VIDEO TITLING EQUIPMENT

P-407

PROJECT REPORT

Submitted by

A.ARUN

B.MANJUNATHAN

S.PRASANNA SRIRAM

R.SIVA GANESH

1999 - 2000

GUIDED BY

Mrs. RANI THOTTUNGAL, M.A.,M.E., M.I.S.T.E.
Lecturer in Electrical & Electronics Department

Submitted in partial fulfillment of the requirement

for the award of the degree of

BACHELOR OF ENGINNERING IN
ELECTRICAL & ELECTRONICS ENGNIEERING
BRANCH OF THE BHARATHIAR UNIVERSITY

DEPARTMENT OF ELECTRICAL & ELECTRONICS ENGNIEERING

KUMARAGURU COLLEGE OF TECHNOLOGY

COIMBATORE – 641006.

MICRO COMMUNICATION SYSTEMS

2, Lakshmi Mills Colony,
N. Palayam, Coimbatore - 641 037
h : 0422 - 215192, 215206
-Mail : microcoms@yahoo.com
microcoms@vsnl.com

PROJECT CERTIFICATE

This is to certify that the following Final BE (EEE) students of Kumaraguru college of Technology, Coimbatore have carried out a project work in our organization, and completed satisfactorily.

1. Mr. A. Arun
2. Mr. B. Manjunathan
3. Mr. S. Prasanna Sriram
4. Mr. R. Siva Ganesh

Title of the project : Video Titling Equipment

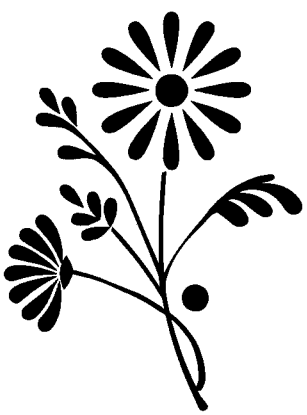
Period of project : 29.11.1999 To 29.02.2000

During this period, their attendance and conduct were found to be good.

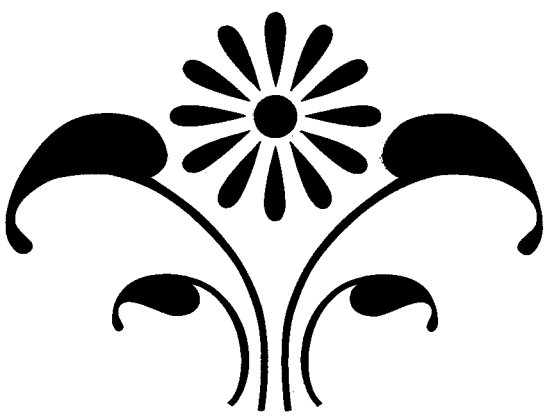
We wish them the very best for a bright future.

For Micro Communication Systems

Coimbatore
11.03.2000



ACKNOWLEDGEMENT



SYNOPSIS

SYNOPSIS

This Project Titled "Video Titling Equipment" concerns with providing Scrolling text and graphics over an available Composite Video signal.

In this project it is proposed to develop a Microprocessor based system capable of producing multilingual Text display over the available Composite Video and a PC interface to enable Graphics and Text overlay.

The Microprocessor based system uses an Intel 8085 microprocessor, which is versatile in operation. This system uses the Synchronizing signals from the input Composite Video to generate its timing signals thus reducing the complexity.

The PC based system uses a Composite Video overlay Synchronizer Chip that is used as an interface. This system converts the RGB output from a VGA/AGP card to Composite Video and controls the Overlay of the same.

CONTENTS

CONTENTS

CHAPTER III

MICROPROCESSOR BASED SYSTEM

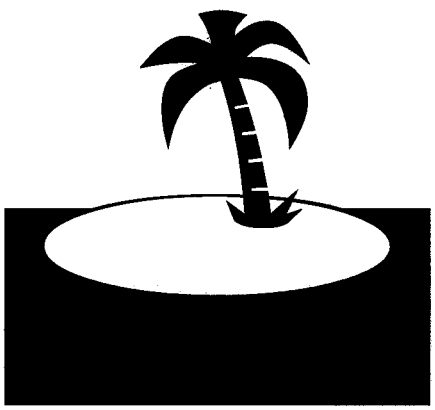
CHAPTER IV

DEVELOPMENT OF SOFTWARE

CHAPTER V

PC BASED SYSTEM

CHAPTER VI



INTRODUTION

CHAPTER I

INTRODUCTION

The invention of the television was a mile stone to mankind which has brought about the communication revolution there by providing infinite information , education and entertainment. The first used in the past were Monochrome or Black & white Television. As technology advanced color television came into existence. But the television signals sent have not changed much though color television signals have Color information in them.

In INDIA we use the CCIR PAL standard for television. This standard concerns with the Bandwidth , Frequency etc. of the video signals transmitted.

1.1 Video Titling

The addition of locally generated information like text, Graphics to an already available Composite Video from any Source is Titling.

Text information can be generated and overlaid over the available Composite Video much easily than complex Graphics. Thus a Microprocessor based system can be used with much ease to generate

multilingual text and add the generated information over the available Composite Video. But Graphics are complex and needs a PC based system to add them. This kind of a PC based system can produce high resolution Graphics and multilingual text and can be overlaid.

The systems described , are such that they do not disturb the existing set up except add an additional block for overlay on the existing setup.

1.2 Need for Titling Equipment

- The latest trend in CATV operation is the addition of local advertisements on an available signal

- The high resolution Graphics produced can be used to Produce televised entertainment video's.

- The Graphics and text generated can be used in adding subtitles to a video coverage.

- Local news and important events can be announced over the network.

- Local advertisements can also be overlaid over the existing channels. This becomes an additional income to the operator.

TELEVISION SIGNALS

CHAPTER II

TELEVISION SIGNALS

Television means seeing at a distance. To be successful, a television system may be required to reproduced faithfully:

1. The shape of each object, or tonal content
2. The relative brightness of each object, or tonal content
3. Motion, or kinematics content
4. Sound
5. Color, or chromatic content
6. Preparative, or stereoscopic content

If only the structural content of each object in a scene was shown, we would have truly black-and-white TV (without any shades of gray). If tonal content were added, we would have black-and-white still pictures. With items 3 and 4 we would have, respectively, "movies" and "talkies". The last two items are not essential, although most people consider color TV desirable, and the next generation will probably get item 6 and talk blithely about "three-see."

The human eye contains many millions of photosensitive elements, in the shape of rods and cones, which are connected to the brain by some 800,000 nerve fibers (i.e., channels). A similar process by the camera tube is used at the transmitting station and the picture tube in the TV receiver. Indeed, some 150,000 effective elements are displayed in each scene. However, the use that number of channel is out of the displayed

in each element being scanned in succession, to convey the total information in the scene. This is done such a high rate that the eye sees the whole scene, without being aware of the scanning motion. A single static picture thus results.

The problem of indicating motion is solved as in the cinema. A succession of picture is shown, each with the scene slightly altered from the previous one – the eye is thus fooled in to seeing continuous motion, through the property known as the persistence of the vision. There are 30 pictures (or "frames", as they are called) per second in the U.S. television system and is above the minimum required (about 18 frames per second) to make the eye believe that it sees continuous motion. Commercial films are run at 24 frames per second; while the perception of smooth motion still results, the flicker due to the light cutoff between frames would be obvious and distracting. In motion pictures, passing the shutter across the lens a second time, while the frame is still being screened, so that a light cutoff occurs 48 times per second circumvents this. This is too fast for the eye to notice the flicker. The same effect could be obtained by running film at 48 per second, but this would results in all films being twice as long as they need be (to indicate smooth motion).

Video band width requirement

The frequency band needed for the video frequencies may be estimated (actually, overestimated) as follows. Consider at first that the lowest frequency required corresponds to a line across the screen, which is of uniform brightness. This represents a period of 1/15, 750=0.0000635=63.5us during which the brightness of the beam does not change. If a large number of lines of that brightness followed in succession,

the frequency during that time would be zero. This is too awkward to arrange, since it requires dc coupling. Thus the lowest frequency transmitted in practice is higher than zero, approximately 60 Hz in fact. As regards the highest required frequency, this will of course correspond to the highest possible variation in the brightness of the beam along a line.

Consider now that the picture has been divided into 525 lines from top to bottom, so that the maximum resolution in the vertical direction corresponds to 525 change (e.g., from black to white) din the picture. It is desirable that horizontal and vertical resolution be the same. However, because of the 4:3 aspect ratio, the picture is 4/3 times as wide as it is high, so that 525 *4/3 =700 transitions from black to white during the length of a horizontal line is the maximum required. This, of course, occurring in 63.5 us.

The period of this maximum transition is thus 63.5/350 = 0.1814us. if each transition is made gradual (i.e., sine wave) rather than abrupt (square wave), 0.01814us is the period of this sine wave, whose frequency therefore is $1/0.1814*10^{-6}$ =5.51MHz.

2.1 CCIR PAL Standards

The letters CCIR stand for committee consultative international Des radio communications, which is an international standards committee in the field of telecommunications. The principal specifications of the European TV standards are:

Number of lines per frame 625

Line frequency 15625Hz

Field frequency 50 Hz

Picture (frame) frequency 25Hz

Video band width 5Hz

Channel width 7Hz

Vision-sound carrier separation 5.5 MHz

The details of the above are, shown in fig 2.1and fig 2.2. Even in Europe there are several slight variation on this standard, but these are mainly concerned with minor details such as channel width video bandwidth and vision-sound carrier separation so they do not affect the basic character course, totally different systems (and receivers!) still in use, such as the British 405 lines system, the frequency 819 line system, and the American 525 line system, and these are incompatible with the present design. However, the generator is broadly compatible with many systems based on a 625 line 50 Hz field standard. Discussion of color signals will be limited to the PAL system. Complete information on the standards mention can be found in the CCIR report no.624 vol. XI, Geneva 1975, which as far as we are aware is the must recent publication on the subject of television standards. The present design is based on the CCIR 'B' and 'G' standards discussed in this report.

The basics of any television signal is a complex synchronization wave –forms, which ensures that the scanning circuits of the TV receiver stay synchronized to the transmitted signal. Standards vary throughout the

world, but in Europe the CCIR norm is used, and the basis of this design is a generator to provide a CCIR sync. Signal is of very little use by itself, so the modulator construction of the generator allows the addition of units to generate various video signals, the first of which is a test pattern generator giving a fully interlaced picture.

2.2 Video Signals

A monitor (as a display screen used with a computer is generally called) is really only a 'stripped down' television set. Or a TV set is an expanded version of a monitor! The monitor only contains the display tube and the necessary driver electronics and it is supplied with an actual video signal. The bandwidth of a monitor is much wider than that of an ordinary TV set. Typically a good monitor has a bandwidth of 20 MHz, while the TV only has 5.5 MHz (this is the limit of the transmitter bandwidth). In television the video signal is modulated on to a carrier wave, so that a receiver and a decoder section are also needed to regain the pure video signal from the received signal.

Scanning

As already discussed the complete frame of a television picture is scanned 30 times Per second, in a manner that is very similar to reading this page. The beam in the camera or picture Tube moves at a constant velocity across the screen and, when it reaches the end of the screen on the Right – hand side, it "whips back" to the left – hand edge of the screen and starts again. Meanwhile, it has descended down the screen, so that the next line traced out is somewhat below the first one. This process

Continues until the bottom of the screen is reached. When this happens to the eyes of a person reading this page, the person will then glance up quickly to the top of the next page. In the case of TV, the information at the top of the existing "page" has changed; so the beam "whips up " to the top of the screen, and the whole process begins against. The whole procedure is, however, a little more complex than just described and a more detailed examination is now required.

Horizontal scanning

The total time from the beginning of one line to the instant when the next line begins to be scanned is 63.5. This time obviously includes not only the scan of the picture from left to right but also the rapid return, or retrace, from right to left. Clearly, the retrace cannot take an infinitesimally short time, and in fact a period of 10.2 is allocated to it, i.e., 16 percent of the time allocated to scanning one line. That to say, the retrace time is 0.16H, and the active time is 0.84h, where H=63.5

It is clearly undesirable for the retrace to be visible; we cannot have pattern of bright horizontal lines across the previously scanned picture pattern. The method of preventing this disturbance is Avery simple one. It consist in reducing the scanning beam current to zero, from just before the beginning of the retrace until just after its end .The process is known as blanking. This consist in adding the pulse to the video waveform, at the right time and for the correct period, to ensure that the signal level been raised to the corresponding to black. It sill be recall that negative modulation is used, in which the voltage corresponding to black is much higher than the voltage that indicate white. The sequence of events is as follows:

1. As the active beam is about to reach the right hand side of the picture, the blanking voltage is applied

2. Immediately afterward, the horizontal scanning generator receives sync pules, which initiates the retrace

3. The retrace continues for a period of time that is governed by the time constant of the oscillator generating the scanning waveform but must be less than .16H.

4. The retrace ends, and scanning of next line begins

5. Immediately afterward, after a total time of 0.16H, the horizontal blanking pulse ends, and the picture becomes visible once again.

Note that the size of the displayed image is arranged so that the brief blanked-out periods, just before the retrace begins and after it ends, are just beyond the edges of the screen. This also applies to vertical blanking, and may be absorbed by making the picture in a receiver roll down slowly with a vertical hold control. Black horizontal bar will be seen, which normally belong above and below the picture seen on the screen.

Vertical scanning

Basically, vertical scanning is similar to horizontal scanning, accept for the obvious difference in the direction of movement and the fact that everything happens much more slowly (i.e., 60 rather than 15,750 times per second). However, interlacing introduces a complication, which must now be explored further.

The sequence of events in vertical scanning is as follows:

1.Line 1 starts at the top left hand corner of the picture, at point F. as is the line and the succeeding lines are scanned horizontally, the beam gradually moves downward. This continues until, midway through line 242, then vertical blanking is applied. This situation is illustrated in fig.2.3.a. Note that active horizontal lines are solid, the horizontal retraces are dashed, and the pointed which vertical blanking is applied is labeled A.

2. Soon, but not immediately, after the application of vertical blanking, the vertical scanning, generator receives a (vertical) sync pulse. The causes vertical retrace to comments, at point B in Fig. 2.3.b.

3.vertical retrace continues, for a time corresponding to several H, until the beam reaches the top of the picture, points C in fig.2.3.b. Note that horizontal scanning continued during the vertical retraces – it would be harmful to stop the horizontal oscillator just because vertical retrace is taking place.

4. The beam, still blanked out, begins its descent. The precise point is determined by the time constants in the vertical scanning oscillator, but it is usually 5 or 6H between point B&C. The situation is shown in fig. 2.3.c.

5. Precisely 21H after it was applied, that is midway through line 263, vertical blanking is removed. The first (odd) field is now completed, and the second (even) field begins. This is also illustrated in fig.2.3.c: note that D is the point at which vertical blanking is removed.

6.The visible line 263 begins at the same height, as did line1, i.e., at the top of the screen. However, line 263 when it becomes visible, is already halfway across the screen, whereas line 1 began at the left-hand edge of the screen. Thus line 263 lie above line 1,line 264 is between 1and 2, and so on. This is illustrated in Fig 2.3.

7.The second field continues, until vertical blanking is applied at the beginning of the retrace after line 504. This is point E in fig 2.3.c.

8.The sequence of events, which now take place, is identical to that already described, for the end of the first field. The only difference is that, after the 21lines of vertical blanking, the beam is located at the top left-hand corner of the picture tube, at point F. thus, when vertical blanking is now removed, the next odd field is traced out, as in fig 2.3.a.

Regrettably, the vertical scanning prodder is complicated by the use of interlacing. However, it is basically, in that blanking is applied sometime before retrace begins and removed after sometime it has ended. Both margins are used for safety and to give individual designers of receiver some flexibility. As explained, horizontal scanning continues during vertical retrace, complicating the drawings and the explanation, but actually simplifying the procedure. To stop the horizontal oscillator 21lines, and then to restart it exactly in sync, would simply not be practical. Finally, beginning one field at the start of the line and the next field at the midpoint of the line is stratagem that ensures that interlacing will take place. If this were not done, the lines of the second field would coincide with those of the first, and vertical resolution would immediately be halved!

Blanking and synchronizing pulses

Blanking Video voltage is limited to certain amplitude limits. Thus, for example, the white level corresponds to 12.5 percent (± 2.5 percent) modulation of the carrier, and the black level corresponds to approximately 67.5 percent modulation. Thus at some point along the amplifier chain, the voltage may vary between 1.25 and 6.75 v, depending

on the relative brightness of the picture at that instant. The darker the higher the higher will be the voltage, within those limits. At the receiver, the picture tube is biased to ensure that a received video voltage corresponding to 12.5 percent modulation yields whiteness at that particular point on the screen, and an equivalent arrangement is made for the black level. Besides, set owners are supplied with brightness and contrast controls, to make final adjustments, as they think fit. Note that the lowest 12.5 percent of the modulation range (the whiter –than-white region) is not used, to minimize the effects of noise.

When the picture is blanked out, before the vertical or horizontal retraces a pulse of suitable amplitude and duration is added to the voltage, at the correct instant of time. Video superimposed on top of this pulse is clipped, the pulses are clamped, and the result is video with blanking. As indicated the blanking level corresponds to 75 percent (± 2.5 percent) of maximum modulation. The black level is actually defined relatively rather than absolutely. It is 5 to 10% below the blanking level as shown in fig 2.4. If in a given transmission the blanking level is exactly 75 percent, then the black level will be about 7.5 percent below this, i.e. approximately 67.5 percent as previously stated. At the video point mentioned previously, we thus have white at 1.25 V; black at about 6.75 V and the blanking level at 7.5 V.

The difference between the blanking level and the black level is known as the setup interval. This is made of sufficient to ensure that the black level cannot possibly "poke up" above the blanking level. If it did so, it would intrude into the region devoted exclusively to sync pulses, and it

might conceivably interfere with the synchronization of the scanning generators.

Synchronizing pulses as shown in fig 2.5, the procedure for inserting synchronizing pulses is fundamentally the same as used in blanking pulse insertion. Horizontal and vertical pulses are added appropriately on top of the blanking pulses, and the resulting waveform is again clipped and clamped. It is seen that the tips of the horizontal and vertical synchronizing pulses reach a level that corresponds to 100 percent modulation of the picture carrier. At the hypothetical video point mentioned previously, we may thus have video between 1.5 and 6.75 V, the blanking level at 7.5 V and the sync pulse tips at 10 V. the overall arrangement may be thought of as a kind of voltage-division multiplex.

It should be noted the horizontal sync information is extracted from the overall waveform by differentiation. Indeed, pulses corresponding to the differentiated leading edges of sync pulses are actually used to synchronize the horizontal scanning oscillator. This is the reason why, in figs 2.4. to 2.6, all time intervals are shown between pulse leading edges. Furthermore, receivers often use monostable –type circuits to generate horizontal oscillation in the receiver.

After the start of the vertical blanking period, the leading edges of the three horizontal syncs pulses and the vertical sync pulse shown will trigger the horizontal oscillator in the receiver. However, there are no

leading edges for a time of 3H after that, as shown, so that the receiver horizontal oscillator wills either lose sync or stop oscillating, depending on the design.

It is obvious that three reading edges are required during this 3H – period. By fare the easiest way of providing these leading edges is to cut slots in the vertical sync pulse. The beginning of each slot has no effect, but the end of each provide the desired leading edge. These slots are known as serration. They have widths of 0.04 H each and are shown exaggerated in fig. 2.6 (to ensure that they are visible). Note that, at the end of and even field, serration 2,4&6or,to be precise, the leading edges following these three serration, are actually used to trigger the horizontal oscillator in the receiver.

The situation after an odd field is even worse. As expected, and as shown in fig, 2.5b the vertical blanking period at the end of an odd field begin midway through a horizontal line. Consequently, looking further along this waveform, we see that the leading edge of the vertical sync pulse comes at the wrong time to provide synchronization for the horizontal oscillator. The obvious answer is to have a serration such that the leading edge following it occurs at time H after the leading edge of the last horizontal sync pulse. To more serration will be required, at H intervals after the first one. In fact, this is the reason for the existence of the first, third and fifth serration in fig. 2.6. The overall effect as shown is that there is 6-serration altogether, at 0.5 H intervals from one another.

Note that the leading edge which now occurs midway through horizontal lines do no harm. All leading edges are used some time, either at end of an even field or at the end of an odd one. Those that are not used in a

particular instance come at a time when they cannot trigger the horizontal waveform, and they are thus ignored.

The second shortcoming of the waveforms of fig.2.5 is discussed here. First, it must be mentioned that synchronization is obtained in the receiver from vertical sync pulses by integration. The integrator produces a small output when it receives horizontal sync pulses, and a much larger output from vertical sync pulses, because their energy content is much higher. What happens is that as a result of receiving a vertical pulse, the output level from the integrator eventually rises enough to cause triggering of the vertical oscillator in the receiver?

However, we must note at this stage that a residual charge on this integrating circuit will be different at the start of the vertical sync pulses in fig 2.5.a and 2.5.b. in the former, the vertical sync pulse begins a time H after the last horizontal pulse. In the later, this difference is only 0.5H, so that a higher charge will exist across the capacitor in the integrating circuits. The equalizing pulses shown in the composite video waveform of fig. 2.6 take care of the situation. It is seen that the period immediately preceding each vertical pulse is the same, regard less of whether this pulse follows and even or an odd field. Consequently, charge is equalized and jitter is prevented.

The vertical sync pulse begins 3H after the start of the vertical blanking period, although fig. 2.3 showed the vertical retrace beginning four lines (i.e., 4H) after the start of vertical blanking. The discrepancy can now be explained. It is simply caused by the integrating circuit taking a time approximately equal to H, from the moment when the vertical sync pulse begins to the instant when its output is sufficient to trigger the vertical retrace.

An image is built up of 625 lines at a frequency of 25 Hz (25 images per second.) This frequency is high enough to prevent the human eye from detecting any annoying flicker. Each image is divided into two parts, each of which consists of 312 ½ lines, called raster. One raster consists of all the uneven lines, the other has all the even lines. These moving images on the raster , then appear as one static image with no flickering. This technique is known as interlacing as in fig 2.1. As the diagram shows, one raster begins with a half line and the other raster ends with a half line. By ending with a half line, the raster synchronization pulses come a whole number of times the line period (the time taken to scan one complete line on the screen) after the last line synchronization pulse, whereas otherwise the raster synchronization pulses appear one half of a line period later. That difference of a half line defines at what height the electron beam starts writing the next line after the fly-back. Because a half line period corresponds exactly to the height of a half line on the screen the result is that the lines of the two raster appear precisely between each other.

That is the system used in television but if we have a static image (such as screen full of numbers) then these two interlaced raster cause an annoying 'jumping' effect and this is something to be avoided in monitors for computer systems! .To prevent this effect from occurring, We have more than enough lines on the screen so we simply use half of the total number of lines and writer the same raster on the cathode ray tube 50 times per second. That can quite easily be achieved with 'software' by ensuring that the raster synchronization pulses always appear at the same interval after the last line synchronization pulse. This is called a non-interlaced

image and is possible with a normal TV set or with a monitor and this is the method generally used to produce a flicker-free image.

2.3 RGB Signals

Any color can be reproduced on a picture tube by actuating the correct proportion the basic colors it is composed of. The final color is obtained by controlling the intensity at which the RGB pictures at the inside of the picture tube light up. To the human eye, the individual basic colors in the pixel group appear as one composite, color or hue at a particular saturation.

The need to convey the R, G & B is therefore, obvious. Since the sum of the luminance value of all the three is already contained in the y signal, only two further signal, R-Y & B-Y, are generated by means of differential operation of the therefore, obvious. Since the sum of the luminance value of all the three is already contained in the y signal, only two further signal, R-Y & B-Y are generated by means of differential operation of the Y signal. R-Y and B-Y are therefore referred as the color difference signals. Before these signal are transmitted, they are given relative brightness factors, the resulting chrominance signals may be written as

$$U = 0.49\ (R-Y)$$
$$V = 0.88\ (B-Y)$$

And the luminance, Y as

$$Y = 0.3R + 0.59G + 0.11B$$

The RGB intensity information required to control the respective electron guns in the picture tube is obtained from the R-Y, B-Y and Y information with the y signals, without causing interference on monochrome TV sets. The reduced ability to resolve color contours as compared to brightness values of the human eye can be used . This means that the color information may be transmitted at a relatively low bandwidth without significantly degrading the overall sharpness of the picture. In the PAL system, the color (or chrominance) bandwidth is about 1MHz.

The color difference signals are readily embedded in the frequency spectrum of the Y signal by making use of the fact that the spectral lines of the Y signal occur at even multiples of the line frequency (15,625 Hz). Also, the amplitude of these spectral lines decreases with frequency.

The color difference signals modulate a sub carrier of which the frequency, FC, is an odd multiple of the line frequency divided by four, plus the picture refresh frequency .

$$FC = 1135 \text{ X } (15{,}625/4) + 25 \text{ (Hz)}$$

This causes the spectral lines of the color difference signal to be slotted in between those of the Y signal, the color sub carrier frequency is set at 4.43361875 MHz, and the color difference signals are quadrature-amplitude modulated (QAM). The B-Y & R-Y components modulate the amplitude of the color sub carriers of 0 degrees respectively. The carrier itself is suppressed, so that it has an amplitude of naught in the absence of a color difference signal. This is done to keep the picture free from interference caused by the otherwise continuously present sub carrier.

In order to eliminate the risk of phase shifts in the transmission path, the phase of the R-Y component is inverted every other picture line. The use of amplitude modulation with suppressed carrier requires a phase and frequency- synchronized sub carrier at the receiver side. In a TV set, the modulated R-Y and B-Y components are recovered from the chrominance sub carrier with the aid of a 4.433 MHz quartz crystal oscillator , whose phase and frequency are corrected every 64 by a 2 long burst signal slotted into the rear porch in the blanking period at the end of every picture . The burst consists of 8 to 11 cycles of the color sub carrier frequency and follows the line sync pulse a shown in fig.2.3.a. a phase comparator is used to keep the crystal oscillator synchronized to the received burst, which also contains the PAL switch signal for the line-by-line R-Y phase reversal. This arrangement ensures that the R-Y signal in the receiver is inverted in synchronism with that at the transmitter side to ensure that the demodulation operation can work correctly.

In practice, the packaging of the luminance and the chrominance information into a single CVBS(Chrominance information blanking synchronization) signal is not without problems. Since the color sub carrier falls in the spectrum of the luminance signal, it causes a finely patterned type of interference known as moire. Luminance circuits in all modern TV sets are therefore fitted with a color trap, which is a relatively simple filter that removes most of the moire effects with the exception of those occurring at areas with sharp color transitions. Here, large phase jumps give rise to sub carrier side bands that fall outside the stop band of the 4.43 MHz color trap. Unfortunately, Y signals in this stop band are also suppressed, which results in reduced picture resolution because some of the high-frequency components disappear. Incidentally, most monochrome sets also contain a color trap to eliminate moire.

The interference between chrominance and luminance also works the other way around : since the luminance band includes the frequency range for the color sub carrier, high frequency Y signals can cause interference in the frequency range around 4.43 MHz. The result is a quasi-random type of pattering and coloring in and around picture areas of fine detail. Notorious examples of this happening can be seen virtually every evening in jackets, shirts or ties of people on television.

2.4 Character Generation

The generation of characters are done using the software which is used generate character matrix . Scan codes from a serial keyboard are compared by the software to a look up table in the EPROM containing the character matrix. The corresponding character matrix is sent to the output device for display.

Each character a matrix of dots is used, 5x7 or 7x9 are commonly used. Writing a line of letters or numbers on the screen is achieved as shown in fig 2.7. One row of dots at a time is written for the whole row of characters. So with a 5x7 matrix, 7 image lines are needed to write one row of characters. In fig 2.8 shows the number of these video signals with the modulation needed to write the word shown. Each pulse after the line synchronization pulse means that the electron beam is then lit on the screen. For clarity the pulses are shaded and the lines drawn close together to show how a characters put together. As this fig 2.9 show, the word 'VIDEO' would appear on the screen. The advantages of this extra line at the bottom are that we can make the lower case letters more accurately. An empty line is drawn between every two lines of characters on the screen so that the characters are separated from each other. Each character on the screen is consists of a pattern of light and dark dots. Dot patterns for the desired characters are stored permanently in the character generator.

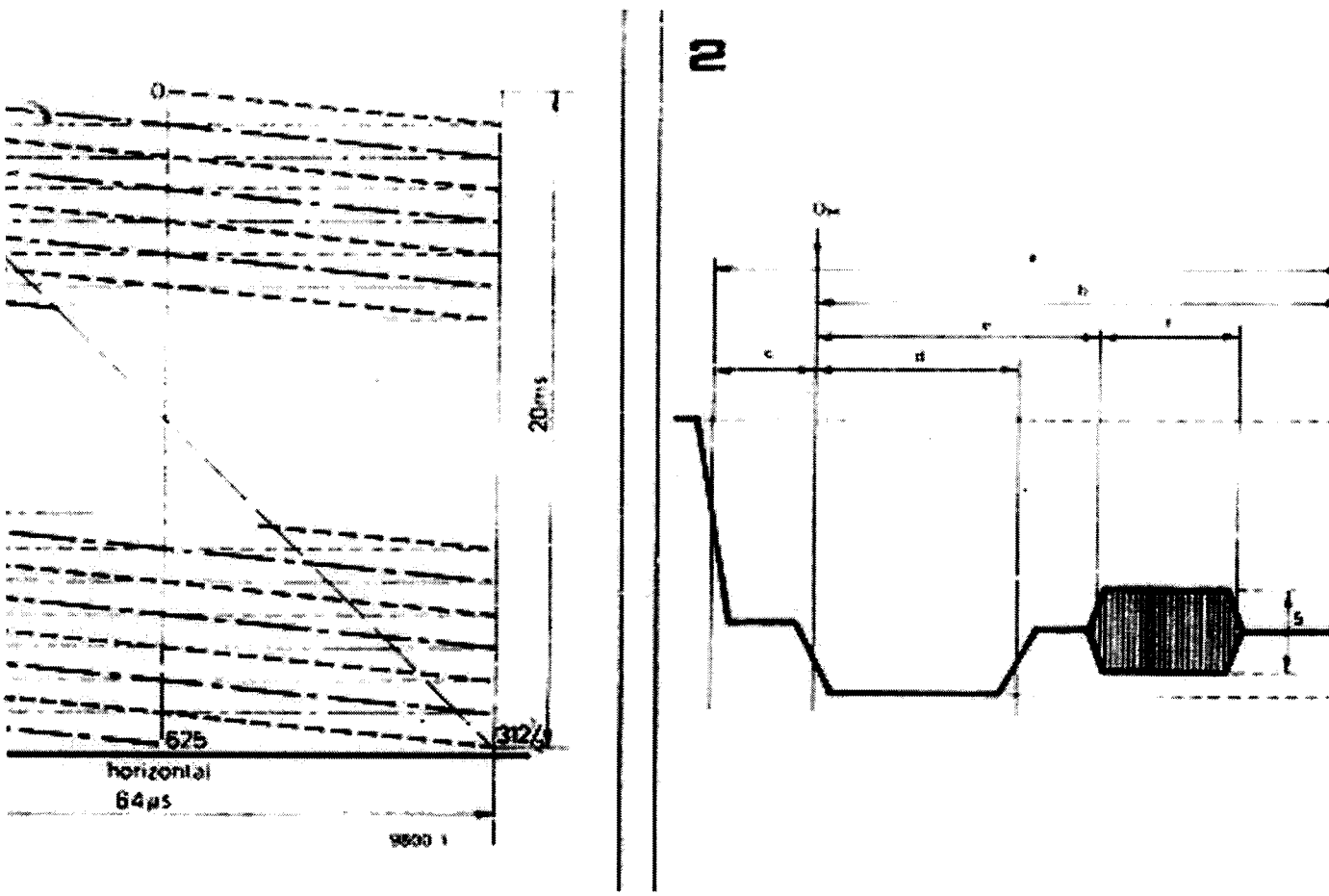


Fig 2.1 CCIR Standards

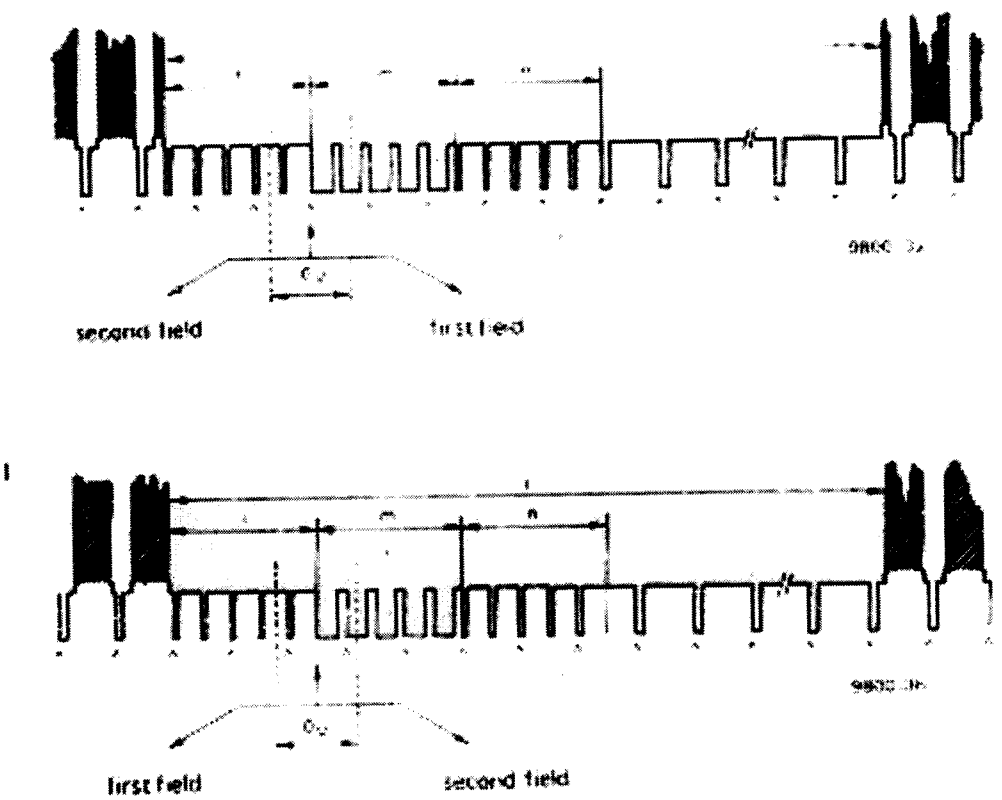


Fig 2.2 CCIR Standards

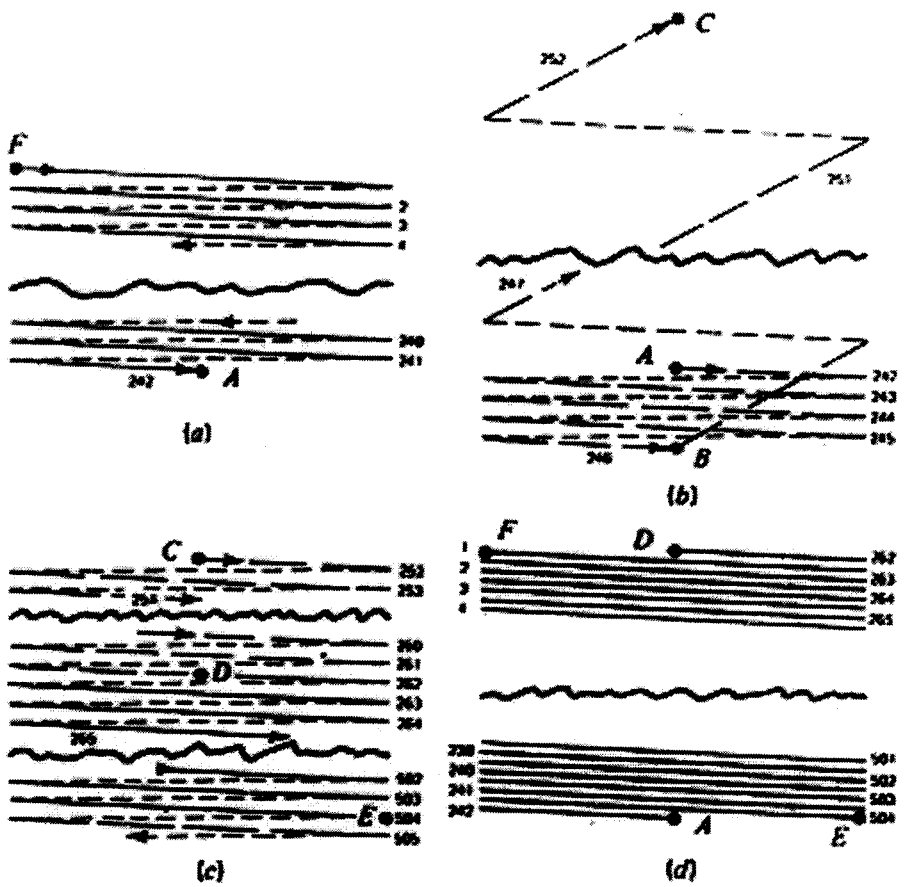


Fig 2.3 Scanning

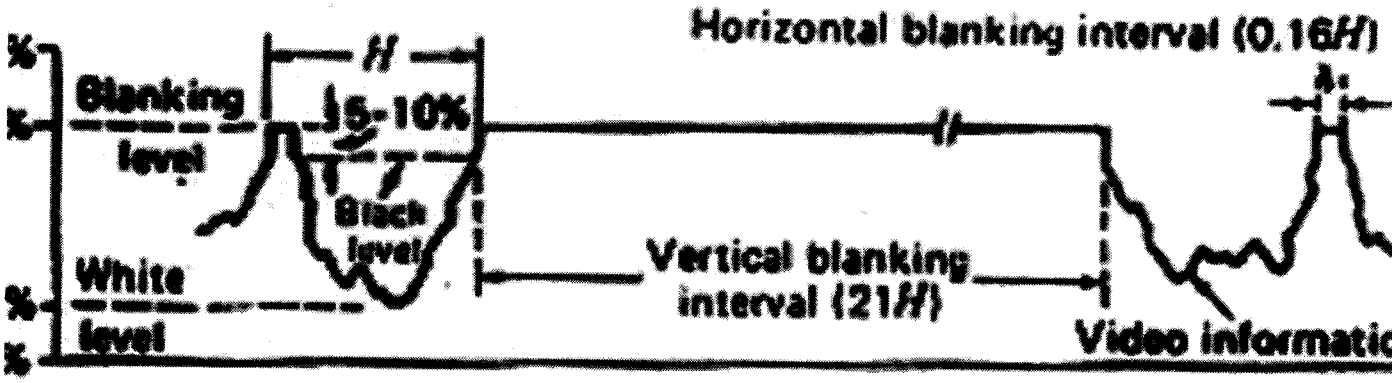
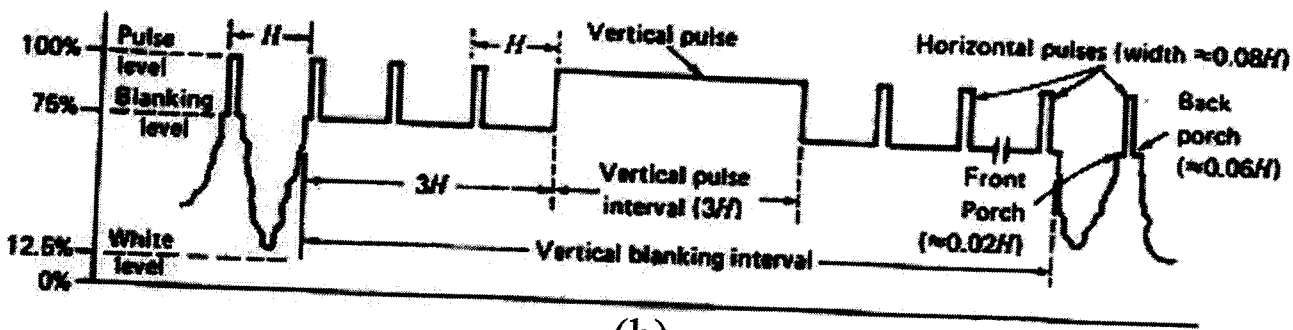


Fig 2.4 Video Blanking Waveform

(a)



(b)

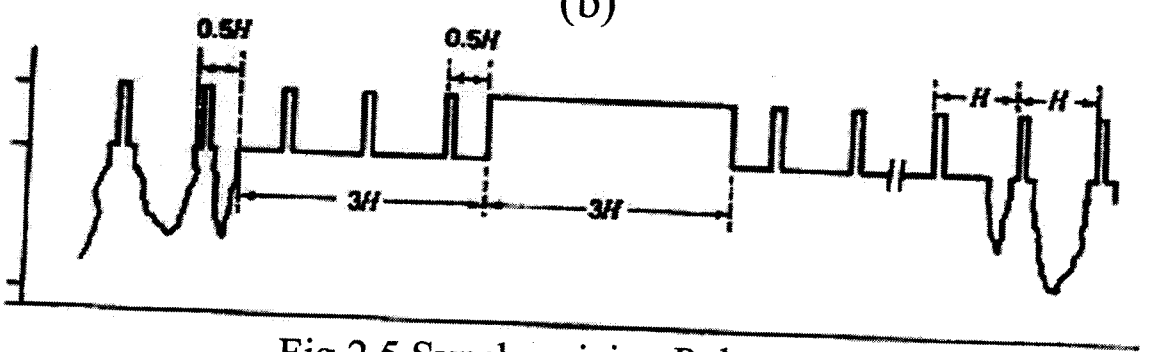
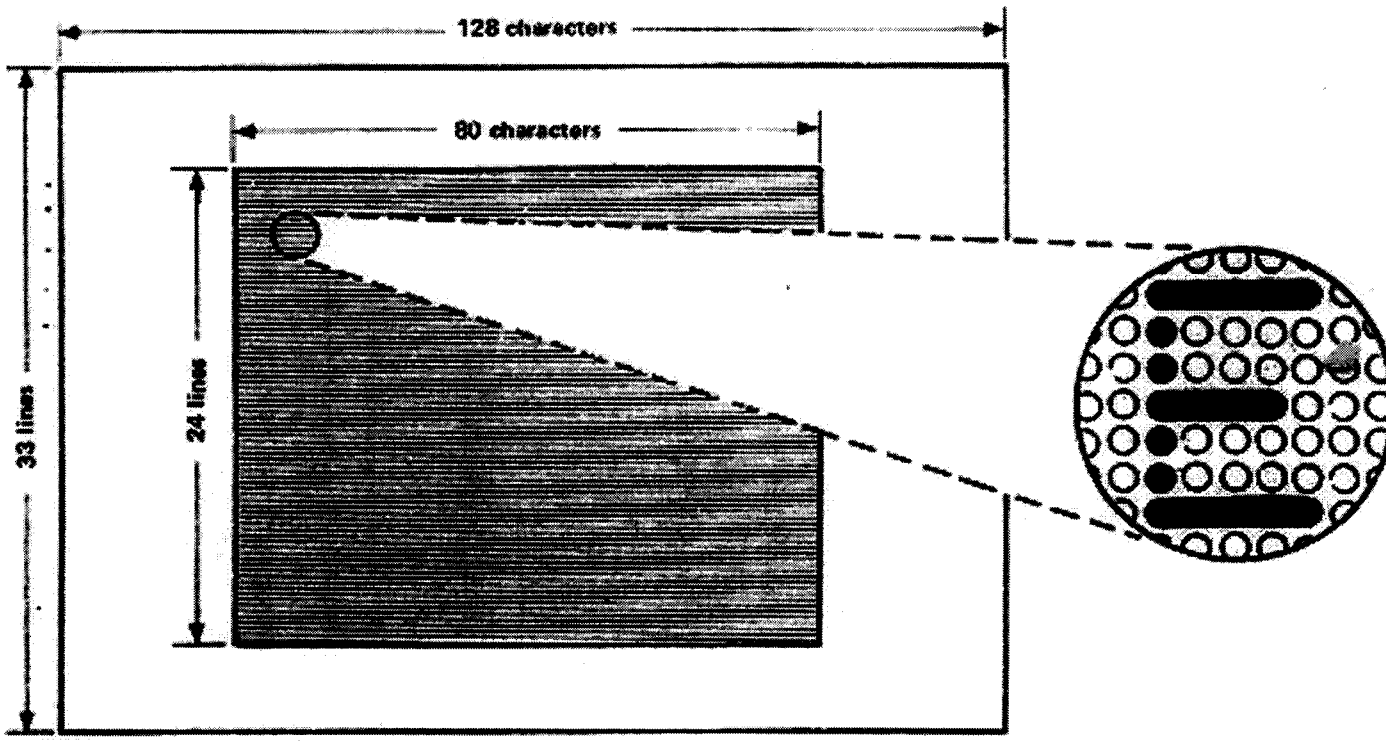


Fig 2.5 Synchronizing Pulses



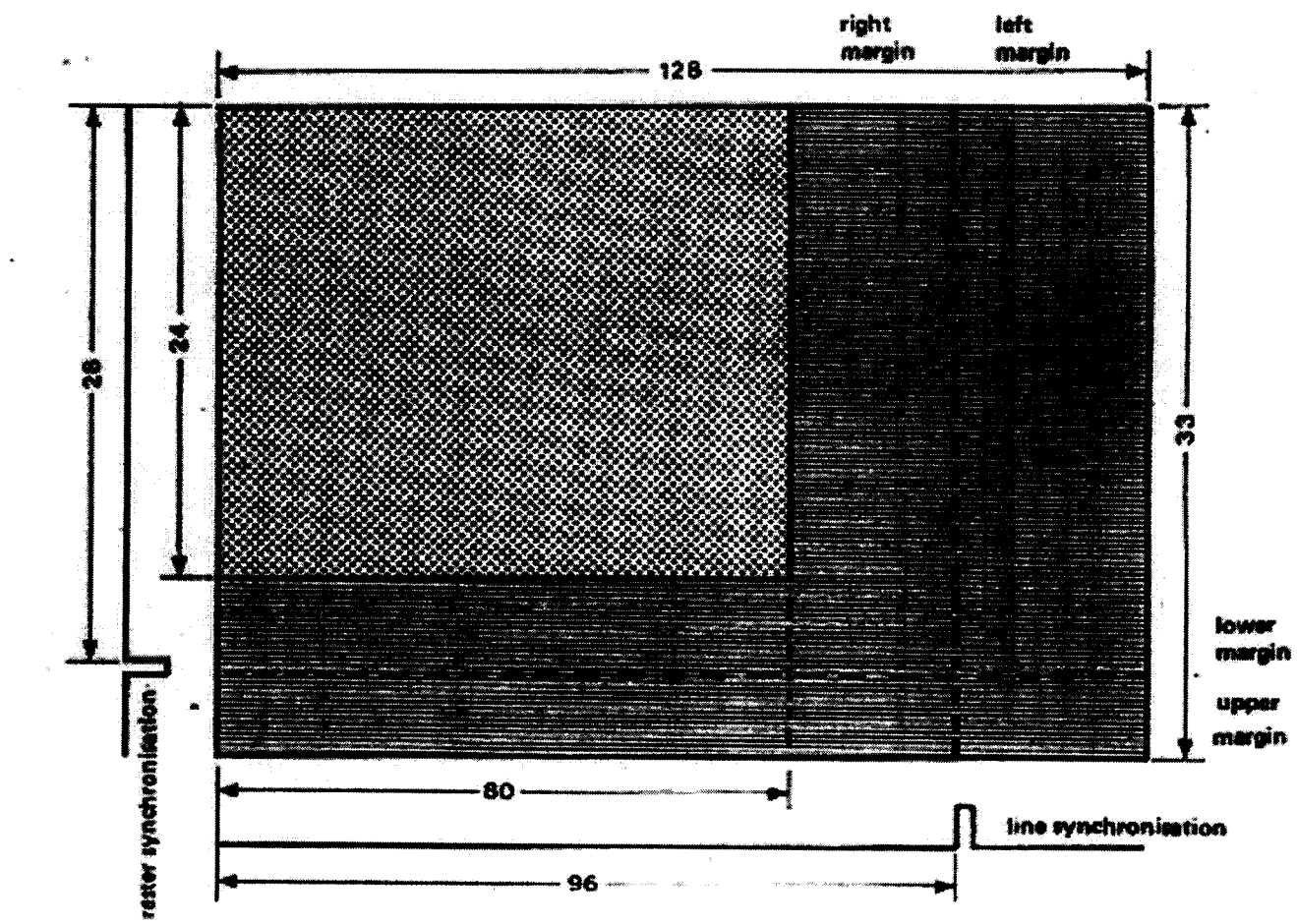


Fig 2.7 Display Pattern

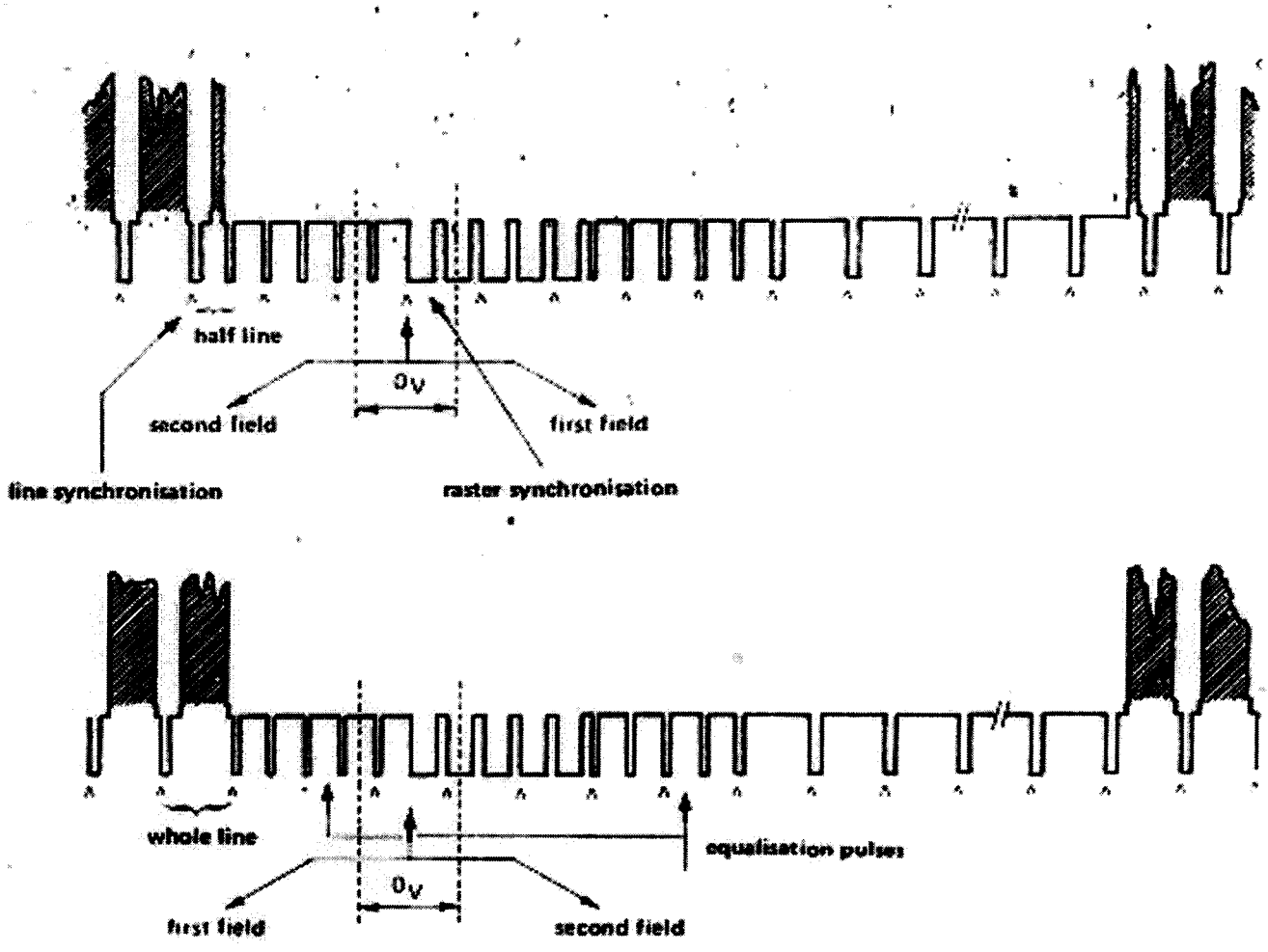


Fig 2.8

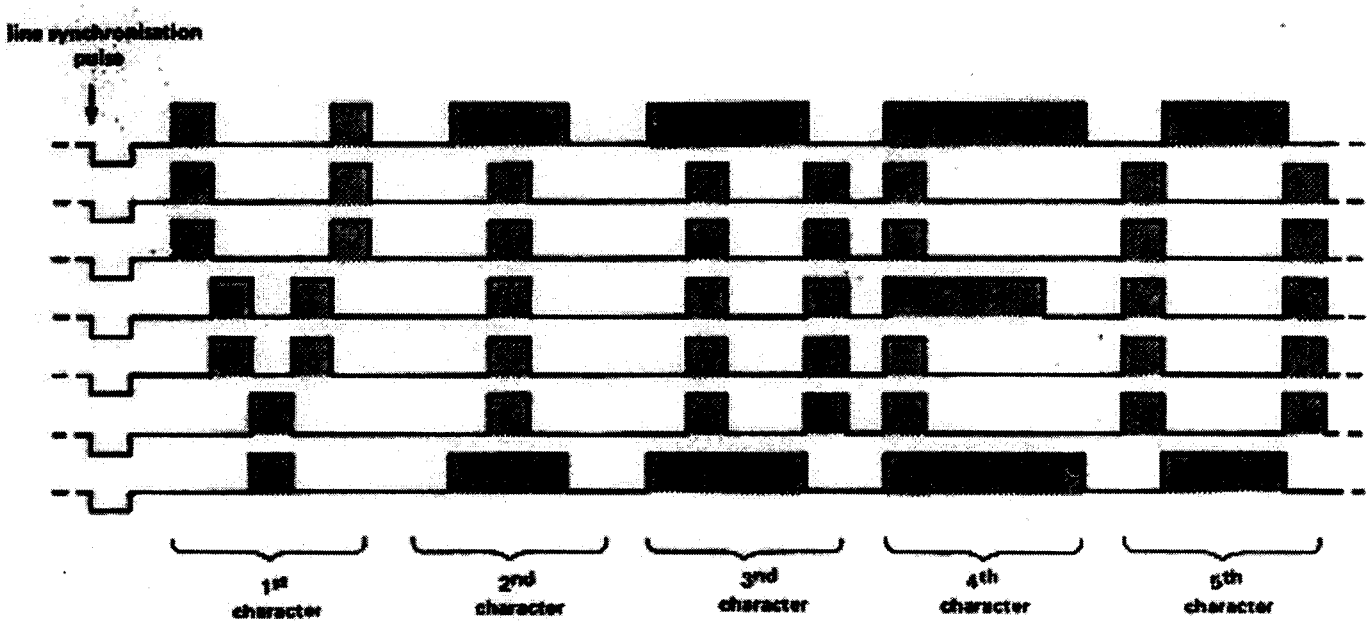
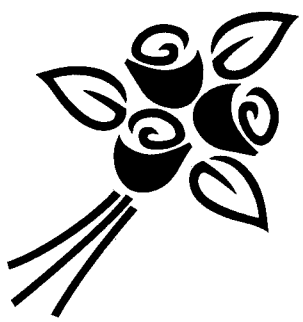


Fig 2.9 Display



MICROPROCESSOR BASED SYSTEM

CHAPTER III

MICROPROCESSOR BASED SYSTEM

3.1 Principle of operation

The circuit described here is a simple Microprocessor based system to provide the additional facility at the least investment. The advantage of the project is that it does not disturb the existing set up, and needs an intermediate block for generation and mixing. Multilingual operation is also incorporated.

The Microprocessor based system uses all the synchronizing signals from the input Composite Video itself. Thus the complexity involved is reduced considerably.

Basically the system has a string counter and flip flops which counts the input synchronizing signal in the analog side. This is used as a reference to where exactly the generated character is to be overlaid on the input Composite Video.

The Characters generated uses the scan codes from a standard serial keyboard (input section) .This scan codes are compared to the video matrix data to generate the characters. The output is obtained in a serial format from a shift register along with a Ram for data storage . The timings

to this register are obtained from the input from the analog section and a reference oscillator. Scrolling is performed by shifting the same data in time in the next cycle.

3.2 Basic Block Diagram

The basic block diagram of the Microprocessor based system is shown in fig 3.1. It consists of a composite video source which can be a satellite receiver or a VCR or a VHS Video Camera . This signal is given as the reference to the timing section. This section generates timing signals with the input composite sync signal as reference. This is used as the shift signal in the parallel load serial shift register.

The input section acts as an interface between the standard PC XT keyboard and the microprocessor system. The output of the interface section, are unique scan codes for every key pressed. These scan codes are converted into video matrix using the look up tables in the memory. These video matrix are stored in the video RAM and acts as input to the parallel load serial shift register. English or Tamil look up tables can be invoked by the use of any special key in the input section. The software takes care of comparison of scan codes with the look up table and sends required outputs to the video RAM. The scrolling of text is obtained by shifting the Video Matrix in time for next cycle. The speed of scrolling is controlled by a combination of Hardware and software while the height at which the display is to start is controlled by Hardware only.

The output of the parallel load serial shift register is the serial format of video matrix , which is just paralleled with the input composite video to produce the overlaid output. Paralleling of the input and output are possible as the input composite video signal is used a reference for the timing pulses.

3.3 Circuit Description

3.3.1 Micro Processor section

The schematic of the microprocessor section is shown in fig 3.2. using the Intel 8085 chip. 8085 has a multiplexed lower order address and data bus which is de multiplexed using the address latch enable signal (ALE). The ALE signal is used as a chip select of an 8 bit latch (74LS373) to accomplish de multiplexed busses to get lower order address bus A0 to A7.

The input device is addressed by the I/O mapped I/O scheme with an address of 00(H) using 74LS139 as shown. The main memory (RAM 62C256) and the EPROM (27C256) are addressed by memory mapped scheme. With starting address of 0000(H).

Two external interrupts namely RST7.5 and TRAP are used. The TRAP line is used to control the speed of the scrolling text . This is accomplished by using an astable multivibrator using IC555 with a variable duty cycle. This invokes the TRAP interrupt service subroutine to vary the scrolling speed. RST 7.5 is used to transfer the input data from the input section buffer.

The clock of the system is a 6 MHz quartz crystal connected to X1 and X2 (not shown in fig). This is internally divided by 2 to obtain a 3 MHz clock speed. This clock speed is not sufficient to work in Video frequency of 5 MHz. So a separate Clock Oscillator of 10 MHz is generated as reference to the output section as well as for generation of timing signals.

The fig also shows a latch 74LS245 ,a parallel load serial shift register 74LS166 and the video RAM 6116 which will be dealt in detail in the output section.

3.3.2 Input section

Introduction to Keyboard

Keyboard circuit even though it is theoretically possible to have a keyboard with one key for each of the 128 functions, this could be rather confusing. To sidestep this difficulty ever key is normally given a double (or triple) function and, in the same manner as scientific calculators, a shift key is added to the keyboard to select which particular function associated with a key is required. When a key is pressed, the corresponding, a coding IC forms ASCII code word. This simply consists of a ROM containing all the ASCII codes, and is addressed by the keyboard via two counter circuits whose outputs form a matrix. The RC network connected between pins 2, 3, and 40 of the decoder/encoder IC determines the frequency at which the matrix is scanned (in fact, the counter clock frequency). One of the

counters delivers its particular code to lines XO.... X7 and the other send its binary code to lines YO.... YO, these then forms the address of the ROM are addressed by the second counter circuit: in fact two of them are tied to the SHIFT and CONTROL keys.

Keyboard consists of a key of switches. There is one key switch for each letter, number, symbol etc., much like a typewriter. When a key is pressed , the key switch is activated . The keyboard has an electronic circuit called keyboard encoder to determine which key has been pressed . Then a standard 8 bit code is generated (scan code) and sent to the computer .Detecting which key is pressed and generating the corresponding code is known as Encoding. The fig 3.3. Shows the basic structure of a keyboard.

There are 2 types of keyboards

1. Serial keyboard
2. Parallel keyboard

In a serial keyboard the data is sent bit by bit in a serial fashion. In a parallel key board the 8 data bits are sent simultaneously . Normally a standard serial keyboard is used for this applications .

Keyboard Interface

The keyboard used here is a standard serial keyboard . The Fig 3.4 shows the basic structure of a keyboard matrix . Each key has a unique

set of coordinates , a row number and a column number . When a key is pressed , the keyboard encoder has to determine the pressed key's row number and the column number . This encoding is performed in matrix keyboards by scanning technique . The scanning method uses rows as inputs and columns as output to the matrix .It scans the matrix row by row to determine the key press and generates 8 bit serial data .

The keyboard interface receives the scan codes in a serial format from the keyboard and generates the interrupt request to the microprocessor. Then the data is stored in a Ram and this data is mixed with the video signal in the analog section which will be discussed later .In a serial keyboard has the following inputs namely +5 volt DC & GND and outputs namely clock signal , data , reset(normally not used). The scan data is processed and stored in a Ram . the corresponding 5*7 dot matrix pattern of the pressed scan code is generated using the character Ram .

The Fig 3.5 shows the block diagram of keyboard interface (Input section). The connector details of serial keyboard are shown in fig 3.6. It consists of a mono flop, shift register and a latch. The serial input data is given to the shift register and the inverted keyboard clock output is used to shift the data. The Parallel data is buffered and transferred to the main memory using the RST 7.5 interrupt service subroutine. The Mono flop is used to generate the Interrupt signal after the serial to Parallel conversion is over. This data (the scan codes) are used in the main program to generate characters.

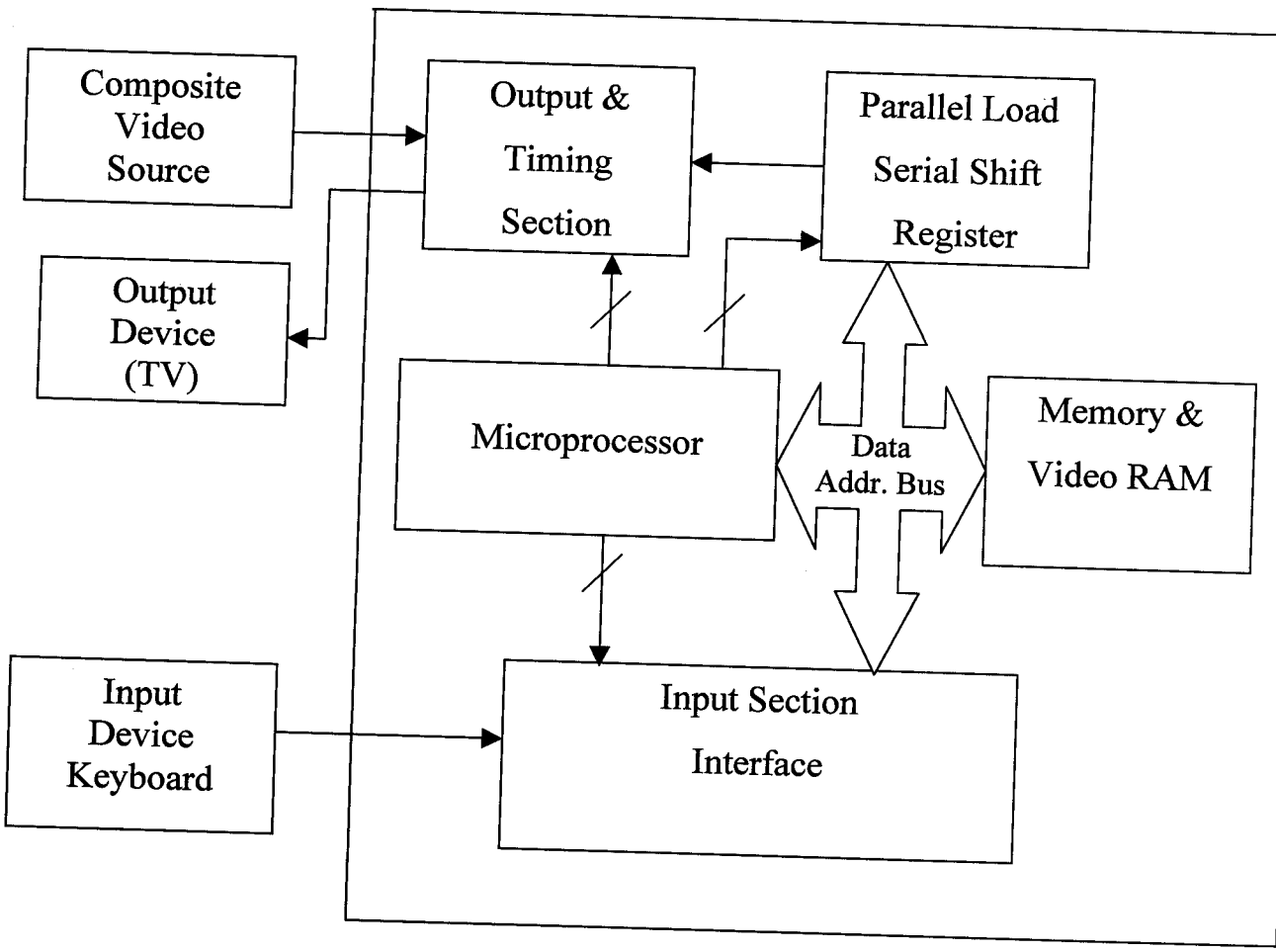


Fig 3.1 Basic Block Diagram of Microprocessor Based System

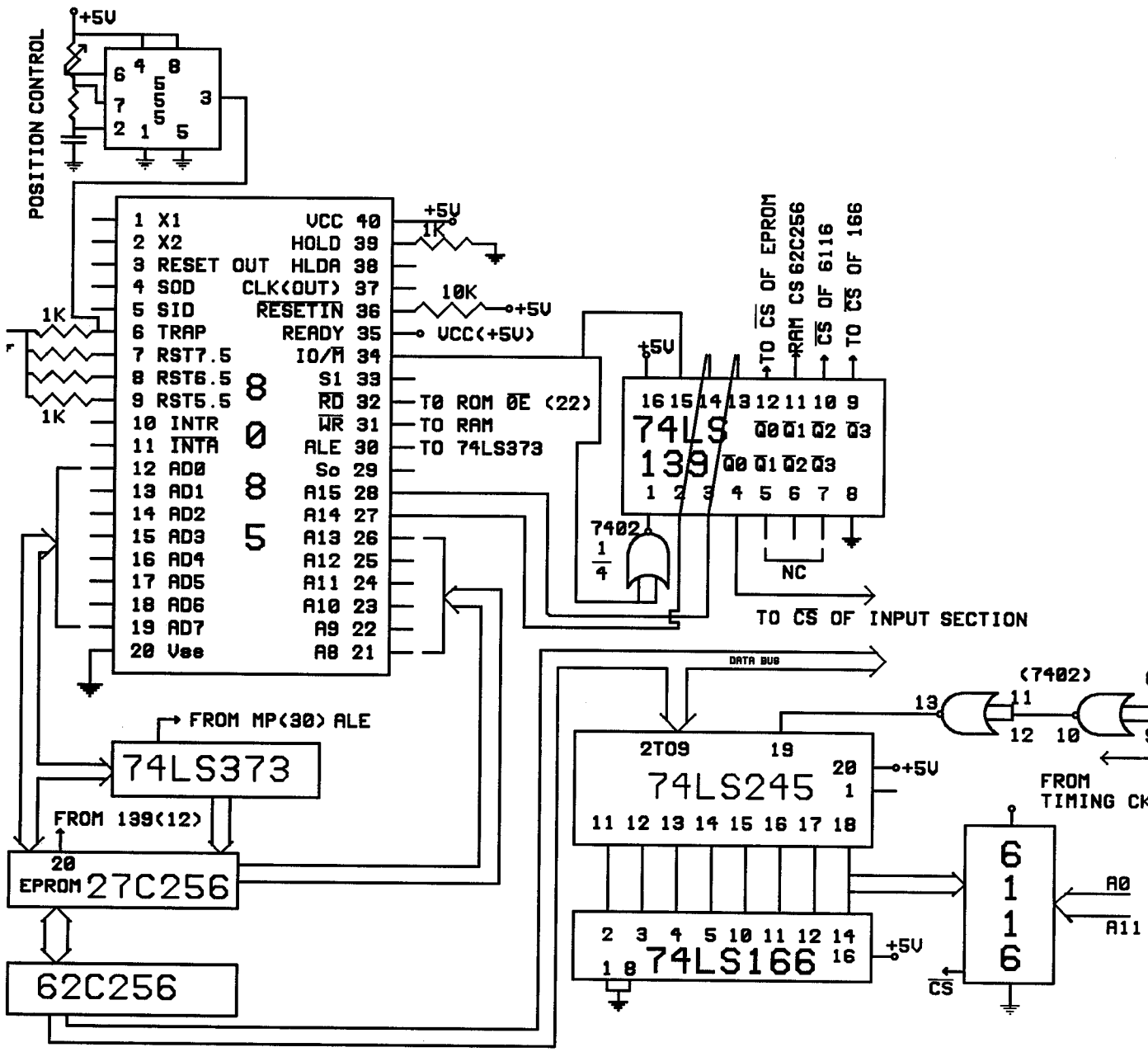


Fig 3.2 Microprocessor Section

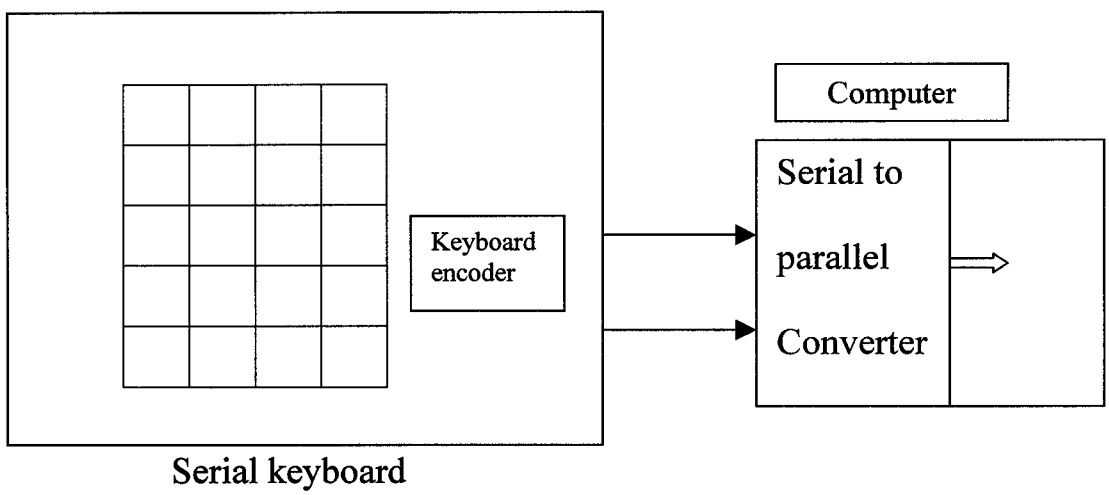


Fig3.3 Basic structure of keyboard

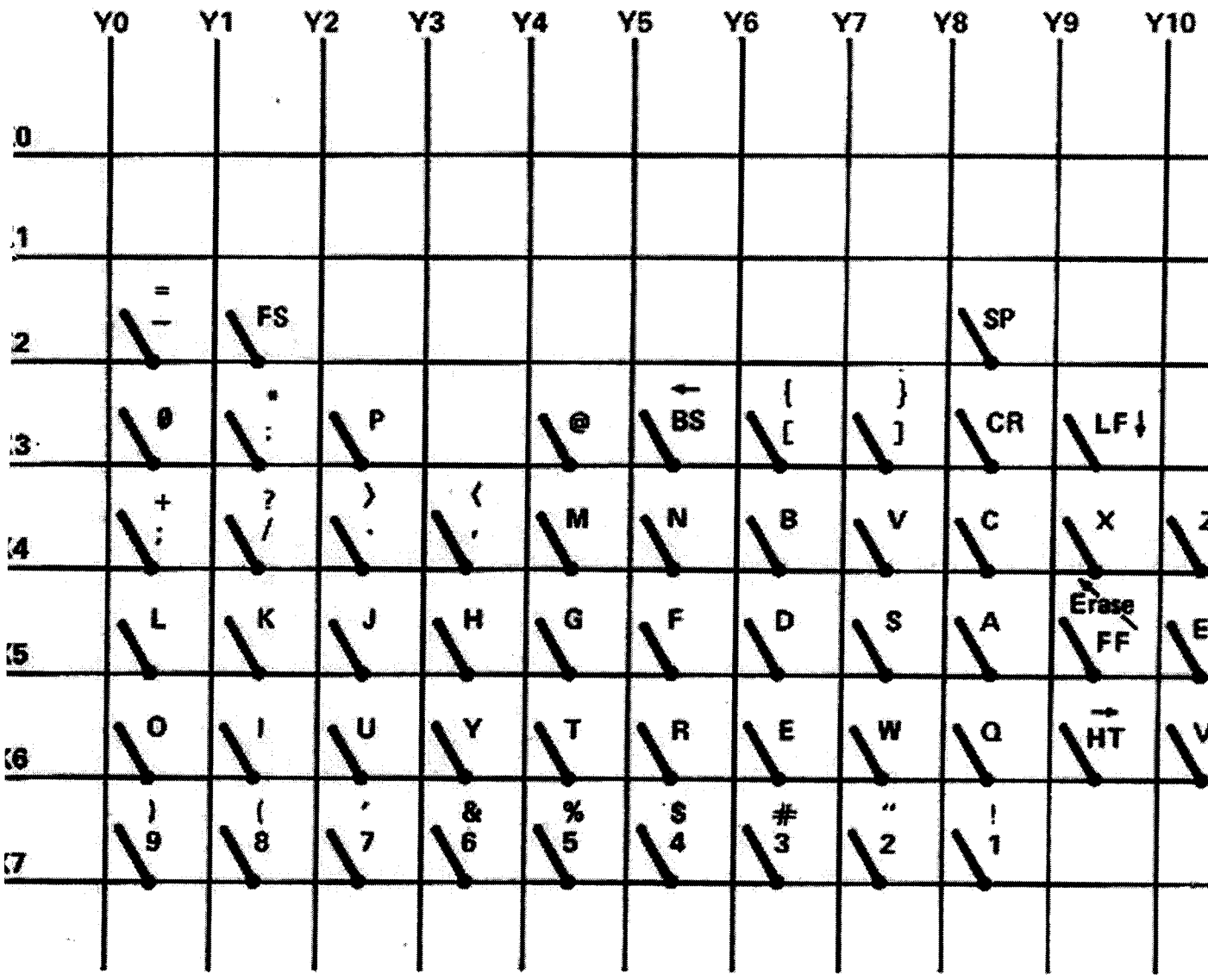


Fig 3.4 Keyboard Matrix

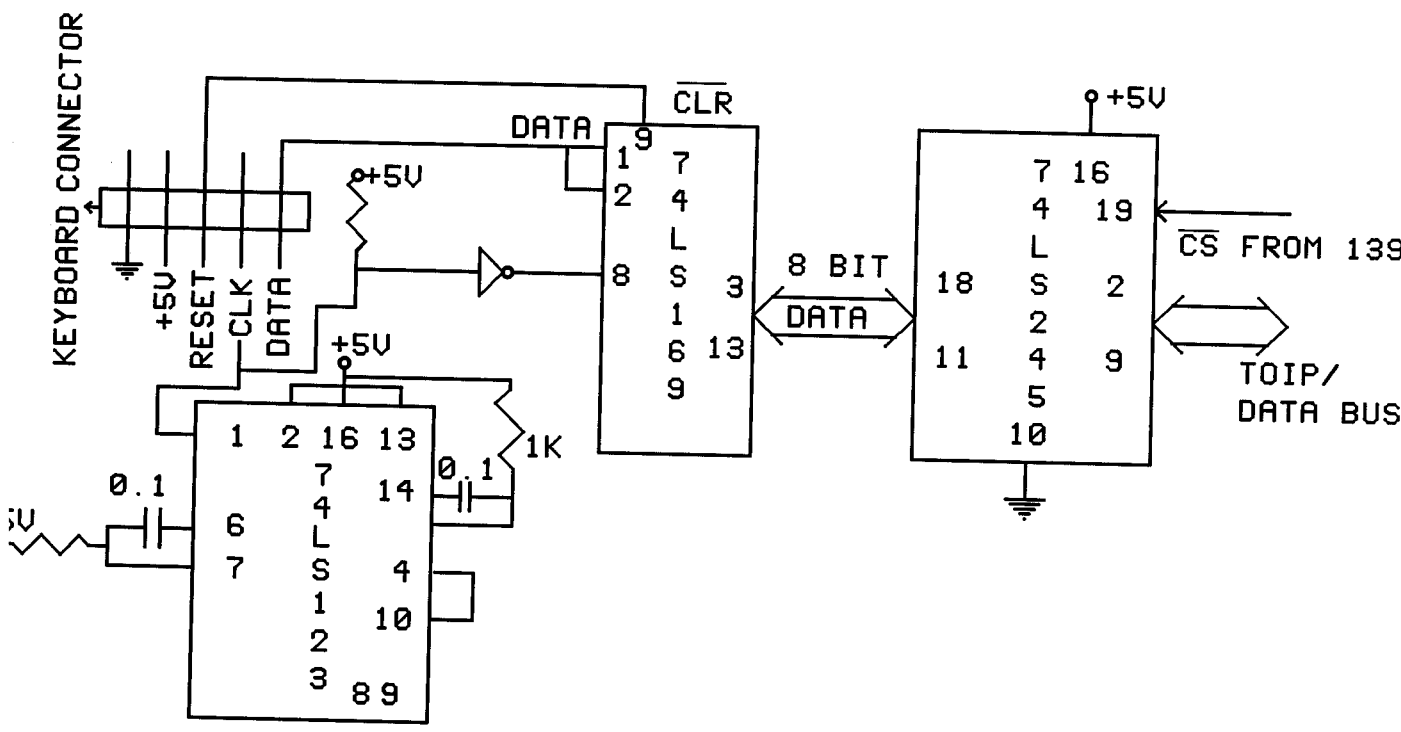
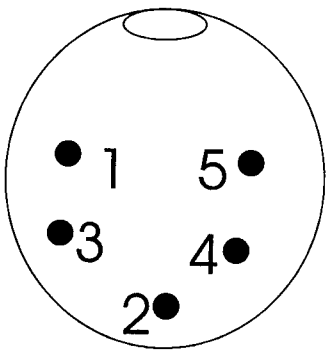


Fig 3.5 Input Section



Pin no.	Signal
1.	Keyboard clock
2	Keyboard data
3	Reset
4.	Ground
5.	+5v

Fig 3.6 Keyboard connector details (DIN)

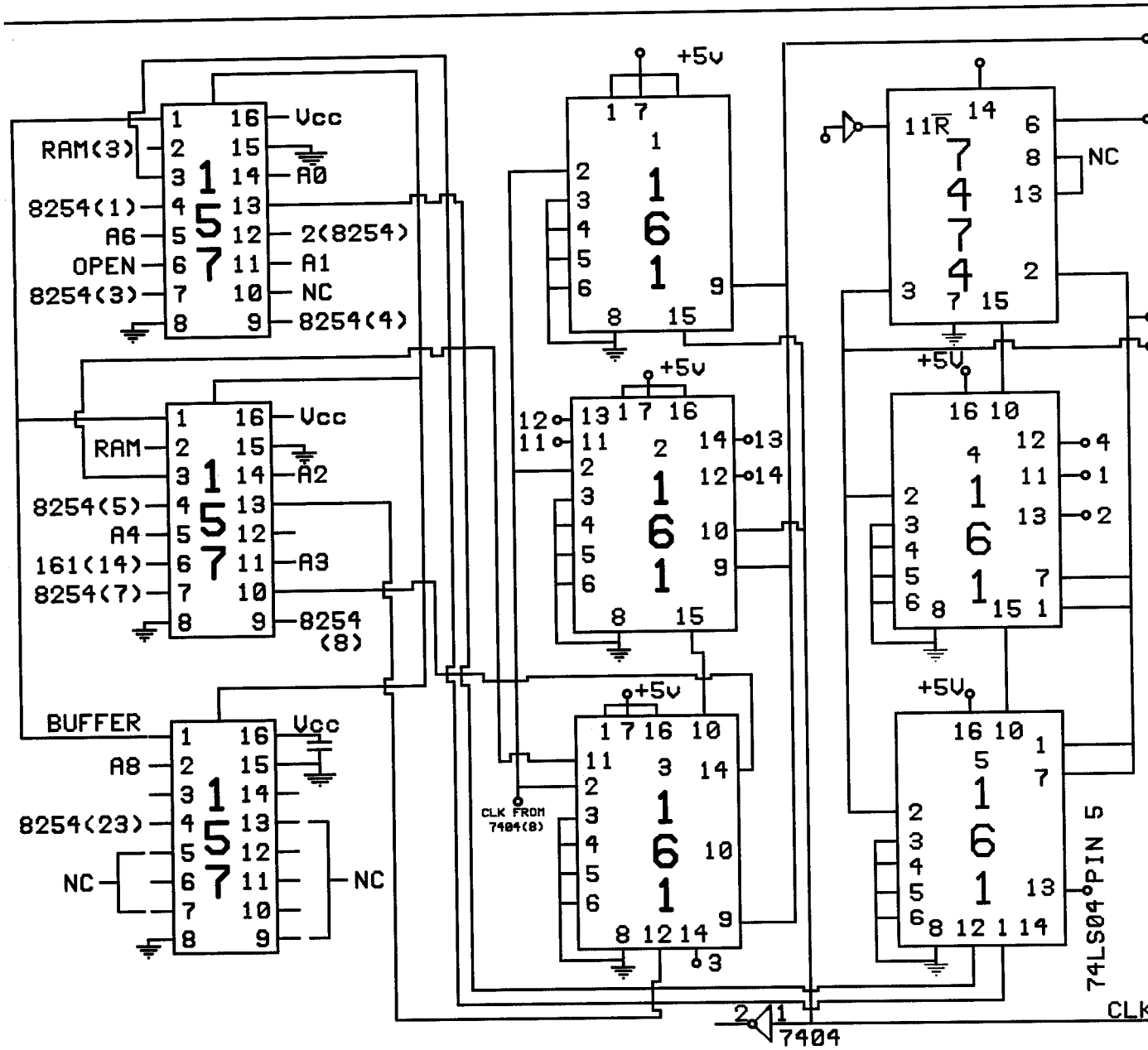


Fig 3.7 Output & Timing Section

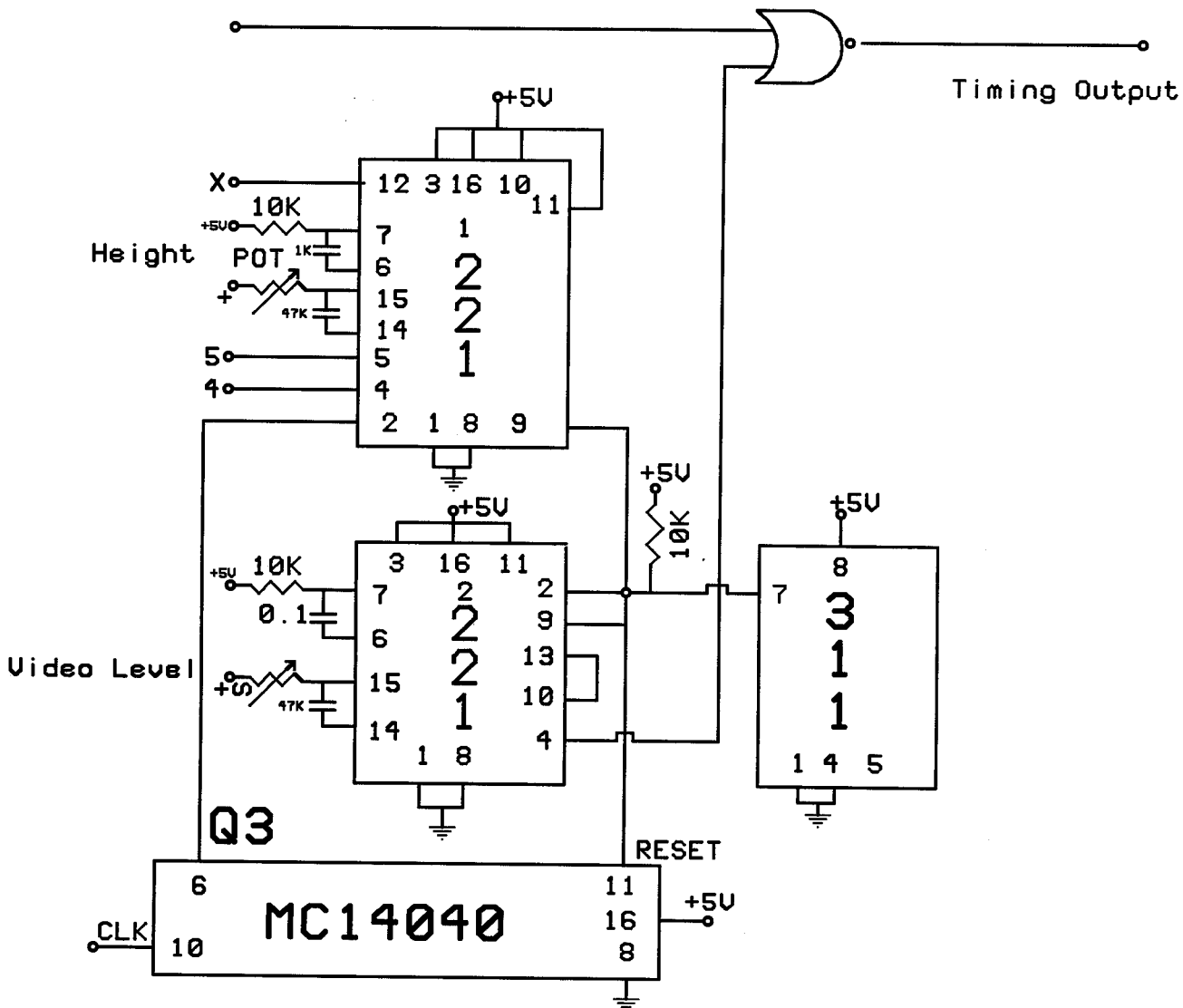


Fig 3.9 Output & Timing Section

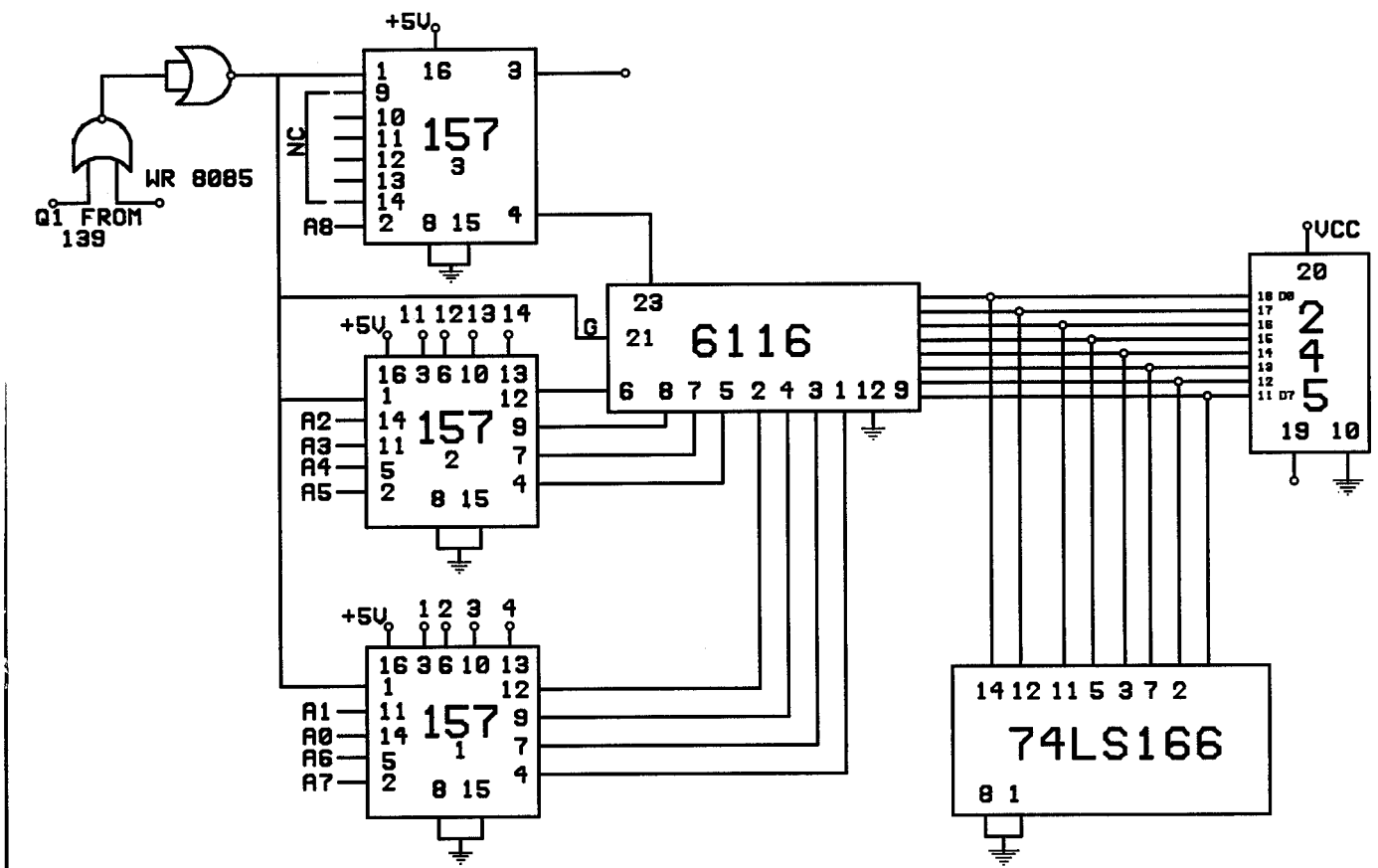
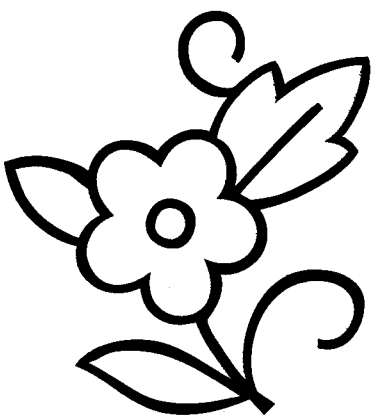


Fig 3.8 **Output & Timing Section**



DEVELOPMENT OF SOFTWARE

CHAPTER IV

DEVOLPMENT OF SOFTWARE

4.1 Program Description

The programming sequence used here is

- Main Program initializes all registers and checks for input data
- When input data is sensed RST 7.5 subroutine is invoked using Hardware interrupt to the Microprocessor
- This routine fetches the parallel converted scan codes
- It processes them using the look up table in either English or Tamil as per the key sequence.
- The data is in the form of 5 x 8-video matrix.
- These video matrixes are sent to the output parallel load serial shift register using the TRAP subroutine.
- The TRAP routine is invoked using Hardware interrupts to the microprocessor
- The TRAP routine sends the video matrixes according to the timing signals from the timing section
- The frequencies of the timing signals are decided by hardware settings of the speed control Potentiometer.
- The Height and the Position of the scroll is decided by hardware settings of the height and position controls.

The program is given in Appendix A

4.2 Algorithm

Main program

Step 1 Disable interrupts

Step 2 Initialize all devices

Step 3 Initialize stack pointer ,program counter and other registers

Step 4 The content of video RAM which is on the form of character matrix Is moved to the output section

Step 5 The timing signals are used to convert the character matrix into time domain so as to obtain scrolling

Step 6 Enable interrupts

Step 7 If there is an input data, call RST 7.5 Interrupt Service Routine

Step 8 Go to Step 4

RST 7.5 Routine

Step 1 Disable interrupts

Step 2 If scroll lock is pressed , then stop scroll and data is stored in main memory

Step 3 If alt key is pressed, then TAMIL character look up table is invoked

Step 4 Else ENGLISH character look up table is invoked

Step 5 Using lookup table, the input data is converted into a character matrix using Scan codes from the keyboard

Step 6 The character matrix thus formed is stored in the video RAM

Step 7 Return

TRAP Routine

Step 1 Disable interrupts

Step 2 Push PSW and H to stack

Step 3 Load and increment HL content with data from character matrix

Step 4 Store HL content to output device

Step 5 Retrieve H and PSW content

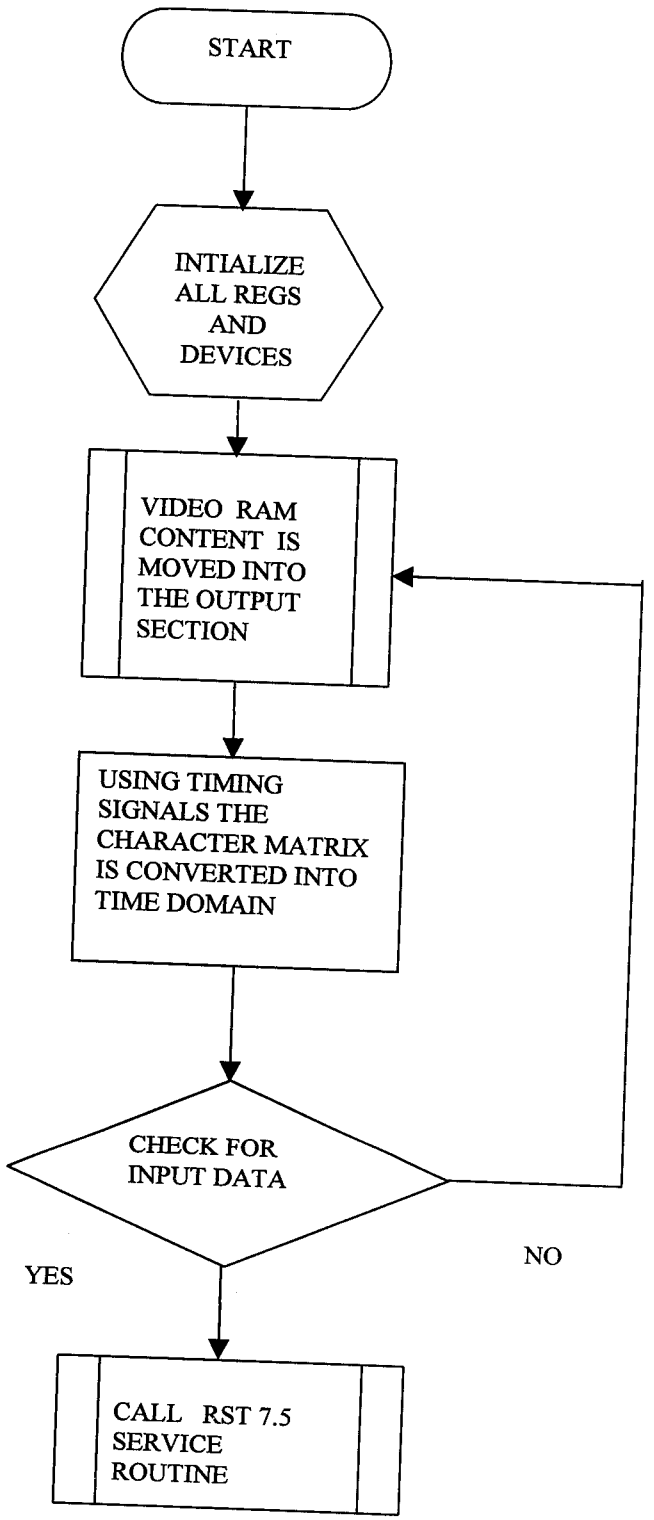
Step 6 Enable interrupt

Step 7 Return

4.3 Flow Chart

The various steps listed in the algorithm are shown in the flow charts in fig 4.1.

Main Program



TRAP Routine

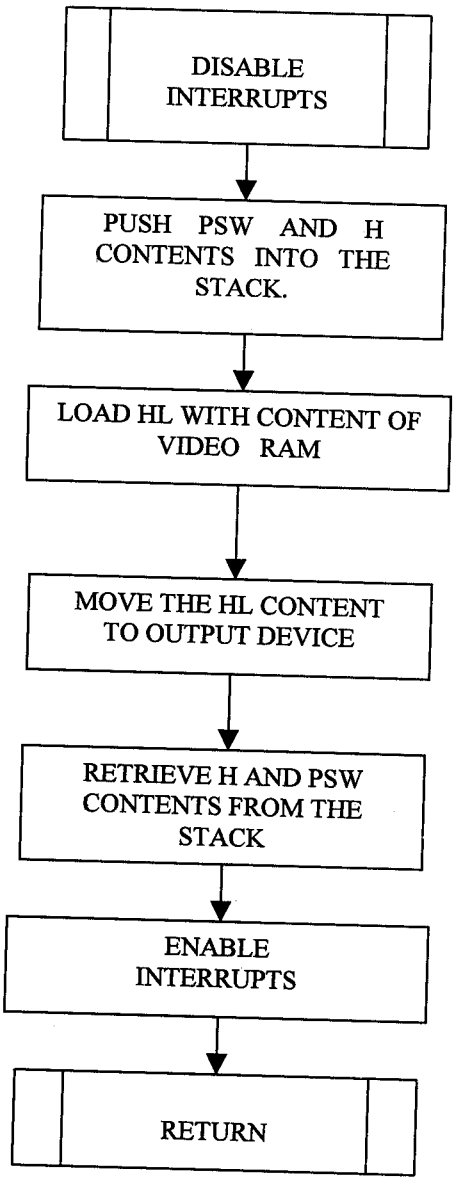
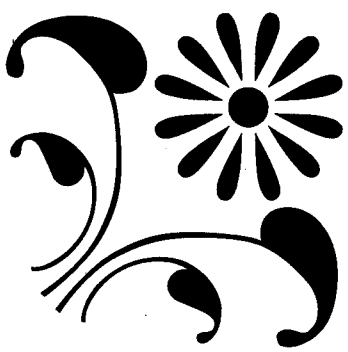


Fig 4.1 Flowcharts



PC BASED SYSTEM

CHAPTER V

PC BASED SYSTEM

5.1 Principle of operation

The principle of operation is illustrated in fig 5.1. The system can described by two blocks namely a Video System and a Overlay synchronizer circuitry. The video system provides the following outputs namely

- Horizontal Synchronizing pulse
- Vertical Synchronizing pulse
- R,G & B Signals
- Video Enable
- Local / Remote signal and
- Overlay Enable
- 4.43 MHz signal for PAL

The Overlay synchronizing system provides a master clock (36 MHz) as a reference to the Video system. These signals are utilized in the overlay section to generate local Video.

The Remote source (Composite Video Source) provides the Video over which the local video is to be overlayed. The system provides the overlayed composite video output. The overlay synchronizing circuit has a mode selector to get either a Remote / Local /Overlayed Video output.

5.2 Basic Architecture

Figure 5.2. shows the basic block diagram of the PC based system. The system consists of a computer to provide VGA/ AGP output namely Horizontal Sync, Vertical sync, R, G & B .(Required for the system) and a remote video source over which the generated Graphics/text have to be overlaid. These two source acts as input to the video overlay synchronizer, which along with the timing circuits synchronizes the two sources.

There is mode selector, which is used to switch between Remote (Composite Video Source), Local (Computer source) or overlay of both. The Timing circuit is used to compare the syncs of the two sources to provide timing signals to the overlay synchronizer.

5.3 Circuit Description

The circuit has the IC MC1378p which acts as an interface between remote composite color video sources and a locally controlled RGB source of video, and it contains the necessary synchronizing circuits, plus a complete color encoder. This is shown fig 5.3.

The MC1378p contains all the necessary circuitry to lock a computer to a remote color composite video source and to switch between the remote and the locally generated signals to create overlays in composite

video. Because the MC1378p, when operated in the remotely locked mode, passes the remote signal directly to its output without decoding and re-encoding, no loss in picture quality is experienced as can happen in less sophisticated system.

5.3.1 Local mode

Fig 5.4 shows the block diagram of the IC MC1378p and its external components. Because the MC1378p operates in two basic modes, local and remote, it is logical to describe them separately.

In the Local mode no external video is required and the main function of the MC1378 is to encode the 1 V RGB signal. A double balanced balance detector, PD5, is used to compare the now free running 4X subscriber oscillator divided by four with the entering sub carrier signal at pin 8 and control the clock oscillator. The clock is divided down by the appropriate number within the graphics system to sub carrier frequency. This forms a PLL using the crystal oscillator at pins 10 and 11 as a reference.

A separate clock could be used if, it is not a multiple of the sub carrier frequency – the disadvantage being that the encoded Video signal's sub carrier will not be related to the horizontal frequency, and unpleasant dot crawl or beating on the display may result.

PD1 is a digital phase detector that compare the horizontal TTL sync at pin 40 with the MC1378p internal horizontal sync and control 4 MHz VCO form a PLL. The 4 MHz VCO signals are internally divided by 256 to horizontal frequency. The eight stage divider is also used to develop

the burst gate and burst flag signals by decoding the countdown. Burst gate is used extensively within the device for gating and clamping the Chroma and video signals. Burst gate is 4 ℓs wide and local oscillator centered about the 2.2 ℓs burst flag signal. Burst gate also fed out of pin 5 to drive other devices that should locked to horizontal frequency. Phase detectors PD2, 3 & 4 are not actively used in the Local mode but PD4 sets an arbitrary oscillator phase to the two electronic phase shifters.

In the PAL mode the R.Y modulator is phase inverted line by line and burst flag is sent to both from B-Y modulator. The PAL flip – flop runs at an arbitrary phase in the local model when the ident circuits enabled by and external diode connected to pin 29.If a particular PAL phase is required, the PAL flip-flop can be reflect at this pin. Fig 5.5 shows timing circuitry needed in local mode.

The overlay enable (pin 25) should be set low in the Local mode, to view the NTSC or PAL enabled RGB signals at pin 27.

5.3.2 Remote mode

In the remote mode all phase detectors and active except PD5. An external valid video signal or remote signal must be fed into pin 24 to provide the timing information to the host computer. Composite sync is separated from the remote signal and fed the vertical sync separator to detect vertical sync. The separated composite sync is used to lock the 4MHz VCO using PD1, the vertical sync being fed out to the graphic

System to lock its sync generator. The 4 MHz is divided by 256 to horizontal frequency and this compared in PD2 with the TTL negative going H - sync signal at pin 4. The output of PD2 used to lock the system clock VCO, the frequency of which can range from 14 to 36 MHz depending upon the host computers requirement. The system clock is divided down to Horizontal Sync frequency with in the host system and fed into pin 40.

The color burst from the remote signals is used to lock the 4X color sub carrier oscillator using PD3, which is gated with burst gate. By using PD4 and comparing the burst of the locally generated composite video from the encoder section with the same sub carrier reference used to lock PD3, the sub carrier phases of both the local and the remotes signals are made essentially equal. Similarly, the two burst amplitudes and compared in the ACC detector and made equal using a variable gain ACC amplifier in the locally generated chroma path.

The absolute burst amplitude of the remote signal only is detected by the kill detector, the chroma of the locally generated signal being turned off when the remote burst falls below a pre determined level. The kill level can be adjusted by changing the value of the resistor pin 31. 470k kills at about 10-20 mV p-p remote burst ,(normal =300mvp-p).

In the Pal mode the phase of the ident of the remote burst is compared with the half line signals from the PAL flip-flop. If an error is detected, indicating that the local ident is not compatible with the remote ident, the flip - flop is reset using the ident processor, if a continuos ident

error is detected, i.e. fixed or not burst on remote signal the chroma in the local signal is killed.

Because the black levels, burst phases, burst amplitudes, and in the case of PAL. Ident states are compatible between local and remote signals, the fast video switch operated by the overlay enable signal fed into pin 25 can be used to switch from one signal to the other to create overlays is composite video. Even portions of the timing waveforms (sync, burst, etc.,) can be selected from either the local or remote sources for specific purposes, such as noise reduction due to weak signal remote, or VCR tape jitter reduction.

5.4 Phase detector operation

Local Mode

PD1 – compares the internal horizontal frequency derived from the 4 MHz with the Horizontal sync derived from the master clock from the host computer. The PLL formed locks the internal horizontal signals to the host computer's signal.

PD2 – not used in Local mode

PD3 – not used in Local mode.

PD4 – active, but providing an arbitrary phase- shift setting between the sub carrier references and the output chroma phase of the locally generated composite video.

PD5 – locks the master the host computer to the sub carrier frequency within the host computer to the four times sub carrier crystal oscillator. The crystal oscillator becomes the systems timing standard in the Local mode.

Remote Mode

PD1- compares and locks the internally counted down 4 MHz VCO to the incoming remote horizontal sync. It is fast acting to follow VCR source fluctuations, etc.

PD2 – locks the master clock oscillator by comparing the internal horizontal signal with the H sync returning from the host computer.

PD-3 a gated phase detector, which locks the crystal oscillator frequency divided by four to the incoming remote signal burst.

PD4 – controls an internal phase shifter to assure that the outgoing local color burst the same phase as the incoming remote burst at PD3.

5.5 MC1378 Set – Up Procedure

1. Switch is Local mode (pin 1 = 0 V) Ground pin 25.
2. Using a source accurate sub carrier frequency as an oscilloscope trigger, adjust the variable capacitor at pin 10 so that the burst appearing at pin 20 is the correct frequency to with 10 Hz.

NTSC – 3.5795454

PAL – 4.4333619

3. Disconnect the signal feeding into pin8 (3.58/ 4.43 MHz) Measure this frequency and adjust the Clock Oscillator until the meter reads 3.58 MHz

(NTSC) or 4.43 MHz (PAL) – 10 kHz. Reconnect the signal to pin 8. This signal should now be phase – locked to the burst frequency at pin 20.

4. If a coils used in the MHz oscillator adjust to give the correct horizontal frequency at pin5 (use pin40 as a scope trigger) when the oscillator has phase locked, adjust the coil to give the correct waveform at pin 2.

5. Switch to "REMOTE" MODE (pin 1 = +5 V)
6. Adjust the 100 k potentiometer at pin 9 to give the correct sub carrier frequency to within 50 Hz at pin20 as in pin2.
7. Feed 1 v p-p composite color video into pin 24. Color burst and composite sync should now appear at pin 27. The color burst will be absent if the 100 k pot was incorrectly adjusted.

5.5.1 MC1378p Clock Oscillator Alignment

Two new circuits are shown in fig 5.3.c to prove the pull-in range and speed of the clock-Oscillator phase locked loop. Fig 5.3.d shows a circuit that has no compromise between the characteristics in both this local and remote modes. Both circuits allow much wider tolerance on the alignment of the Clock Oscillator.

5.5.2 MC1378p Sub Carrier Notch Filter

Cross color can cause annoying rainbow effects on fast luminance edges especially in non interlaced pictures. Fig 5.3.d shows a simple sub carrier notch filter in the luminance delay path of the IC to remove some of the offending cross color artifacts at the expense of luminance bandwidth. The cross color problem can be especially bad when attempting to record on consumer type VCRs because on playback the chroma-horizontal interleaving becomes random. Th notch method is equally effective on PAL or NTSC.

5.5.3 Improved Remote Video Isolation

Fig 5.6 shows the circuit, which helps improve the isolation of the remote Video. Because of certain limitations in the device and its packaging, the cross talk from remote composite video input to composite video output can be troublesome when operating in the Local Mode with a video signal present at the composite video input. Typically, the cross talk is about – 35dB at 4.43 MHz and better at 3.58 MHz Low frequencies are better than – 60 dB. The circuit shows in fig 5.6 will improve the isolation in the Local Mode by an additional 20 – dB.

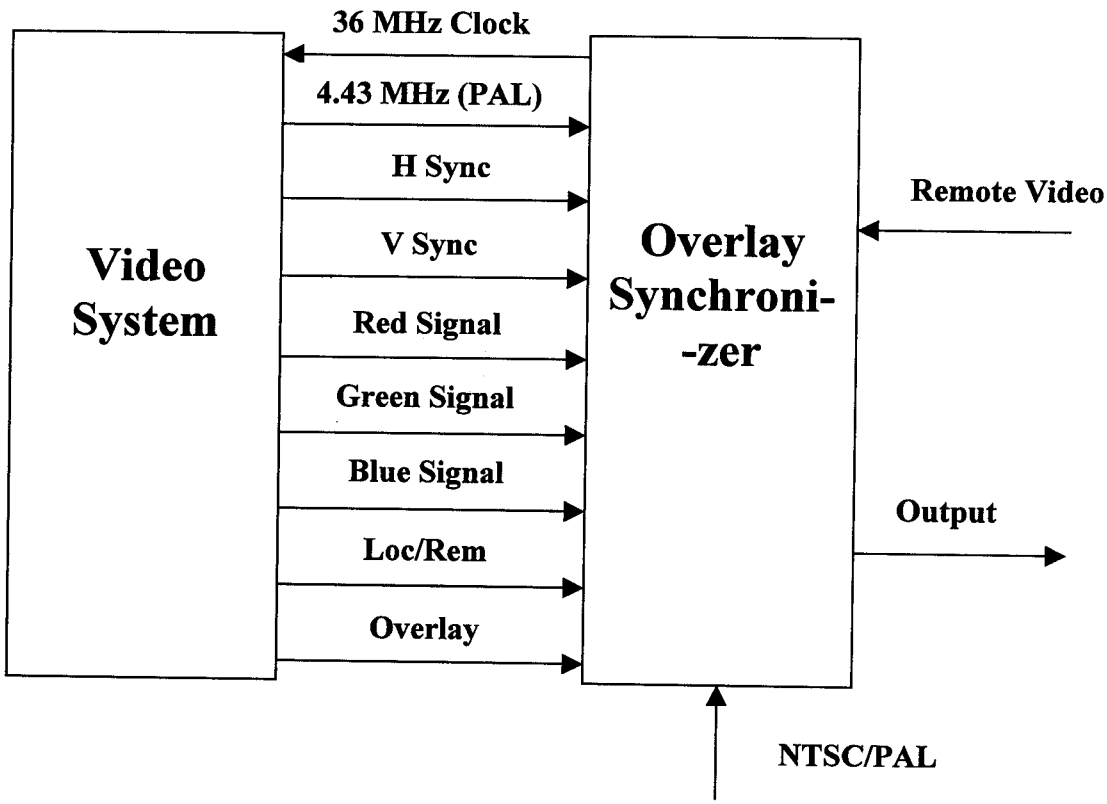


Fig 5.1 Basic Architecture of PC based System

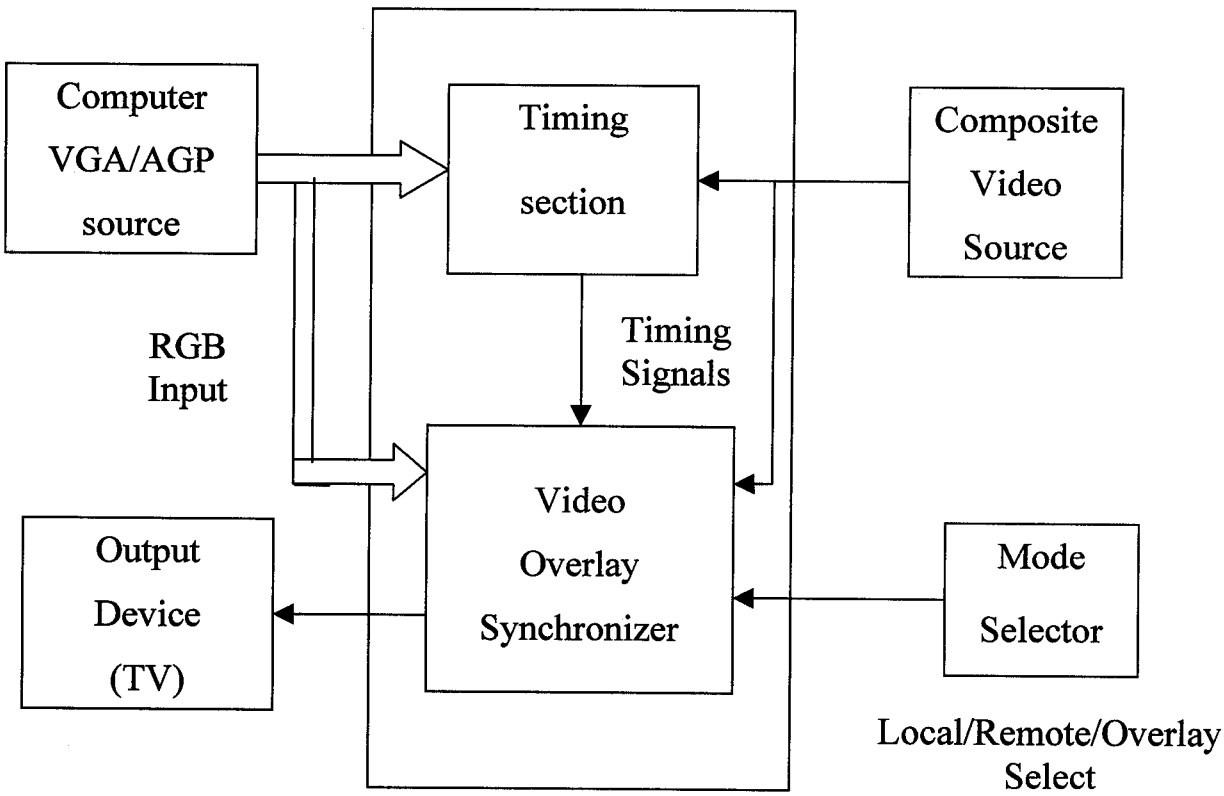


Fig 5.2 Basic Block Diagram of PC based system

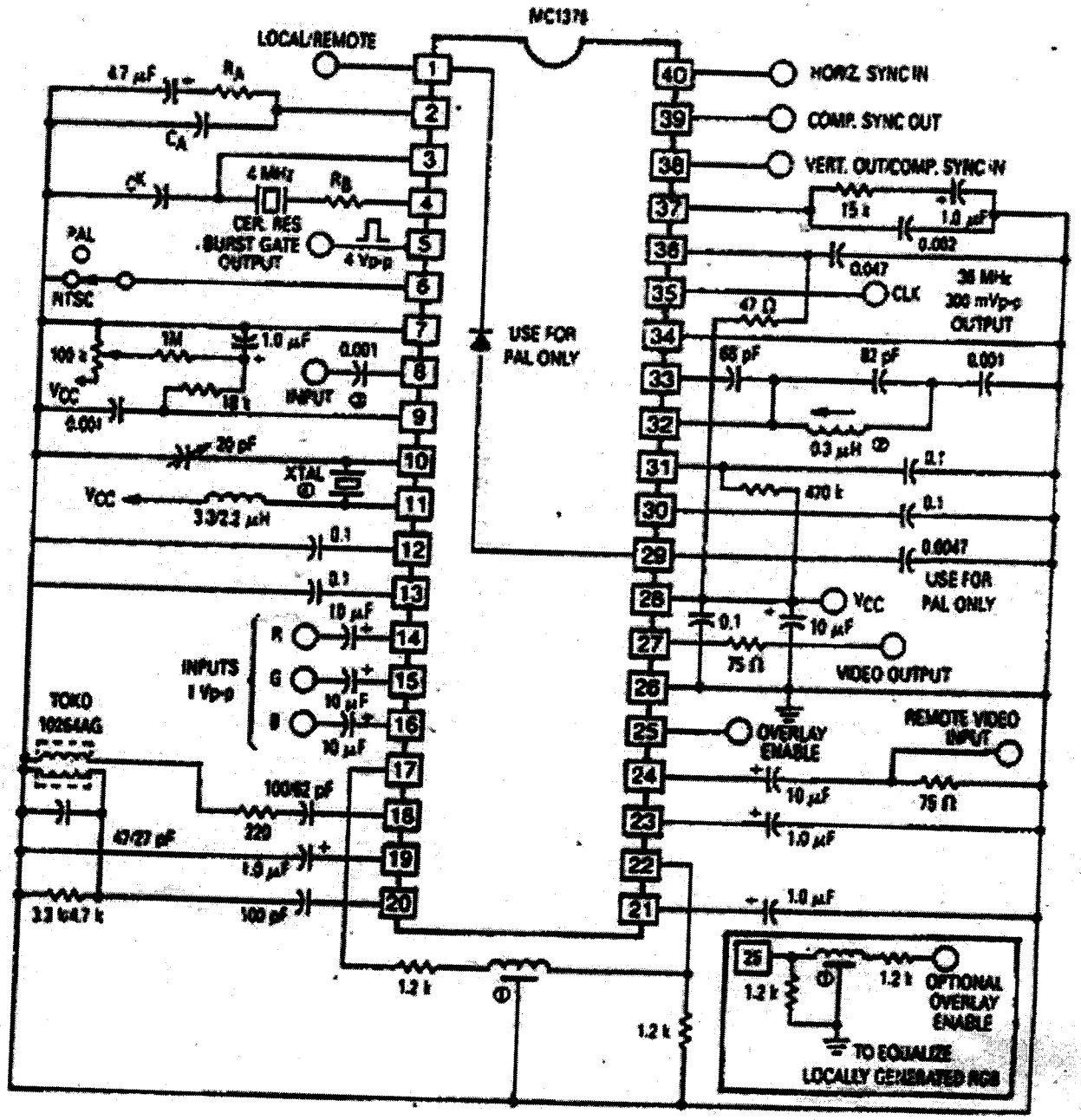


Fig 5.3 Schematic Diagram

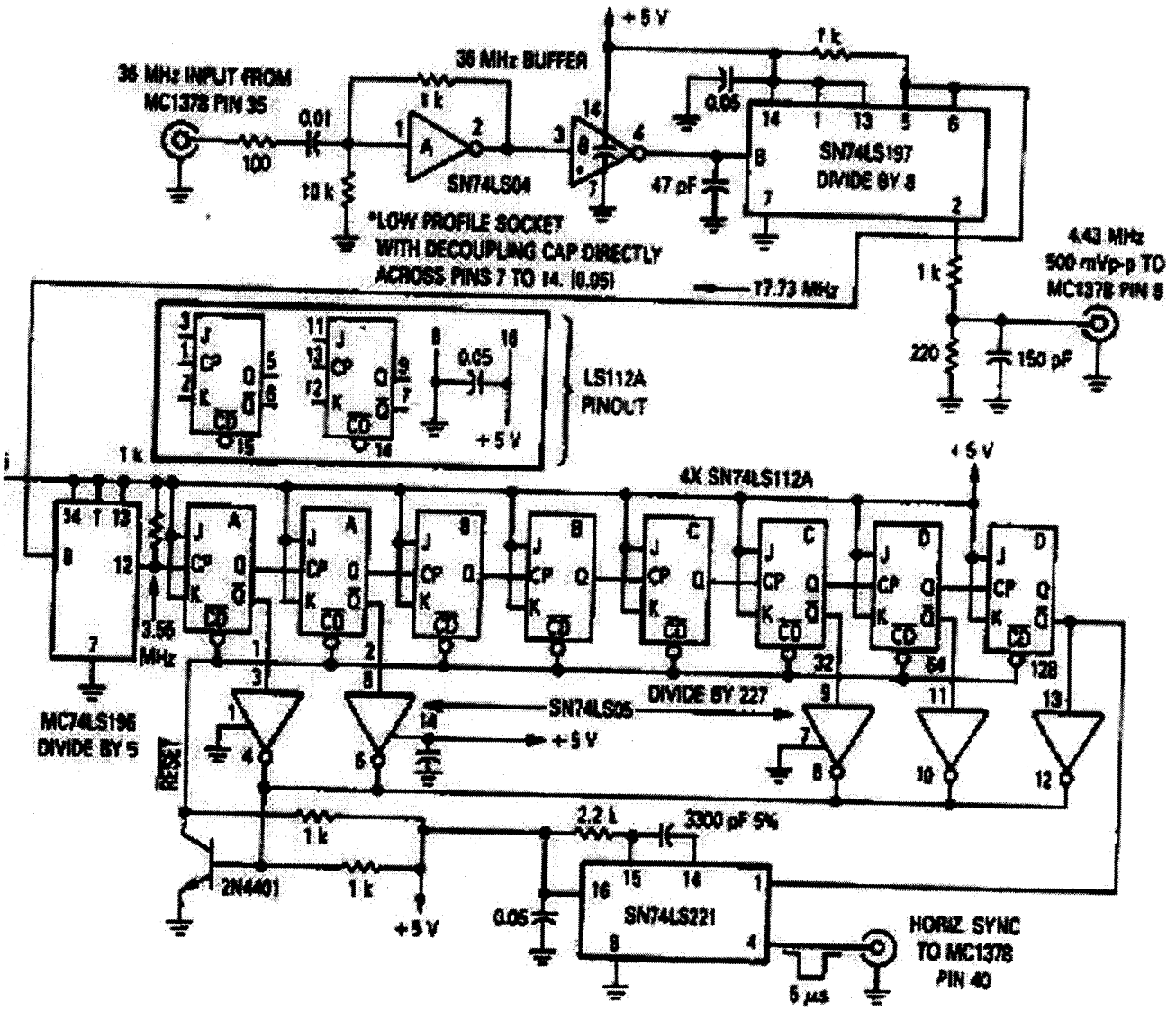


Fig 5.5 Timing Circuitry

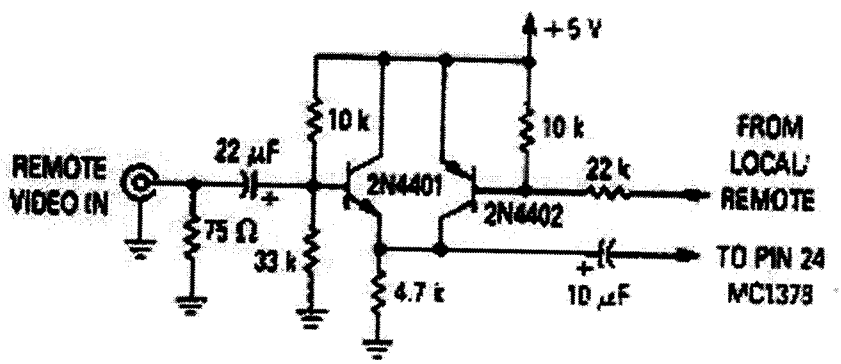


Fig 5.6 Improved Remote Video Isolation Circuit

CHAPTER VI
CONCLUSION

A Video titling equipment using Microprocessor and PC has been developed in the project. The Hardware for PC based system and hardware and software for the Microprocessor based system has been designed and developed.

Multilingual Text can be generated and added to any composite video source by the use of the Microprocessor based system. This system is very economical as it needs the bare minimum capital. But it cannot produce Complex Graphics.

On the other hand Graphics of any sort generated in a computer can be overlaid on any composite video source by the use of the PC based system. This system is thus much flexible and versatile. But the investment on such a system is higher.

The advantages of the developed system are

- The existing set up is not disturbed
- Multilingual text can be generated using the microprocessor system with control on scroll speed, height of the overlay and starting position of the same.

REFERENCES

REFERENCES

1. Ramesh S. Goankar, " Microprocessor Architecture, Programming and Applications with 8085/8080A", New Age International Limited, Publishers, 1995.
2. B. Govindarajulu, " IBM PC and Clones , Hardware, Trouble Shooting and maintenance" , Tata McGraw – Hill Publishing Company Limited, 1993.
3. Albert Paul Malvino, Donald P. Leach , " Digital Principles And Applications", Tata McGraw – Hill Publishing Company Limited, 1994.
4. William H. Gothmann , "Digital Electronics", Printice Hall Of India Pvt. Ltd., 1987.
5. George Kennedy, " Electronic Communication Systems", Tata McGraw – Hill Publishing Company Limited, 1994.
6. " Motorola Semiconductors Master Selection Guide", Motorola Inc. 1994.
7. "MCS 85 user manual" Intel corporation 1976-77.

APPENDIX A

APPENDIX A

Program

LOOP 1 :	DI:	disable interrupts
	LXI SP,FFFF:	load stack pointer
	LXI H,0000 :	load HL pair
	SHLD ,FE24 :	store HL content
	MVI A,00 :	initialize Acc
	STA ,FE 1E :	store Acc content
	STA ,FE 1F :	store Acc content
	STA ,FE26 :	store Acc content
	LXI H, FE00 :	load HL pair
	MVI C,1E :	move data to C reg
LOOP 1 :	MOV M, A :	move Acc to memory
	INX H :	increment HL pair
	DCR C :	decrement C reg
	JNZ LOOP 1 :	jump on no zero
	STA ,FE20 :	store Acc content
	STA ,FE22 :	store Acc content
	STA ,FE23 :	store Acc content
	MVI A, OB :	move data to Acc
	SIM :	set interrupt mask
	EI :	enable interrupt
	MVI A, 00 :	initialize Acc
	LXI H, 8000 :	load HL pair

	MVI C, 07 :	move content to C reg
LOOP 2 :	MOV M, A :	move Acc to memory
	INX H :	increment HL pair
	DCR C :	decrement C reg
	JNZ LOOP 2 :	jump on no zero
	LXI H , 8000 :	load HL pair
	MOV A , H :	move H reg to Acc
	ADI , 70 :	add immediately
	MOV H, A :	move Acc to H reg
	MVI M ,FE :	move content to memory
	MVI C , 10 :	move content to C reg
LOOP 3 :	INX H :	increment HL pair
	MVI M , 00 :	initialize memory
	DCR C :	decrement C reg
	JNZ LOOP 3 :	jump on no zero
	LXI H, 0000 :	initialize HL pair
	SHLD , FE29 :	store HL content in location
	CALL SUB 1 :	call subroutine
	CALL SUB 2 :	call subroutine
	MVI B,09 :	move content B reg
	MVI C , 00 :	initialize C reg
LOOP 4 :	DCR C :	decrement C reg
	JNZ LOOP 4 :	jump on no zero
	DCR B :	decrement B reg
	JNZ LOOP 4 :	jump on no zero
	DI :	disable interrupt
	MVI A, 00 :	initialize Acc

	STA , FE1E :	store Acc content
	STA , FE1F :	store Acc content
	EI :	enable interrupt
	MVI A, 0F :	move content to Acc
	STA ,FE21 :	store Acc content in location
	LHLD , FE24 :	load HL content
	SHLD , FE27 :	store HL content
	LDA , FE20 :	load Acc with content location
	ANI ,10 :	logical AND Acc with data
	JZ LOOP 5:	jump on zero
	JMP LOOP 6 :	jump to location
LOOP 5 :	LXI H , FDFF :	load HL pair with content
	MVI B , 10 :	load B reg with content
LOOP 8 :	ORA A :	logical OR Acc with content
	MVI C, 20 :	move content to C reg
LOOP 7 :	MOV A,M :	move HL content to Acc
	RAR :	rotate right the Acc content
	MOV M, A:	load Acc to memory
	DCX H :	decrement HL content
	DCR C :	decrement C reg
	JNZ LOOP 7 :	jump on no zero
	DCR B :	decrement B reg
	JNZ LOOP 8 :	jump on no zero
	LDA , FE21 :	load Acc
	ORA A :	logical OR with Acc
	JZ LOOP 9:	jump on zero
	DCR A :	decrement Acc

	STA , FE21 :	store Acc content
	JMP LOOP 10 :	jump to location
LOOP 9 :	LHLD FE29 :	load HL content
	INX H :	increment HL pair
	MOV A, H :	move H content to Acc
	CPI ,70 :	compare Acc content with data
	JNZ LOOP 11 :	jump on no zero
	LXI H, 0000 :	initialize HL pair
	JMP LOOP 12 :	jump to location
LOOP 11 :	LXI D, 8000 :	load DE pair
	MOV A, L :	move L content to Acc
	ADD E :	add Acc with E reg
	MOV E, A :	move Acc to E reg
	MOV A, H :	move H data to Acc
	ADC D :	add with carry the D reg
	MOV D,A :	move Acc to D reg
	LDAX D :	load Acc with DE content
	CPI , FE :	compare Acc with data
	JNZ LOOP 13 :	jump on no zero
	LXI H , 0000 :	initialize HL pair
LOOP 12 :	SHLD , FE29 :	store HL content
	CALL SUB 3 :	call subroutine
	CALL SUB 4 :	call subroutine
	MVI A, 0F :	move content to Acc
	STA , FE21 :	store Acc content
	JMP LOOP 10 :	jump to location
LOOP 13 :	SHLD , FE29 :	store HL content

	MVI A,0F :	move content to Acc
	ADD E :	add Acc with E
	MOV L , A :	move Acc to L REG
	MOV A , D :	move D content to Acc
	ACI ,00 :	add immediately
	MOV H, A ;	move Acc to H reg
	MOV A, M ;	move content of memory to Acc
	LXI D , FC1E :	load DE pair with content
	CALL SUB 5 :	call subroutine
	MVI A , 0F :	move content to Acc
	STA , FE21 :	store Acc content
LOOP 10 :	RIM :	read interrupt mask
	ORA A :	logical OR with Acc
	JP LOOP 14 :	jump on parity
	LDA , 4000 :	load Acc content
LOOP 14 :	LDA , FE20 :	load Acc content
	ANI ,10 :	logical AND with data
	JZ LOOP 15 :	jump on zero
	JMP LOOP 6 :	jump to location
LOOP 15 :	LHLD FE24 :	load HL pair with content
	XCHG :	exchange DE with HL
	LHLD FE27 :	load HL content
	MOV A, L :	move L to Acc
	XRA E :	logical XOR with e reg
	JNZ LOOP 16 :	jump on no zero
	MOV A, H :	move H content to Acc
	XRA D :	logical XOR with D reg

	JZ LOOP 10 :	jump on zero
LOOP 16 :	LHLD FE24 :	load HL content
	SHLD FE27 :	store HL content
	LDA , FE21 :	load Acc with content of location
	ANI ,01 :	logical AND with data
	JNZ LOOP 5 :	jump on no zero
	CALL SUB 6 :	call subroutine
	JMP LOOP 5 :	jump to location
LOOP 6 :	DI :	disable interrupt
	MVI A, 00 :	initialize Acc
	STA , FE1E :	store Acc content
	STA , FE1F :	store Acc content
	EI :	enable interrupt
	LHLD , FE29 :	load HL content
	CALL SUB 7 :	call subroutine
	CALL SUB 8 :	call subroutine
	CALL SUB 9 :	call subroutine
LOOP 23 :	LDA , FE20 :	load Acc content
	ANI , 10 :	logical AND with data
	JZ LOOP 17 :	jump on zero
	MVI A , 01 :	move content to Acc
	RST 1 :	jump to vector location
	JZ LOOP 18 :	jump on zero
LOOP 26 :	MVI A, 00 :	initialize Acc
	RST 1 :	jump to vector location
	CPI , 2E :	compare data with Acc
	JNZ LOOP 19 ;	jump on no zero

	LHLD ,FE29 :	load HL content
	INX H :	increment HL pair
	MOV A,H :	move H content to Acc
	CPI ,70 :	compare data with Acc
	JZ LOOP 18 :	jump on zero to location
	SHLD ,FE29 :	store HL content
	JMP LOOP 20 :	jump to location
LOOP 19 :	CPI ,2F :	compare data with Acc
	JNZ LOOP 21 :	jump on no zero
	LHLD ,FE29 :	load HL content
	MOV A , H ;	move H content to Acc
	ORA L :	logical OR with L
	JZ LOOP 18 :	jump on zero
	DCX H :	decrement HL pair
	SHLD ,FE29 :	store HL content
	JMP LOOP 20 :	jump to location
LOOP 21 :	CPI , FF :	compare data with Acc
	JNZ LOOP 22 :	jump on no zero
	LHLD FE29 :	load HL content
	MOV A, L :	move L content to Acc
	ADI , 08 :	add immediately
	MOV L, A :	move Acc to L
	MOV A, H :	move H to Acc
	ACI , 00 :	add on carry
	MOV H, A :	move Acc to H reg
	CPI , 70 :	compare data with Acc
	JZ LOOP 23 :	jump on zero

	JMP LOOP 24 :	jump to location
LOOP 22 :	MOV D, A :	move Acc to D reg
	LXI B , 8000 :	load BC pair
	LHLD FE29 :	load HL content
	MOV A, L :	move L to Acc
	ADI , 07 :	add immediately the data
	MOV L , A :	move Acc to L reg
	MOV A, H :	move H content to Acc
	ACI , 00 :	add on carry
	MOV H , A :	move Acc to H
	CPI , 70 :	compare data with Acc
	JZ LOOP 20 :	jump on zero
	MOV A , L :	move L to Acc
	ADD C :	add C reg with Acc
	MOV L, A :	move Acc to L
	MOV A, H :	move H to Acc
	ADC B :	add B with Acc
	MOV H, A :	move Acc to H reg
	MOV M, D :	move D reg to memory
	LHLD FE29 :	load HL content
	INX H :	increment HL pair
	SHLD FE29 :	store HL content
LOOP 20 :	MVI A, 01 :	move data to Acc
	RST 1 :	jump to vector location
	JZ LOOP 25 :	jump on zero
	JMP LOOP 26 :	jump to location
LOOP 25 :	LHLD FE29 :	load HL content

	CALL SUB 1 :	call subroutine
	CALL SUB 5 :	call subroutine
	CALL SUB 4 :	call subroutine
LOOP 18 :	JMP LOOP 23 :	jump to location
	SHLD FE29 :	store HL content
	CALL SUB1 :	call subroutine
	CALL SUB5:	call subroutine
	JMP LOOP 23:	jump to location
	CALL SUB5:	call subroutine
LOOP 17:	MVI A, 0F :	move data to Acc
	STA , FE21:	store Acc content
	JMP , LOOP 5:	jump to location
	LXI H,FC0E :	load HL pair
SUB 5 :	MVI C, 10:	move data to C reg
	MOV A,M :	move memory to Acc
	CMA :	complement Acc
	MOV M , A :	move Acc to memory
LOOP 27 :	INX H :	increment HL pair
	MOV A , M :	move memory to Acc
	CMA :	complement Acc
	MOV M , A :	move Acc to memory
	MVI A,1F :	move data to Acc
	MOV M,A :	move Acc to memory
	MVI A,1F :	move data to Acc
	ADD L:	add with L content
	MOV L,A :	move Acc to L reg
	MOV A,H :	move H content to Acc

	ACI , 00 :	add with carry
	MOV H , A :	move Acc to H reg
	DCR C :	decrement C reg
	JNZ LOOP 27 :	jump on no zero
	RET :	return to main
SUB 4:	PUSH PSW:	push PSW to stack
	PUSH B :	push B to stack
	PUSH D :	push D to stack
	PUSH H :	push H to stack
	LXI H, FC00 :	load HL pair
	LXI D, 4000 :	load DE pair
	MVI C. 00 :	initialize C reg
LOOP 28 :	RIM :	read interrupt mask
	ORA A :	logical OR with Acc
	JM LOOP 28 :	jump on negative
LOOP 29 :	RIM :	read interrupt mask
	ORA A :	logical OR with Acc itself
	JP LOOP 29 :	jump on parity
LOOP 30 :	MOV A , M :	move memory content to Acc
	STAX D :	store Acc data in DE pair
	INX H :	increment HL pair
	INX D :	increment DE pair
	DCR C:	decrement C reg
	JNZ LOOP 30 :	jump on no zero
LOOP 31 :	MOV A, M :	move memory content to Acc
	STAX D :	store Acc to DE pair
	INX H :	increment HL pair

```
                INX  D :          increment DE PAIR
                DCR  C :          decrement C reg
                JNZ  LOOP 31 :    jump on no zero
                POP  H :          retrieve H from stack
                POP  D :          retrieve D from stack
                POP  B :          retrieve B from stack
                POP  PSW :        retrieve PSW from stack
                RET :             return
SUB 1 :         PUSH PSW :        push PSW to stack
                PUSH B :          push B to stack
                PUSH D :          push D to stack
                PUSH H :          push H to stack
                LXI  B, 8000 :    load BE pair
                MOV  A,L :        move L to Acc reg
                ADD C :           add C reg to Acc
                MOV  L,A :        move Acc to L reg
                MOV  A,H :        move H to Acc reg
                ADC  B :          add B reg & carry to Acc
                MOV  H,A :        move Acc to H reg
                LXI  D,FC00 :     load DE pair
                MVI  C,10 :       move data to C reg
LOOP 32 :       MOV  A,M :        move memory  to Acc
                CALL SUB 3 :      call subroutine
                INX  H:           increment HL pair
                INX  D :          increment DE pair
                INX  D :          increment DE pair
                DCR  C :          decrement C reg
```

	MOV A,H :	move H to Acc reg
	ADC B :	add B with Acc
	MOV H,A :	move Acc to H reg
	MVI C,10 :	move data to C reg
LOOP 33 :	MOV A,M :	move Memory to Acc reg
	STAX D :	store Acc
	INX H :	increment HL pair
	INX D :	increment DE pair
	MOV A,M :	move Memory to Acc reg
	STAX D :	store Acc
	INX H:	increment HL pair
	MOV A,E :	move E to Acc reg
	ADI , 1F :	add Data with Acc
	MOV E,A :	move Acc to E reg
	MOV A,D :	move D to Acc reg
	ACI , 00 :	add data & carry with Acc
	MOV D,A :	move Acc to D reg
	DCR C :	decrement C reg
	JNZ , LOOP 33 :	jump on no zero
	POP B :	retrieve B from stack
	POP D :	retrieve D from stack
	POP H :	retrieve H from stack
	RET :	return

TRAP :	DI :	disable interrupts
	PUSH PSW :	push PSW to stack
	PUSH H:	push H to stack
	LHLD FE24 :	load HL content
	INX H:	increment HL pair
	SHLD FE24 :	store HL content
	POP H :	retrieve H content
	POP PSW :	retrieve PSW content
	EI :	enable interrupt
	RET :	return
RST 1 :	EI :	enable interrupt
	PUSH H :	push H to stack
	ORA A :	logical OR with Acc
	JZ LOOP 34 :	jump on zero
	DCR A:	decrement Acc
	JZ LOOP 35 :	jump on zero
	JMP LOOP 36 :	jump on location
LOOP 34 :	EI :	enable interrupt
	NOP :	no operation
	NOP :	no operation
	NOP :	no operation
	NOP :	no operation
	DI :	disable interrupt
	LDA , FE1E :	load Acc
	LXI H,FE1E :	load HL pair
	CMP M :	compare memory with Acc

	JZ LOOP 34 :	jump on zero
	MOV L,A :	move Acc to L reg
	INR A :	increment Acc
	CPI , 1E :	compare data
	JNZ 0126 :	jump on no zero
	MVI A, 00 :	initialize Acc
LOOP 37 :	STA , FE1E :	store Acc content
	MOV A , M :	move memory to Acc
	JMP LOOP 36 :	jump to location
LOOP 35 :	DI :	disable interrupt
	LDA , FE1E :	load Acc
	LXI H,FE1E :	load HL pair
	CMP M :	compare with memory
	MOV L,A :	move Acc to L reg
	MOV A,M :	move memory to Acc
LOOP 36 :	POP H :	retrieve H from stack
	EI :	enable interrupt
	RET :	return

RST 6.5 :	DI	disable interrupt
	PUSH PSW :	push PSW to stack
	PUSH B :	push B to stack
	PUSH D :	push D to stack
	PUSH H :	push H to stack
	IN , 00 :	input data
	MOV B,A :	move Acc to B

	CPI , FF :	compare data
	JNZ LOOP 38 :	jump on no zero
	JMP LOOP 39 :	jump to location
LOOP 38 :	LDA , FE26 :	load Acc
	CMP B :	compare with B
	JZ LOOP 39 :	jump on zero
	MOV A,B :	move B to Acc
	STA , FE26 :	store Acc content
	MVI C,08 :	move data to C
	MVI D,00 :	move data to D
LOOP 40 :	MOV A,B :	move B content to Acc
	RAR :	rotate right
	MOV B,A :	move Acc to B
	MOV A,D :	move D to Acc
	RAL :	rotate left
	MOV D,A :	move Acc to D reg
	DCR C :	decrement C reg
	JNZ LOOP 40 :	jump on no zero
	MOV A,D :	move D to Acc
	MOV B,A :	move Acc to B
	ANI , 7F :	logical AND with data
	LXI H , 013A :	load HL pair
	MVI C,08 :	load C reg
LOOP 42 :	CMP M :	compare with memory
	JZ LOOP 41 :	jump on zero
	INX H :	increment HL pair
	DCR C :	decrement C reg

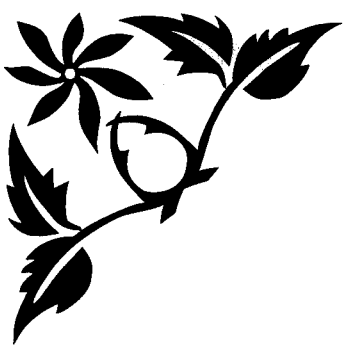
	JNZ LOOP 42 :	jump on no zero
	MOV A,B :	move B to Acc
	CPI , 0F :	compare with data
	JNZ LOOP 43 :	jump on no zero
	LDA , FE20 :	load Acc
	ANI , 03 :	logical AND with data
	MVI A,2F :	move data to Acc
	JZ LOOP 44 :	jump on zero
	MVI A,FF :	move data to Acc
	JMP LOOP 44 :	jump to location
LOOP 43 :	MOV A,B :	move B to Acc
	JMP LOOP 45 :	jump to location
LOOP 41 :	MVI A , 08 :	move data to Acc
	SUB C:	subtract C from Acc
	LXI H , 0142 :	load HL pair
	ADD L :	add L with Acc
	MOV L,A :	move Acc to L
	MOV A,B :	move B to Acc
	ANI , 80 :	logical AND with data
	JNZ LOOP 46 :	jump on no zero
	MOV A,M :	move memory to Acc
	CPI , 10 :	compare data
	JZ LOOP 47 :	jump on zero
	JNC LOOP 47 :	jump on no carry
	LDA , FE20 :	load Acc
	ORA M :	logical OR with memory
	STA , FE20 :	store Acc content

	JMP LOOP 39 :	jump to location
LOOP 47 :	LDA FE20 :	load Acc
	XRA M :	logical XOR with memory
	STA , FE20 :	store Acc content
	JMP LOOP 39 :	jump to location
LOOP 46 :	MOV A,M :	move memory to Acc
	CPI , 10 :	compare data
	JZ LOOP 48 :	jump on zero
	JNC LOOP 48 :	jump on no carry
	CMA :	complement Acc
	MOV C,A :	move Acc to C
	LDA , FE20 :	load Acc
	ANA C :	logical AND with C
	STA , FE20 :	store Acc content
LOOP 48 :	JMP LOOP 39 :	jump to location
LOOP 45 :	ANI , 80 :	logical AND
	JNZ LOOP 39 :	jump on no zero
	LDA , FE20 :	load Acc
	MOV C,A :	move Acc to C reg
	ANI , 20 :	logical AND
	JZ LOOP 49 :	jump on zero
	MOV A,B :	move B to Acc
	CPI , 19 :	compare data with Acc
	JZ LOOP 50 :	jump on zero
	CPI , 1A :	compare data with Acc
	JZ LOOP 51 :	jump on zero
	CPI , 27 :	compare data with Acc

	JZ LOOP 52 :	jump on zero
	ANI , C0 :	logical AND
	JNZ LOOP 39 :	jump on no zero
	MOV A,C :	move C to Acc
	ANI , 03 :	logical AND with data
	JNZ LOOP 53 :	jump on no zero
	LDA , FE22 :	load Acc
	ORA B :	logical OR
	MOV L,A :	move Acc to L reg
	LDA FE23 :	load Acc
	MOV H,A :	move Acc to H reg
	JMP LOOP 54 :	jump to location
LOOP 53 :	LDA , FE22 :	load Acc
	ORA B :	logical OR with B reg
	MOV L,A :	move Acc to L reg
	LDA , FE23 :	load Acc
	ORI , 02 :	logical OR with data
	MOV H,A :	move Acc to H reg
LOOP 54 :	MVI A,00 :	initialize Acc
	STA , FE23:	store Acc content
	STA , FE22 :	store Acc
	LXI D , 0350 :	load DE pair
	MOV A,L :	move L to Acc
	ADD E :	add Acc with E reg
	MOV L,A :	move Acc to L reg
	MOV A,H :	move H reg to Acc
	ADC D :	add D reg with carry

	MOV H,A :	move Acc to H reg
	MOV A,M :	move memory to Acc
	ORA A:	logical OR with Acc
	JZ LOOP 39 :	jump on zero
	CPI , FF :	compare data with Acc
	JZ LOOP 39 :	jump on zero
	JMP LOOP 44 :	jump to location
LOOP 50 :	MVI A,80 :	move data to Acc
	STA , FE22 :	store Acc content
	MOV A,C :	move C reg to Acc
	ANI , 03 :	logical AND with data
	MVI A,00 :	initialize Acc
	JZ LOOP 55 :	jump on zero
	MVI A,01 :	move data to Acc
LOOP 55 :	STA , FE23 :	store Acc content
	JMP LOOP 39 :	jump to location
LOOP 51 :	MVI A,40 :	move data to Acc
	STA , FE22 :	store Acc content
	MOV A,C :	move c reg to Acc
	ANI , 03 :	logical AND with data
	MVI A,00 :	initialize Acc
	JZ LOOP 56 :	jump on zero
	MVI A,01 :	move data to Acc
LOOP 56 :	STA , FE23 :	store Acc content
	JMP LOOP 39 :	jump to location
LOOP 52 :	MVI A,C0 :	move data to Acc
	STA , FE22 :	store Acc content

	MOV A,H :	move H reg to Acc
	ACI , 00 :	add data with carry
	MOV H,A :	move Acc to H reg
	MOV A,M :	move memory to Acc
	ORA A :	logical OR with Acc
	JZ LOOP 39 :	jump on zero
	CPI , FF :	compare data with Acc
	JZ LOOP 39 :	jump on zero
LOOP 44 :	MOV B,A :	move Acc to B reg
	LDA , FEIF :	load Acc
	LXI H , FE1E :	load HL pair
	MOV D,A :	move Acc to D reg
	INR A:	increment Acc
	CPI , 1E :	compare data with Acc
	JNZ LOOP 61 :	jump on no zero
	MVI A,00 :	initialize Acc
LOOP 61 :	CMP M :	compare memory with Acc
	JZ LOOP 39 :	jump on zero
	STA , FE1F :	store Acc content
	MOV L,D :	move D reg to L reg
	MOV M,B :	move memory to B reg
LOOP 39 :	POP H :	retrieve H content from stack
	POP D :	retrieve D content from stack
	POP B :	retrieve B content from stack
	POP PSW :	retrieve PSW from stack
	RET :	return

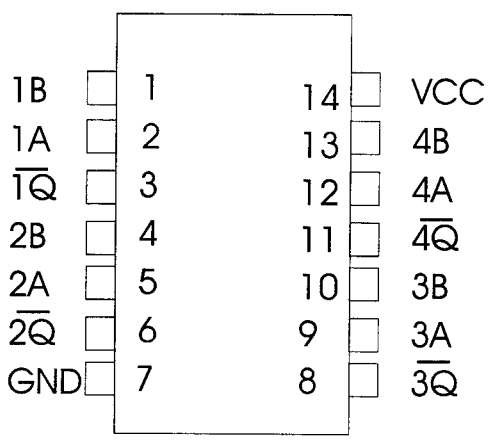


APPENDIX B

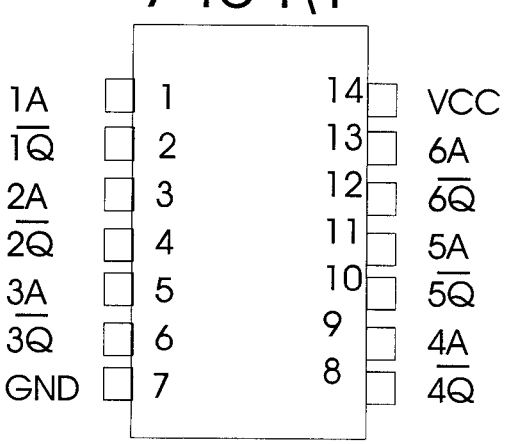
APPENDIX B

PIN DETAILS

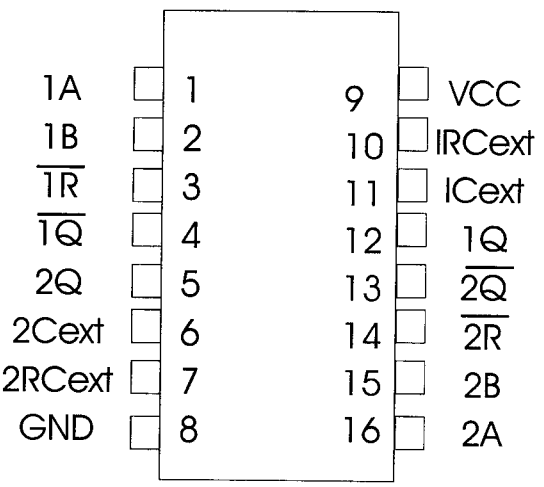
7400\1



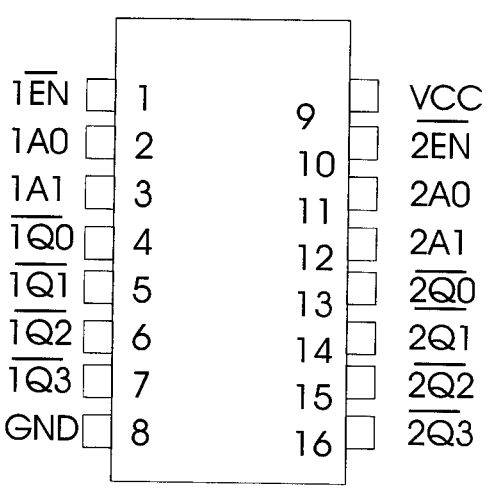
7404\1



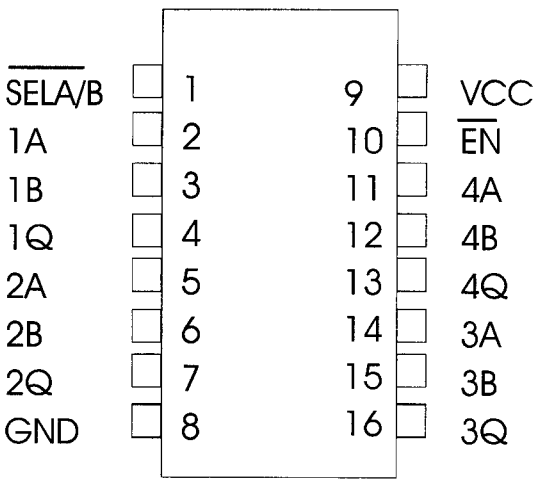
74123


1A 1
1B 2
1R 3
1Q 4
2Q 5
2Cext 6
2RCext 7
GND 8
9 VCC
10 IRCext
11 ICext
12 1Q
13 2Q
14 2R
15 2B
16 2A

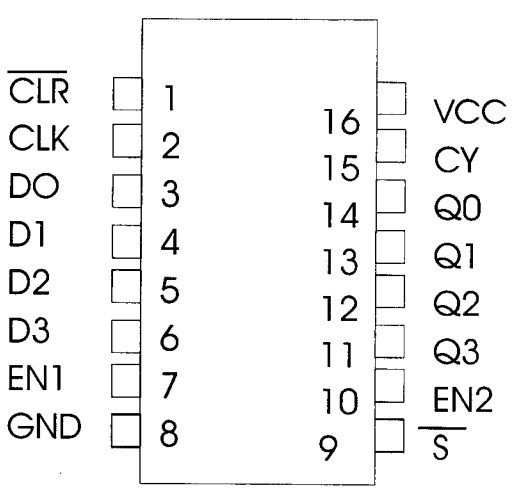
74139


1EN 1
1A0 2
1A1 3
1Q0 4
1Q1 5
1Q2 6
1Q3 7
GND 8
9 VCC
10 2EN
11 2A0
12 2A1
13 2Q0
14 2Q1
15 2Q2
16 2Q3

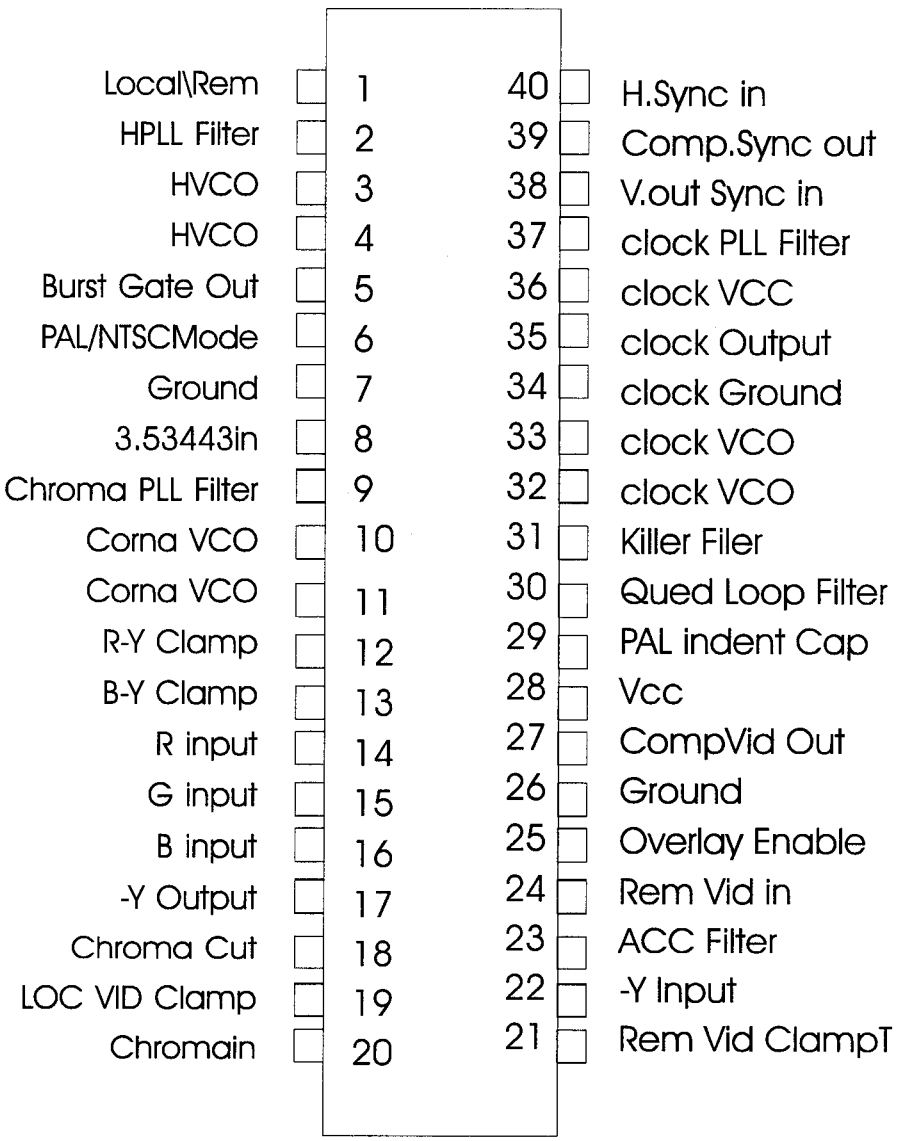
74157


SELA/B 1
1A 2
1B 3
1Q 4
2A 5
2B 6
2Q 7
GND 8
9 VCC
10 EN
11 4A
12 4B
13 4Q
14 3A
15 3B
16 3Q

74161


CLR 1
CLK 2
DO 3
D1 4
D2 5
D3 6
EN1 7
GND 8
16 VCC
15 CY
14 Q0
13 Q1
12 Q2
11 Q3
10 EN2
9 S

MC 1378 P


Local\Rem 1
HPLL Filter 2
HVCO 3
HVCO 4
Burst Gate Out 5
PAL/NTSCMode 6
Ground 7
3.53443in 8
Chroma PLL Filter 9
Corna VCO 10
Corna VCO 11
R-Y Clamp 12
B-Y Clamp 13
R input 14
G input 15
B input 16
-Y Output 17
Chroma Cut 18
LOC VID Clamp 19
Chromain 20
40 H.Sync in
39 Comp.Sync out
38 V.out Sync in
37 clock PLL Filter
36 clock VCC
35 clock Output
34 clock Ground
33 clock VCO
32 clock VCO
31 Killer Filer
30 Qued Loop Filter
29 PAL indent Cap
28 Vcc
27 CompVid Out
26 Ground
25 Overlay Enable
24 Rem Vid in
23 ACC Filter
22 -Y Input
21 Rem Vid ClampT