address of the line
4,5	Number of pixels per line = N
6,7	=0 if monochrome, =1 if colour
.	Lower byte of first pixel
.	
N+8,N+9	Upper byte of first pixel
.	

After transfering all the requested lines the remote station transmit end-of-frame Basic Block.

Bytes	Content
0 - 5	=0
6,7	=0 if monochrome; =1 if colour

(b) Slow-scan transfer

The format of the basic block requesting for frames, i.e INITF, is as follows:

Basic Block

Bytes	Content
0,1	=4

2,3	Remote framestore display frame number
4,5	Horizontal address of the first block
6,7	Vertical address of the first block
8,9	Minimum width of the block
10,11	Minimum height of the block
12,13	Minimum block size multiplication factor
14,15	Number of overlapping pixels in the horizontal direction
16,17	Number of overlapping pixels in the vertical direction

The image data from the remote station is formatted as follows

Basic Block

Bytes	Content
0,1	BBNUMBER, i.e. Basic Block number
2	M : defines the coding mode 1 = 4:1 subsampling 2 = 2:1 horizontal subsampling 3 = Normal 4 = Unconditional block update
3	Block size multiplication factor(N)
4,5	Vertical address of the block
6,7	Horizontal address of the block
8 - 8+bytes per	image data block' upper byte = chrominance lower byte = luminance

The remote(sending) image station tries to fill each Basic Block with as many image data blocks as possible but it will not split an image data block data between two successive Basic Blocks.

The format of the end-fo-frame Basic Block is the same as that of the Freeze frame transfer.

The controller station sends a stop-transfer Basic Block to the receiver station when the user wants to stop the slow-scan transfer. The format of the stop-transfer Basic Block is as shown below.

SSP Port =1

Bytes	Content
0	=5
1 - 5	=0

3. Nameserver protocol

The protocol is used for accessing the address of the remote framestore. The station requesting the table lookup service sends an SSP to the Nameserver. The format of the Basic Block is as shown below.

SSP

Basic Block:

Bytes	Contents
0 - 5	Header
6 on	Name of the remote station

The format of the Basic Block from the Nameserver is as follows.

Basic Block:

Bytes	Contents
0 - 5	Header
6	Remote store node address
7	Flags
8,9	Destination public port address
10,11	Open function

Digital video simulation system

The Digital Video simulation system, shown in figure B.1., was developed for the purpose of analysing of long image sequences. It consist of three main components:

(i) A laser disc to provide the image source
(ii) A colour framestore [26], consisting of three R16.4.2. framestores, is used for the acquisition and display of images
(iii) BBC microcomputer, to control the laser disc as well as data transfer to and from the framestore

The laser disc produces images at 25 frames per second. It allows the user to step through the frames individually under the control of the BBC microcomputer. This enables the BBC microcomputer to read data out of the framestore and process it at leisure.

The BBC microcomputer is interfaced to both the laser disc and the framestore. Table B.1. shows the register mapping on the BBC microcomputer 1MHz. bus. Figure B.2. shows a flowchart for the colour framestore read and write operation. Figure B.3. shows the circuit diagram of the interface between the BBC microcomputer and the framestore.

The laser disc allows the BBC microcomputer to control it via its remote control port. The commands and data from the BBC microcomputer are encoded into a serial code. This format is shown in figure B.4. The format of the continous pattern beginning with the fixed code '001' followed by D0 to D4 to indicate the content. It ends with a stop code '000'. A '0' has the low interval width of 1.05 milliseconds while a '1' has 2.11 milliseconds. The relationship between the D0-D4 codes and the commands and frame number is shown in table B.2.

The pulse width is controlled by the BBC microcomputer software. Every time the laser disc port is accessed a positive pulse will be generated. The individual code, 0 and 1, are generated by timing the interval between successive laser port access. Figure B.5. shows a listing of the section of the BBC microcomputer laser disc control program which times the period between each pulse. The circuit diagram of the laser disc interface is shown in figure B.6.

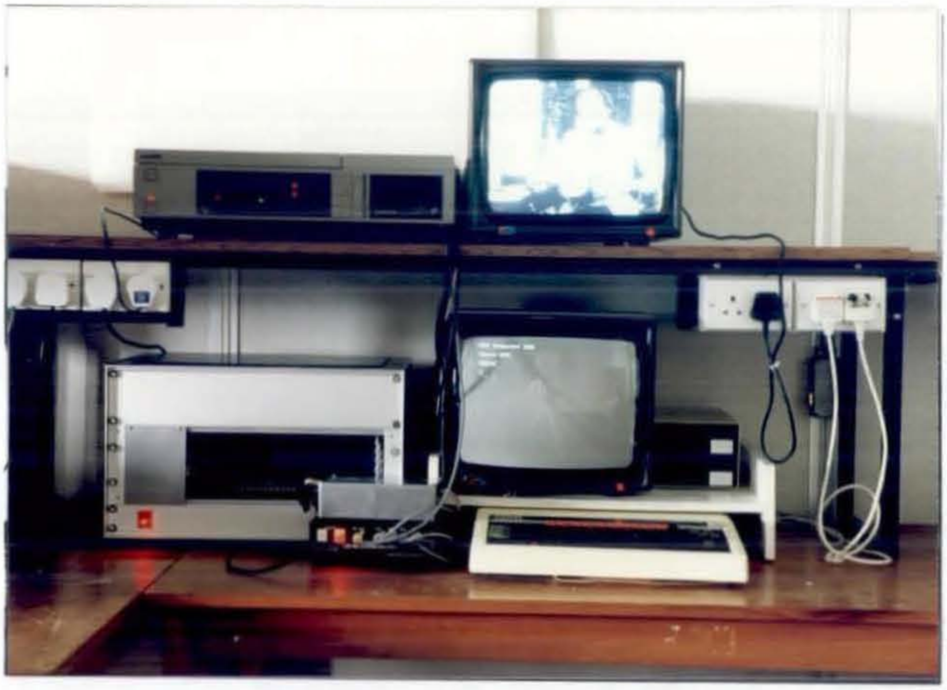


Figure B.0 : Digital video simulation system

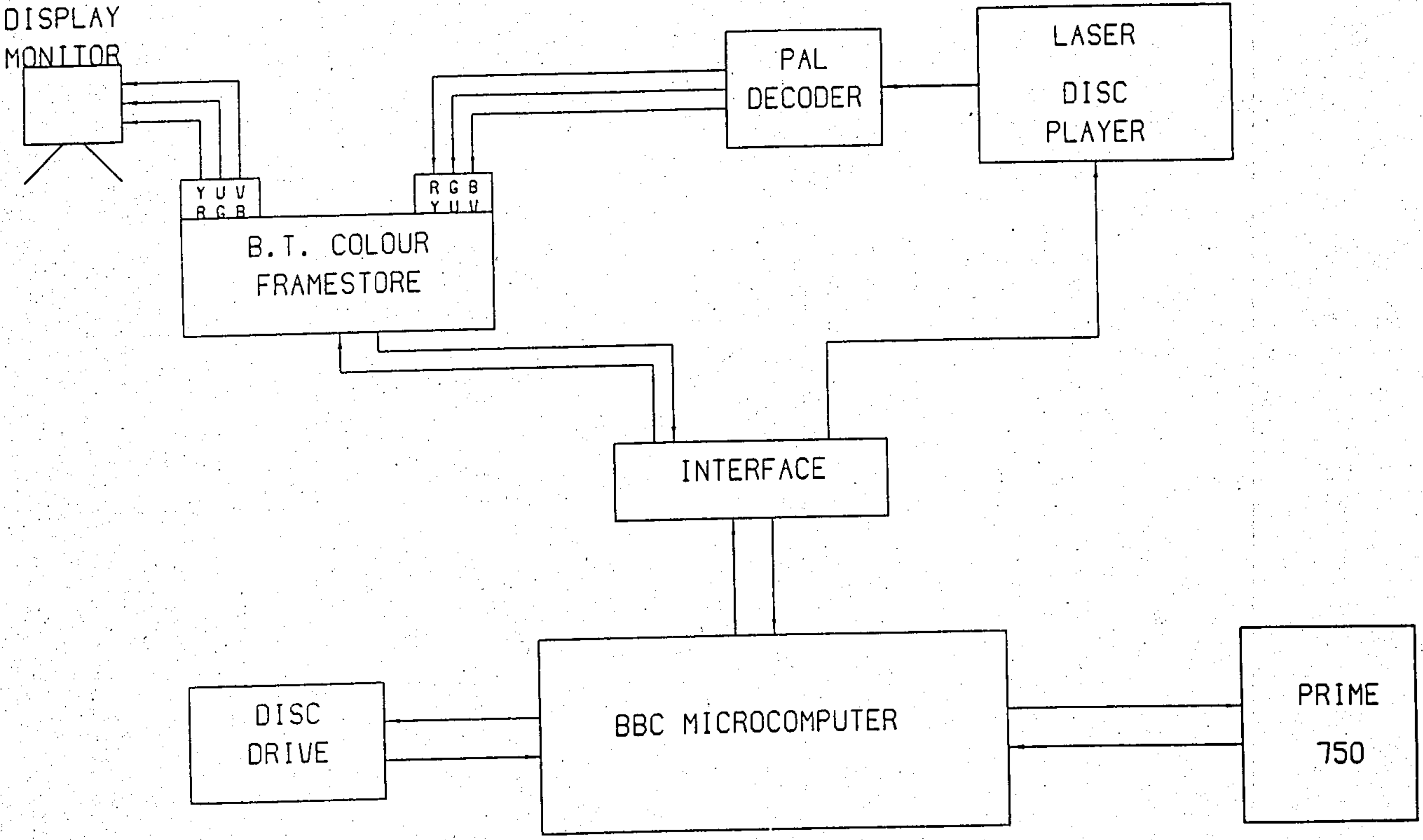
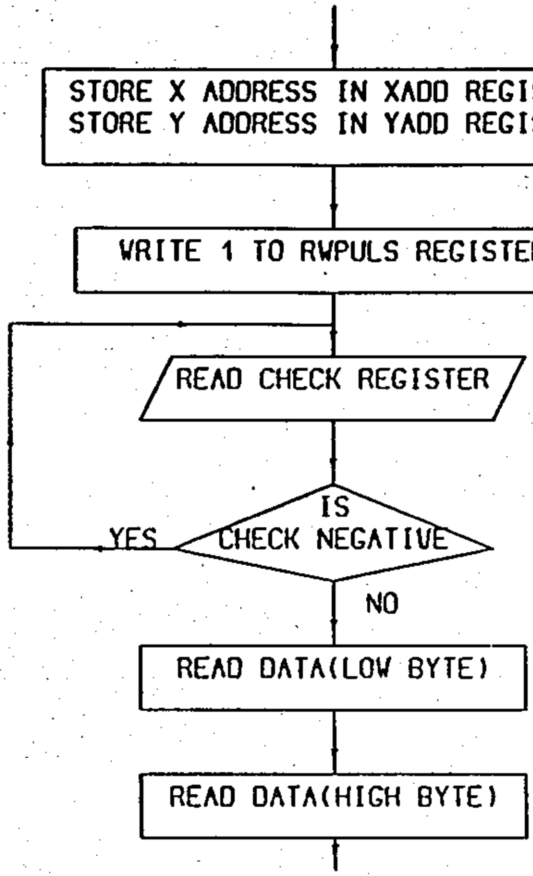
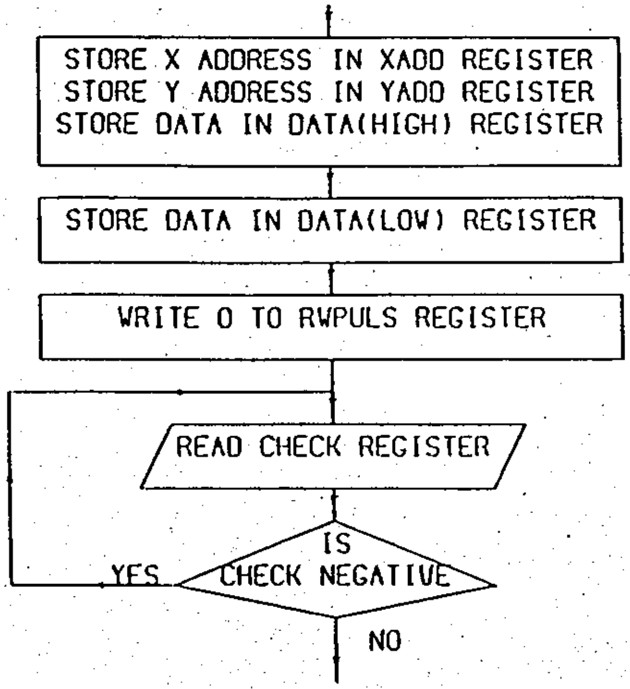


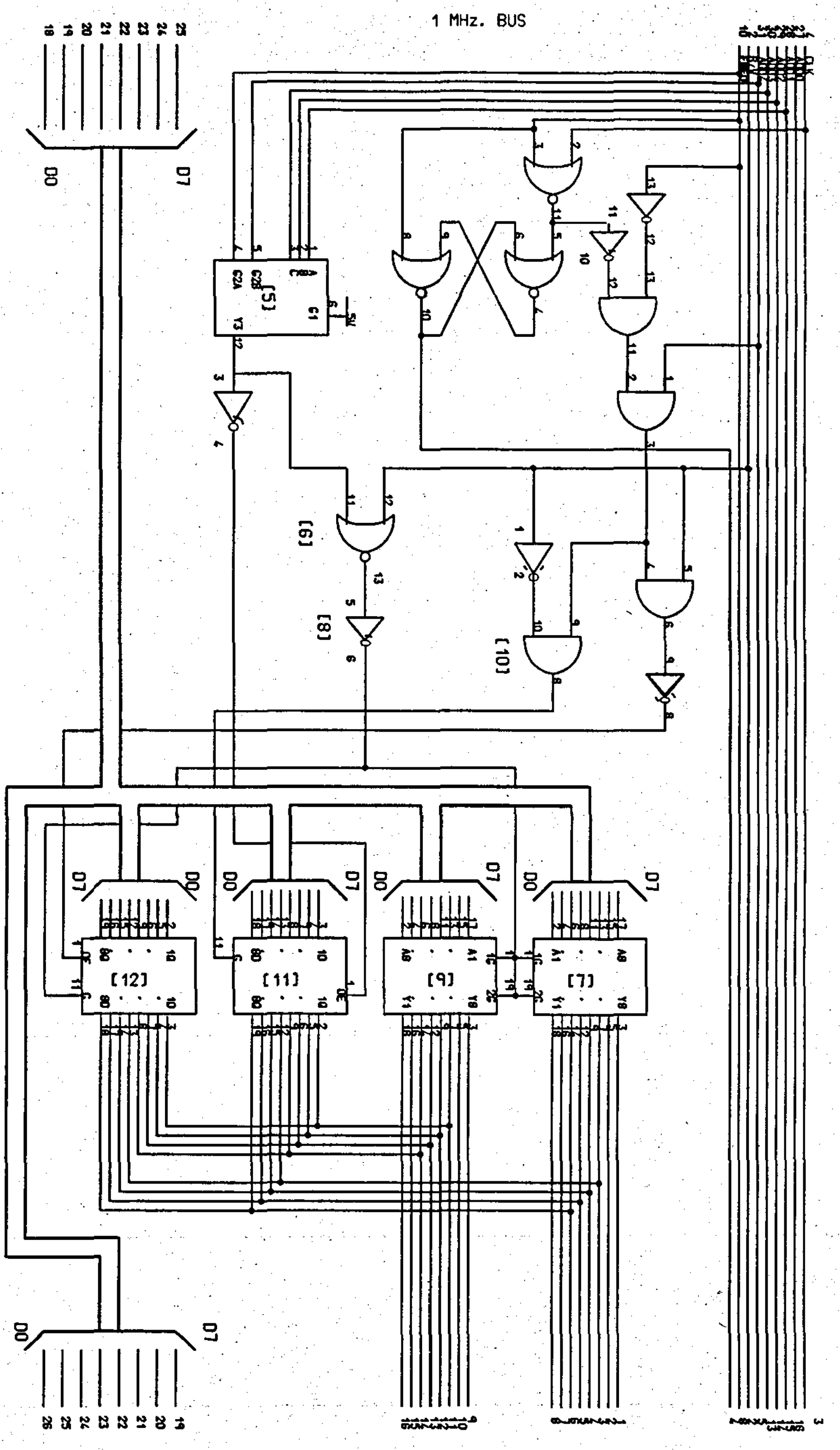
Figure B.1 : Digital Video simulation system


STORE X ADDRESS IN XADD REGISTER
STORE Y ADDRESS IN YADD REGISTER
WRITE 1 TO RWPULS REGISTER
READ CHECK REGISTER
IS CHECK NEGATIVE
YES
NO
READ DATA(LOW BYTE)
READ DATA(HIGH BYTE)

(a) Flowchart for read routine


STORE X ADDRESS IN XADD REGISTER
STORE Y ADDRESS IN YADD REGISTER
STORE DATA IN DATA(HIGH) REGISTER
STORE DATA IN DATA(LOW) REGISTER
WRITE 0 TO RWPULS REGISTER
READ CHECK REGISTER
IS CHECK NEGATIVE
YES
NO

(b) Flowchart for write routine



COLOUR FRAMESTORE

ADDRESSS (HEX)	REGISTER
FC00	LASER DISC
FD00	XHIGH ADDRESS
FD01	XLOW ADDRESS
FD02	YHIGH ADDRESS
FD03	YLOW ADDRESS
FD04	STORE SELECT (SSELECT)
FD06	DATA REGISTER (low byte)
FD08	READ/WRITE REGISTER
FD0A	CHECK REGISTER
FD0C	SNATCH REGISTER
FD0E	CTRC REGISTER
FD0F	RA4
FD10	DATA REGISTER (high byte)

Table B.1 : 1MHz. register address

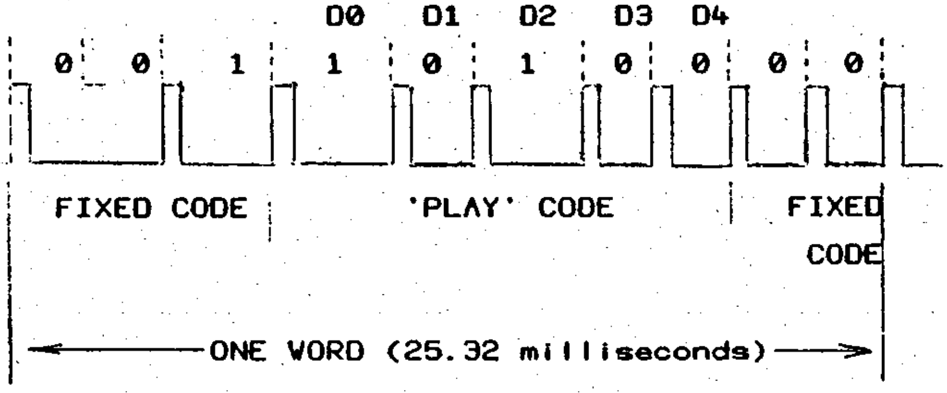


Figure B.4 : Laser disc data format

D0	D1	D2	D3	D4	FUNCTION
0	0	0	0	1	x 3 play forward
0	0	0	1	0	scan forward
0	0	0	1	1	slow forward
0	0	1	0	0	step forward
0	0	1	0	1	play
0	0	1	1	0	x 3 play reverse
0	0	1	1	1	scan reverse
0	1	0	0	0	slow reverse
0	1	0	0	1	step reverse
0	1	0	1	0	pause
0	1	0	1	1	search
0	1	1	0	0	chapter
0	1	1	0	1	audio 2/R
0	1	1	1	0	audio 1/L
0	1	1	1	1	reject
1	0	0	0	0	0 (numeral)
1	0	0	0	1	1
1	0	0	1	0	2
1	0	0	1	1	3
1	0	1	0	0	4
1	0	1	0	1	5
1	0	1	1	0	6
1	0	1	1	1	7
1	1	0	0	0	8
1	1	0	0	1	9
1	1	0	1	0	frame

Table B.2 : Laser disc codes

```
; The x register is the outer
; delay loop counter while y register is
; inner delay loop.
; The delay time is set by A register
; If A register
; = 1 ; delay time is 1.05 milliseconds
; = 2 ; delay time is 2.11 milliseconds
;
.PULSE  LDA &80,X
        SEI
        STA &FC00
.LOOP   LDY £&0E6
.DELAY  NOP
        NOP
        DEY
        BNE DELAY
        NOP
        NOP
        SEC
        NOP
        NOP
        SBC £1
        BNE LOOP
        DEX
        BNE PULSE
        RTS
;
```

Figure B.5 : Listing of timing program

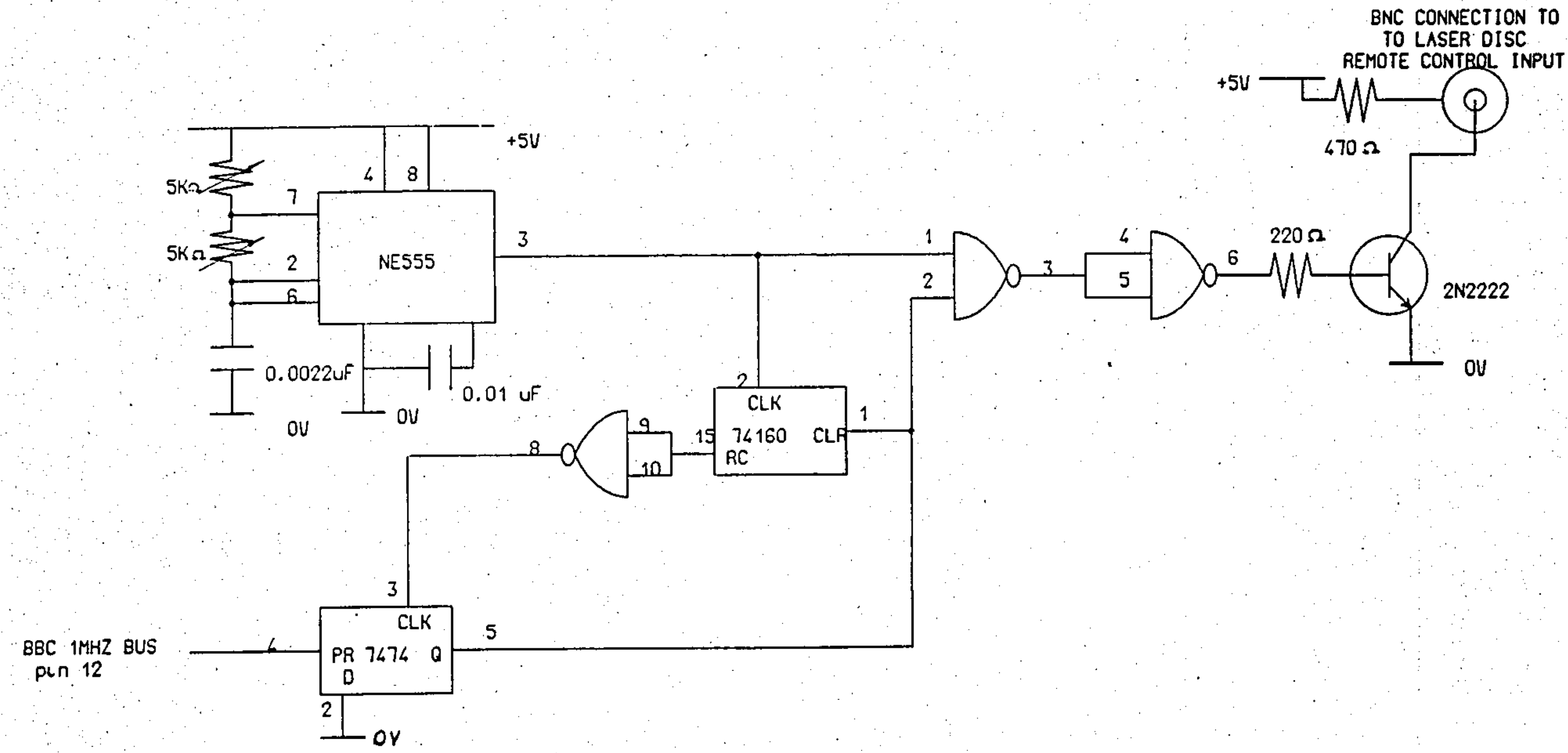


Figure B.6 : Laser Disc interface circuit diagram

APPENDIX C

Reference Images

FRAME 0

FRAME 1

FRAME 2

FRAME 3

FRAME 4

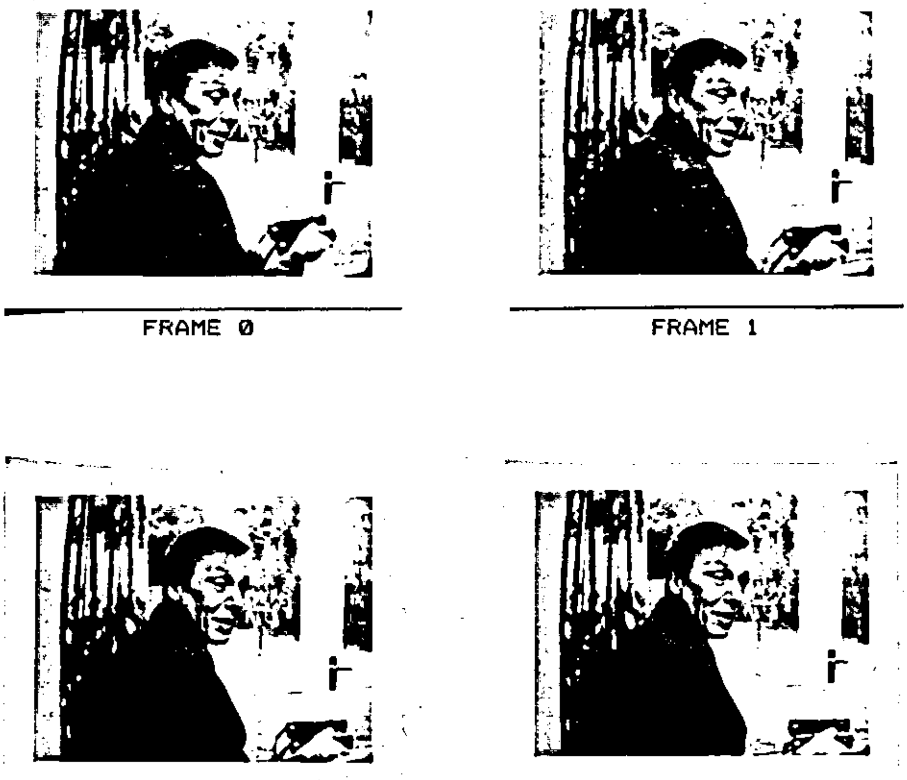
FRAME 5

FRAME 6

FRAME 7

FRAME 8

FRAME 9



FRAME 0

FRAME 1

FRAME 2

FRAME 3

IMAGE SEQUENCE A2

LUT controller board component listing

IC no.	Type	Manufacturer
24	7400	
17,32	74LS00	
12,13,15,29,60	7404	
49,61	74LS08	
30	74LS10	
26,27,28	74LS14	
31,55,56,74	74LS32	
21,36,46,54,64	74LS74	
41	74LS86	
25,50	74LS123	
20,34,57	74LS138	
6,7,10	74LS153	
42,43,44,45,59	74LS157	
2	74LS158	
3,11,33,35,39,40	74LS161	
14,18,22,23	74LS242	
16,63	74LS244	
52,62,65,66,67	74LS373	
68,73		
1,4,5,19,51,53,71	74LS374	
37	25LS2521	AMI
72	74S436	
47,48,69,70	IMS1420-S70	INMOS
8	82S128	TI
9	EF9365	Thomson EFCIS